



Reference Manual

Replication Server®

15.5

DOCUMENT ID: DC32410-01-1550-01

LAST REVISED: March 2010

Copyright © 2010 by Sybase, Inc. All rights reserved.

This publication pertains to Sybase software and to any subsequent release until otherwise indicated in new editions or technical notes. Information in this document is subject to change without notice. The software described herein is furnished under a license agreement, and it may be used or copied only in accordance with the terms of that agreement.

To order additional documents, U.S. and Canadian customers should call Customer Fulfillment at (800) 685-8225, fax (617) 229-9845.

Customers in other countries with a U.S. license agreement may contact Customer Fulfillment via the above fax number. All other international customers should contact their Sybase subsidiary or local distributor. Upgrades are provided only at regularly scheduled software release dates. No part of this publication may be reproduced, transmitted, or translated in any form or by any means, electronic, mechanical, manual, optical, or otherwise, without the prior written permission of Sybase, Inc.

Sybase trademarks can be viewed at the Sybase trademarks page at <http://www.sybase.com/detail?id=1011207>. Sybase and the marks listed are trademarks of Sybase, Inc. ® indicates registration in the United States of America.

Java and all Java-based marks are trademarks or registered trademarks of Sun Microsystems, Inc. in the U.S. and other countries.

Unicode and the Unicode Logo are registered trademarks of Unicode, Inc.

All other company and product names used herein may be trademarks or registered trademarks of their respective companies.

Use, duplication, or disclosure by the government is subject to the restrictions set forth in subparagraph (c)(1)(ii) of DFARS 52.227-7013 for the DOD and as set forth in FAR 52.227-19(a)-(d) for civilian agencies.

Sybase, Inc., One Sybase Drive, Dublin, CA 94568.

Contents

About This Book	xv
 CHAPTER 1	
Introduction to the Replication Command Language	1
Data replication commands	2
Table replication definition commands	2
Function replication definition commands	3
Database replication definition commands	4
Publication commands	5
Subscription commands	6
User commands	9
Database interface commands	10
Database connection commands	10
Error class commands	11
Function and function string commands	11
Warm standby database commands	13
Gateway commands	14
Route commands	14
System information commands	15
Partition commands	17
Configuration commands	17
System administration commands	18
Recovery commands	20
 CHAPTER 2	
Topics	21
Datatypes	21
Exact numeric (integer) datatypes	22
Exact numeric (decimal) datatypes	23
Approximate numeric (floating point) datatypes	24
Character datatypes	24
Money datatypes	25
Date/time, and date and time datatypes	25
Binary datatypes	29
Bit datatype	31

Unicode datatypes.....	31
Java datatypes	33
opaque datatypes.....	34
Datatype definitions.....	34
Identifiers.....	35
Name space for identifiers.....	37
Reserved words	38
Support for Adaptive Server	40
Character set support.....	40
Sort order support	41
Message language support.....	42
Extended page- and column-size support.....	42
Mixed-version environments	42

CHAPTER 3	Replication Server Commands.....	43
	abort switch	50
	activate subscription.....	51
	add partition	54
	admin config.....	54
	admin disk_space	58
	admin echo.....	59
	admin get_generation	60
	admin health.....	61
	admin log_name.....	62
	admin logical_status.....	63
	admin pid.....	65
	admin quiesce_check.....	66
	admin quiesce_force_rsi	67
	admin rssid_name.....	68
	admin schedule	69
	admin security_property.....	70
	admin security_setting	71
	admin set_log_name.....	72
	admin show_connection_profiles	72
	admin show_connections.....	76
	admin show_function_classes	78
	admin show_route_versions	79
	admin show_site_version.....	80
	admin sqm_readers	81
	admin stats.....	83
	admin stats, backlog	87
	admin stats, cancel	89
	admin stats, {md mem mem_in_use}.....	89
	admin stats, reset.....	90

admin stats, status	91
admin stats, {tps cps bps}	91
admin time.....	94
admin translate.....	95
admin verify_repserver_cmd	97
admin version.....	99
admin who.....	100
admin who_is_down.....	117
admin who_is_up	118
allow connections	119
alter applied function replication definition	119
alter connection	123
alter connector	146
alter database replication definition.....	148
alter error class	150
alter function.....	153
alter function replication definition	155
alter function string.....	159
alter function string class.....	160
alter logical connection.....	162
alter partition	166
alter queue	168
alter replication definition	170
alter request function replication definition	180
alter route	184
alter schedule.....	192
alter user	193
assign action	194
check publication.....	199
check subscription.....	201
configure connection	204
configure logical connection	205
configure replication server	206
configure route	221
connect.....	221
create applied function replication definition	224
create article.....	231
create connection	235
create connection using profile	241
create database replication definition.....	248
create error class	254
create function.....	257
create function replication definition.....	259
create function string.....	265

create function string class.....	279
create logical connection.....	282
create partition	284
create publication	286
create request function replication definition	290
create replication definition	296
create route	313
create schedule.....	318
create subscription	322
create user	334
define subscription	335
disconnect	342
drop article	343
drop connection.....	345
drop database replication definition	345
drop error class	347
drop function	348
drop function replication definition.....	349
drop function string.....	350
drop function string class	352
drop logical connection	353
drop partition	354
drop publication	355
drop replication definition	357
drop route.....	359
drop schedule.....	361
drop subscription.....	362
drop user	367
grant	368
ignore loss.....	369
move primary	370
rebuild queues.....	372
resume connection	374
resume distributor	377
resume log transfer	378
resume queue	379
resume route	380
revoke	381
set	382
set log recovery	385
set proxy.....	386
show connection	386
show server	387
shutdown.....	388

suspend connection	389
suspend distributor	390
suspend log transfer	391
suspend route	392
switch active	393
sysadmin apply_truncate_table	394
sysadmin cdb	396
sysadmin dropdb	404
sysadmin dropldb	405
sysadmin drop_queue	406
sysadmin drops	407
sysadmin dump_file	408
sysadmin dump_queue	409
sysadmin dump_thread_stacks	414
sysadmin dump_tran	416
sysadmin erssd	419
sysadmin fast_route_upgrade	422
sysadmin hibernate_off	423
sysadmin hibernate_on	425
sysadmin issue_ticket	426
sysadmin log_first_tran	428
sysadmin purge_all_open	429
sysadmin purge_first_open	431
sysadmin purge_route_at_replicate	433
sysadmin restore_dsi_saved_segments	434
sysadmin set_dsi_generation	435
sysadmin site_version	436
sysadmin skip_bad_repserver_cmd	439
sysadmin sqm_purge_queue	441
sysadmin sqm_unzap_command	442
sysadmin sqm_unzap_tran	443
sysadmin sqm_zap_command	446
sysadmin sqm_zap_tran	447
sysadmin sqt_dump_queue	450
sysadmin system_version	454
validate publication	457
validate subscription	458
wait for create standby	461
wait for delay	462
wait for switch	463
wait for time	464

CHAPTER 4	Replication Server System Functions	465
	rs_batch_end	465

rs_batch_start	467
rs_begin	468
rs_check_repl.....	469
rs_commit.....	469
rs_datarow_for_writetext.....	472
rs_delete	474
rs_dsi_check_thread_lock.....	474
rs_dumpdb	476
rs_dumptran	479
rs_get_charset	483
rs_get_errormode.....	484
rs_get_lastcommit.....	485
rs_get_sortorder.....	487
rs_get_textptr	488
rs_get_thread_seq	489
rs_get_thread_seq_noholdlock	490
rs_initialize_threads	491
rs_insert	492
rs_marker	493
rs_non_blocking_commit	494
rs_non_blocking_commit_flush.....	495
rs_raw_object_serialization.....	496
rs_repl_off	497
rs_repl_on	498
rs_rollback.....	499
rs_select.....	500
rs_select_with_lock	502
rs_set_ciphertext.....	503
rs_set_dml_on_computed.....	504
rs_set_isolation_level.....	505
rs_set_quoted_identifier.....	506
rs_setproxy.....	506
rs_sqldml.....	507
rs_textptr_init.....	509
rs_ticket_report	510
rs_triggers_reset	511
rs_truncate	513
rs_update	516
rs_update_threads	518
rs_usedb	519
rs_writetext.....	520

CHAPTER 5	Adaptive Server Commands and System Procedures	523
	dbcc dbrepair	524

dbcc gettrunc.....	525
dbcc settrunc.....	527
set replication	530
set repmode	531
set repthreshold	532
sp_configure 'enable rep agent threads'	536
sp_config_rep_agent.....	537
sp_help_rep_agent	547
sp_reptostandby.....	553
sp_setrepcol.....	560
sp_setrepdbmode	563
sp_setrepdefmode	566
sp_setreplicate	569
sp_setrepproc	571
sp_setreptable.....	574
sp_start_rep_agent	577
sp_stop_rep_agent	580

CHAPTER 6

RSSD Stored Procedures	581
rs_capacity	582
rs_delexception	583
rs_dump_stats.....	583
rs_fillcaptable	587
rs_helpclass	590
rs_helpclassfstring	592
rs_helpcounter	593
rs_helpdb	596
rs_helpdbrep	597
rs_helpdbsub.....	599
rs_helperror.....	600
rs_helpexception	601
rs_helpfstring.....	602
rs_helpfunc.....	604
rs_helppartition.....	606
rs_helppub	607
rs_helppubsub.....	609
rs_helpprep	612
rs_helpprepdb	619
rs_helpprepversion.....	620
rs_helproute	622
rs_helpsub.....	624
rs_helpuser	626
rs_helpreptable	627
rs_init_erroractions.....	628

	rs_send_repserver_cmd	629
	rs_ticket.....	632
	rs_zeroltm	634
CHAPTER 7	Executable Programs	635
	repserver	636
	rs_subcmp.....	642
CHAPTER 8	Replication Server System Tables	661
	rs_articles	662
	rs_classes	663
	rs_columns	664
	rs_config.....	667
	rs_databases	668
	rs_datatype	669
	rs_dbreps	673
	rs_dbsubsets	674
	rs_diskaffinity	676
	rs_diskpartitions	677
	rs_erroractions	678
	rs_exceptscmd	679
	rs_exceptshdr.....	680
	rs_exceptslast	682
	rs_funcstrings.....	683
	rs_functions	685
	rs_idnames.....	686
	rs_ids.....	687
	rs_lastcommit	688
	rs_locator	689
	rs_maintusers.....	690
	rs_msgs.....	691
	rs_objects	692
	rs_oqid	696
	rs_profdetail	696
	rs_profile	697
	rs_publications	698
	rs_queuemsg	699
	rs_queuemsgtxt.....	700
	rs_queues	701
	rs_recovery	702
	rs_repdb	703
	rs_repobjs	704
	rs_routes	705

rs_routeversions.....	706
rs_rules	707
rs_schedule.....	709
rs_scheduletxt	710
rs_segments.....	711
rs_sites.....	712
rs_statcounters.....	712
rs_statdetail.....	713
rs_statrun	713
rs_subscriptions	715
rs_systext	719
rs_tbconfig.....	720
rs_threads	721
rs_ticket_history	722
rs_translation.....	723
rs_users	724
rs_version.....	725
rs_whereclauses	726

CHAPTER 9	Replication Monitoring Services API	727
	add event trigger	729
	add server	732
	configure component.....	735
	configure RMS	737
	configure server	740
	connect to server.....	742
	create group	742
	delete group	743
	disconnect server	744
	drop event trigger	744
	drop server	746
	filter connection	747
	get component	748
	get group	751
	get heartbeat	753
	get heartbeat tickets	754
	get network spec	756
	get rmiaddress	757
	get servers	757
	get status descriptions	759
	get threads	760
	get triggers	760
	get version.....	762
	log level	763

	resume component	763
	resume Replication Agent	765
	shutdown server	765
	start heartbeat	766
	stop heartbeat	767
	suspend component	768
	suspend Replication Agent	769
	trace	770
APPENDIX A	Acronyms and Abbreviations.....	773
APPENDIX B	Replication Server Design Limits.....	775
	Replication Server limits.....	775
	Platform-specific limits	776
	Replication definition and subscription limits.....	776
	Function string limits	776
	Programming limits and parameters	777
APPENDIX C	RMS Server and Component States	779
	Server states	779
	Replication Server.....	782
	Adaptive Server Enterprise	783
	IQ.....	784
	DirectConnect.....	784
	Open Server.....	785
	Replication Agent	785
	RMS	786
	Component states	786
	Connections	787
	Logical Connections	788
	Queues.....	789
	Routes	789
	Partitions	790
	RepAgent threads	790
APPENDIX D	Event Trigger Arguments.....	791
	Connection status event arguments.....	791
	Partition status event arguments.....	792
	Route status event arguments	793
	Server status event arguments	794
	Database connection latency event arguments	795
	Queue latency event arguments	796

Partition and queue size threshold event arguments	797
Index.....	799

About This Book

Replication Server® maintains replicated data at multiple sites on a network. Organizations with geographically distant sites can use Replication Server to create distributed database applications with better performance and data availability than a centralized database system can provide.

This book, *Replication Server Reference Manual*, describes these Replication Server features:

- The Replication Command Language (RCL) used by Replication Server
- System functions for Replication Server
- Adaptive Server® commands and system procedures that you use with Replication Server
- Replication Server System Database (RSSD) stored procedures that you use to manage the Replication Server system tables
- Replication Server executable programs, which you invoke directly from the operating system
- Replication Server system tables

Audience

The *Replication Server Reference Manual* is intended for anyone who uses Replication Server. It assumes that you have basic knowledge of how to use Replication Server.

This book is also for replication system Administrators who manage the routine operation of Replication Servers. Any user who has sa permission can be a replication system Administrator, although each Replication Server usually has just one.

How to use this book

The information in this book is organized as follows:

- Chapter 1, “Introduction to the Replication Command Language” categorizes the commands and what they do.
- Chapter 2, “Topics” discusses datatypes, identifiers, reserved words, and support for Adaptive Server.

-
- Chapter 3, “Replication Server Commands” consists of reference pages for all Replication Server commands.
 - Chapter 4, “Replication Server System Functions” provides reference pages for each system function Replication Server propagates from primary to replicate databases.
 - Chapter 5, “Adaptive Server Commands and System Procedures” contains reference pages for the Adaptive Server commands and system procedures used with Replication Server.
 - Chapter 6, “RSSD Stored Procedures” contains reference pages for the Adaptive Server stored procedures used to manage the Replication Server system tables.
 - Chapter 7, “Executable Programs” contains reference pages for the Replication Server executable programs and the rs_subcmp subscription comparison program.
 - Chapter 8, “Replication Server System Tables” describes each Replication Server system table.
 - Chapter 9, “Replication Monitoring Services API” contains reference pages for the Replication Monitor Service (RMS) API.
 - Appendix A, “Acronyms and Abbreviations” lists the acronyms and abbreviations used in the Replication Server documentation and system messages.
 - Appendix B, “Replication Server Design Limits” lists the maximum and minimum parameters and values for various replication system objects.
 - Appendix C, “RMS Server and Component States” lists the RMS server and component states.
 - Appendix D, “Event Trigger Arguments” lists the information RMS passes concerning the execution of a certain event.

Related documents

The Replication Server documentation set consists of the following:

- The release bulletin for your platform – contains last-minute information that was too late to be included in the books.

A more recent version of the release bulletin may be available on the World Wide Web. To check for critical product or document information that was added after the release of the product CD, use the Sybase Product Manuals at <http://www.sybase.com/support/manuals/>.

- *Installation Guide* for your platform – describes installation and upgrade procedures for all Replication Server and related products.
- *New Features Guide* – describes the new features in Replication Server version 15.5 and the system changes added to support those features.
- *Administration Guide* – contains an introduction to replication systems. This manual includes information and guidelines for creating and managing a replication system, setting up security, recovering from system failures, and improving performance.
- *Configuration Guide* for your platform – describes configuration procedures for Replication Server and related products.
- *Design Guide* – contains information about designing a replication system and integrating heterogeneous data servers into a replication system.
- *Getting Started with Replication Server* – provides step-by-step instructions for installing and setting up a simple replication system.
- *ASE-to-ASE Replication Quick Start Guide* – provides information for Adaptive Server users who want to set up a Replication Server to replicate data from one Adaptive Server database to another.
- *Heterogeneous Replication Guide* and the Replication Server Options documentation set – describes how to use Replication Server to replicate data between databases supplied by different vendors.
- *Reference Manual* (this book) – contains the syntax and detailed descriptions of Replication Server commands in the Replication Command Language (RCL); Replication Server system functions; Adaptive Server commands, system procedures, and stored procedures used with Replication Server; Replication Server executable programs; and Replication Server system tables.
- *System Tables Diagram* – illustrates system tables and their entity relationships in a poster format. Available only in print version.
- *Troubleshooting Guide* – contains information to aid in diagnosing and correcting problems in the replication system.
- Replication Manager plug-in help, which contains information about using Sybase Central™ to manage Replication Server.

Other sources of information

Use the Sybase Getting Started CD and the Sybase Product Manuals Web site to learn more about your product:

-
- The Getting Started CD is included with your software and contains release bulletins, installation guides in PDF format, and other documents or updated information. To read or print documents on the Getting Started CD, you need Adobe Acrobat Reader, which you can download at no charge from the Adobe Web site using a link provided on the CD.

You can also access the documents available on the Getting Started CD from the Sybase Product Manuals Web site.

- The Sybase Product Manuals Web site, which can be accessed using a standard Web browser, includes the Replication Server documents that are not included in the Getting Started CD. In addition to product manuals, you will find links to EBFs/Maintenance, Technical Documents, Case Management, Solved Cases, newsgroups, and the Sybase Developer Network.

To access the Sybase Product Manuals Web site, go to Product Manuals at <http://www.sybase.com/support/manuals/>.

Sybase certifications on the Web

Technical documentation at the Sybase Web site is updated frequently.

❖ Finding the latest information on product certifications

- 1 Point your Web browser to Technical Documents at <http://www.sybase.com/support/techdocs/>.
- 2 Click Partner Certification Report.
- 3 In the Partner Certification Report filter select a product, platform, and timeframe and then click Go.
- 4 Click a Partner Certification Report title to display the report.

❖ Finding the latest information on component certifications

- 1 Point your Web browser to Availability and Certification Reports at <http://certification.sybase.com/>.
- 2 Either select the product family and product under Search by Base Product; or select the platform and product under Search by Platform.
- 3 Select Search to display the availability and certification report for the selection.

❖ Creating a personalized view of the Sybase Web site (including support pages)

Set up a MySybase profile. MySybase is a free service that allows you to create a personalized view of Sybase Web pages.

Sybase EBFs and software maintenance

- 1 Point your Web browser to Technical Documents at <http://www.sybase.com/support/techdocs/>.
- 2 Click MySybase and create a MySybase profile.

❖ Finding the latest information on EBFs and software maintenance

- 1 Point your Web browser to the Sybase Support Page at <http://www.sybase.com/support>.
- 2 Select EBFs/Maintenance. If prompted, enter your MySybase user name and password.
- 3 Select a product.
- 4 Specify a time frame and click Go. A list of EBF/Maintenance releases is displayed.

Padlock icons indicate that you do not have download authorization for certain EBF/Maintenance releases because you are not registered as a Technical Support Contact. If you have not registered, but have valid information provided by your Sybase representative or through your support contract, click Edit Roles to add the “Technical Support Contact” role to your MySybase profile.

- 5 Click the Info icon to display the EBF/Maintenance report, or click the product description to download the software.

Conventions

This section describes style and syntax conventions, RCL command formatting conventions, and graphic icons used in this book.

Style conventions Syntax statements (displaying the syntax and options for a command) are printed as follows:

```
alter user user
set password new_passwd
[verify password old_passwd]
```

See “Syntax conventions” on page xx for more information.

Examples that show the use of Replication Server commands are printed as follows:

```
alter user louise
set password somNific
verify password EnnuI
```

Command names, command option names, program names, program flags, keywords, configuration parameters, functions, and stored procedures are printed as follows:

Use `alter user` to change the password for a login name.

Variables, parameters to functions and stored procedures, and user-supplied words are in italics in syntax and in paragraph text, as follows:

The `set password new_passwd` clause specifies a new password.

Names of database objects, such as databases, tables, columns, and datatypes, are in italics in paragraph text, as follows:

The `base_price` column in the `Items` table is a money datatype.

Names of replication objects, such as function-string classes, error classes, replication definitions, and subscriptions, are in italics, as follows:

`rs_default_function_class` is a default function-string class.

Syntax conventions Syntax formatting conventions are summarized in the following table. Examples combining these elements follow.

Table 1: Syntax formatting conventions

Key	Definition
<i>variable</i>	Variables (words standing for values that you fill in) are in italics.
{ }	Curly braces mean you must choose at least one of the enclosed options. Do not include braces in the command.
[]	Brackets mean you may choose or omit enclosed options. Do not include brackets in the command.
	Vertical bars mean you may choose no more than one option (enclosed in braces or brackets).
,	Commas mean you may choose as many options as you need (enclosed in braces or brackets). Separate your choices with commas, to be typed as part of the command. Commas may also be required in other syntax contexts.
()	Parentheses are to be typed as part of the command.
...	An ellipsis (three dots) means you may repeat the last unit as many times as you need. Do not include ellipses in the command.

Obligatory choices

- Curly braces and vertical bars – choose only one option.
`{red | yellow | blue}`
- Curly braces and commas – choose one or more options. If you choose more than one, separate your choices with commas.

	<pre>{cash, check, credit}</pre>
Optional choices	<ul style="list-style-type: none"> One item in square brackets – choose it or omit it. <pre>[anchovies]</pre> Square brackets and vertical bars – choose none or only one. <pre>[beans rice sweet_potatoes]</pre> Square brackets and commas – choose none, one, or more options. If you choose more than one, separate your choices with commas. <pre>[extra_cheese, avocados, sour_cream]</pre>
Repeating elements	<p>An ellipsis (...) means that you may repeat the last unit as many times as you need. For the alter function replication definition command, for example, you can list one or more parameters and their datatypes for either the add clause or the add searchable parameters clause:</p> <pre>alter function replication definition <i>function_rep_def</i> {deliver as '<i>proc_name</i>' add @<i>parameter</i> <i>datatype</i> [, @<i>parameter</i> <i>datatype</i>]... add searchable parameters @<i>parameter</i> [, @<i>parameter</i>]... send standby {all replication definition} parameters}</pre>
RCL command formatting	<p>RCL commands are similar to Transact-SQL® commands. The following sections present the formatting rules.</p>
Command format and command batches	<ul style="list-style-type: none"> You can break a line anywhere except in the middle of a keyword or identifier. You can continue a character string on the next line by typing a backslash (\) at the end of the line. Extra space characters on a line are ignored, except after a backslash. Do not enter any spaces after a backslash. You can enter more than one command in a batch, unless otherwise noted. RCL commands are not transactional. Replication Server executes each command in a batch without regard for the completion status of other commands in the batch. Syntax errors in a command prevent Replication Server from parsing subsequent commands in a batch.
Case sensitivity	<ul style="list-style-type: none"> Keywords in RCL commands are not case-sensitive. You can enter them with any combination of uppercase or lowercase letters. Identifiers and character data may be case-sensitive, depending on the sort order that is in effect.

-
- If you are using a case-sensitive sort order, such as “binary,” you must enter identifiers and character data with the correct combination of uppercase and lowercase letters.
 - If you are using a sort order that is not case-sensitive, such as “nocase,” you can enter identifiers and character data with any combination of uppercase or lowercase letters.

Identifiers

Identifiers are names you give to servers, databases, variables, parameters, database objects, and replication objects. Database object names include names for tables, columns, and views. Replication object names include names for replication definitions, subscriptions, functions, and publications.

- Identifiers can be 1 – 255 bytes long (equivalent to 1 – 255 single-byte characters) and must begin with a letter, the @ sign, or the _ character. See “Identifiers,” in Chapter 2, “Topics” for a list of identifiers that have been extended to 255 bytes.
- Replication Server function parameters are the only identifiers that can begin with the @ character. Function parameter names can include 255 characters *after* the @ character.
- After the first character, identifiers can include letters, digits, and the #, \$, or _ characters. Spaces are not allowed.
- Parameters in function strings have the same rules as identifiers, except:
 - They are enclosed in question marks (?), allowing Replication Server to locate them in the function string. Use two consecutive question marks (??) to represent a literal question mark in a function string.
 - The exclamation point (!) introduces a parameter modifier that indicates the source of the data that will be substituted for a parameter at runtime. For a complete list of modifiers, see create function string on page 265.

Parameters in function strings

Data support Replication Server supports all Adaptive Server datatypes.






User-defined datatypes are not supported. The double precision, nchar, and nvarchar datatypes are indirectly supported by mapping them to other datatypes.

For more information about the supported datatypes, including how to format them, see “Datatypes” on page 21.

Replication Server supports a set of datatype definitions for non-Sybase data servers that lets you replicate column values of one datatype to a column of a different datatype in the replicate database. See the *Replication Server Administration Guide Volume 1* for more information about heterogeneous datatype support (HDS).

Icons

Illustrations in this book use icons to represent the components of a replication system.

	Description
	This icon represents Replication Server, the Sybase server program maintains replicated data on a local-area network (LAN) and processes data transactions received from other Replication Servers on wide-area network (WAN).
	This icon represents Adaptive Server, the Sybase data server. Data servers manage databases containing primary or replicated data. Replication Server also works with heterogeneous data servers, so, unless otherwise noted, this icon can represent any data server in a replication system.
	This icon represents Replication Agent™, a replication system process or module that transfers transaction log information for primary database to a Replication Server. The Replication Agent for Adaptive Server is RepAgent. Sybase provides Replication Agent products for Adaptive Server® Anywhere, DB2, Microsoft SQL Server, and Oracle data servers. Except for RepAgent, which is an Adaptive Server thread, all Replication Agents are separate processes. In general, this icon only appears when representing a Replication Agent that is a separate process.
	This icon represents client application. A client application is a user process or application connected to a data server. It may be a front-end application program executed by a user or a program that executes as an extension of the system.
	This icon represents the Sybase Central Replication Manager plug-in (RM), a management utility that lets a replication system administrator develop, manage, and monitor a Sybase Replication Server environment.

Accessibility features

This document is available in an HTML version that is specialized for accessibility. You can navigate the HTML with an adaptive technology such as a screen reader, or view it with a screen enlarger.

Replication Server HTML documentation has been tested for compliance with U.S. government Section 508 Accessibility requirements. Documents that comply with Section 508 generally also meet non-U.S. accessibility guidelines, such as the World Wide Web Consortium (W3C) guidelines for Web sites.

Note You might need to configure your accessibility tool for optimal use. Some screen readers pronounce text based on its case; for example, they pronounce ALL UPPERCASE TEXT as initials, and MixedCase Text as words. You might find it helpful to configure your tool to announce syntax conventions. Consult the documentation for your tool.

For information about how Sybase supports accessibility, see Sybase Accessibility at <http://www.sybase.com/accessibility>. The Sybase Accessibility site includes links to information on Section 508 and W3C standards.

If you need help

Each Sybase installation that has purchased a support contract has one or more designated people who are authorized to contact Sybase Technical Support. If you cannot resolve a problem using the manuals or online help, please have the designated person contact Sybase Technical Support or the Sybase subsidiary in your area.

Introduction to the Replication Command Language

The Replication Command Language (RCL) is divided into these categories:

Topic	Page
Data replication commands	2
User commands	9
Database interface commands	10
Gateway commands	14
Route commands	14
System information commands	15
Partition commands	17
Configuration commands	17
System administration commands	18
Recovery commands	20

This chapter lists and summarizes the commands in each category. Some commands are included in multiple categories. For complete command syntax and usage information, see Chapter 3, “Replication Server Commands.”

For detailed information on datatypes, identifiers, reserved words, and support for Adaptive Server, see Chapter 2, “Topics.”

For an introduction to Replication Server architecture, see Chapter 1, “Introduction,” and Chapter 2, “Replication Server Technical Overview” in the *Replication Server Administration Guide Volume 1*.

Some Replication Server procedures may require you to execute Adaptive Server system procedures such as `sp_setreptable` or `sp_setrepproc`. For complete syntax and usage information, see Chapter 5, “Adaptive Server Commands and System Procedures.”

The Replication Manager (RM) provides another way to perform many of the tasks that RCL commands perform. See *Replication Server Administration Guide Volume 1* for more information.

Data replication commands

Data replication commands create and manage the replication definitions, publications, and subscriptions that make it possible to replicate tables or stored procedures.

Table replication definition commands

A table replication definition describes the table and the columns that are to be replicated. A primary table is the replication source; a replicate table is the destination. You can create one or more replication definitions for each primary table.

Create a replication definition in the Replication Server that manages the database where the primary table is stored.

The replication definition includes:

- A name for the replication definition
- The names of the primary and replicate tables, if they are different from each other and from the replication definition name
- The location of the primary table
- The names and datatypes of the primary columns to be replicated and the corresponding replicate column names
- The names of the columns that form the primary key for the table

The replication definition can optionally include:

- The names of columns that can be referenced in where clauses for subscriptions
- Whether the replication definition and its columns will be used in replicating to a standby database
- Whether to replicate all columns or the minimum number of required columns for update and delete operations
- Replication status for text, unitext, image, and rawobject columns
- Whether to change the datatype of replicated values from the datatype of the primary database to the datatype of the replicate database.

No data is distributed when you create a replication definition. You must create a copy of the table in each replicate database and then create subscriptions to begin replicating data.

Use these commands to work with table replication definitions:

- create replication definition – creates a replication definition for a table.
- alter replication definition – changes a replication definition.
- drop replication definition – removes a replication definition.

For commands that you use in subscribing to replication definitions, see “Subscription commands” on page 6.

Function replication definition commands

A function replication definition specifies information about a stored procedure that is to be replicated.

Create a function replication definition in the Replication Server that manages the primary database.

The function replication definition includes:

- A name for the function replication definition.
- The location of the primary data.
- The names and datatypes of the stored procedure parameters to be replicated.

The function replication definition can optionally include:

- The name of the stored procedure executed in the source database and the name of the stored procedure to be executed in the destination database, if the stored procedure name is different from the name of the function replication definition.
- The names of parameters that can be referenced in where clauses for subscriptions.
- Whether the function replication definition and its parameters will be used in replicating to a standby database.

Use these commands to work with function replication definitions:

- create applied function replication definition – creates an applied function replication definition for a stored procedure.

- alter applied function replication definition – changes an applied function replication definition.
- create request function replication definition – creates a request function replication definition for a stored procedure.
- alter request function replication definition – changes a request function replication definition.
- drop function replication definition – removes a function replication definition.

No data is distributed when you create a function replication definition. You must create stored procedures in both the primary and replicate databases, and you must create a subscription at the replicate Replication Server.

See “Subscription commands” on page 6 for commands that you use in subscribing to replication definitions.

Database replication definition commands

A database replication definition describes the database or database objects to be replicated. You can choose to replicate the entire database, or you can choose to replicate—or not replicate—particular tables, functions, transactions, DDL, and system stored procedures in that database.

The database replication definition includes:

- The name of the database replication definition
- The name of the primary server where the database to be replicated is located
- The name of the database to be replicated

The database replication definition can optionally include:

- An indicator on whether to replicate the DDL to subscribing databases
- An indicator on whether to replicate tables, stored procedures, user-defined functions, transaction, or system procedures to subscribing databases

Use these commands to work with database replication definitions:

- create database replication definition – creates a replication definition for replicating a database or a database object.

- alter database replication definition – changes an existing database replication definition.
- drop database replication definition – deletes an existing database replication definition.

See “Subscription commands” on page 6 for commands that you use in subscribing to replication definitions.

Publication commands

The publications feature of Replication Server lets you group the tables and procedures you want to subscribe to, and their replication definitions, and create one subscription for the group.

A **publication** is a set of articles from the same primary database. Each **article** is a replication definition for a table or stored procedure and a set of where clauses that specify which rows are of interest. An article can contain zero, one, or multiple where clauses. Multiple clauses are separated by the *or* keyword.

Use these commands to work with publications and articles:

- create publication – creates a publication.
- drop publication – removes a publication and its articles. The *drop_repdef* option drops the associated replication definitions.
- validate publication – verifies that a publication has at least one article and marks the publication so that new subscriptions can be created for it.
- check publication – indicates whether subscriptions can be created for a publication, and reports the number of articles it contains.
- create article – creates an article and assigns it to a publication.
- drop article – removes an article from a publication. The *drop_repdef* option also drops the associated replication definition.

See “Publication subscription commands” on page 8 for information about the commands used in subscribing to publications.

Subscription commands

Subscriptions initiate the replication of data or stored procedures. A subscription specifies a table or function replication definition name, or a publication, and the database into which the data is to be replicated.

- A subscription for a table replication definition data.
- A subscription for a function replication definition replicates stored procedures.
- A subscription for a database replication definition replicates databases or database objects.
- A subscription for a publication replicates the data represented by each article in the publication. Publications can also have articles for stored procedures.

A subscription to a table or function replication definition may include a where clause, which determines the rows that are replicated or whether a stored procedure is replicated.

A subscription to a database replication definition subscribes to all data. You cannot use a where clause to set the criteria for subscribed data. If you need to subscribe to particular tables or functions, you can add table or function subscriptions. See the “Using database, table, and function subscription concurrently” section in the *Replication Server Administration Guide Volume I* for more information.

Note A subscription to a publication cannot include a where clause. where clauses are contained in the publication’s articles.

Subscription materialization

When you create a subscription for a table replication definition, rows that fit the subscription are copied from the primary to the replicate table in a process called **materialization**. After materialization is complete, Replication Server distributes row changes in the primary database through normal replication.

If a subscription involves many rows, materialization can hold locks for a long time and overload the network. Replication Server queues may also fill with data. To avoid these problems, Replication Server provides four different ways to materialize a subscription.

You can use any method for subscriptions to table replication definitions or to publications. Use nonmaterialization or bulk materialization for subscriptions to function replication definitions or database replication definitions.

- **Atomic materialization** is the default method for table replication definitions. Replication Server selects rows at the primary table, using a holdlock, and copies them over the network. The primary table is locked during materialization and data is consistent between the primary and replicate tables.
- In **nonatomic materialization**, Replication Server selects rows at the primary table, without using a holdlock, and copies them over the network. Because the primary table is not locked, the replicate may go through visible steps that did not exist at the primary while nonatomic materialization is in progress.
- In nonmaterialization, the primary and replicate data is already in sync. You do not need to copy data over the network or load it from media. No updates can be in process while such a subscription is created.
- In **bulk materialization**, data is manually unloaded and loaded from media. This is the most efficient way to materialize subscriptions that involve a large amount of data.

For more information about subscription materialization methods, see the *Replication Server Administration Guide Volume 1*.

Atomic and nonatomic
materialization
commands

Use these commands to create a subscription and initialize data at the replicate database:

- `create subscription` – creates and materializes a subscription using atomic materialization.
- `create subscription ... without holdlock` – creates and materializes a subscription using nonatomic materialization.

If you use nonatomic materialization, which selects primary data without a holdlock, you must also use:

- `set autocorrection` – prevents failures caused by missing or duplicate rows in a replicate table. When primary data is selected without a holdlock, it might be updated before materialization is complete and before normal transaction replication begins.

Nonmaterialization
command

Use this command to create a subscription when data is already in sync at the replicate database:

Bulk materialization commands

- `create subscription ... without materialization` – creates a subscription without materializing data at the replicate database.

Bulk materialization is used to manually coordinate subscription status and to transfer data for function replication definitions or database replication definitions.

Use these commands for bulk materialization:

- `define subscription` – adds a subscription to the system tables at the primary and replicate Replication Server.
- `activate subscription` – starts the distribution of updates from the primary database to the replicate database and sets the subscription status to **ACTIVE**.

After you use this command and verify status, manually load initial data from media into the replicate database. Use the `with suspension` option to prevent data from being applied to the replicate database until the load from media is complete.

- `validate subscription` – completes bulk materialization and changes the subscription status to **VALID**. Replication Server is notified that materialization is complete.

Other subscription commands

To monitor the materialization or dematerialization of a subscription, use:

- `check subscription` – finds the status of a subscription at the primary or replicate database.

To drop a subscription from a replicate database, use:

- `drop subscription` – clears subscription information from system tables.

Optionally, you can use `drop subscription with purge` to remove the replicate data associated with a subscription. This process is called **dematerialization**.

Publication subscription commands

Publication subscriptions use the same commands as subscriptions for replication definitions.

- To create a publication subscription using atomic materialization, nonatomic materialization, or nonmaterialization, use `create subscription`.

- To create a publication subscription using bulk materialization, use `define subscription` and the other bulk materialization commands.

When you add an article to a publication that has a subscription, you must refresh the publication subscription to include subscriptions for the new article. This process is called **rematerialization**.

- For atomic or nonatomic rematerialization, use `create subscription` with the `for new articles` clause.
- If data is in sync at the primary and replicate databases, use `create subscription` with the `for new articles` clause and the `without materialization keywords`.
- For bulk rematerialization, use `define subscription` with the `for new articles` clause, then use the other bulk materialization commands.

User commands

Users must have Replication Server login accounts to execute Replication Server commands. An account consists of a login name and a password, both of which must be supplied to connect to a Replication Server.

Use these commands to administer user login accounts:

- `create user` – adds a new user to a Replication Server.
- `alter user` – changes a user's password.
- `drop user` – drops a Replication Server user account.

Use these commands to manage user permissions:

- `grant` – assigns permissions.
- `revoke` – revokes permissions.

Use the `set proxy` command to switch to another user login account with different permissions.

Each permission allows a user to execute a set of commands. For example, to create a replication definition, a user must have `create object` permission. A user with “sa” permission can execute any Replication Server command.

Database interface commands

Replication Server provides several ways to connect to databases and to customize the operations performed in them. Replication Server's open architecture supports primary or replicate databases managed by heterogeneous data servers, including Adaptive Server and several other data servers.

For each database, you can:

- Create or modify a Replication Server connection to a database. See “Database connection commands” on page 10 for details.
- Customize error handling methods. See “Error class commands” on page 11 for details.
- Customize database operations. See “Function and function string commands” on page 11 for details.
- Create or modify a logical database connection used in a warm standby application. See “Warm standby database commands” on page 13 for details.
- Set configuration parameters for the connection or logical connection. See “Configuration commands” on page 17 for details.

Each database that will be a source of replicated transactions or stored procedures must have a Replication Agent. For details, see the *Replication Server Administration Guide Volume 1*.

Database connection commands

A physical database **connection** connects a Replication Server to a local database that contains primary or replicate data. A Replication Server distributes messages to and from a database via a connection.

Use these commands to manage database connections:

- `create connection` – creates a database connection from Replication Server to a non-Sybase database. Adaptive Server database connections are added with `rs_init`.
- `create connection using profile clause` – uses predefined information to configure the connection between Replication Server and a replicate non-Adaptive Server database, and, if needed, to modify the RSSD, and the replicate data server and database.

- alter connection – changes or configures a database connection.
- drop connection – removes a database connection.
- suspend connection – suspends a database connection.
- resume connection – resumes a suspended connection.

Error class commands

An **error class** is a name under which error handling actions—such as retry and ignore—are assigned to specific data server errors.

Use the create connection command to associate an error class with a database. Use alter connection to change an error class. You can often create one error class for all databases for a specified data server.

Note The default error class for an Adaptive Server database is assigned when you add a connection using `rs_init`.

Use these commands to manage error handling actions and error classes:

- create error class – creates an error class.
- alter error class – modifies an existing error class by copying error actions from another error class.
- move primary – moves an error class or function-string class and any of the function-string class' derived classes to a different primary site.
- drop error class – drops an error class.
- assign action – assigns actions to data server error codes.

Use the stored procedure `rs_init_erroractions` to initialize a new error class created with error actions from an existing error class. For details, see Chapter 5, “Adaptive Server Commands and System Procedures.”

Function and function string commands

You can use function strings to program Replication Server to execute customized commands at destination databases.

A **function** is a name associated with a data server operation. For example, `rs_insert` is the system function that inserts a row in a table, and `rs_begin` is the system function that initiates a transaction. System functions can manipulate data, as does `rs_insert`, or control transactions as does `rs_begin`.

Replication Server uses a template called a **function string** to construct the commands it submits to a database. At runtime, variables in the function string are replaced with values from the function.

A **function-string class** groups function strings for use with a database. For example, a function-string class might group all of the function strings for a vendor's data server or for a department's tables. Replication Server provides function-string classes for Adaptive Server and DB2 databases.

Use `create connection` to associate a function-string class with a database. Use `alter connection` to change a function-string class.

Note The default function-string class for Adaptive Server databases, `rs_sqlserver_function_class`, is assigned when you add a connection using `rs_init`.

You can create a new function-string class that inherits function strings from an existing class. Then you can customize only the function strings for which you want to specify non-default behavior, as your database or application requires.

Function-string class commands

Use these commands to work with function-string classes:

- `create function string class` – creates a function-string class.
- `alter function string class` – changes the inheritance relationships of a function-string class.
- `move primary` – moves an error class or function-string class and any of the function-string class' derived classes to a different primary site.
- `drop function string class` – drops a function-string class.

Function string commands

Use these commands to work with the function strings in a function-string class:

- create function string – creates a function string.
- alter function string – replaces an existing function string.
- drop function string – drops a function string.

Function commands

Use these commands to work with user-defined functions. They are only necessary for Asynchronous Procedure Calls.

- create function – creates a function.
- alter function – adds parameters to a user-defined function.
- drop function – drops a function.

Warm standby database commands

A Replication Server **warm standby application** maintains two Adaptive Server databases, one of which functions as a standby or backup copy of the other. Replication Server's connection to the active and standby databases is called a **logical connection**.

Use these commands to manage logical database connections:

- create logical connection – creates a logical connection.
- alter logical connection – changes the characteristics of a logical connection.
- drop logical connection – removes a logical connection.
- configure logical connection – configures a logical connection.

Use these commands to perform tasks associated with warm standby applications:

- switch active – changes the active database.
- abort switch – aborts the switch active command, if possible.
- wait for switch – in an interactive or script-based Replication Server session, prevents commands from executing until the switch to a new active database is complete.

- wait for create standby – in an interactive or script-based Replication Server session, prevents Replication Server from accepting commands until the standby database is ready for operation.

Gateway commands

The Replication Server gateway minimizes explicit log in to multiple replication servers, ID servers, and the RSSD. The Replication Server gateway uses your RSSD primary user name and password to log in to RSSD, your ID server user name and password to log in to ID Server, your remote server identification (RSI) to log in to a remote Replication Server, and your maintenance user ID to log in to the remote Adaptive Server. You do not need to supply this information more than once, when you access Replication Server itself.

The Replication Server gateway also supports cascading connections, which allow your Replication Server to communicate with servers that it is not directly connected to. It also allows you to manage a replication domain using a single client connection.

Use these commands to manage Replication Server gateways:

- connect – turns Replication Server into a gateway to its RSSD, ID server, a remote Replication Server, or a remote data server.
- show connection – lists the contents of the connection stack.
- show server – displays the current working server.
- disconnect – terminates a connection to a server.

Route commands

A **route** is a *one-way* message stream from the source (primary) Replication Server to the destination (target) Replication Server. A Replication Server sends messages to, or receives messages from, another Replication Server via a route. Such messages include data for replicated transactions. A route may connect Replication Servers across a local-area network or a wide-area network.

Use these commands to manage routes:

- `create route` – creates and configures a route from the current Replication Server to another.
- `alter route` – changes or reconfigures the route from the current Replication Server to another.
- `drop route` – removes the route to another Replication Server.
- `suspend route` – suspends the route to another Replication Server.
- `resume route` – resumes a suspended route.

System information commands

Use these commands to obtain information about the Replication Server. Any user can execute these commands.

- `admin disk_space` – displays the usage statistics of each disk partition accessed by the Replication Server.
- `admin echo` – returns the text you enter to verify that the Replication Server is running.
- `admin get_generation` – retrieves the generation number for a primary database.
- `admin health` – displays the overall status of the Replication Server.
- `admin log_name` – displays the path to the current log file.
- `admin logical_status` – displays status information for a logical connection in a warm standby application.
- `admin pid` – displays the process ID of the Replication Server.
- `admin quiesce_check` – determines if the queues in the Replication Server have been quiesced.
- `admin quiesce_force_rsi` – determines whether a Replication Server is quiesced and forces it to deliver outbound messages.
- `admin rssid_name` – displays the names of the data server and database for the Replication Server System Database (RSSD).
- `admin security_property` – displays network-based security mechanisms and features supported by Replication Server.

- `admin security_setting` – displays the status of network-based security features supported by Replication Server.
- `admin set_log_name` – closes the existing Replication Server log file and opens a new log file.
- `admin show_connections` – displays information about all connections from the Replication Server.
- `admin show_function_classes` – displays the names of existing function-string classes and their parent classes, and indicates the number of levels of inheritance.
- `admin show_route_versions` – displays the version number of routes that originate and terminate at the Replication Server.
- `admin show_site_version` – displays the site version of the Replication Server.
- `admin sqm_readers` – displays the read point and delete point for each Replication Server thread that is reading an inbound queue.
- `admin stats` – displays information and statistics about Replication Server counters.
- `admin stats, backlog` – reports the current transaction backlog in the stable queues.
- `admin stats, {md | mem | mem_in_use}` – reports information about memory usage.
- `admin stats, status` – displays the flushing status for all counters.
- `admin stats, reset` – resets all counters that can be reset.
- `admin stats, {tps | cps | bps}` – reports the number of transactions, commands, or bytes of throughput per second.
- `admin time` – displays the current time of Replication Server.
- `admin translate` – performs a datatype translation on a specific data value, displaying the results in literal format with delimiters.
- `admin version` – displays the Replication Server software version.
- `admin who` – displays information about threads running in the Replication Server.
- `admin who_is_down` – displays a subset of information about Replication Server threads that are not running.

- `admin who_is_up` – displays a subset of information about Replication Server threads that are running.

Partition commands

Replication Server stores messages in stable queues, which are stored on disk partitions. Inbound queues store messages received from Replication Agents; outbound queues store messages to be transmitted to data servers or other Replication Servers.

Use `rs_init` to create Replication Server's initial partitions. For more information about working with partitions in `rs_init`, see the Replication Server installation and configuration guides.

Use these commands to add, drop or change the size of partitions:

- `create partition` – makes a partition available to Replication Server. You must create a partition before you can add it.

Note `create partition` replaces the existing `add partition` command. For backward compatibility, `add partition` is still supported as an alias for `create partition` but it will be depreciated in the future.

- `drop partition` – removes a partition from Replication Server.
- `alter partition` – changes the size of a partition.

For more information about stable queues and partitions, see the *Replication Server Administration Guide Volume 1*.

Configuration commands

When Replication Server starts, configuration parameters are read from system tables or from a configuration file. Configuration parameters may be static or dynamic. You can change dynamic parameters while Replication Server is running, but you must restart Replication Server after you change static parameters.

Use these commands to configure Replication Server:

- alter connection and configure connection – change the characteristics of a Replication Server connection to a database.
- configure logical connection – changes the Replication Server configuration for a logical connection in a warm standby application.
- configure replication server – changes Replication Server parameters and default parameters for routes and connections.
- alter route and configure route – change the characteristics of a route. A route connects one Replication Server to another.

Configuration parameters are also set when you create routes and connections using `create route` and `create connection`.

For more information, see the *Replication Server Administration Guide Volume 1*.

System administration commands

Use these commands to perform system administration tasks, and to troubleshoot problems that follow system failures. You must have “sa” permission to execute these commands.

Warning! Many of these commands should be used with caution and only in very restricted circumstances. Please check the associated documentation carefully before you use them.

- alter queue – specifies the behavior of a stable queue that encounters a large message of greater than 16K bytes. Use only if the Replication Server version is 12.5 or later and the site version is 12.1 or earlier.
- resume distributor – resumes a suspended distributor thread for a connection to a database.
- shutdown – shuts down a Replication Server.
- suspend distributor – suspends the distributor thread for a connection to a database.
- sysadmin apply_truncate_table – turns the “subscribe to truncate table” option on or off for existing subscriptions to a particular table, enabling or disabling replication of truncate table.

- `sysadmin dropdb` – drops references to a database from the ID Server.
- `sysadmin dropldb` – drops references to a logical database from the ID Server.
- `sysadmin drop_queue` – deletes a stable queue.
- `sysadmin dropsr` – drops references to a Replication Server from the ID Server.
- `sysadmin dump_file` – specifies an alternate log file for use when a stable queue is dumped.
- `sysadmin dump_queue` – dumps the contents of a stable queue.
- `sysadmin erssd` – allows you to check ERSSD file locations and backup configurations, defragment ERSSD files, move ERSSD files or perform an unscheduled backup of the ERSSD.
- `sysadmin fast_route_upgrade` – updates the route version to the site version of the lower of the primary or replicate Replication Server.
- `sysadmin hibernate_off` – turns off hibernation mode for the Replication Server and returns it to an active state.
- `sysadmin hibernate_on` – turns on hibernation mode for (or suspends) the Replication Server.
- `sysadmin log_first_tran` – writes the first transaction in a Data Server Interface (DSI) queue to the exceptions log.
- `sysadmin purge_all_open` – purges all open transactions from the inbound queue.
- `sysadmin purge_first_open` – purges the first open transaction from the inbound queue.
- `sysadmin purge_route_at_replicate` – removes all references to the primary Replication Server from a Replication Server at a replicate site.
- `sysadmin restore_dsi_saved_segments` – restores backlogged transactions so that they can be reapplied to the database.
- `sysadmin set_dsi_generation` – changes a database generation number in the RSSD to prevent Replication Server from reapplying transactions in the stable queue after a replicate database is restored.
- `sysadmin site_version` – sets the site version level.
- `sysadmin sqm_purge_queue` – removes all messages from a Replication Server Interface (RSI) stable queue.

- `sysadmin sqm_unzap_command` – restores a deleted message in a stable queue.
- `sysadmin sqm_zap_command` – deletes a single message in a stable queue.
- `sysadmin sqt_dump_queue` – dumps the transaction cache for each inbound or DSI queue.
- `sysadmin system_version` – sets the minimum Replication Server version level for the replication system.

Recovery commands

Use these commands to coordinate recovery after a database is reloaded or when Replication Server stable queues fail.

Warning! Many of these commands should be used with caution and only in very restricted circumstances. Make sure to check the associated documentation carefully before you use them.

- `allow connections` – places Replication Server in recovery mode for specified databases.
- `ignore loss` – allows Replication Server to accept messages after a loss is detected.
- `rebuild queues` – rebuilds Replication Server stable queues.
- `resume log transfer` – allows a RepAgent thread to connect to the Replication Server.
- `resume queue` – restarts a stable queue stopped after receiving a messenger larger than 16K bytes. Applicable only when the Replication Server version is 12.5 or later and the site version is 12.1 or earlier.
- `set log recovery` – places Replication Server in log recovery mode for a database.
- `suspend log transfer` – disconnects a RepAgent from a Replication Server and prevents either from connecting.

For detailed recovery procedures, see the *Replication Server Administration Guide Volume 2*.

This chapter contains information on these topics:

Topic	Page
Datatypes	21
Identifiers	35
Reserved words	38
Support for Adaptive Server	40
Mixed-version environments	42

Datatypes

Replication Server directly supports the Sybase datatypes shown in Table 2-1.

Table 2-1: Replication Server-supported datatypes

Datatype class	Datatypes
Exact numeric (integer)	bigint, int, smallint, tinyint, unsigned bigint, unsigned int, unsigned smallint, unsigned tinyint, rs_address
Exact numeric (decimal)	decimal, numeric, identity
Approximate numeric (floating point)	float, real
Character	char(n), varchar(n), text, opaque
Money	money, smallmoney
Date/time	datetime, smalldatetime, date, time, timestamp, bigdatetime, bigtime
Binary	binary(n), varbinary(n), image, rawobject, rawobject in row
Bit	bit
Unicode	unichar(n), univarchar(n), unitext
Java	rawobject, rawobject in row
Datatype definitions	See “Datatype definitions” on page 34.

RCL indirectly supports these Sybase datatypes:

- double precision

- `nchar`, `nvarchar`

These datatypes are not supported:

- The optional precision argument of the float datatype
- The optional precision and scale arguments of the exact decimal datatypes

Data in columns with unsupported datatypes can be replicated if you create the replication definition using one of the supported datatypes shown in Table 2-1. For example, to replicate a double precision column, define the column as `float` in the replication definition. To replicate a column with a user-defined datatype, use the underlying datatype in the replication definition.

To replicate data stored in columns of type `nchar` or `nvarchar` in the Adaptive Server, use the `char` and `varchar` Replication Server datatypes, respectively. The only difference is that the length units in `nchar` and `nvarchar` refer to the number of characters in the native character set of the Adaptive Server, and the length units in `char` and `varchar` always refer to bytes.

To get the length of the corresponding Replication Server `char` and `varchar` datatypes, multiply the declared length of the `nchar` or `nvarchar` datatype by the value of the Adaptive Server global variable `@@ncharsize`.

For example, if `@@ncharsize` is 1 (true for all single-byte character sets like `iso_1`, `cp850`, `cp437`, `roman8`, and `mac`), there is a one-to-one correspondence and the declared lengths are the same. If `@@ncharsize` is 2 (true for some multibyte character sets like `Shift-JIS` and `EUC-JIS`), multiply the declared length of the `nchar` and `nvarchar` datatypes by 2 and declare them as `char` and `varchar` in the replication definition.

The following sections describe the supported datatypes. For more information about Adaptive Server datatypes, see the *Adaptive Server Enterprise Reference Manual*.

Exact numeric (integer) datatypes

Replication Server supports these exact numeric (integer) datatypes:

- `bigint` – whole numbers between -2^{63} and $+2^{63} - 1$ (-9,233,372,036,854,775,808 and +9,233,372,036,854,775,807), inclusive
- `int` – whole numbers between -2^{31} and $+2^{31} - 1$ (-2,147,483,648 and +2,147,483,647), inclusive

- `smallint` – whole numbers between -2^{15} and $+2^{15} - 1$ (-32,768 and +32,767), inclusive
- `tinyint` – positive whole numbers between 0 and 255, inclusive
- `unsigned bigint` – whole numbers between 0 and 18,446,744, 073, 709,551,615, inclusive
- `unsigned int` – whole numbers between 0 and 4,294,967,295, inclusive
- `unsigned smallint` – whole numbers between 0 and 65535, inclusive
- `unsigned tinyint` – whole numbers between 0 and 255, inclusive.

The `rs_address` datatype, which uses the underlying datatype `int`, is used in a special method of subscription resolution. For more information about the `rs_address` datatype, see `create subscription`. See also the *Replication Server Administration Guide Volume 1*.

Exact numeric (decimal) datatypes

Replication Server supports the following exact numeric (decimal) datatypes:

- `decimal` – exact decimal numbers between -10^{38} and $10^{38} - 1$, inclusive.
- `numeric` – exact decimal numbers between -10^{38} and $10^{38} - 1$, inclusive.

When you create a replication definition, omit the length and precision from numeric datatype declarations. Replication Server processes numeric values without affecting precision.

Identity columns use `numeric` as the underlying datatype, with exact decimal numbers of scale 0 between 1 and $10^{38} - 1$, inclusive.

When you create a replication definition for a table that contains an identity column, specify “identity” as the datatype for the column.

This command is applied to the replicated table *before* an insert command:

```
set identity_insert table_name on
```

This command is applied to the replicated table *after* an insert command:

```
set identity_insert table_name off
```

Identity columns are never updated by the update command.

If the replicate data server is Adaptive Server and a table contains an identity column, the maintenance user must be the owner of the table (or must be the “dbo” user or aliased to the “dbo” login name) at the replicate database in order to use the Transact-SQL `identity_insert` option.

Approximate numeric (floating point) datatypes

There are two approximate numeric (floating point) datatypes:

- `float` – positive or negative floating point numbers. Precision and number of significant digits are machine-dependent. Storage size is 8 bytes.
- `real` – like `float` except the storage size is 4 bytes.

Character datatypes

Note The Unicode datatypes `unichar`, `univarchar`, and `unitext` have the same attributes as their `char`, `varchar`, and `text` equivalents. See “Unicode datatypes” on page 31.

- `char(n)` – any combination of up to 32,768 single-byte letters, symbols, and numbers. Specify the maximum size of the string with *n*. A `char` value can contain 0 characters, but *n* must be between 1 and 32,768. A multibyte string cannot exceed 32,768 bytes.
- `varchar(n)` – any combination of up to 32,768 single-byte letters, symbols, and numbers. A `varchar` value can contain 0 characters if it is defined to allow null values, but *n* must be between 1 and 32,768.

The difference between `char` and `varchar` data is the way the values are stored in Adaptive Server databases. Replication Server treats them as equivalent types, but maintains the distinction so that the storage method is the same in primary and replicate databases.

- `text` – variable-length character columns up to 2,147,483,647 bytes in length.

Replication Server 15.1 supports datatype conversion between large object (LOB) datatypes such as `text`, `unitext`, and image datatypes with text pointer and `text`, `unitext`, and image datatypes without text pointer.

Entry format for character data

Literal `char`, `varchar`, and `text` values—or their equivalents—must be enclosed in single quotation marks.

You can embed single quotation marks in `char` and `varchar` literals in two ways. Use two consecutive quotation marks to represent a single embedded quotation mark, as in this example:

```
'''You can have cake if you bake it,' Ed claims.'
```

The first and last quotation marks delimit the character string. The two internal pairs of quotation marks are interpreted as embedded single quotation marks.

Replication Server generates single quotation marks when it substitutes a character value for a variable in a function-string template. For more information, see [create function string](#).

Money datatypes

The money datatypes hold fixed precision values for currency or monetary values:

- `money` – monetary values between -922,337,203,685,477.5808 and 922,337,203,685,477.5807, with accuracy to 1/10000 of a monetary unit. Storage size is 8 bytes.
- `smallmoney` – monetary values between -214,748.3648 and 214,748.3647, with accuracy to 1/10000 of a monetary unit. Storage size is 4 bytes.

Entry format for money data

Precede money and smallmoney literal values with a U.S. dollar sign (\$) to distinguish them from the floating point datatypes. For negative values, place the minus sign after the dollar sign.

Replication Server outputs a dollar sign when it substitutes money and smallmoney values into function-string output templates.

Date/time, and date and time datatypes

Replication Server supports these datatypes for date and time data:

- `datetime` – dates and times of day between January 1, 1753 and December 31, 9999. Storage size is 8 bytes: 4 bytes for the number of days before or after the base date of January 1, 1900, and 4 bytes for the time, to 1/300 second. Dates before the base date are stored as negative values.
- `smalldatetime` – dates and times of day between January 1, 1900 and June 6, 2079, with accuracy to one minute. Storage size is 4 bytes: one small integer for the number of days after January 1, 1900, and one small integer for the number of minutes since midnight.

- **date** – dates between January 1, 0001, and December 31, 9999. Storage size is 4 bytes. Dates before the base date are stored as negative values.
- **time** – time between 12:00:00 AM and 11:59:59.999 PM. Storage size is 4 bytes.
- **bigtime** – time of day, containing hour, minute, second, and fraction of a second corresponding to the TIME datatype in Sybase IQ. The fraction is stored to 6 decimal places. A bigtime value requires 8 bytes of storage. ODBC standards restrict bigtime datatype to an accuracy of seconds. For this reason, do not use bigtime datatypes in WHERE clause comparisons that rely on a higher accuracy than seconds.

The valid range of bigtime is from 12:00:00.000000AM to 11:59:59.999999PM

- **bigdatetime** – point in time, containing year, month, day, hour, minute, second, and fraction of a second corresponding to the TIMESTAMP datatype in Sybase IQ. The fraction is stored to 6 decimal places. The day must be a nonzero value. A bigdatetime value requires 8 bytes of storage.

The valid range of bigdatetime is from January 1, 0001 to December 31, 9999 and from 12:00:00.000000AM to 11:59:59.999999PM. The display of bigdatetime data outside the range of 1600-02-28 23:59:59 to 7911-01-01 00:00:00 might be incomplete, but the complete bigdatetime value is stored in the database.

- **timestamp** – uses varbinary(8) as the underlying datatype. A status bit differentiates timestamp from varbinary.

timestamp is propagated as timestamp to Replication Server 15.1 and as varbinary to Replication Server 15.0.1 or earlier.

Note Replication into a timestamp column is supported only in ASE 15.0.2 or later.

Entry format for date/time values

Enter datetime and smalldatetime values as character strings, enclosed in single quotation marks.

Replication Server encloses datetime values in single quotation marks when it substitutes datetime values into function-string output templates. Be sure to consider this when you create function strings that include datetime variables.

The date and time portions of the data are recognized separately; therefore, the time can precede or follow the date. If you omit the time, Replication Server assumes midnight (12:00:00:000AM). If you omit the date, Replication Server assumes January 1, 1900.

Enter times according to these general rules:

- Hours range from 0 to 23; minutes and seconds range from 0 to 59; milliseconds range from 0 to 999.
- A value must have a colon or an “AM” or “PM” indicator to be recognized as a time value.
- You can append “AM” or “PM,” with or without an intervening space. 12AM is midnight and 12PM is noon. If you specify AM, the hour must be between 1 and 12 (0 is acceptable in place of 12). If you specify PM, the hour must be between 13 and 23.
- Milliseconds can be preceded by either a colon or a period. If preceded by a colon, the number means thousandths of a second. If preceded by a period, a single digit means tenths of a second, two digits mean hundredths of a second, and three digits mean thousandths of a second. For example, “12:30:20:1” means twenty and one-thousandth of a second past 12:30; “12:30:20.1” means twenty and one-tenth of a second past 12:30.
- You can omit any portion of a time value. If you omit seconds, you must also omit milliseconds. If you omit minutes, you must also omit seconds and milliseconds. Replication Server assumes zero for any omitted part.

Here are some examples of time literals:

```
2:00
14.30
14:30:20
14:30:20:500
4pm
11:41:36 AM
12:48:5.333 pm
```

Enter dates with the month, day, and year in any order, subject to the following rules:

- You can enter the month as a number from 1–12, or use the U.S. English month name or its three-character abbreviation.
- If you use the numeric month, the date parts must be separated with slashes (/), hyphens (-), or periods (.). The date parts must be given in month-day-year order.

- These examples show different ways to enter the date March 15, 1998:

```
3-15-1998
March-15-1998
March 15 1998
15/March/1998
March.15.1998
```

- You can abbreviate U.S. English months to 3 characters. Case is not significant.

```
JAN 9 1998
31 oct 1997
```

- When you use an alphabetic month, the month and day can be followed by a comma. These are valid dates:

```
Nov 17, 1997
1997 Nov, 17,
17 Nov, 1997
```

- You can enter the year with one, two, or four digits. A one- or two-digit year less than 50 is assumed to be in the current (twenty-first) century. A two-digit year greater than or equal to 50 is in the last (twentieth) century.
- Four-digit years are recognized anywhere in a date value. Two-digit years must appear after the day of the month.
- You can omit the day of the month if you use the alphabetic month and a four-digit year. The day defaults to the first of the month. You cannot use separators other than commas after the month name.

Replication Server interprets these dates as May 1, 1998:

```
May 1998
1998 MAY
may, 1998
```

These examples show how to use `bigdatetime` and `bigtime` in a replication definition, a function replication definition, and a subscription. In these examples:

- PDS – primary data server
- pdb1 – primary database
- RDS – replicate data server
- rdb1 – replicate database
- tb1 – table

- col1, col2, col3 – columns
- repl – replication definition
- func1 – function replication definition
- sub1 – subscription

Example 1 Using the datatypes in a replication definition.

```
create replication definition repl
with primary at PDS.pdb1
with all tables named tb1
(col1 int, col2 bigdatetime, col3 bigtime)
primary key (col1)
```

Example 2 Using the datatypes in a function replication definition.

```
create function replication definition func1
with primary at PDS.pdb1
(@par1 int, @par2 bigdatetime, @par3 bigtime)
searchable parameters (@par1)
```

Example 3 Using the datatypes in a subscription.

```
create subscription sub1 for repl
with replicate at RDS.rdb1
where col3 = '14:20:00.010101'
without materialization
```

Binary datatypes

The binary datatypes are:

- binary(*n*) – up to 32,768 bytes of fixed-length binary data. The binary datatypes are used for storing programming code or pictures, not for numeric values. Specify the maximum byte length of the value with *n*. A binary value can contain 0 bytes, but *n* must be between 1 and 32,768.
- varbinary(*n*) – up to 32,768 bytes of variable-length binary data. The varbinary datatypes are used for storing programming code or pictures, not for numeric values. Specify the maximum byte length of the value with *n*. A varbinary value can contain 0 bytes, but *n* must be between 1 and 32,768.

The difference between binary and varbinary data is the way the values are stored in Adaptive Server databases. Replication Server treats them as equivalent types, but maintains the distinction so that the storage method is the same in primary and replicate databases.

- **rawobject in row** – 255 bytes of variable-length binary data. The rawobject in row datatype is used to store serialized Java values within the data pages allocated to the table.

Replication Server handles rawobject in row data exactly as it handles varbinary data. The base datatype for rawobject in row is varbinary(255). See also “Java datatypes” on page 33.

- **rawobject large in row** – 32,768 bytes of variable-length binary data. The rawobject large in row datatype is used to store serialized Java values within the data pages allocated to the table.

Replication Server handles rawobject large in row data the same as it handles varbinary data. The base datatype for rawobject large in row is varbinary(32768). See also “Java datatypes” on page 33.

- **image** – variable-length binary columns up to 2,147,483,647 bytes in length.

Replication Server 15.1 supports datatype conversion between LOB datatypes such as text, untext, and image datatypes with text pointer and text, untext, and image datatypes without text pointer.

- **rawobject** – variable-length binary columns up to 2,147,483,647 bytes in length. The rawobject datatype is used to store serialized Java values. Replication Server does not support datatype conversion of rawobject data. This means if your replication definition declares a column as rawobject, the primary table’s column must be rawobject.

Replication Server handles rawobject data exactly as it handles image data. The base datatype for rawobject is image. See also “Java datatypes” on page 33.

Entry format for binary data

Enter binary, varbinary, image, rawobject, rawobject in row, and rawobject large in row literal values using the hexadecimal digits 0-9 and A-F (or a-f). Each byte is represented by 2 hexadecimal digits, and the entire value is preceded by “0x”. The following example is a 10-byte binary string:

```
0x010305070B0D1113171D
```

Replication Server outputs the “0x” prefix when it substitutes binary values in function-string output templates.

Bit datatype

The bit datatype is used for Boolean values:

bit – either 1 or 0. Integer values other than 1 or 0 are interpreted as 1.

Unicode datatypes

Replication Server supports three Unicode datatypes, `unichar(n)`, `univarchar(n)`, and `unitext`. Unicode allows you to mix languages from different language groups in the same data server.

The Unicode datatypes behave exactly like their equivalent Replication Server datatypes. See “Character datatypes” on page 24 for more information.

- `unichar` → `char`
- `univarchar` → `varchar`
- `unitext` → `text`

The Unicode datatypes share the syntax and semantics of their equivalent datatypes, except Unicode values are always stored in UTF-16, regardless of the Replication Server default character set. `unichar(n)` is a fixed-width, non-nullable datatype. `univarchar(n)` is a variable-width, nullable datatype. For `unichar(n)` and `univarchar(n)`, use `n` to specify the number of Unicode characters. `unitext` is variable-width, nullable datatype.

You can:

- Replicate `unichar(n)`, `univarchar(n)`, and `unitext` columns to replicate and standby databases
- Use `unichar(n)` and `univarchar(n)` columns in the primary key of a replication definition
- Use `unichar(n)` and `univarchar(n)` columns as searchable columns in a replication definition and in the where clauses of associated subscriptions and articles
- Use `unichar(n)` and `univarchar(n)` columns as searchable columns in a function replication definition and in the where clauses of associated subscriptions and articles
- Use `unichar(n)`, `univarchar(n)`, and `unitext` columns when replicating to or from heterogeneous data servers

In the same way as text:

- `unitext` columns cannot be part of the primary key in the replication definition.
- `unitext` columns cannot be specified as searchable columns in a replication definition.
- `unitext` columns cannot be specified as searchable columns in a function replication definition.
- `unitext` datatype cannot be used as a base datatype or a datatype definition or as a source or target of either a column-level or class-level translation.

To correctly replicate the `unichar` and `univarchar` columns, the Replication Server must be configured:

```
RS_charset=utf8
```

If the Replication Server default character set is not UTF-8, Replication Server can replicate only `unichar` and `univarchar` characters in the ASCII-7 code range.

Upgrade issues

To fully support the `unichar` and `univarchar` datatypes, both the primary and replicate Replication Server must be running version 12.5 or later.

To fully support the `unitext` datatype, both the primary and replicate Replication Servers must be running version 15.0.1 or later, the route version must be 15.0.1 or later, and the LTL version must be 700 or above. If the LTL version is less than 700 at connect-source time, RepAgent converts `unitext` columns to image.

The RM route upgrade feature copies replication definitions referencing `unichar`, `univarchar`, and `unitext` datatypes from upstream Replication Servers.

Mixed-version issues

In a mixed-version environment, the route version between the primary and replicate Replication Servers determines which features are supported.

- Only Adaptive Server versions 15.5 and later support `bigdatetime` and `bigtime`. If the primary data server is at least Adaptive Server 15.5, and:
 - Primary and replicate Replication Server are version 15.5 or later, and the replicate Adaptive Server does not support the datatypes, you can create a replication definition containing a mapping for each of the two datatypes to the `varchar` datatype. Alternatively, use the `varchar` datatype instead of the two datatypes in the replication definition.

- Primary Replication Server is version 15.5 or later, and the replicate Replication Server and Adaptive Server do not support the datatypes, use the varchar datatype instead of the two datatypes in the replication definition.
- Primary and replicate Replication Server, and the replicate Adaptive Server do not support the datatypes, RepAgent automatically sends the varchar datatype to Replication Server.
- For replication of a quoted identifier to succeed, the primary Replication Server and the Replication Server that connects to the replicate data server version must be 15.2. However, intermediate Replication Servers in a route can be earlier versions.
- A replication definition created with untext columns is not propagated to Replication Server version 12.6 and earlier.
- A replication definition subscribed by Replication Server version 12.6 and earlier cannot be altered to add untext columns.
- A replication definition created with untext columns is propagated to Replication Server version 12.6 or earlier if the untext columns are removed.

Java datatypes

Java columns pass through the replication system as any of three Replication Server datatypes:

- As rawobject, in which the information is stored in the database in a separate location in the same way that image data is stored. The base datatype of rawobject is image. rawobject is the default datatype for Java columns in Replication Server.
- As rawobject in row, in which the information is stored in the database on consecutive data pages allocated to the table in the same way that char data is stored. The base datatype of rawobject in row is varbinary(255).
- As rawobject large in row, in which the information is stored in the database on consecutive data pages allocated to the table in the same way that char data is stored. The base datatype of rawobject large in row is varbinary(32768).

rawobject, rawobject in row, and rawobject large in row datatypes are compatible only with their base datatypes. They are not compatible with each other. You cannot replicate one Java datatype to the other Java datatype, or vice versa. See also “Binary datatypes” on page 29.

The rs_subcmp reconciliation utility treats Java datatypes as their base datatypes.

opaque datatypes

The opaque datatype handles datatypes currently not supported by Replication Server. RepAgent provides the formatted data for Replication Server to apply directly to the target data server. Examples of such datatypes are the anydata datatype of Oracle and the sql_variant datatype of Microsoft SQL Server.

Limitations

The limitations of the opaque datatypes are:

- You cannot use opaque datatypes in searchable columns and where clauses of replication definitions, subscriptions, and articles.
- You cannot use a map to clause on opaque datatypes.
- You cannot use the dynamic SQL feature when an opaque datatype column or parameter exists in your replication definition.
- You cannot use opaque datatype if your function string has a remote procedure call (RPC).
- You cannot apply character conversion or byte-order conversion to opaque data.

Mixed-version support

To support opaque datatype, the primary and replicate Replication Server must have a site version of at least 15.1 and an LTL version of at least 710.

Datatype definitions

Sybase provides a set of user defined datatypes and datatype classes. You can use them to change the datatype of column values when you replicate between:

- Sybase data servers
- Sybase data servers and non-Sybase data servers

- Homogeneous non-Sybase data servers
- Heterogeneous non-Sybase data servers

A datatype definition describes a non-Sybase datatype in terms of a base Replication Server native datatype. The base datatype determines the maximum and minimum length associated with the datatype definition and provides defaults for other datatype attributes. The base datatype also defines the delimiters associated with the datatype definition.

Each datatype class contains datatype definitions for a specific data server. The datatype classes are:

- Adaptive Server – `rs_sqlserver_udd_class`
- SQL Anywhere® – `rs_asa_udd_class`
- DB2 – `rs_db2_udd_class`
- Microsoft SQL Server – `rs_mssql_udd_class`
- Oracle – `rs_oracle_udd_class`

For a list and description of supported datatype definitions for each datatype class, see the *Replication Server Heterogeneous Replication Guide*.

Identifiers

Identifiers are symbolic names for objects—databases, tables, replication definitions, publications, subscriptions, functions, parameters, function string variables, and so on.

Identifiers are 1–255 bytes long for these objects:

- Tables
- Columns
- Procedures
- Parameters

- Functions - as part of function replication definition or internal functions

Note The create function, alter function, and drop function commands do not support long identifiers. The name of the function and the parameters of these commands cannot exceed 30 bytes.

- Function strings
- Replication definitions – including table replication definitions, function replication definitions, and database replication definitions
- Articles
- Publications

All other identifiers are 1–30 bytes long.

If an identifier is not enclosed in quotes, its first character must be an ASCII letter. Subsequent characters can be ASCII letters, digits, or the \$ or _ character. Embedded spaces are not allowed.

Identifiers that begin with the characters “rs_” are reserved for Replication Server. See “Reserved words” on page 38 for a list of other reserved words.

Parameter names for Replication Server functions and Adaptive Server stored procedures are the only identifiers that can begin with the @ character.

- Replication Server function parameter names can be up to 256 bytes including the @ character.
- Adaptive Server stored procedure parameter names can be up to 255 bytes including the @ character.

You can use reserved words for identifiers by enclosing the identifiers in double quotes. When you use quotes, you can also use embedded spaces and otherwise prohibited characters, such as !@#%&*(), and 8-bit and multibyte characters. Replication Server strips any trailing blanks from the end of the identifier, even if you have placed it within quotes. For example:

```
check subscription "publishers_sub"
for "publishers_rep"
```

```
with replicate at "SYDNEY_DS"."pubs2"
```

Warning! Adaptive Server allows you to place identifiers within quotes when you set `quoted_identifier` to `on`. This lets you use reserved words for Adaptive Server object names. However, Replication Server does not recognize identifiers in quotes in the commands that it sends to Adaptive Server, so you cannot use Transact-SQL keywords as names for replicated Adaptive Server objects.

If necessary, you can alter function strings to place quotes around identifiers for replicated objects.

Enclose variable names in function-string templates in question marks. For example, this variable name could be used in a function string to refer to a primary database:

```
?rs_origin_db!sys?
```

or, using quoted identifiers:

```
? "rs_origin_db" !sys?
```

Name space for identifiers

The name space of an identifier is the scope in which Replication Server recognizes it. A data server name, for example, has a global name space because the name can be used for only one data server in the entire replicated data system. A column name, on the other hand, has table scope; it must be qualified with the name of the table because more than one table can have a column with the same name.

Table 2-2 shows the Replication Server name space for each identifier.

Table 2-2: Name space for Replication Server identifiers

Identifier type	Name space
Article	Publication
Column	Table
Data server	Global
Database	Data server
Error class	Global
Function-string class	Global

Identifier type	Name space
Function	Replication definition. User-defined functions used for asynchronous procedures executed in Adaptive Server databases must have globally unique names, unless a table replication definition is specified in the procedure.
Function replication definition	Global
Parameter	Function
Publication	Primary data server and database
Replication definition	Global
Replication Server	Global
Subscription	Replication definition, replicate data server, and database. Subscriptions must have globally unique names.
User	Replication Server
Variable	Function or table

You should adopt a naming convention for replication definitions and other Replication Server objects with global scope to ensure that names remain unique in the global name space.

Warning! Identifiers with global name space must be managed carefully. Replication Server cannot detect all duplications in the global name space immediately, but errors may occur later.

Identifiers with a name space other than global sometimes must be qualified. For example, the syntax for many Replication Server commands includes an **at** clause, which identifies the data server and database where a table is located:

at *data_server.database*

In a correctly configured system, all servers will use the same sort order. If servers do not use the same sort order, different servers will compare identifiers inconsistently, which can lead to abnormal behavior in the network.

Reserved words

The words in Table 2-3 are reserved Replication Server keywords. Although the words are shown in lowercase, Replication Server is not case-sensitive. Therefore, all combinations of uppercase and lowercase letters are reserved. Replication Server also reserves all keywords and identifiers beginning with “rs_”.

Table 2-3: Replication Server reserved words

	Words
<i>A</i>	abort, _aco, action, activate, active, add, _add_recov_pending, admin, _af, after, all, allow, alter, always_rep, always_replicate, _alt_attr2, _alter_attributes2, _alter_col_objid, and, _ap, _apd, article, articles, _apd, append, applied, _ar, _arp, article, articles, as, assign, at
<i>B</i>	before, begin, _bf, _bg
<i>C</i>	changed, _ch, check, ci, class, _cm, columns, commit, configure, connect, connection, connections, connector, controller, create
<i>D</i>	database, datarow, dataserver, ddl, debug, define, definition, deletelen, deliver, description, disconnect, display_only, distribute, distribution, distributor, _dln, _dr, drop, drop_repdef, _ds, dsi_suspended, dump, dynamic
<i>E</i>	enable, error, exec, execute, expand
<i>F</i>	_fi, first, for, from, function, functions
<i>G</i>	get, grant
<i>H</i>	_ha, hastext, holdlock
<i>I</i>	ignore, in, incrementally, init, installjava, internal_use_only, into, _instj, _isb, _isbinary
<i>J</i>	_jar
<i>K</i>	key
<i>L</i>	language, large, last, load, log, logical, loss
<i>M</i>	maintenance, map, marker, materialization, message, _mbf, min_before, min_row, minimal, move, _mr
<i>N</i>	name, named, _ne, never_rep, new, next, no, no_password, none, not, notrep, nowait, npw, _nr, _nu, null, nullable
<i>O</i>	of, off, offset, on, only, open_xact, or, _os, osid, output, overwrite, owner
<i>P</i>	parameters, parent, partialupd, partition, passthru, password, primary, procedure, procedures, profile, _pu, public, publication, purge
<i>Q</i>	queue, queues, quoted
<i>R</i>	_rar, rebuild, reconfigure, recover, recovery, references, reject, remove, _rename_phystable_name, _reorder_columns, repfunc, replay, rep_if_changed, replicate, replicate_if_changed, replication, request, _resetq, _resetqueue, resetqueue, resume, resync, retry, revoke, _rc, _rf, _rl, _roc, rollback, route, row, rpc, _rpn, _rs_alterrepdef, rs_rcl, rs_ticket, _rsc, rsrpc
<i>S</i>	scan, schedule, searchable, segment, select, send, sendallxacts, seq, server, set, shutdown, site, size, skip, source, sql, sqlddl, sqldml, _st, standby, starting, status, stdb, string, subscribe, subscription, suspend, suspension, switch, sys_sp, sysadmin, system
<i>T</i>	table, tables, template, textcol, textlen, _tl, _tn, to, _tp, tpinit, tnull, _tr, trace, tran, transaction, transactions, transfer, truncate, truncation, twosave
<i>U</i>	_up, unsigned, update, use, user, username, using
<i>V</i>	validate, verify, verify_repserver_cmd, vers
<i>W</i>	wait, warmstdb, _wh, where, with, without, withouttp, _wo, writetext
<i>Y</i>	_yd, yielding
<i>Z</i>	_zl, zerolen

Support for Adaptive Server

This section outlines specific Replication Server support for Adaptive Server.

Replication Server supports international customers by providing:

- Support for all Sybase-supported character sets, including 8-bit, multibyte character sets, and Unicode character sets
- Support for all Sybase-supported sort orders, including non-binary sort orders and Unicode sort orders
- Localization of Replication Server messages into English, French, German, and Japanese languages
- Support for Replication Server logical page size, number of columns and columns size, number of arguments for stored procedures

The following information describes these features. For guidelines on designing a replication system in an international environment, refer to Chapter 7, “International Replication Design Considerations,” in the *Replication Server Design Guide*.

Character set support

Replication Server supports all Sybase-supported character sets and performs character set conversion of data and identifiers, as needed. The following guidelines apply to character set conversion:

- Replication Server, like all Sybase software, cannot convert between single-byte character data and multibyte character data.
- Identifiers, such as table and column names, that contain multibyte characters or single-byte characters with the high bit set must be enclosed in double quotes.
- XML text data must either be encoded in a single-byte character set, or must be encoded in the same character set as Adaptive Server.

Specifying character sets

You specify character sets with the *RS_charset* parameter in the Replication Server configuration file. You can also specify a character set for writing the Replication Server configuration file. This parameter is *CONFIG_charset*.

For replication to work properly, the Replication Server's character set must be the same as the character set of the data servers it controls. It should also be compatible with the character sets of all other Replication Servers in your system.

Character set conversion

Replication Server performs character set conversion on data and identifiers between primary and replicate databases. However, Replication Server does not perform character set conversion between incompatible character sets. If the character sets are compatible, but one or more characters are not common to both character sets, a question mark (?) is substituted for the unrecognized characters.

A configuration parameter in the `rs_config` system table, `dsi_charset_convert`, gives you options for how Replication Server handles character set conversion. You set this parameter with the `alter connection` command. For more information about these options, see `alter connection` on page 123.

rs_get_charset
system function

Each time Replication Server connects to a data server, it executes `rs_get_charset`, which obtains the character set used by the data server. If it is not what is expected, Replication Server prints a warning message to the error log file. For more information, see `rs_get_charset` on page 483.

Sort order support

Replication Server uses sort order, or collating sequence, to determine how character data and identifiers are compared and ordered. Replication Server supports all Sybase-supported sort orders, including non-binary sort orders. Non-binary sort orders are necessary for the correct ordering of character data and identifiers in European languages.

You specify sort orders with the `RS_sortorder` parameter in the Replication Server configuration file. You can specify any Sybase-supported sort order that is compatible with your character set.

For replication to work properly, all sort orders in your replication system should be the same.

rs_get_sortorder
system function

Each time Replication Server connects to a data server, it executes `rs_get_sortorder`, which obtains the sort order used by the data server. If it is not what is expected, Replication Server prints a warning message to the error log file. For more information, see `rs_get_sortorder` on page 487.

Message language support

Replication Server can print messages in French, German, and Japanese to the error log and to clients. You specify languages with the *RS_language* parameter in the Replication Server configuration file.

You can specify any language to which the Replication Server has been localized that is compatible with your character set. English is the default language and is compatible with all Sybase character sets.

Stored procedure
messages

The *rs_msgs* system table stores localized error messages used during installation and by the Replication Server stored procedures that manage the RSSD. For details about this system table, see *rs_queuemsg*.

Extended page- and column-size support

Replication Server version 12.5 and later supports the extended limits supported by Adaptive Server version 12.5 and later. Replication Server supports:

- A choice of logical page sizes: 2K, 4K, 8K, or 16K
- Larger rows (to the limit of the page size)
- Wider columns (to the limit of the page size)
- Wider index keys
- More columns per table
- Larger messages (greater than 16K bytes)

For more information about extended limits in Replication Server, see the *Replication Server Administration Guide Volume 1*.

Mixed-version environments

If a replication system domain has Replication Server 15.5 and later, all Replication Servers, and all site and route versions in the replication system domain must be version 12.6 and later.

You must upgrade your Replication Server to version 12.6 or later before you can upgrade to version 15.5.

Replication Server Commands

This chapter contains the reference pages for the RCL commands.

Table 3-1 provides a brief description of the commands in this chapter.

Table 3-1: RCL commands

Command	Description
abort switch	Aborts the switch active command, unless Replication Server has gone too far in the active switch process to abort it.
activate subscription	For a subscription to a replication definition or a publication, starts the distribution of updates from the primary to the replicate database and sets the subscription status to ACTIVE.
add partition	Makes a partition available to Replication Server. A partition can be a disk partition or an operating system file. See create partition .
admin config	Retrieves Replication Server parameters such as global, connection, logical connection, and route parameters.
admin disk_space	Displays use of each disk partition accessed by the Replication Server.
admin echo	Returns the string entered by the user.
admin get_generation	Retrieves the generation number for a primary database.
admin health	Displays the status of the Replication Server.
admin log_name	Displays the path to the current log file.
admin logical_status	Displays status information for logical connections.
admin pid	Displays the process ID of the Replication Server.
admin quiesce_check	Determines if the queues in the Replication Server have been quiesced.
admin quiesce_force_rsi	Determines whether a Replication Server is quiescent and forces it to deliver and obtain acknowledgments for messages in RSI queues.
admin rssid_name	Displays the names of the data server and database for the RSSD.
admin schedule	Displays information on a schedule.
admin security_property	Displays information about supported network-based security mechanisms and security services.
admin security_setting	Displays network-based security parameters and values for the Replication Server.
admin set_log_name	Closes the existing Replication Server log file and opens a new log file.
admin show_connection_profiles	Lists the profile name, version, and comments for each profile defined in Replication Server.

Command	Description
admin show_connections	Displays information about all connections from the Replication Server to data servers and to other Replication Servers.
admin show_function_classes	Displays the names of existing function-string classes and their parent classes, and indicates the number of levels of inheritance.
admin show_route_versions	Displays the version number of routes that originate at the Replication Server and routes that terminate at the Replication Server.
admin show_site_version	Displays the site version of the Replication Server.
admin sqm_readers	Displays the read and delete points of the threads that are reading a stable queue.
admin stats	Displays information and statistics about Replication Server counters.
admin stats, backlog	Reports the current transaction backlog in the stable queues.
admin stats, cancel	Cancels the currently running asynchronous command.
admin stats, {md mem mem_in_use}	Reports information about memory usage.
admin stats, reset	Resets all counters that can be reset.
admin stats, status	Displays the flushing status for all counters.
admin stats, {tps cps bps}	Reports the number of transactions, commands, or bytes of throughput per second.
admin time	Displays the current time of Replication Server.
admin translate	Performs a datatype translation on a value, displaying the results in delimited literal format.
admin verify_repserver_cmd	Verifies that Replication Server can successfully execute a replication definition request.
admin version	Displays the version number of the Replication Server software.
admin who	Displays information about threads running in the Replication Server.
admin who_is_down	Displays information about Replication Server threads that are not running.
admin who_is_up	Displays information about Replication Server threads that are running.
allow connections	Places Replication Server in recovery mode for specified databases.
alter applied function replication definition	Changes an existing applied function replication definition.
alter connection	Changes the attributes of a database connection.
alter connector	Changes the attributes of a database connector.
alter database replication definition	Changes an existing database replication definition.
alter error class	Changes an existing error class by copying error actions from another error class.
alter function	Adds parameters to a user-defined function.
alter function replication definition	Changes an existing function replication definition.
alter function string	Replaces an existing function string.

Command	Description
alter function string class	Alters a function-string class, specifying whether it should be a base class or a derived class.
alter logical connection	Disables or enables the Distributor thread for a logical connection, changes attributes of a logical connection, and enables or disables replication of truncate table to the standby database.
alter partition	Alters the size of a partition.
alter queue	Specifies the behavior of the stable queue that encounters a large message greater than 16K bytes.
alter replication definition	Changes an existing replication definition.
alter request function replication definition	Changes an existing request function replication definition.
alter route	Changes the attributes of a route from the current Replication Server to a remote Replication Server.
alter schedule	Enables or disables a schedule that executes commands.
alter user	Changes a user's password.
assign action	Assigns Replication Server error-handling actions to data server errors received by the DSI thread.
check publication	Finds the status of a publication and the number of articles the publication contains.
check subscription	Finds the materialization status of a subscription to a replication definition or a publication.
configure connection	Changes the attributes of a database connection. See <code>alter connection</code> .
configure logical connection	Changes attributes of a logical connection. See <code>alter logical connection</code> .
configure replication server	Sets characteristics of the Replication Server, including network-based security.
configure route	Changes the attributes of a route from the current Replication Server to a remote Replication Server. See <code>alter route</code> .
connect	Turns Replication Server into a gateway to its RSSD, ID server, a remote Replication Server, or to a remote data server.
create applied function replication definition	Creates an applied function replication definition and user-defined function for a stored procedure that is to be replicated.
create article	Creates an article for a table or function replication definition and specifies the publication that is to contain the article.
create connection	Adds a database to the replication system and sets configuration parameters for the connection. To create a connection for an Adaptive Server database, use <code>Sybase Central</code> or <code>rs_init</code> .
create connection using profile	Uses predefined information to configure the connection between Replication Server and a non-Adaptive Server database, and, if needed, to modify the RSSD and the named <code>data_server.database</code> .
create database replication definition	Creates a replication definition for replicating a database or a database object.

Command	Description
create error class	Creates an error class.
create function	Creates a user-defined function.
create function replication definition	Creates a function replication definition and user-defined function for a stored procedure that is to be replicated.
create function string	Adds a function string to a function-string class. Replication Server uses function strings to generate instructions for data servers.
create function string class	Creates a function-string class.
create logical connection	Creates a logical connection. Replication Server uses logical connections to manage warm standby applications.
create partition	Makes a partition available to Replication Server. A partition can be a disk partition or an operating system file.
create publication	Creates a publication for tables or stored procedures that are to be replicated as a group to one or more subscribing replicate databases.
create request function replication definition	Creates a request function replication definition and user-defined function for a stored procedure that is to be replicated.
create replication definition	Creates a replication definition for a table that is to be replicated.
create route	Designates the route to use for a connection from the current Replication Server to a remote Replication Server.
create schedule	Creates a schedule to execute shell commands at a time you specify.
create subscription	Creates and initializes a subscription and materializes subscription data. The subscription may be for a database replication definition, a table replication definition, a function replication definition, or a publication.
create user	Adds a new user login name to a Replication Server.
define subscription	Adds a subscription to the Replication Server system tables, but does not materialize or activate the subscription. The subscription may be for a database replication definition, a table replication definition, a function replication definition, or for a publication. This command begins the process of bulk subscription materialization, or the process of refreshing a publication subscription.
disc	Terminates connection to a server. See disconnect .
disconnect	Terminates connection to a server.
drop article	Drops an article and optionally drops its replication definition.
drop connection	Removes a database from the replication system.
drop database replication definition	Drops an existing database replication definition.
drop error class	Drops an error class and any actions associated with it.
drop function	Drops a user-defined function and its function strings.
drop function replication definition	Drops a function replication definition and its user-defined function.

Command	Description
drop function string	Drops a function string for a function-string class.
drop function string class	Drops a function-string class.
drop logical connection	Drops a logical connection. Logical connections are used to manage warm standby applications.
drop partition	Removes a disk partition from the Replication Server.
drop publication	Drops a publication and all of its articles, and optionally drops the replication definitions for the articles.
drop replication definition	Drops a replication definition and its functions.
drop route	Closes the route to another Replication Server.
drop schedule	Drops a schedule that executes commands.
drop subscription	Drops a subscription to a database replication definition, table replication definition, function replication definition, article, or publication.
drop user	Drops a Replication Server user login name.
grant	Assigns permissions to users.
ignore loss	Allows Replication Server to accept messages after it detects a loss.
move primary	Changes the primary Replication Server for an error class or a function-string class.
rebuild queues	Rebuilds Replication Server stable queues.
resume connection	Resumes a suspended connection.
resume distributor	Resumes a suspended Distributor thread for a connection to a database.
resume log transfer	Allows the RepAgent to connect to the Replication Server.
resume queue	Restarts a stable queue stopped after being passed a message larger than 16K bytes.
resume route	Resumes a suspended route.
revoke	Revokes permissions from users.
set	Prevents failures that would otherwise be caused by missing or duplicate rows in a replicated table.
set log recovery	Specifies databases whose logs are to be recovered from offline dumps.
set proxy	Switches to another user.
show connection	Lists the contents of the connection stack.
show server	Displays the current working server.
shutdown	Shuts down a Replication Server.
suspend connection	Suspends a connection to a database.
suspend distributor	Suspends the Distributor thread for a connection to a primary database.
suspend log transfer	Disconnects a RepAgent from a Replication Server and prevents a RepAgent from connecting.
suspend route	Suspends a route to another Replication Server.
switch active	Changes the active database in a warm standby application.

Command	Description
sysadmin apply_truncate_table	Turns on or off the “subscribe to truncate table” option for all existing subscriptions to a particular table, enabling or disabling replication of truncate table.
sysadmin cdb	Administer the net-change database in real-time loading (RTL) replication to Sybase IQ and high volume adaptive replication (HVAR) into Adaptive Server.
sysadmin dropdb	Drops a database from the ID Server.
sysadmin dropldb	Drops a logical database from the ID Server.
sysadmin drop_queue	Deletes a stable queue. Use this command to drop a failed materialization queue.
sysadmin dropsrs	Drops a Replication Server from the ID Server.
sysadmin dump_file	Specifies an alternative log file name for use when dumping a Replication Server stable queue.
sysadmin dump_queue	Dumps the contents of a Replication Server stable queue.
sysadmin dump_thread_stacks	Dumps Replication Server stacks.
sysadmin dump_tran	Dumps the commands of a stable queue transaction into a log file.
sysadmin erssd	Displays ERSSD name, schedule, backup directory, and ERSSD file locations. Used with options, this command performs unscheduled backups and moves ERSSD files.
sysadmin fast_route_upgrade	Updates the route version to the site version of the lower of the primary or replicate Replication Server.
sysadmin hibernate_off	Turns off hibernation mode for the Replication Server and returns it to an active state.
sysadmin hibernate_on	Turns on hibernation mode for (or suspends) the Replication Server.
sysadmin issue_ticket	Injects an rs_ticket marker in to the inbound queue.
sysadmin log_first_tran	Writes the first transaction in a DSI queue into the exceptions log.
sysadmin purge_all_open	Purges all open transactions from an inbound queue of a Replication Server.
sysadmin purge_first_open	Purges the first open transaction from the inbound queue of a Replication Server.
sysadmin purge_route_at_replicate	Removes all references to a primary Replication Server from a replicate Replication Server.
sysadmin restore_dsi_saved_segments	Restores backlogged transactions.
sysadmin set_dsi_generation	Changes a database generation number in the Replication Server to prevent the application of transactions in the DSI stable queue after a replicate database is restored.
sysadmin site_version	Sets the site version number for the Replication Server. This lets you use the software features in the corresponding release, and prevents you from downgrading to an earlier release.
sysadmin skip_bad_repserver_cmd	Instructs Replication Server to skip a failed replication definition request the next time Replication Agent starts.

Command	Description
sysadmin sqm_purge_queue	Purges all messages from a stable queue.
sysadmin sqm_unzap_command	Restores a message into a stable queue.
sysadmin sqm_unzap_tran	Restores a transaction into the stable queue.
sysadmin sqm_zap_command	Deletes a single message in a stable queue.
sysadmin sqm_zap_tran	Deletes a transaction from the stable queue.
sysadmin sqt_dump_queue	Dumps the transaction cache for an inbound queue or a DSI queue.
sysadmin system_version	Displays or sets the system-wide version number for the replication system, allowing you to use the software features in the corresponding release level.
validate publication	Sets the status of a publication to VALID, allowing new subscriptions to be created for the publication.
validate subscription	For a subscription to a replication definition or a publication, sets the subscription status to VALID. This command is part of the bulk materialization process, or part of the process of refreshing a publication subscription.
wait for create standby	A blocking command that allows a client session in the Replication Server to wait for the standby database creation process to complete.
wait for delay	Specifies a time interval at which this command is blocked.
wait for switch	A blocking command that allows a client session in the Replication Server to wait for the switch to the new active database to complete.
wait for time	Specifies a time of day at which to unblock this command.

abort switch

Description	Aborts the switch active command, unless Replication Server has gone too far in the active switch process to abort it. The switch active command changes the active database in a warm standby application.
Syntax	<code>abort switch for <i>logical_ds.logical_db</i></code>
Parameters	<p><i>logical_ds</i></p> <p>The data server name for the logical connection.</p> <p><i>logical_db</i></p> <p>The database name for the logical connection.</p>
Examples	<p>Example 1 Replication Server has gone too far in the active switch process to cancel. Wait for the switch to complete and enter another switch active command to return to the original active database.</p> <pre> abort switch for LDS.pubs2 Switch for logical connection LDS.pubs2 is beyond the point where it can be aborted. Abort command fails. </pre> <p>Example 2 Replication Server has aborted the active switch. The active database has not changed.</p> <pre> abort switch for LDS.pubs2 Switch for logical connection LDS.pubs2 has been aborted. </pre>
Usage	<ul style="list-style-type: none"> • The abort switch command attempts to cancel the switch active command. • If there is no switch in progress for the logical connection, Replication Server returns an error message. • If the command cancels the active switch successfully, you may have to restart the RepAgent for the active database. • The switch active command cannot be cancelled after it reaches a certain point. If this is the case, you must wait for the switch active to complete. Then use switch active again to return to the original active database.
Permissions	abort switch requires "sa" permissions.
See also	switch active, admin logical_status, wait for switch

activate subscription

Description	For a subscription to a replication definition or a publication, starts the distribution of updates from the primary to the replicate database and sets the subscription status to ACTIVE. This command is part of the bulk materialization process, or part of the process of refreshing a publication subscription.
Syntax	<pre>activate subscription <i>sub_name</i> for {<i>table_rep_def</i> <i>function_rep_def</i> publication <i>pub_name</i> with primary at <i>data_server.database</i>} with replicate at <i>data_server.database</i> [with suspension [at active replicate only]]</pre>
Parameters	<p><i>sub_name</i> The name of the subscription to be activated.</p> <p>for <i>table_rep_def</i> Specifies the name of the table replication definition the subscription is for.</p> <p>for <i>function_rep_def</i> Specifies the name of the function replication definition the subscription is for.</p> <p>for publication <i>pub_name</i> Specifies the name of the publication the subscription is for.</p> <p>with primary at <i>data_server.database</i> Specifies the location of the primary data. If the primary database is part of a warm standby application, <i>data_server.database</i> is the name of the logical data server and database. Use this clause only with a subscription for a publication.</p> <p>with replicate at <i>data_server.database</i> Specifies the location of the replicate data. If the replicate database is part of a warm standby application that uses logical connections, <i>data_server.database</i> is the name of the logical data server and database.</p> <p>with suspension Suspends the Data Server Interface (DSI) for the replicate database after changing the subscription status. While the DSI is suspended, Replication Server holds updates for the replicate database in a stable queue. After you load the initial data and resume the DSI, Replication Server applies the updates. In a warm standby application, this clause suspends the active database DSI and the standby DSI.</p>

with suspension at active replicate only

In a warm standby application, suspends the active database DSI but not the standby DSI.

Examples

Example 1 Activates the subscription titles_sub for the table replication definition titles_rep, where the replicate database is SYDNEY_DS.pubs2. This command suspends the DSI.

```
activate subscription titles_sub
for titles_rep
with replicate at SYDNEY_DS.pubs2
with suspension
```

Example 2 Activates the subscription myproc_sub for the function replication definition myproc_rep, where the replicate database is SYDNEY_DS.pubs2.

```
activate subscription myproc_sub
for myproc_rep
with replicate at SYDNEY_DS.pubs2
```

Example 3 Activates the subscription pubs2_sub for the publication pubs2_pub, where the primary database is TOKYO_DS.pubs2 and the replicate database is SYDNEY_DS.pubs2.

```
activate subscription pubs2_sub
for publication pubs2_pub
with primary at TOKYO_DS.pubs2
with replicate at SYDNEY_DS.pubs2
```

Usage

- Use activate subscription to activate a subscription at the primary and replicate Replication Servers. The subscription can be to a table replication definition, function replication definition, database replication definition, or publication.
- This command begins the second step in the bulk materialization process. The first step is the creation of the subscription using define subscription.
- To complete bulk materialization, load the data from media, resume the connection to the replicate database if it was suspended, and execute validate subscription.
- Execute activate subscription at the Replication Server where you created the subscription.
- activate subscription changes the status of a subscription from DEFINED to ACTIVE. Subsequent updates at the primary data server are distributed through the primary Replication Server.

- If you have added any new articles to a publication with an existing subscription, you must refresh the publication subscription by materializing the new data in order to create subscriptions for the new articles.

After using `define subscription` to begin this process, use `activate subscription` to activate the new article subscriptions. Then manually load the subscription data for the new article subscriptions, and use `validate subscription` to validate the publication subscription.

- When you activate a publication subscription, all of its article subscriptions are activated at the same time, rather than one at a time.
- This command modifies RSSD tables at multiple sites. Use `check subscription` at the primary and replicate Replication Servers to see the effects on each.
- For more information about subscription materialization, see the *Replication Server Administration Guide Volume 1*.

The *with suspension* clause

- When you use the *with suspension* clause, `activate subscription` suspends the DSI after changing the subscription status. This prevents the replicate Replication Server from sending updates for the replicated table before the subscription data is loaded.

After the data is loaded at the replicate site, execute `resume connection` to apply the updates. If you do not use *with suspension*, you should prohibit updates to the primary version until after the subscription is materialized.

- If the database is part of a warm standby application, the *with suspension* clause suspends the DSI for the active database and standby DSI after changing the subscription status. This allows you to load the data into both databases before allowing updates to continue in the active database.

If you load the data into the active database with logging (for example, by using `logged bcp` or by executing transactions in the active database), use the clause *with suspension* at active replicate only, so that the standby DSI is not suspended. In this case, you do not have to load the subscription data into the standby database because it is replicated from the active database.

Permissions

`activate subscription` can be executed by users with “create object” permission at the replicate Replication Server and “primary subscribe” permission at the primary Replication Server.

See also

`check subscription`, `create subscription`, `define subscription`, `drop subscription`, `resume connection`, `validate subscription`

add partition

Description	Makes a partition available to Replication Server. A partition can be a disk partition or an operating system file.
	<hr/> Note add partition and create partition are identical except for the command name. For backward compatibility, add partition is still supported as an alias for create partition but it will be deprecated in the future. <hr/>
Syntax	For syntax information, see create partition.
Usage	For usage information, see create partition.

admin config

Description	Displays all Replication Server configuration parameters.
Syntax	admin config [.,[[[{"connection" logical_connection} , data_server, database] ["route", repserver]] [, configuration_name] ["table", data_server, database, [, table_name [, table_owner], [, configuration_name]]]]]
Parameters	<p>"connection" Displays connection configuration parameters.</p> <p>logical_connection Displays logical connection configuration parameters.</p> <p>"table" Specifies the name of a table being queried on. Use together with <i>table_name</i> which is a character string of up to 200 characters. <i>table_owner</i> is an optional qualifier for the table name, representing the table owner.</p> <p> If you do not specify a table name, admin config displays configuration parameters for all tables.</p> <p><i>data_server, database</i> The data server and database being queried on.</p> <p> If the configuration parameters to be displayed are related to a connection, the server must be a data server, and <i>database</i> must be supplied. If the parameters to be displayed are related to a route, server must be a Replication Server, and you cannot supply <i>database</i>.</p>

"route"

Displays route configuration parameters.

repserver

Specifies the target Replication Server of the route.

configuration_name

The configuration parameter whose values and status you want to display.

Examples

Example 1 Displays all Replication Server global configuration parameters:

```
admin config
go
Configuration          Config  Run    Default
                        Value   Value  Value
-----
cm_max_connections     65     65     64
dsi_cmd_batch_size     8193   8193   8192

Legal Values           Datatype  Status
-----
range: 1,2147483647   integer   Restart required
range: 1,2147483647   integer   Restart required
(2 rows affected)
```

Example 2 Displays all configuration parameters for route to Replication Server, TOKYO_RS:

```
admin config, "route", TOKYO_RS
```

Example 3 Displays all configuration parameters for connection to pdb1:

```
admin config, "connection", ost_wasatch_04, pdb1
go
```

```
Configuration          Config  Run    Default
                        Value   Value  Value
-----
dsi_cmd_batch_size     NULL    NULL   8192

Legal Values           Datatype  Status
-----
range: 1,2147483647   integer   Connection/Route
                                      restart required
(1 row affected)
```

Example 4

Displays all configuration parameters after using dsi_command_convert to set d2none on the tb1 table in the pubs2 database of the SYDNEY_DS data server:

admin config, "table", SYDNEY_DS, pubs2

admin config displays:

Configuration	Config Value	Run Value	Default Value
-----	-----	-----	-----
dsi_compile_enable	<server default>	<server default>	on
dsi_command_convert	d2none	d2none	none
Legal Values			Datatype
-----			-----
list: on,of			string
list: none, i2none, d2none, u2none, i2di, u2di, t2none			string
Status	Table		
-----	-----		
Restart not required	dbo.tb1		
Restart not required	dbo.tb1		
(2 rows affected)			

Example 5 Displays the configuration parameters only for dsi_command_convert after using dsi_command_convert on the tb1 table in the pubs2 database of the SYDNEY_DS dataserver:

admin config, "table", SYDNEY_DS, pubs2, tb1, dsi_command_convert

admin config displays:

Configuration	Config Value	Run Value	Default Value
-----	-----	-----	-----
dsi_command_convert	d2none	d2none	none
Legal Values			Datatype
-----			-----
list: none, i2none, d2none, u2none, i2di, u2di, t2none			string
Status	Table		
-----	-----		
Restart not required	dbo.tb1		
(1 row affected)			

Usage

Use `admin config` to retrieve the different types of configuration parameters—server, connection, logical connection, route—used to customize and tune the Replication Server:

For more information on configuring and tuning Replication Server parameters, refer to *Replication Server Administration Guide Volume 1 and Volume 2*.

admin disk_space

Description Displays use of each disk partition accessed by the Replication Server.

Syntax admin disk_space

Examples Displays information about the disk partition:

```
admin disk_space

Partition      Logical      Part.Id
-----
/dev/hdb2      partition_1  101

Total Segs     Used Segs    State
-----
                20          3      ON-LINE
```

Usage Table 3-2 describes the output columns.

Table 3-2: Column descriptions for admin disk_space output

Column	Description
Partition	Device name used by the Replication Server
Logical	Logical name assigned to the partition
Part.Id	Partition ID
Total Segs	Total number of 1MB segments on a partition
Used Segs	Total segments currently in use by the Replication Server
State	State of this device. Can be: <ul style="list-style-type: none">• ON-LINE – The device is normal• OFF-LINE – The device cannot be found• DROPPED – The device has been dropped but has not disappeared (some queues are using it)

Permissions Any user may execute this command.

See also admin who, alter partition, create partition, drop partition

admin echo

Description	Returns the string entered by the user.
Syntax	admin echo, <i>character_string</i> [, with_log]
Parameters	<i>character_string</i> The character string entered by the user. with_log Writes the string entered by the user to the Replication Server log.
Examples	<p>Example 1 The Replication Server returns “hello”, the character string entered by the user.</p> <pre>admin echo, hello echo ----- hello</pre> <p>Example 2 The Replication Server returns “Hello world!” and writes “Hello world!” to the Replication Server log.</p> <pre>admin echo, 'Hello world!', with_log echo ----- Hello world!</pre>
Usage	<ul style="list-style-type: none">• Use admin echo to determine if the local Replication Server is running.• This command does not function as a network echo. If you do not enter an argument, nothing is returned.
Permissions	Any user may execute this command.

admin get_generation

Description	Retrieves the generation number for a primary database.
Syntax	admin get_generation, <i>data_server</i> , <i>database</i>
Parameters	<p><i>data_server</i></p> <p>The data server with the primary database.</p> <p><i>database</i></p> <p>The database whose generation number you are retrieving.</p>
Examples	<pre>admin get_generation, TOKYO_DS, pubs2</pre> <p>Current generation number for TOKYO_DS.pubs2 is 0</p>
Usage	<ul style="list-style-type: none">• The database generation number is the first 2 bytes of the origin queue ID generated by a RepAgent for log records. The generation number is a parameter of the Log Transfer Language (LTL) distribute command. <p>For more information about the distribute command, refer to “distribute” in Chapter 5, “Creating a Replication Agent,” in the <i>Replication Server Design Guide</i>.</p> <ul style="list-style-type: none">• The generation number should be incremented following a load for the primary database. Incrementing the number prevents Replication Server from ignoring (as duplicates) any transactions applied after the load.• Increment the generation number by executing Adaptive Server dbcc settrunc in the Adaptive Server database.
Permissions	Any user may execute this command.
See also	dbcc settrunc

admin health

Description Displays the status of the Replication Server.

Syntax admin health

Examples Displays the status of the Replication Server.

```
admin health
Mode           Quiesce      Status
-----
NORMAL         TRUE          HEALTHY
```

Usage • Table 3-3 describes the output columns.

Table 3-3: Column descriptions for admin health output

Column	Description
Mode	<p>The state of the Replication Server with regard to recovery. It is one of these values:</p> <ul style="list-style-type: none"> • NORMAL – Replication Server is operating normally. • REBUILDING – This is a transient state while Replication Server executes the rebuild queues command. • RECOVERY – The Replication Server is in stand-alone mode and the rebuild queues command has been executed. • STANDALONE – Replication Server is not accepting or starting any connections. You can only enter this state by starting Replication Server with the -M flag. Exit from stand-alone mode by shutting down the Replication Server and restarting it without the -M flag.
Quiesce	<p>Indicates if the Replication Server is quiesced. It is either:</p> <ul style="list-style-type: none"> • TRUE – Replication Server is quiesced, that is, all messages have been flushed. • FALSE – Replication Server is not quiesced.
Status	<p>Overall status of the Replication Server. It is either:</p> <p>HEALTHY – All threads are executing as expected.</p> <p>SUSPECT – A thread is down and the Replication Server expected it to be running. Or, a thread is in a “Connecting” state. The “Connecting” state means that either the server to which Replication Server is connecting is unavailable and a problem exists, or the Replication Server will connect successfully in a moment and the suspect status is transitory.</p> <p>You can see threads that are not running by executing admin who_is_down.</p>

Permissions Any user may execute this command.

See also admin quiesce_check, admin quiesce_force_rsi, admin who, admin who_is_down, admin who_is_up, rebuild queues

admin log_name

Description	Displays the path to the current log file.
Syntax	admin log_name
Examples	Displays the path to the log file for the current Replication Server. <pre>admin log_name Log File Name ----- /work/log/TOKYO_RS.log</pre>
Usage	If you start Replication Server with the -e flag and give a full path name for the error log, admin log_name returns the full path. If you give a relative path name, admin log_name returns the relative path name in the Replication Server's current working directory.
Permissions	Any user may execute this command.
See also	admin set_log_name

admin logical_status

Description	Displays status information for logical connections.
Syntax	admin logical_status [, <i>logical_ds</i> , <i>logical_db</i>]
Parameters	<p><i>logical_ds</i> The data server name for the logical connection.</p> <p><i>logical_db</i> The database name for the logical connection.</p>
Examples	This output shows the LDS.pubs2 logical connection in its normal, active state. The current active database is the pubs2 database in the TOKYO_DS data server. The standby database is the pubs2 database in the SYDNEY_DS data server. The TOKYO_RS Replication Server manages the logical connection. Both physical connections are active. No special operations are in progress.

```
admin logical_status, LDS, pubs2
```

Logical Connection Name -----	Active Connection Name -----	Active Conn State -----	Standby Connection Name -----	Standby Conn State -----
[109] LDS.pubs2	[115] TOKYO_DS.pubs2	Active/	[116] SYDNEY_DS.pubs2	Active/

Controller RS -----	Operation in Progress -----	State of Operation in Progress -----	Spid -----
[16777317] TOKYO_RS	None	None	

Usage	<ul style="list-style-type: none"> Use admin logical_status to find the status of logical connections for an active database and a standby database in a warm standby application. If you do not specify <i>logical_ds</i> and <i>logical_db</i>, admin logical_status displays information about all logical connections controlled by this Replication Server. Table 3-4 describes the output columns.
-------	---

Table 3-4: Column descriptions for admin logical_status output

Column	Description
Logical Connection Name	The DBID (database ID) for the logical connection and the logical data server and database names.
Active Connection Name	The DBID, the data server, and the database name for the current active database.

Column	Description
Active Connection State	A description of the status of the active connection. Can be active, suspended, or suspended by error.
Standby Connection Name	The DBID, the data server, and the database name for the current standby database.
Standby Connection State	A description of the status of the standby connection. Can be active, suspended, suspended by error, or waiting for marker.
Controller RS	The RSID (Replication Server ID) and name of the Replication Server that manages the logical, active, and standby databases.
Operation in Progress	A description of the operation in progress. Can be None, Switch Active, or Create Standby.
State of Operation in Progress	The current step in the operation.
Spid	The process ID for the server thread that is executing the operation.

Permissions Any user may execute this command.

See also abort switch, admin sqm_readers, admin who, create connection, create logical connection, switch active, wait for create standby, wait for switch

admin pid

Description	Displays the process ID of the Replication Server.
Syntax	admin pid
Examples	<p>The process ID for the current Replication Server is 12032.</p> <pre>admin pid pid ----- 12032</pre>
Usage	Display the process ID of the Replication Server.
Permissions	Any user may execute this command.

admin quiesce_check

Description Determines if the queues in the Replication Server have been quiesced.

Syntax admin quiesce_check

Examples **Example 1** The TOKYO_RS Replication Server is quiescent.

```
admin quiesce_check
Replication Server TOKYO_RS is quiesced
```

Example 2 This message indicates that the system is not quiescent because there are unread messages in queue 103:1. The reported Read location (30.2) and Write location (32.1) show that more blocks in the queue have been written than read. Assuming no more blocks are written, the Read location must advance to segment 32, block 2, before the system becomes quiescent.

```
admin quiesce_check
Can't Quiesce. Queue 103:1 has not been read out.
Write=32.1 Read=30.2
```

Usage

- admin quiesce_check determines if a Replication Server is quiescent.
- The Replication Server is quiescent if:
 - There are no subscription materialization queues.
 - Replication Server has read and processed all messages in all queues.
 - No inbound (RepAgent) queues contain undelivered committed transactions.
 - All messages in RSI queues have been sent to their destination Replication Servers and acknowledgments have been received.
 - All messages in DSI queues have been applied and acknowledgments received from data servers.

Permissions Any user may execute this command.

See also admin quiesce_force_rsi, suspend connection, suspend log transfer

admin quiesce_force_rsi

Description	Determines whether a Replication Server is quiescent and forces it to deliver and obtain acknowledgments for messages in RSI queues.
Syntax	admin quiesce_force_rsi
Examples	<p>Example 1 The TOKYO_RS Replication Server is quiescent.</p> <pre>admin quiesce_force_rsi Replication Server TOKYO_RS is quiesced</pre> <p>Example 2 This message indicates that the system is not quiescent because there are unread messages in queue 103:1. The reported Write location (32.1) and Read location (30.2) show that more blocks in the queue have been written than read.</p> <pre>admin quiesce_force_rsi Can't Quiesce. Queue 103:1 has not been read out. Write=32.1 Read=30.2</pre>
Usage	<ul style="list-style-type: none">• Execute suspend log transfer from all before you execute admin quiesce_force_rsi. This prevents RepAgents from connecting with the Replication Server.• Execute this command after all inbound queues are quiescent.• The Replication Server is quiescent if:<ul style="list-style-type: none">• There are no subscription materialization queues• Replication Server has read all messages in all queues• No inbound (RepAgent) queues contain undelivered committed transactions• All messages in RSI queues have been sent to their destination Replication Servers and acknowledgments have been received• All messages in DSI queues have been applied and acknowledgments have been received from data servers• RSI normally empties its queue every 30 seconds.
Permissions	Any user may execute this command.
See also	admin quiesce_check, suspend connection, suspend log transfer

admin rssd_name

Description	Displays the names of the data server and database for the RSSD.
Syntax	admin rssd_name
Examples	<p>In the example, TOKYO_DS is the name of the data server, and TOKYO_RSSD is the name of the RSSD.</p> <pre>admin rssd_name RSSD Dataserver RSSD Database ----- TOKYO_DS TOKYO_RSSD</pre>
Usage	Display the names of the data server and database for the RSSD.
Permissions	Any user may execute this command.

admin schedule

Description Displays information on task schedules in Replication Server.

Syntax **admin “schedule”**[, ‘*sched_name*’]

Parameters ‘*sched_name*’
The name of the schedule to display.

Examples To display a schedule named schedule1, enter:

```
admin “schedule”, ‘schedule1’
```

The output is:

Schedule Name	Schedule	Time	Status	Type	Owner	Sequence	Command
-----	-----	----	-----	----	-----	-----	-----
s1	27 * * * *	*	1	0	sa	1	conn_suspend.sh

Usage You must enclose the “schedule” clause in double quotes as schedule is a Replication Server keyword.

If you do not specify any schedule name, executing only admin “schedule” displays information on all existing schedules in Replication Server.

Permissions admin “schedule” requires “sa” permission.

See also alter schedule, drop schedule, create schedule

admin security_property

Description	Displays information about supported network-based security mechanisms and security services.		
Syntax	admin security_property [, <i>mechanism_name</i>]		
Parameters	<i>mechanism_name</i> A supported network-based security mechanism.		
Examples	<pre>admin security_property Mechanism Feature Supported ----- DCE Unified Login yes DCE Confidentiality yes DCE Integrity no ...</pre>		
Usage	<ul style="list-style-type: none">When executed without options, displays the name of the default security mechanisms, the security services available for that mechanism, and whether available services are supported at your site.To execute admin security_property, network-based security must be enabled—use configure replication server to set the use_security_services parameter on—at the current Replication Server.		
Permissions	Any user can execute this command.		
See also	admin security_setting, alter connection, alter route, configure replication server, create connection, create route, set proxy		

admin security_setting

Description Displays network-based security parameters and values for the Replication Server.

Syntax `admin security_setting [, rs_idserver |, rs_server |, data_server.database]`

Parameters *rs_idserver*

The ID Server to which the current Replication Server connects.

rs_server

The Replication Server to which the current Replication Server connects.

data_server

The data server for the target database to which the current Replication Server connects.

database

The target database to which the current Replication Server connects.

Examples

```
admin security_setting
Server      Feature      Status
-----
Global      Unified Login  required
Global      Confidentiality not_required
Global      Integrity      not_required
...
```

Usage

- To execute `admin security_setting`, network-based security must be enabled—use `configure replication server` to set the `use_security_services` parameter “on”—at the current Replication Server.
- If you execute `admin security_setting` without options, Replication Server displays default values configured with `configure replication server`.

Permissions

Any user may execute this command.

See also

`admin security_property`, `alter connection`, `alter route`, `configure replication server`, `create connection`, `create route`, `set proxy`

admin set_log_name

Description	Closes the existing Replication Server log file and opens a new log file.
Syntax	admin set_log_name, <i>log_file</i>
Parameters	<i>log_file</i> The name of the new log file.
Examples	Opens a new log file called SYDNEY_RS.log. You can verify the path and log file name with the admin log_name command. <pre>admin set_log_name, '/work/log/SYDNEY_RS.log'</pre>
Usage	<ul style="list-style-type: none">• If this command fails, the original log file remains open.• If the Replication Server is restarted, the log file name specified in the command line is used. If no name is specified in the command line, the default log file name is used.• If you enter a log file name containing characters other than letters and numerals, enclose it in quotes. Do this, for example, if the log file name contains a period (.), as in the example above.• admin set_log_name displays the name you enter. Enter an absolute path name to make the output most useful.
Permissions	Any user may execute this command.
See also	admin log_name

admin show_connection_profiles

Description	Lists the profile name, version, and comments for each profile defined in Replication Server.
Syntax	admin show_connection_profiles[, " <i>match_string</i> "]
Parameters	<i>match_string</i> Filters the connection profiles displayed. Only those connection profiles whose names contain the string provided are displayed.
Examples	Example 1 Lists the names of all connection profiles currently defined in Replication Server: <pre>admin show_connection_profiles go</pre>

Profile Name	Version	Comments
-----	-----	-----
rs_ase_to_db2	standard	Standard ASE to DB2 replication connection profile.
rs_ase_to_udb	standard	Standard ASE to UDB replication connection profile.
rs_ase_to_oracle	standard	Standard ASE to Oracle replication connection profile.
rs_ase_to_msss	standard	Standard ASE to Microsoft SQLServer replication connection profile.
rs_ase_to_iq	standard	Standard ASE to Sybase IQ replication connection profile.
rs_ase_to_ase	standard	Standard ASE to ASE replication connection profile.
rs_db2_to_msss	standard	Standard DB2 to Microsoft SQLServer replication connection profile.
rs_db2_to_oracle	standard	Standard DB2 to Oracle replication connection profile.
rs_db2_to_udb	standard	Standard DB2 to UDB replication connection profile.
rs_db2_to_ase	standard	Standard DB2 to ASE replication connection profile.
rs_oracle_to_db2	standard	Standard Oracle to DB2 replication connection profile.
rs_oracle_to_udb	standard	Standard Oracle to UDB replication connection profile.
rs_oracle_to_msss	standard	Standard Oracle to Microsoft SQLServer replication connection profile.
rs_oracle_to_ase	standard	Standard Oracle to ASE replication connection profile.
rs_msss_to_db2	standard	Standard Microsoft SQLServer to DB2 replication connection profile.
rs_msss_to_oracle	standard	Standard Microsoft SQLServer to Oracle replication connection profile.
rs_msss_to_udb	standard	Standard Microsoft SQL Server to UDB replication connection profile.
rs_msss_to_sqlany	standard	Standard Microsoft SQLServer to SQL Anywhere replication connection profile.
rs_msss_to_ase	standard	Standard MicrosoftSQL Server to ASE replication connection profile.
rs_udb_to_db2	standard	Standard udb to db2 replication connection profile.
rs_udb_to_msss	standard	Standard UDB to Microsoft SQLServer replication connection profile.

rs_udb_to_oracle	standard	Standard UDB to Oracle replication connection profile.
rs_udb_to_ase	standard	Standard UDB to ASE replication connection profile.
rs_db2_to_db2	standard	Standard DB2 to DB2 replication connection profile.
rs_oracle_to_oracle	standard	Standard Oracle to Oracle replication connection profile.
rs_udb_to_udb	standard	Standard UDB to UDB replication connection profile.
rs_msss_to_msss	standard	Standard Microsoft SQLServer to Microsoft SQLServer replication connection profile.

(36 rows affected)

Example 2 Lists the names of all connection profiles currently defined in Replication Server that have the string “oracle” in the connection profile name:

```
admin show_connection_profiles, "oracle"
go
```

Profile Name	Version	Comments
rs_ase_to_oracle	standard	Standard ASE to Oracle replication connection profile.
rs_db2_to_oracle	standard	Standard DB2 to Oracle replication connection profile.
rs_oracle_to_db2	standard	Standard Oracle to DB2 replication connection profile.
rs_oracle_to_udb	standard	Standard Oracle to UDB replication connection profile.
rs_oracle_to_msss	standard	Standard Oracle to Microsoft SQLServer replication connection profile.
rs_oracle_to_ase	standard	Standard Oracle to ASE replication connection profile.
rs_msss_to_oracle	standard	Standard Microsoft SQLServer to Oracle replication connection profile.
rs_udb_to_oracle	standard	Standard UDB to Oracle replication connection profile.
rs_oracle_to_oracle	standard	Standard Oracle to Oracle replication connection profile.

Usage	When creating connections with the using profile option, use admin show_connection_profiles to determine the name and version of the available profiles.
See also	create connection using profile

admin show_connections

Description Displays information about all connections from the Replication Server to data servers and to other Replication Servers.

Syntax admin show_connections

Examples Displays connection data for this Replication Server.

```
admin show_connections

Server      User      Database
-----
SYDNEY_DS   pubs2_maint  pubs2sb
SYDNEY_RS   SYDNEY_RS_rsi  NULL

State      Owner      Spid
-----
already_faded_out  DSI      89
active        RSI      53

connection state  number  comments
-----
connecting        0      in the process of connecting to a server
active            2      established connections owned and used by
                        threads
idle              0      established connections owned but not being
                        used
being_faded_out   0      idle connections that are being closed
already_faded_out 0      idle connections that have been closed
free              1      established connections not owned by any
                        threads
closed            61     closed connections not owned by any threads
limbo             0      connection handles in state transition
total             64     total number of connection handlers available
```

- Usage
- This command displays information about database connections and routes from the current Replication Server.
 - Table 3-5 describes the output from this command.

Table 3-5: Column descriptions for admin show_connections output

Column	Description
Server	The name of the data server or Replication Server to which this Replication Server is connected
User	The login name for this client
Database	The name of the database to which this Replication Server is connected (null for routes)

Column	Description
State	The state of this connection
Owner	Indicates the owner of the thread. One of these: DSI – Data Server Interface (to a database) RSI – Replication Server Interface (to a Replication Server)
Spid	Unique identifier for this thread
connection state	One of these values: <ul style="list-style-type: none"> • active – the connection is being used • already_faded_out – the connection is owned and closed • being_faded_out – the connection is owned and is being closed • closed – closed connections are not owned by any threads • connecting – connecting to a server • free – the connection is open and not owned by anyone • idle – the connection is owned but is not used • limbo – connection handles are in a state transition • total – the total number of connections
number	The number of connections of this type
comments	A description of the connection state field

Permissions Any user may execute this command.

See also alter connection, alter logical connection, alter route, create connection, create logical connection, create route, drop connection, drop logical connection, drop route, resume connection, suspend connection

admin show_function_classes

Description Displays the names of existing function-string classes and their parent classes, and indicates the number of levels of inheritance.

Syntax admin show_function_classes

Examples

```
admin show_function_classes
```

Class	ParentClass	Level
-----	-----	-----
sql_derived_class	rs_default_function_class	1
DB2_derived_class	rs_db2_function_class	2
rs_db2_function_class	rs_default_function_class	1
rs_default_function_class	BASE_CLASS	0
(and so on)		

Usage Level 0 is a base class such as rs_default_function_class, level 1 is a derived class that inherits from a base class, and so on.

Permissions Any user may execute this command.

See also alter connection, alter function string class, create connection, create function, create function string, create function string class, drop function string class, move primary

admin show_route_versions

Description	Displays the version number of routes that originate at the Replication Server and routes that terminate at the Replication Server.									
Syntax	admin show_route_versions									
Examples	<p>In the example, the route version of repserver_1510.repserver_1500 is 15.0.0.</p> <pre>admin show_route_versions</pre> <table><tr><th>Source RepServer</th><th>Dest. RepServer</th><th>Route Version</th></tr><tr><td colspan="3">-----</td></tr><tr><td>repserver_1510</td><td>repserver_1510</td><td>1500</td></tr></table>	Source RepServer	Dest. RepServer	Route Version	-----			repserver_1510	repserver_1510	1500
Source RepServer	Dest. RepServer	Route Version								

repserver_1510	repserver_1510	1500								
Usage	<ul style="list-style-type: none">• The route version is the earliest site version of the source and destination Replication Server. If the route version is lower than the earliest site version, you need to perform route upgrade.• The version number determines which feature set in a mixed-version environment you can use with the route.• For each route, admin show_route_versions displays the name of the source Replication Server, the name of the destination Replication Server, and the version of the route.									
Permissions	Any user may execute this command.									
See also	admin show_site_version, sysadmin fast_route_upgrade									

admin show_site_version

Description	Displays the site version of the Replication Server.
Syntax	<code>admin show_site_version</code>
Examples	<p>In the example, the Replication site version is 15.1.0.</p> <pre>admin show_site_version Site Version ----- 1510</pre>
Usage	Displays the site version of the Replication Server. The site version determines which Replication Server features you can use. Once the site version is set, you cannot downgrade to an earlier release.
Permissions	Any user may execute this command.
See also	<code>sysadmin site_version</code>

admin sqm_readers

Description	Displays the read and delete points of the threads that are reading a stable queue.
Syntax	admin sqm_readers, <i>q_number</i> , <i>q_type</i>
Parameters	<p><i>q_number</i></p> <p>The ID number that Replication Server assigns to the queue. The number can be found in the output of the admin who, sqm command.</p> <p><i>q_type</i></p> <p>The type of the queue. Inbound queues have a type of 1. Outbound queues have a type of 0.</p>

Examples admin sqm_readers, 103, 1

RdrSpid	RdrType	Reader	Index
-----	-----	-----	-----
46	SQT	103:1 DIST LDS.pubs	0
57	SQT	103:1 DSI 107 SYDNEY_DS.pubs2	1

First Seg.Block	Next Read	Last Seg.Block	Delete	WriteWait
-----	-----	-----	-----	-----
14.43	14.44	14.43	1	1
14.43	14.44	14.43	1	0

- Usage
- admin sqm_readers reports the read point and the delete point for each Replication Server thread that is reading an inbound queue. You can use this information to help identify the cause when Replication Server fails to delete messages from queues.
 - Replication Server cannot delete points beyond the minimum delete point of all threads that are reading the queue. The deletion point is the first segment block.
 - Use the admin who, sqm command to find the *q_number*.
 - Table 3-6 describes the output columns for the admin sqm_readers command.

Table 3-6: Column descriptions for admin sqm_readers output

Column	Description
RdrSpid	Unique identifier for this reader.
RdrType	The type of thread that is reading the queue.
Reader	Information about the reader. For a complete description of this information, see Table 3-8 on page 108.

Column	Description
Index	The index for this reader.
First Seg.Block	The first undeleted segment and block number in the queue. This information is useful when dumping queues.
Next Read	The next segment, block, and row to be read from the queue.
Last Seg.Block	The last segment and block written to the queue. This information is useful when dumping queues.
Delete	Whether or not the reader is allowed a delete. A value of “1” indicates that the reader is allowed a delete.
WriteWait	Whether or not the reader is waiting for a write. A value of “1” indicates that the reader is waiting for a write.

Permissions Any user may execute this command.

See also admin who, admin stats

admin stats

Description	Displays information and statistics about Replication Server operations.
Syntax	<pre>admin {stats statistics} [, sysmon "all" <i>module_name</i> [, inbound outbound] [, <i>display_name</i>]] [, <i>server</i> [, <i>database</i>]] [<i>instance_id</i>] [, {display , save} [, <i>obs_interval</i>] [, <i>sample_period</i>]]</pre>
Parameters	<p>sysmon Displays statistics only for those counters identified as particularly important for performance and tuning purposes. Counters are selected from nearly all modules. This is the default.</p> <p>"all" Displays statistics from all counters.</p> <p><i>module_name</i> Displays statistics from the named module's counters, where <i>module_name</i> is cm, dsi, dist, dsixec, repagent, rsi, rsuser, serv, sqm, sqt, sts, sync, and others. Use <code>rs_helpcounter</code> to obtain valid module names.</p> <p>inbound outbound inbound and outbound are types of sqt or sqm. If neither inbound nor outbound is supplied for the sqt or sqm module, Replication Server reports statistics for both types of queues.</p> <p><i>display_name</i> Is the name of a counter. Use <code>rs_helpcounter</code> to obtain valid display names. <i>display_name</i> is only used in conjunction with <i>module_name</i>.</p> <p><i>server</i> [, <i>database</i>] If the statistics to be collected are related to a connection, <i>server</i> must be a dataserver and <i>database</i> must be supplied. If the statistics to be collected are related to a route, <i>server</i> must be a Replication Server and you are not allowed to supply <i>database</i>.</p> <p><i>instance_id</i> Identifies a particular instance of a module such as SQT or SQM. To view instance IDs, execute <code>admin who</code> and view the Info column.</p> <p>The instance ID 0 indicates Replication Server-wide statistics. It is the instance ID of the Replication Server.</p> <p>display Displays statistics on the computer screen. This is the default.</p>

save
Saves statistics in the RSSD. Old sampling data is truncated or preserved, depending on the current setting of stats_reset_rssd.

obs_interval
Specifies the length of each observation interval during the sampling period. If you do not specify an interval, there will be only one observation interval with a length equal to the sampling period. Each observation interval must be at least 15 seconds. Format can either be a numeric value in seconds, or “hh:mm[:ss]”.

sample_period
Indicates the total sampling duration. The default value is zero, which reports the current counter values. With a non-zero value, the current counter values are reset and then collected for the specified sample period. Format can either be a numeric value in seconds, or “hh:mm[:ss]”.

Examples

Example 1 Collects outbound SQT statistics for connection 108 for two minutes and sends the data to the RSSD.

```
admin stats, sqt, outbound, 108, save, 120
```

Example 2 Collects outbound SQT statistics for connection 108 for two hours and sends data to the RSSD. In addition, the sample period is divided into observation intervals of 30 seconds each.

```
admin stats, sqt, outbound, 108, save, 30, "02:00:00"
```

Example 3 Displays statistics for the SQM and SQMR modules for the inbound queue for connection 102.

```
admin stats, sqm, inbound, 102
```

Report Time: 10/31/05 02:14:17 PM

Instance	Instance ID	ModType/InstVal
SQM, 102:1 pds01.tpcc	102	1

Monitor	Obs	Last	Max	Avg ttl/obs
#*SegsActive	1	1	1	1

=====

Instance	Instance ID	ModType/InstVal
SQMR, 102:1 pds01.tpcc, 0 SQT	102	11

Observer	Obs	Rate x/sec
-----	-----	-----

SleepsWriteQ

4

0

Note In output, prefixes that precede counter names provide information about the counter. For example, a preceding # indicates a counter that is not reset, even if `admin stats, reset` is executed, and a preceding * indicates a counter that must be sampled, irrespective of the setting of `stats_sampling`. In this example, the `SegsActive` counter is always sampled and never reset.

Example 4 Collects statistics for all instances of the `SleepsWriteQ` counter in the `SQM` module.

```
admin stats, sqm, SleepsWriteQ
```

Report Time 10/31/05 02:17:03 PM

Instance	Observer	Obs	Rate x/sec
-----	-----	----	-----
SQMR, 101:0 edsprsr01.edbprs01, 0, DSI	SleepsWriteQ	0	0
SQMR, 102:0 pds01.tpcc, 0, DSI	SleepsWriteQ	0	0
SQMR, 102:1 pds01.tpcc, 0, DSI	SleepsWriteQ	20	0
SQMR, 103:0 rds01.tpcc, 0, DSI	SleepsWriteQ	0	0

Example 5 Starts sampling and saving statistics to the `RSSD` for one hour and thirty minutes at 20-second intervals:

```
admin stats, "all", save, 20, "01:30:00"
```

Usage

- `admin stats` collects and displays statistics from Replication Server modules. There are three types of statistics collectors:
 - **Observer** – counts the number of times an event occurs. For example, Replication Server uses an observer to count the number of times a command from the `RepAgent` is observed.
 - **Monitor** – periodically samples a value. For example, Replication Server uses a monitor to sample the sizes of sent commands.
 - **Counter** – collects statistics not observed by monitors and observers. Counters are usually used to accumulate a running total of a particular value, including the total number of milliseconds required to complete a particular task. For example, Replication Server uses a counter to accumulate the elapsed time between receiving two commands from the `RepAgent`.

Observer, monitor, and counter watch four types of statistics: number of observations, total observed values, last observed values, and maximum observed values.

- admin stats prints a report that includes this information:
 - Instance – a specific occurrence of a module.
 - Instance ID – the numeric identifier for a given module instance. For example, two different SQM instances may have instance IDs 102 and 103 respectively.
 - ModType/InstVal – in some cases, an instance may have multiple versions or module types. For example, a given SQM instance may have an inbound type and an outbound type. For SQM instances, inbound versions have a module type of 1 and outbound versions have a module type of 0.
 - Monitor, Observer, or Counter – displays the name of the statistics collector being observed. For example, SleepsWriteQ.
 - Obs – the number of observations of a statistics collector during an observation period.
 - Last – the last value observed during an observation period.
 - Max – the maximum value observed during an observation period.
 - Total – the sum of the observed values during an observation period.
 - Avg ttl/obs – the average value observed in an observation period. This is calculated as Total/Obs.
 - Rate x/sec – the change, in a period of 1 second, observed during the given observation period. Observers calculate this as Obs/seconds in an observation period. Monitors and counters calculate this as Total/second in an observation period.
- By default, admin stats reports values for the sysmon counters.
- By default, admin stats does not report counters that show 0 (zero) observation. To change this behavior, set the stats_show_zero_counters configuration parameter on.
- If statistics are displayed on the computer screen, they are not stored in the RSSD. Similarly, if statistics are stored in the RSSD, they are not displayed on screen.
- If you use admin stats...*display_name* to display statistics for a particular counter, Replication Server always displays statistics for that counter, even if stats_sampling is off and the number of observations is zero.

- Use `admin stats` with the independent module name to collect statistics for dependent modules. You cannot collect statistics using the dependent module name in the `admin stats` command.

Independent module	Dependent module
Data Server Interface (DSI)	DSI Executor (DSIEXEC)
Data Server Interface (DSI)	Active Object Statistics (AOBJ)
Stable Queue Manager (SQM)	SQM Reader (SQMR)
Thread Synchronization (SYNC)	SYNC Element (SYNCELE)

For more information about Replication Server modules, see the *Replication Server Administration Guide Volume 2*.

Permissions Any user may execute this command.

See also `configure replication server`

admin stats, backlog

Description Reports the volume of replicated transactions awaiting distribution in the inbound and outbound queues in terms of segments and blocks.

Syntax `admin {stats | statistics}, backlog`

Examples Reports the transaction backlog for the inbound and outbound queues.

```
admin stats, backlog
```

```
Report Time: 10/31/05 02:17:01 PM
```

Instance	Monitor	Obs	Last	Max	Avg	t1/obs
SQMR 101:0 edsprs01.edbprs01, 0, DSI	*SQMRBacklogSeg	0	0	0		0
SQMR 102:0 pds01.tpcc, 0, DSI	*SQMRBacklogSeg	0	0	0		0
SQMR 102:1 pds01.tpcc, 0, SQT	*SQMRBacklogSeg	695	3	3		1
SQMR 103:0 rds01.tpcc, 0, DSI	*SQMRBacklogSeg	0	0	0		0

```
=====
Report Time: 10/31/05 02:56:11 PM
```

Instance	Monitor	Obs	Last	Max	Avg	t1/obs
SQMR 101:0 edsprs01.edbprs01, 0, DSI	*SQMRBacklogBlock	0	0	0		0
SQMR 102:0 pds01.tpcc, 0, DSI	*SQMRBacklogBlock	0	0	0		0
SQMR 102:1 pds01.tpcc, 0, SQT	*SQMRBacklogBlock	692	50	64		29
SQMR 103:0 rds01.tpcc, 0, DSI	*SQMRBacklogBlock	251	0	2		0

=====

Usage

- admin stats, backlog prints this information:
 - Instance – a specific occurrence of a module.
 - Monitor – the name of the monitor or counter.
 - Obs – the number of segments or blocks observed during the observation period.
 - Last – the number of segments or blocks made during the last observation period.
 - Max – the largest number of segments or blocks observed in any observation made in the observation period.
 - Total – the sum of all the segments or blocks observed during the observation period.
- admin stats, backlog collects data from the SQMRBacklogSeg and SQMRBacklogBlock counters.
- A segment is 1MB and a block is 16K.

Permissions

Any user may execute this command.

admin stats, cancel

Description	Cancels the currently running asynchronous command. For multiple observation intervals, data already saved at the time of cancel is not deleted.
Syntax	admin {stats statistics}, cancel
Usage	You can use admin stats, cancel to explicitly terminate the currently running asynchronous command. Replication Server does not allow other sampling commands when a sampling is already running in the background.
Permissions	Any user may execute admin stats, cancel.

admin stats, {md | mem | mem_in_use}

Description	Reports information about memory usage.
Syntax	admin {stats statistics}, {md mem mem_in_use}
Parameters	<p>md</p> <p>Reports Message Delivery statistics associated with the DIST and RSI users.</p> <p>mem</p> <p>Reports current memory segment use according to segment size.</p> <p>mem_in_use</p> <p>Reports current memory use in bytes.</p>
Examples	<p>Reports total memory use in bytes.</p> <pre>admin stats, mem_in_use Memory_in_Use ----- 14215074</pre>
Usage	Message Delivery statistics are associated with the DIST threads and RSI users.
Permissions	Any user may execute this command.

admin stats, reset

Description	Resets all counters that can be reset.
Syntax	<code>admin {stats statistics}, reset</code>
Examples	Resets all counters to zero. This command does not generate an output. <code>admin stats, reset</code>
Usage	Bit 0x10 of the <code>rs_statcounter.counter_status</code> column indicates whether a counter can be reset or not. When this bit is set for a counter, you cannot use <code>admin stats, reset</code> or any other command to reset the counter.
Permissions	Any user may execute this command.
See also	<code>admin stats</code> , <code>admin stats, status</code>

admin stats, status

Description	Displays configuration settings for monitors and counters.
Syntax	admin {stats statistics}, status
Examples	<pre>1> admin stats, status 2> go</pre> <p>Command in progress, sampling period 00:30:00, time elapsed 00:02:32</p> <pre>Sybase Replication Server Statistics Configuration ===== Configuration Default Current ----- stats_sampling off on stats_show_zero_counters off off stats_reset_rssd on on</pre>
Usage	<ul style="list-style-type: none"> Displays the default and current values of these configuration parameters: <ul style="list-style-type: none"> stats_sampling – indicates whether sampling is on or off. stats_show_zero_counters – specifies whether or not to display counters with zero observation since the last reset.
Permissions	Any user may execute admin stats, status.

admin stats, {tps | cps | bps}

Description	Reports the current throughput in terms of transactions, commands, or bytes per second.
Syntax	admin {stats statistics}, {tps cps bps}
Parameters	<p>tps</p> <p>Specifies that Replication Server reports the current throughput in transactions per second.</p> <p>cps</p> <p>Specifies that Replication Server reports the current throughput in commands per second.</p>

bps

Specifies that Replication Server reports the current throughput in bytes per second.

Examples

Displays counters that calculate throughput in commands per second. Due to the length of the output, only a portion is shown here:

admin stats, cps

Report Time: 10/31/05 02:58:54 PM

Instance

Observer	Obs	Rate x/sec
REP AGENT, pds01.tpcc *CmdsRecv	69876	0

(1 row affected)

Report Time: 10/31/05 02:58:54 PM

Instance

Observer	Obs	Rate x/sec
SQM, 101:0 edsprs01.edbprs01 *CmdsWritten	0	0
SQM, 102:0 pds01.tpcc *CmdsWritten	0	0
SQM, 102:1 pds01.tpcc *CmdsWritten	69886	25
SQM, 103:0 rds01.tpcc *CmdsWritten	48174	17

(4 rows affected)

Report Time: 10/31/05 02:58:54 PM

Instance

Observer	Obs	Rate x/sec
SQMR, 101:0 edsprs01.edbprs01, 0, DSI *CmdsRead	0	0
SQMR, 102:0 pds01.tpcc, 0, DSI *CmdsRead	0	0
SQMR, 102:1 pds01.tpcc, 0, SQT *CmdsRead	50499	18
SQMR, 103:0 rds01.tpcc, 0, DSI *CmdsRead	48144	17

(4 rows affected)

...

Usage

- When calculating throughput per second, Replication Server bases the calculation on the number of processed transactions and the number of elapsed seconds since the counters were last reset using `admin stats, reset`.
- Different modules report throughput for each type of calculation:
 - Transactions per second – are reported by the SQT, DIST, DSI, and other modules.
 - Commands per second – are reported by the RepAgent, RSIUSER, SQM, DIST, DSI, and RSI modules.
 - Bytes per second – are reported by the RepAgent, RSIUSER, SQM, DSI, and RSI modules. The SQM reports transactions in both bytes and blocks per second.

Permissions

Any user may execute this command.

admin time

Description Displays the current time of Replication Server.

Syntax admin time

Parameters None

Examples admin time

```
Time
-----
      Feb 15 2001 9:28PM
```

Usage • admin time is useful for figuring out machine time, or time-zone differences while debugging or examining latency issues.

 • This command is also useful in scripting, to figure out what time Replication Server initiates or completes tasks.

Permissions Any user may execute this command.

admin translate

Description	Performs a datatype translation on a value, displaying the results in delimited literal format.
Syntax	<code>admin translate, <i>value</i>, <i>source_datatype</i>, <i>target_datatype</i></code>
Parameters	<p><i>value</i></p> <p>The literal representation of the value that is to be translated.</p> <p><i>source_datatype</i></p> <p>The name of a datatype (either a Replication Server native datatype or a datatype definition that describes the content and format of <i>value</i>).</p> <p><i>target_datatype</i></p> <p>The name of a datatype (either a Replication Server base datatype or a datatype definition that is the requested output for the translation).</p>
Examples	<p>Example 1 This examples translates the DB2 TIMESTAMP value '1999-06-22-14.35.23.123456' to the Oracle DATE value '22-Jan-99.'</p> <pre>admin translate, '1999-06-22-14.35.23.123456', rs_db2_timestamp, rs_oracle_date</pre> <p>Example 2 This example translates the Adaptive Server binary value 0x1122aabb to the Oracle binary value '1122aabb.'</p> <pre>admin translate, 0x1122aabb, 'binary(4)', 'rs_oracle_binary(4)'</pre>
Usage	<ul style="list-style-type: none"> • Delimit <i>value</i> according to the delimitation requirements of the base datatype of the source datatype. • If <i>source_datatype</i> or <i>target_datatype</i> requires a length specification, for example <code>char(255)</code>, enclose the datatype name in single quotes. • The source and target datatypes may differ depending on whether you want to test class-level or column-level translations. Thus: <ul style="list-style-type: none"> • For class-level translations – use the published datatype for <i>source_datatype</i>. • For column-level datatypes – use the declared datatype for <i>source_datatype</i> and the published datatype for <i>target_datatype</i>. • Use <code>admin translate</code> with the diagnostic version of Replication Server to trace errors in translations.

- For information about supported datatype translations, see the *Replication Server Heterogeneous Replication Guide*. For information about translating datatypes using heterogeneous datatype support (HDS), see the *Replication Server Administration Guide Volume 1*.

Permissions

Any user may execute this command.

See also

alter replication definition, create replication definition, alter connection, create connection

admin verify_repserver_cmd

Description	Verifies that Replication Server can successfully execute a replication definition request.
Syntax	admin verify_repserver_cmd , ' <i>rs_api</i> '
Parameters	<i>rs_api</i> The string containing the Replication Command Language (RCL) command and all corresponding parameters you want to verify. Enclose <i>rs_api</i> in single quotes, and replace each single quote inside the string with two single quotes:
Examples	Example 1 In this example, admin verify_repserver_cmd tests if you can use alter replication definition to drop columns from a replication definition and suspend the target DSI successfully after the data for the old replication definition version is replicated to a target, such as a standby or replicate database:

```
admin verify_repserver_cmd, 'alter replication
definition authors drop address, city, state, zip
with DSI_suspended'
```

If Replication Server can execute the alter replication definition command, Replication Server returns with this message:

```
The replication definition command can be executed
successfully.
```

Example 2 This example shows what happens if you use admin verify_repserver_cmd to see whether you can drop columns from a replication definition that does not exist:

```
admin verify_repserver_cmd, 'alter replication
definition authors_does_not_exist
drop address, city, state, zip'
```

Replication Server returns with a message that the replication definition named “authors_does_not_exist” does not exist.

Example 3 This example shows that admin verify_repserver_cmd can detect syntax errors, such as using the “columns” keyword in the command line:

```
admin verify_repserver_cmd, 'alter replication
definition authors drop columns address, city, state, zip
with DSI_suspended'
```

Replication Server returns with a message, such as:

```
Line 1, character 71: Incorrect syntax with the keyword
```

'columns'.

Example 4 This example shows that admin verify_repserver_cmd can detect if you are using quotes incorrectly, such as using double quotes to enclose 'off':

```
admin verify_repserver_cmd, 'alter replication
definition authors replicate sqldml "off"'
```

Replication Server returns with a message, such as:

Line 1, Incorrect syntax with the keyword 'off'.

The correct syntax is:

```
admin verify_repserver_cmd, 'alter replication
definition authors replicate sqldml `off`'
```

Usage

- When Replication Agent sends a replication definition RCL to Replication Server to execute, and the replication definition RCL fails to execute, Replication Agent shuts down. To avoid this situation, use admin verify_repserver_cmd to verify that Replication Server can successfully execute a replication definition request before you execute the RCL directly from the primary database. Replication Server returns an error if it cannot successfully execute the request.
- Replication Server supports admin verify_repserver_cmd for the same replication definition commands as rs_send_repserver_cmd:
 - alter replication definition
 - create replication definition
 - drop replication definition
 - alter applied function replication definition
 - create applied function replication definition
 - alter request function replication definition
 - create request function replication definition

Permissions

Any user may execute this command.

See also

admin verify_repserver_cmd, alter replication definition,
rs_send_repserver_cmd,
sysadmin skip_bad_repserver_cmd

admin version

Description	Displays the version number of the Replication Server software.
Syntax	admin version
Examples	<pre>admin version Version ----- Replication Server/15.0/P/Sun_svr4/OS 5.8/1/OPT/Wed Jan 4 17:47:58 2006 Copyright 1992, 2006</pre>
Usage	<ul style="list-style-type: none">• The software version number of the Replication Server is the release level of the software product.• The software version number does not, by itself, determine which capabilities you can use in the Replication Server. The system version number for the replication system and the site version number for the Replication Server also determine what features you can use.• The Replication Server's site version number may be equal to or lower than the software version number. See <code>sysadmin site_version</code> for more information.• The system version number for the replication system may be equal to or lower than the software version number. See <code>sysadmin site_version</code> for more information.
Permissions	Any user may execute this command.
See also	<code>sysadmin site_version</code> , <code>sysadmin system_version</code>

admin who

Description	Displays information about threads running in the Replication Server.
Syntax	admin who [, {dist dsi rsi sqm sqt}[, no_trunc ,connection identifier1 [, connection identifier2] ...]]
Parameters	<div><div>dist</div><div>Returns information about Distributor threads. These threads distribute transactions in the inbound queue to replicate databases and Replication Servers.</div><div>dsi</div><div>Returns information about DSI threads. These threads apply replicated transactions to databases.</div><div>rsi</div><div>Returns information about RSI threads. These threads send messages to other Replication Servers.</div><div>sqm</div><div>Returns information about SQM threads. These threads manage Replication Server stable queues.</div><div>sqt</div><div>Returns information about SQT threads. These threads read queues and group functions into transactions.</div><div>no_trunc</div><div>Increases the size of the Info column from 40 characters to 80 characters. This is useful in displaying long data server or database names.</div></div>

Note You cannot use no_trunc if you specify connection identifiers.

connection identifier

Filters the admin who output for a thread module. Depending on the thread module, you can compose a connection identifier with one or more of these

- *db_id* – database identifier, which is a number
- *db_name* – database name
- *ds_name* – data server name
- *q_number* – stable queue number
- *q_type* – stable queue type, where 0 is for an outbound queue and 1 is for an inbound queue
- *rs_id* – Replication Server identifier, which is a number
- *rs_name* – Replication Server name

Table 3-7 lists the connection identifiers you can specify for each type of thread module.

Table 3-7: admin who threads and corresponding connection identifiers

Thread	Connection identifier	Example
DIST	<p>Display distributor (DIST) thread information for one or more specific connections by providing any of:</p> <ul style="list-style-type: none"> • <i>db_id</i> • <i>ds_name</i> • <i>ds_name</i> and <i>db_name</i> <p>If you specify only the data server name as the connection identifier, admin who lists the distributor information for all the databases that belong to the data server.</p>	<p>Display DIST information on dataserver “ASE_01” and database “DB01”:</p> <pre>admin who, dist, ASE_01, DB01</pre>
DSI	<p>Display Data Server Interface (DSI) thread information for one or more specific connections by providing any of:</p> <ul style="list-style-type: none"> • <i>db_id</i> • <i>ds_name</i> • <i>ds_name</i> and <i>db_name</i> <p>If you specify only the data server name as the connection identifier, admin who lists the DSI connection information for all the databases that belong to the data server.</p>	<p>Display DSI information for <i>db_id</i> = 101:</p> <pre>admin who, dsi, 101</pre>

Thread	Connection identifier	Example
RSI	Display Replication Server Interface (RSI) thread information for a specific connection by providing <i>one</i> of: <ul style="list-style-type: none"><i>rs_id</i><i>rs_name</i> You can either specify the connection by replication server name or replication server identifier.	Display the RSI information for <i>rs_id</i> = 16777318: admin who, rsi, 16777318
SQM	Display Stable Queue Manager (SQM) thread information for one or more specific connections by providing any of: <ul style="list-style-type: none"><i>db_name</i> and <i>db_name</i> with or without <i>q_type</i><i>ds_name</i> with or without <i>q_type</i><i>q_number</i> with or without <i>q_type</i><i>rs_id</i> with or without <i>q_type</i><i>rs_name</i> with or without <i>q_type</i> If you do not specify <i>q_type</i> , the command lists both the inbound and outbound queue types of all the admin who, sqm entries of the specified data server.	Display SQM information on data server “ASE_01” for <i>q_type</i> =1, which are inbound queues: admin who, sqm, ASE_01, 1
SQT	Display Stable Queue Transaction (SQT) thread information for one or more specific connections by providing any of: <ul style="list-style-type: none"><i>db_name</i> and <i>db_name</i> with or without <i>q_type</i><i>ds_name</i> with or without <i>q_type</i><i>q_number</i> with or without <i>q_type</i><i>rs_id</i> with or without <i>q_type</i><i>rs_name</i> with or without <i>q_type</i> If you do not specify <i>q_type</i> , the command lists both the inbound and outbound queue types of all the admin who, sqt entries of the specified data server.	Display SQT information on dataserver “ASE_01”, database “DB01” for inbound queues: admin who, sqt, ASE01, DB01, 1

Examples

Example 1 In the following example, admin who displays the state of all threads in the Replication Server. DSI scheduler threads are shown as “DSI” in the output. DSI executor threads are shown as “DSI EXEC.” If the DSI is suspended when Replication Server starts up, the output shows only one DSI executor thread, even if more are configured.

```
admin who
```

Spid	Name	State	Info
97	DIST	Active	103 LDS.pubs2
98	SQT	Awaiting Wakeup	103:1 DIST LDS.pubs2
68	SQM	Awaiting Message	103:0 LDS.pubs2
89	DSI EXEC	Awaiting Message	106(1) SYDNEY_DS.pubs2sb

```

91 DSI          Awaiting Command      106 SYDNEY_DS.pubs2sb
21 DSI EXEC     Awaiting Message      101(1) TOKYO_DS.TOKYO_RSSD
10 DSI          Awaiting Command      101 TOKYO_DS.TOKYO_RSSD
16 DIST         Active                 101 TOKYO_DS.TOKYO_RSSD
17 SQT          Awaiting Wakeup       101:1 DIST TOKYO_DS.TOKYO_RSSD
15 SQM          Awaiting Message      101:1 TOKYO_DS.TOKYO_RSSD
14 SQM          Awaiting Message      101:0 TOKYO_DS.TOKYO_RSSD
30 REP AGENT    Awaiting Command      TOKYO_DS.TOKYO_RSS
   USER
 4 DSI EXEC     Awaiting Message      104(1) TOKYO_DS.pubs2
 0 DSI          Awaiting Command      104 TOKYO_DS.pubs2
 8 REP AGENT    Awaiting Command      TOKYO_DS.pubs2
   USER
53 RSI          Awaiting Wakeup       SYDNEY_RS
52 SQM          Awaiting Message      16777318:0 SYDNEY_RS
   RSI USER     Inactive              TOKYO_RS
11 dSUB         Active
 6 dCM          Awaiting Message
 9 dAIO         Awaiting Message
12 dREC         Active                dREC
71 USER         Active                sa
47 GATEWAY      Awaiting Command      SYDNEY_RS
 5 dALARM       Awaiting Wakeup
13 dSYSAM       Sleeping

```

Example 2 In the following example, the admin who, dist command displays information about each DIST thread in the Replication Server.

```
admin who, dist
```

Spid	State	Info
21	Active	102 SYDNEY_DS.SYDNEY_RSSD
22	Active	106 SYDNEY_DS.pubs2

PrimarySite	Type	Status	PendingCmds	SqtBlocked
102	P	Normal	0	1
106	P	Normal	0	1

Duplicates	TransProcessed	CmdsProcessed	MaintUserCmds
0	715	1430	0
290	1	293	0

NoRepdefCmds	CmdsIgnored	CmdMarkers	RSTicket	SqtMaxCache
0	0	0	0	0

0

0

1

0

0

Example 3 In this example, admin who, dsi displays information about each DSI scheduler thread running in the Replication Server.

admin who, dsi			
Spid	State	Info	
8	Awaiting Message	101	TOKYO_DS.TOKYO_RSSD
79	Awaiting Message	104	TOKYO_DS.pubs2
145	Awaiting Message	105	SYDNEY_DS.pubs2sb

Maintenance User	Xact_retry_times	Batch	Cmd_batch_size
TOKYO_RSSD_maint	3	on	8192
pubs2_maint	3	on	8192
pubs2_maint	3	on	8192

Xact_group_size	Dump_load	Max_cmds_to_log
65536	off	-1
65536	off	-1
65536	off	-1

Xacts_read	Xacts_ignored	Xacts_skipped
39	0	0
0	0	0
1294	2	0

Xacts_succeeded	Xacts_failed	Xacts_retried	Current Origin DB
0	28	0	102
0	0	0	0
0	93	0	104

Current Origin QID	Subscription Name	Sub Command
0x000000000...	NULL	NULL
0x000000000...	NULL	NULL
0x000000000...	NULL	NULL

Current Secondary QID	Cmds_read	Cmds_parsed_by_sqt
NULL	129	0

	NULL	0	0
	NULL	6740	0

IgnoringStatus	Xacts_Sec_Ignored	GroupingStatus	TriggerStatus
-----	-----	-----	-----
Applying	0	on	on
Applying	0	on	on
Applying	0	off	off

ReplStatus	NumThreads	NumLargeThreads	LargeThreshold
-----	-----	-----	-----
on	1	0	100
on	1	0	100
off	3	1	20

CacheSize	Serialization	Max_Xacts_in_group	Xacts_retried_blk
-----	-----	-----	-----
0	wait_for_commit	20	0
0	wait_for_commit	200	0
0	wait_for_start	20	0

CommitControl	CommitMaxChecks	CommitLogChecks
-----	-----	-----
on	400	200
on	400	200
on	400	200

CommitCheckIntvl	IsolationLevel	dsi_rs_ticket_report	RSTicket
-----	-----	-----	-----
1000	default	on	0
1000	default	on	0
1000	default	on	0

Example 4 In this example, admin who, rsi displays information about RSI threads.

```

admin who, rsi

Spid      State          Info
-----
  53      Awaiting Wakeup  SYDNEY_RS

Packets Sent      Bytes Sent      Blocking Reads
-----
  3008.000000      624678.000000      269

Locater Sent      Locater Deleted

```

0x000000...

0x000000...

Example 5 In this example, admin who, sqm displays information about SQM threads.

admin who, sqm				
Spid	State		Info	
-----	-----		-----	
14	Awaiting Message		101:0 TOKYO_DS.TOKO_RSSD	
15	Awaiting Message		101:1 TOKYO_DS.TOKYO_RSSD	
52	Awaiting Message		16777318:0 SYDNEY_RS	
68	Awaiting Message		103:0 LDS.pubs2	
Duplicates	Writes	Reads	Bytes	
-----	-----	-----	-----	
0		0	0	
0		8867	9058	
0	0.1	2037	2037	
0	0.1.0	0	0	
B Writes	B Filled	B Reads	B Cache	Save_Int:Seg
-----	-----	-----	-----	-----
0	0		0	0:0
0	34	44	2132	0:33
0	3	54	268	0:4
0	0	23	0	strict:0
First Seg.Block		Last Seg.Block		Next Read
-----		-----		-----
0.1		0.0		0.1.0
33.10		33.10		33.11.0
4.12		4.12		4.13.0
0.1		0.0		0.1.0
Readers	Truncs			
-----	-----			
1	1			
1	1			
1	1			
1	1			

Example 6 In this example, admin who, sqt displays information about SQT threads.

```

admin who, sqt

Spid      State      Info
-----
17      Awaiting Wakeup  101:1 TOKYO_DS.TOKYO_RSSD
98      Awaiting Wakeup  103:1 DIST LDS.pubs2
10      Awaiting Wakeup  101 TOKYO_DS.TOKYO_RSSD
0       Awaiting Wakeup  106 SYDNEY_DSpubs2sb

Closed    Read      Open      Trunc
-----
0         0         0         0
0         0         0         0
0         0         0         0
0         0         0         0

Removed    Full      SQM Blocked  First Trans  Parsed
-----
0         0         1           0           0
0         0         1           0           0
0         0         0           0           0
0         0         0           0           0

SQM Reader  Change Oqids  Detect Orphans
-----
0           0           0
0           0           0
0           0           1
0           0           1

```

Usage

- If you use `admin who` with an option, you must include a comma before the option.
- If you specify connection identifiers and Replication Server cannot find information that fulfills the criteria, the output does not display any record.
- To display information about all threads in the Replication Server, execute `admin who` with no options.

Output column descriptions for *admin who*

The `spid`, `Name`, `State`, and `Info` columns display when `admin who` is executed without options. The `spid`, `State`, and `Info` columns also display when any option is chosen.

***spid* column**

This is a unique identifier for a thread running in the Replication Server. If a thread is suspended or down, this field is blank.

Name and Info columns

Name is the type of Replication Server thread. The contents of Info varies, depending upon the type of thread. Table 3-8 describes the Name and Info columns for each thread.

Table 3-8: Name and Info column for admin who output

Name	Description	Contents of info
dAlarm	Alarm daemon. This thread keeps track of alarms set by other threads, such as the fade-out time for connections and the subscription daemon retry interval.	Empty
dAIO	The asynchronous I/O daemon. It manages asynchronous I/O to stable queues for the Replication Server.	Empty
dCM	The daemon for the connection manager. It manages connections to data servers and other Replication Servers.	Empty
dREC	The recovery daemon. This thread sleeps for a configurable period of time (rec_daemon_sleep_time configuration parameter) and then initiates any recovery actions specified in the rs_recovery table.	Empty
dSUB	The subscription retry daemon. This thread wakes up after a configurable time-out period (sub_daemon_sleep_time configuration parameter) and attempts to restart any subscriptions that have failed.	Empty
dSYSAM	SySAM daemon. This thread keeps track of checked out licenses.	Empty
dVERSION	The version daemon. This thread activates briefly when the Replication Server is started for the first time after an upgrade. It communicates the Replication Server's new software version number to the ID Server.	The version of this Replication Server.
DIST	Distributor thread. Each primary database has a Distributor thread that reads transactions from the inbound queue, determines which subscriptions are interested, and forwards the transactions.	The names of the data server and database whose updates the thread is distributing.
DSI	DSI scheduler thread. This thread reads a stable queue via SQT and applies the transactions through the DSI Executor threads.	The name of the data server the thread writes to.
DSI EXEC	DSI executor thread. This thread executes the transactions on the replicate database and acts on errors that the replicate data server returns.	The ID of the DSI executor thread and the name of the data server it is connected to.
GATEWAY	The gateway server thread. This thread passes commands from client to the server and returns the server's reply to the client.	The name of the Replication Server that acts as the gateway server.

Name	Description	Contents of info
REP AGENT USER	A client connection that is a RepAgent thread. This thread verifies that RepAgent submissions are valid and writes them into the inbound queue.	The name of the primary data server and database whose log the RepAgent is forwarding.
RSI	RSI sender. This thread sends messages from one Replication Server to another.	The name of the Replication Server where messages are sent.
RSI User	Client connection thread for a Replication Server connected to this one. It writes messages destined for other Replication Servers or databases into outbound queues.	The name of the Replication Server connected to this one as a client.
RS User	Replication Server connection used to create or drop subscriptions at the primary Replication Server.	The name of the subscription owner.
SQM	Stable queue manager. This thread manages a Replication Server stable queue.	<p><i>Queue number:</i> An ID for a Replication Server or database.</p> <p><i>Queue type:</i> 1 for the inbound queue, 0 for outbound queues.</p> <p>Any other number is the ID of a subscription the queue is for.</p> <p><i>Queue identifier:</i> for these queues:</p> <ul style="list-style-type: none"> • For queues used to spool messages to another Replication Server, it is the name of the other Replication Server. • For queues used to spool messages to databases, it is the name of the data server and database. • For queues used to spool messages related to a subscription being created or dropped, it is the name of the replication definition and the name of the subscription.
SQT	Stable queue transaction interface. This thread reads a stream of messages from a stable queue and reassembles the transactions in commit order. The Distributor and DSI use this thread.	Same as the corresponding SQM thread.
USER	Thread for a client executing RCL commands.	The login name of the client.

State column The State column contains the thread execution status. Table 3-9 describes the valid states for Replication Server threads. The states for DSI threads are defined differently, depending on whether they are scheduler threads or executor threads. For the definitions, see the *Replication Server Troubleshooting Guide*.

Table 3-9: State column descriptions for admin who output

State	Description
Active	Actively processing a command.
Active, DSI timer	Actively processing a command. dsi_timer is on.
Awaiting Batch Order	A DSI thread is waiting to submit a command batch to the replicate data server.
Awaiting Command	The thread is waiting for a client to send a command.
Awaiting Command, DSI timer	The thread is waiting for a client to send a command. dsi_timer is on.
Awaiting Commit Order	Thread is waiting for its turn to commit a completed transaction.
Awaiting I/O	The thread is waiting for an I/O operation to finish.
Awaiting Message	The thread is waiting for a message from an Open Server™ message queue.
Awaiting Message, DSI timer	The thread is waiting for a message from an Open Server™ message queue. dsi_timer is on.
Awaiting Wakeup	The thread has posted a sleep and is waiting to be awakened.
Checking Condition	The thread is waiting for an event to occur.
Connecting	The thread is connecting.
Disconnecting	The thread is disconnecting.
Down	The thread has not started or has terminated.
Getting Lock	The thread is waiting on a mutual exclusion lock.
Inactive	The status of an RSI User thread at the destination of a route when the source Replication Server is not connected to the destination Replication Server.
Initializing	The thread is being initialized.
Invalid	The thread is in an unknown status.
Locking Resource	The thread is attempting to lock a shared resource.
Not Running	Thread is cleaning up in preparation for shutdown.
Reading Disk	The thread is preparing for a disk read.
SkipUntil Dump	Thread has received a resync database marker, and this state remains until the DSI has processed a subsequent dump database marker.
Setting Condition	The thread is setting the condition for another thread to wake up.
SkipUntil Resync	Thread is resuming after you execute skip to resync, and this state remains until the thread receives a resync database marker.
Sleeping	Thread is yielding processor time for a finite period.
Suspended	The thread has been suspended by the user.
Unlocking Resource	The thread is releasing a shared resource.

Output column descriptions for admin who, dist

This command returns a table with a row for each DIST thread in the Replication Server. With the Spid, State, and Info columns, the table contains the columns shown in Table 3-10.

Table 3-10: Column descriptions for *admin who, dist* output

Column	Description
PrimarySite	The ID of the primary database for the SQT thread.
Type	The thread is a physical or logical connection.
Status	The thread has a status of “normal” or “ignoring.”
PendingCmds	The number of commands that are pending for the thread.
SqtBlocked	Whether or not the thread is waiting for the SQT.
Duplicates	The number of duplicate commands the thread has seen and dropped.
TransProcessed	The number of transactions that have been processed by the thread.
CmdsProcessed	The number of commands that have been processed by the thread.
MaintUserCmds	The number of commands belonging to the maintenance user.
NoRepdefCmds	<p>The number of commands dropped because no corresponding table replication definitions were defined.</p> <p>In the case of Warm Standby, it is possible to have Rep Server create the replication definition. In multi-site availability (MSA), one defines database replication definitions. In either of these cases, if the replicated data originates from a source without a table replication definition, the counter is increased and replicated data proceeds to the target.</p>
CmdsIgnored	The number of commands dropped before the status became “normal.”
CmdMarkers	The number of special markers that have been processed.
RSTicket	<p>The number of <code>rs_ticket</code> subcommands that have been processed by a DIST thread, if the Replication Server <code>stats_sampling</code> parameter is on.</p> <p>Minimum: 0</p> <p>Maximum: $2^{63}-1$</p> <p>Default: 0</p>
SqtMaxCache	<p>Maximum SQT (Stable Queue Transaction interface) cache memory for the database connection, in bytes.</p> <p>The default, 0, means that the current setting of <code>sqt_max_cache_size</code> is used as the maximum cache size for the connection.</p>

Output column descriptions for *admin who, dsi*

This command returns a table with a row for each running DSI scheduler thread in the Replication Server. If a DSI scheduler thread exists for a database but does not appear in the output of *admin who, dsi*, use *resume connection* to restart the data server interface for the database. Along with the *Spid*, *State*, and *Info* columns, the table contains the columns shown in Table 3-11.

Table 3-11: Column descriptions for *admin who, dsi* output

Column	Description
Maintenance User	The login name of the maintenance user applying the transactions.

Column	Description
Xact_retry_times	The number of times a failed transaction is retried if the error action is RETRY_LOG or RETRY_STOP.
Batch	Indicates if the batch option is on. If it is on, you can submit multiple commands as a batch to the data server.
Cmd_batch_size	The maximum size, in bytes, of a batch of output commands that you can send to the data server.
Xact_group_size	The maximum size, in bytes, of a transaction group consisting of source commands.
Dump_load	Indicates if the dump/load option is on. This configuration option coordinates dumps between primary and replicate databases.
Max_cmds_to_log	Maximum number of commands that can be logged into the exceptions log for a transaction. A value of -1 indicates an unlimited number of commands.
Xacts_read	The number of transactions read by the DSI from the outbound stable queue. This number should increase as the DSI applies transactions. You can use the information to monitor the rate of activity.
Xacts_ignored	The number of transactions determined to be duplicates. Typically, some transactions are ignored at start-up time because they were applied previously. Deletes from the DSI queue are delayed, so at start-up time, duplicates are detected and ignored. If you see a large number of ignored transactions, there is a chance that the rs_lastcommit table is corrupted. For more information, refer to the <i>Replication Server Troubleshooting Guide</i> .
Xacts_skipped	The number of transactions skipped by resuming the connection with skip first transaction.
Xacts_succeeded	The number of transactions applied successfully against the database.
Xacts_failed	The number of transactions that failed. Depending on the error mapping, some transactions may be written into the exceptions log. You should inspect the exceptions log.
Xacts_retried	The number of transactions that were retried.
Current Origin DB	The origin database ID for the current transaction.
Current Origin QID	If the state is Active, it is the Origin Queue ID of the begin log record of the transaction being processed. Otherwise, it is the Origin Queue ID of the begin log record of the last transaction processed.
Subscription Name	If the thread is processing a subscription, this is the name of the subscription.
Sub Command	If the thread is processing a subscription, this is the subscription command: activate, validate, drop, or unknown.
Current Secondary QID	If the thread is processing an atomic subscription applied incrementally, this column holds the queue ID of the current transaction.
Cmds_read	The number of commands read from the DSI queue.
Cmds_parsed_by_sqt	The number of commands parsed by SQT before being read by the DSI queue.
IgnoringStatus	Contains “Ignoring” if the DSI is ignoring transactions while waiting for a marker. Contains “Applying” if the DSI is executing transactions in the database.
Xacts_Sec_ignored	In a warm standby application, the number of transactions that were ignored after the switchover.

Column	Description
GroupingStatus	Contains “on” if the DSI is executing transactions in groups. Contains “off” if the DSI is executing transactions one at a time.
TriggerStatus	Contains “on” if set triggers is on. Contains “off” if set triggers is off.
ReplStatus	Indicates whether the Replication Server replicates transactions in the database. The default is “off” for standby databases. The default is “on” for all other databases.
NumThreads	The number of parallel DSI threads in use.
NumLargeThreads	The number of parallel DSI threads reserved for use with large transactions.
LargeThreshold	In a parallel DSI configuration, the number of commands allowed in a transaction before it is considered large.
CacheSize	The maximum SQT cache memory for the database connection, in bytes. The default, 0, means that the current setting of the <code>sqt_max_cache_size</code> parameter is used as the maximum SQT cache memory.
Serialization	The method used to maintain serial consistency when parallel DSI threads are used.
Max_Xacts_in_group	The maximum number of transactions in a group. The default is 20. You can configure this number using the <code>alter connection</code> command.
Xacts_retried_blk	The number of times the DSI rolled back a transaction due to exceeding maximum number of checks for lock contention.
CommitControl	Indicates if commit control is internal or external. Set to true if internal.
CommitMaxChecks	Indicates the maximum number of lock contention attempts before rolling back transaction and retrying.
CommitLogChecks	Indicates the maximum number of lock contention attempts before logging a message.
CommitCheckIntvl	Amount of time, in milliseconds, a transaction waits before issuing a check for lock contention.
IsolationLevel	Database isolation level for DSI connection.
RSTicket	The number of <code>rs_ticket</code> subcommands that have been processed by a DSI queue manager, if the Replication Server <code>stats_sampling</code> parameter is “on”. The default, 0, means that the current setting of <code>sqt_max_cache_size</code> is used as the maximum cache size for the connection.
dsi_rs_ticket_report	Determines whether to call function string <code>rs_ticket_report</code> . <code>rs_ticket_report</code> function string is invoked when <code>dsi_rs_ticket_report</code> is set to on. Default: off

Output column descriptions for *admin who, rsi*

This command displays information about RSI threads that send messages to other Replication Servers. Along with the `Spid`, `State`, and `Info` columns, `admin who, rsi` contains the columns shown in Table 3-12.

Table 3-12: Column descriptions for *admin who, rsi* output

Column	Description
Packets Sent	The number of network packets sent.

Column	Description
Bytes Sent	The total number of bytes sent.
Blocking Reads	The number of times the stable queue was read with a blocking read.
Locator Sent	The locator of the last message sent (contains the queue segment, block and row).
Locator Deleted	The last locator that the recipient acknowledged and that has been deleted by Replication Server.

Output column descriptions for *admin who, sqm*

This command displays information about SQM threads that manage Replication Server stable queues. Along with the Spid, State, and Info columns, *admin who, sqm* contains the columns shown in Table 3-13.

Table 3-13: Column descriptions for *admin who, sqm* output

Column	Description
Duplicates	The number of duplicate messages detected and ignored. There are usually some duplicate messages at start-up.
Writes	The number of messages written into the queue.
Read	The number of messages read from the queue. This usually exceeds the number of writes because the last segment is read at start-up to determine where writing is to begin. Also, long transactions may cause messages to be reread.
Bytes	The number of bytes written.
B Writes	The number of 16K blocks written. It may be greater than <i>Bytes/16K</i> because not every 16K block written is full. You can determine the density of blocks by dividing <i>Bytes</i> by <i>B Writes</i> .
B Filled	The number of 16K blocks written to disk because they are filled.
B Reads	The number of 16K blocks read.
B Cache	The number of 16K blocks read that are in cache.
Save_Int:Seg	The Save_Int interval and the oldest segment in the Save_Int list. The Save_Int interval is the number of minutes the Replication Server maintains an SQM segment after all messages in the segment have been acknowledged by targets.

For example, a value of 5:88 indicates a Save_Int interval of 5 minutes, where segment 88 is the oldest segment in the Save_Int list.

This feature provides redundancy in the event of replication system failure. For example, a Replication Server could lose its disk partitions while receiving data from another Replication Server. The Save_Int feature lets the sending Replication Server re-create all messages saved during the Save_Int interval.

A *Save_Int* value of “strict” may be used when a queue is read by more than one reader thread. Replication Server maintains the SQM segment until all threads reading the queue have read the messages on the segment and applied them to their destination.

Column	Description
First Seg.Block	The first undeleted segment and block number in the queue. If the figures for First Seg.Block and Last Seg.Block do not match, data remains in the queue for processing. This information is useful when dumping queues. For more information, refer to the <i>Replication Server Troubleshooting Guide</i> .
Last Seg.Block	The last segment and block written to the queue. If the figures for First Seg.Block and Last Seg.Block do not match, data remains in the queue for processing. This information is useful when dumping queues. For more information, refer to the <i>Replication Server Troubleshooting Guide</i> .
Next Read	The next segment, block, and row to be read from the queue.
Readers	The number of threads that are reading the queue.
Truncs	The number of truncation points for the queue.

Output column descriptions for *admin who, sqt*

SQT threads read transactions from a stable queue and pass them to the SQT reader in commit order. The reader can be a DIST or a DSI thread.

SQT stores the transactions it is processing in a memory cache. The Closed, Read, Open, Trunc, and Removed columns shown in Table 3-14 apply to transactions in the SQT cache.

Table 3-14: Column descriptions for *admin who, sqt* output

Column	Description
Closed	The number of committed transactions in the SQT cache. The transactions have been read from the stable queue and await processing.
Read	The number of transactions processed, but not yet deleted from the queue.
Open	The number of uncommitted or unaborted transactions in the SQT cache.
Trunc	The number of transactions in the transaction cache. Trunc is the sum of the Closed, Read, and Open columns.
Removed	The number of transactions whose constituent messages have been removed from memory. This happens when the SQT processes large transactions. The messages are reread from the stable queue.
Full	Indicates that the SQT has exhausted the memory in its cache. This is not a problem as long as there are closed or read transactions still awaiting processing. If the SQT cache is often full, consider raising its configured size. To do this, see <i>alter connection</i> .
SQM Blocked	1 if the SQT is waiting on SQM to read a message. This state should be transitory unless there are no closed transactions.

Column	Description
First Trans	<p>This column contains information about the first transaction in the queue and can be used to determine if it is an unterminated transaction. The column has three pieces of information:</p> <ul style="list-style-type: none">• ST: Followed by O (open), C (closed), R (read), or D (deleted)• Cmds: Followed by the number of commands in the first transaction• qid: Followed by the segment, block, and row of the first transaction
Parsed	The number of transactions that have been parsed.
SQM Reader	The index of the SQM reader handle.
Change Oqids	Indicates that the origin queue ID has changed.
Detect Orphans	Indicates that it is doing orphan detection.

Permissions Any user may execute this command.

admin who_is_down

Description	Displays information about Replication Server threads that are not running.												
Syntax	admin who_is_down [, no_trunc]												
Parameters	<p>no_trunc</p> <p>Increases the size of the Info column from 40 characters to 80 characters. This is useful in displaying long data server or database names.</p>												
Examples	<pre>admin who_is_down</pre> <table><tr><th>Spid</th><th>Name</th><th>State</th><th>Info</th></tr><tr><td>----</td><td>-----</td><td>-----</td><td>-----</td></tr><tr><td></td><td>RSI</td><td>Suspended</td><td>SYDNEY_RS</td></tr></table>	Spid	Name	State	Info	----	-----	-----	-----		RSI	Suspended	SYDNEY_RS
Spid	Name	State	Info										
----	-----	-----	-----										
	RSI	Suspended	SYDNEY_RS										
Usage	<ul style="list-style-type: none">• The Spid column in the output of admin who_is_down is always empty. There are no processes for threads that are not running.• Execute admin who_is_down when admin health shows that the Replication Server is suspect. The output for this command does not list threads that are in a state of “Connecting,” which could be the cause of the suspect health.• For a description of the output from this command, see admin who.												
Permissions	Any user may execute this command.												
See also	admin health, admin who, admin who_is_up												

admin who_is_up

Description Displays information about Replication Server threads that are running.

Syntax admin who_is_up [, no_trunc]

Parameters no_trunc
 Increases the size of the Info column from 40 characters to 80 characters.
 This is useful in displaying long data server or database names.

Examples admin who_is_up

Spid	Name	State	Info
----	-----	-----	-----
97	DIST	Active	103 LDS.pubs2
98	SQT	Awaiting Wakeup	103:1 DIST LDS.pubs2
96	SQM	Awaiting Message	103:1 LDS.pubs2
68	SQM	Awaiting Message	103:0 LDS.pubs2
89	DSI EXEC	Awaiting Message	106(1) SYDNEY_DS.pubs2sb
91	DSI	Awaiting Command	106 SYDNEY_DS.pubs2sb
21	DSI EXEC	Awaiting Message	101(1) TOKYO_DS.TOKYO_RSSD
10	DSI	Awaiting Command	101 TOKYO_DS.TOKYO_RSSD
16	DIST	Active	101 TOKYO_DS.TOKYO_RSSD
17	SQT	Active	101:1 DIST TOKYO_DS.TOKYO
15	SQM	Awaiting Message	101:1 TOKYO_DS.TOKYO_RSSD
14	SQM	Awaiting Message	103:1 TOKYO_DS.TOKYO_RSSD
30	REP AGENT USER	Awaiting Command	TOKYO_DS.TOKYO_RSSD
4	DSI EXEC	Awaiting Message	104(1) TOKYO_DS.pubs2
9	dAIO	Awaiting Message	
12	dREC	Active	dREC
61	USER	Active	sa
5	dALARM	Awaiting Wakeup	

Usage For a description of the output, see admin who.

Permissions Any user may execute this command.

See also admin who, admin who_is_down

allow connections

Description	Places Replication Server in recovery mode for specified databases.
Syntax	<code>allow connections</code>
Usage	<ul style="list-style-type: none">• Execute <code>allow connections</code> to begin replaying log records from reloaded dumps.• Start Replication Server in stand-alone mode and execute <code>set log recovery</code> for each database whose log you are replaying.• After executing <code>allow connections</code>, Replication Server accepts connect requests only from RepAgents started in recovery mode for the specified databases. This ensures that Replication Server receives the replayed log records before current transactions.• If you restart Replication Server in stand-alone mode and execute <code>allow connections</code> without first executing <code>set log recovery</code> commands, Replication Server moves from stand-alone mode to normal mode.• For detailed recovery procedures, see the <i>Replication Server Administration Guide Volume 2</i>.
Permissions	<code>allow connections</code> requires “sa” permission.
See also	<code>ignore loss</code> , <code>rebuild queues</code> , <code>set log recovery</code>

alter applied function replication definition

Description	Changes the function replication definition created by the <code>create applied function replication definition</code> command.
Syntax	<pre>alter applied function replication definition <i>repdef_name</i> {with replicate function named '<i>func_name</i>' add @<i>param_name</i> <i>datatype</i>[, @<i>param_name</i> <i>datatype</i>]... add searchable parameters @<i>param_name</i>[, @<i>param_name</i>]... send standby {all replication definition} parameters} [with DSI_suspended]</pre>
Parameters	<p><i>repdef_name</i> The name of the applied function replication definition to change.</p> <p>with replicate function named '<i>func_name</i>' Specifies the name of the stored procedure to execute at the replicate database. <i>func_name</i> is a character string with a maximum length of 255 characters.</p>

add

Adds parameters and their datatypes to the applied function replication definition.

@*param_name*

The name of a parameter you are adding to the list of replicated or searchable parameters. Each parameter name must begin with the @ character.

datatype

The datatype of the parameter you are adding to the parameter list. See “Datatypes” on page 21 for a list of the datatypes and their syntax. Adaptive Server stored procedures and function replication definitions cannot contain parameters with the text, unitext, rawobject, and image datatypes.

add searchable parameters

Specifies additional parameters that you can use in the where clause of the create subscription or define subscription command.

send standby

In a warm standby application, specifies whether to send all parameters in the function (send standby all parameters) or only those specified in the replication definition (send standby replication definition parameters), to a standby database. The default is send standby all parameters.

with DSI_suspended

Allows you to suspend the standby DSI, if there is one, and each of the subscribing replicate DSI threads. Replication Server suspends the DSI thread in the standby or replicate database after Replication Server applies all the data for the old replication definition version to the standby or replicate database.

After Replication Server suspends a DSI thread, you can make changes to the target stored procedures, and to any customized function strings. When you resume the DSI thread, Replication Server replicates the primary updates using the altered replication definition.

You do not need to use with DSI_suspended if:

- There is no subscription to the replication definition.
- You do not need to change customized function strings.
- You do not need to change the replicate or standby stored procedure.

Note If there is a subscription from a replicate Replication Server with a site version earlier than 1550, the replicate DSI threads for that Replication Server are not suspended.

Examples

Example 1 Adds the @notes, @pubdate, and @contract parameters to the titles_frep function replication definition:

```
alter applied function replication definition titles_frep
add @notes varchar(200), @pubdate datetime, @contract bit
```

Example 2 Adds the @type and @pubdate parameters to the list of searchable parameters in the titles_frep function replication definition:

```
alter applied function replication definition titles_frep
add searchable parameters @type, @pubdate
```

Example 3 Changes the titles_frep function replication definition to be replicated as the newtitles stored procedure at the replicate database, and instructs Replication Server to suspend the target DSI after primary data that exists before you execute alter applied replication definition is replicated to the replicate database:

```
alter applied function replication definition titles_frep
with replicate function named 'newtitles'
with DSI suspended
```

Usage

- Use alter applied function replication definition to change an existing applied function replication definition. You can add replicated parameters and searchable parameters, select which parameters to send to the warm standby, and specify a different name for the stored procedure to execute in the replicate database.
- alter applied function replication definition can alter only the replication definition created with the create applied function replication definition command.
- When you alter a function replication definition, the name, parameters, and datatypes that you specify for the function replication definition must match the stored procedure that you are replicating. Only the parameters specified in the function replication definitions are replicated.
- Multiple function replication definitions for the same stored procedure must have the same parameter list. If you add a new parameter, the new parameter is automatically added to all the function replication definitions created for that stored procedure.
- You must execute alter applied function replication definition at the primary Replication Server.
- A parameter name cannot appear more than once in any clause.
- When adding parameters, you *must* instruct Replication Server to coordinate alter applied function replication definition with distributions for the function replication definition. In addition, you must instruct Replication Server to coordinate changes to stored procedures and replication definitions.

See “Replication definition change request process,” in Chapter 9, “Managing Replicated Tables” in the *Replication Server Administration Guide Volume 1* to alter replication definitions.

- Use the with replicate function named clause to specify the stored procedure name you want to execute at the replicate database. See create applied function replication definition.

See the *Replication Server Administration Guide Volume 1* for more information about alter applied function replication definition.

Permissions

alter applied function replication definition requires “create object” permission.

See also

alter function string, alter replication definition, alter function replication definition, alter request function replication definition, create applied function replication definition, create request function replication definition, rs_send_repserver_cmd, rs_helprepversion

alter connection

Description	Changes the attributes of a database connection.
Syntax	<pre>alter connection to <i>data_server.database</i> { [for replicate table named [<i>table_owner</i>.]<i>table_name</i> [set <i>table_param</i> [to] '<i>value</i>']] set function string class [to] <i>function_class</i> set error class [to] <i>error_class</i> set replication server error class [to] <i>rs_error_class</i> set password [to] <i>passwd</i> set log transfer [to] {on off} set <i>database_param</i> [to] '<i>value</i>' set <i>security_param</i> [to] '<i>value</i>' set security_services [to] 'default' set dataserver and database name [to] <i>new_ds.new_db</i> set trace [to] '<i>trace_value</i>'}</pre>
Parameters	<p><i>data_server</i></p> <p>The data server that holds the database whose connection is to be altered.</p> <p><i>database</i></p> <p>The database whose connection is to be altered.</p> <p>for replicate table named</p> <p>Specifies the name of the table at the replicate database. <i>table_name</i> is a character string of up to 200 characters. <i>table_owner</i> is an optional qualifier for the table name, representing the table owner. Data server operations may fail if actual table owners do not correspond to what you specify in the replication definition.</p> <p><i>table_param</i></p> <p>The table-level parameter that affects a table you specify with for replicate table name.</p> <p>Valid values: <i>dsi_compile_enable</i> and <i>dsi_command_convert</i>. See Table 3-15 for descriptions.</p> <p><i>function_class</i></p> <p>The function-string class to use with the data server. See Table 3-25 on page 267 for a list of function classes that Replication Server provides for database connections.</p> <p><i>error_class</i></p> <p>The error class that handles database errors. See Table 3-24 on page 238 for a list of error classes that Replication Server provides for database connections.</p>

- rs_error_class*
The error class that handles Replication Server errors for a database. See Table 3-24 on page 238 for a list of Replication Server error classes.
- passwd*
The new password to use with the login name for the database connection. You must specify a password if network-based security is not enabled.
- log transfer on
Allows the connection to send transactions from a RepAgent to the Replication Server.
- log transfer off
Stops the connection from sending transactions from a primary database RepAgent.
- database_param*
The parameter that affects database connections from the Replication Server.
- value*
A character string containing a new value for the option.

Note Parameters and values are described in Table 3-15.

Table 3-15: Parameters affecting database connections

database_param	Description and value
batch	Specifies how Replication Server sends commands to data servers. When batch is “on,” Replication Server may send multiple commands to the data server as a single command batch. When batch is “off,” Replication Server sends commands to the data server one at a time. Default: on
batch_begin	Indicates whether a begin transaction can be sent in the same batch as other commands (such as insert, delete, and so on). Default: on
command_retry	The number of times to retry a failed transaction. The value must be greater than or equal to 0. Default: 3

database_param	Description and value
deferred_name_resolution	<p>Enable deferred name resolution in Replication Server to support deferred name resolution in Adaptive Server. Deferred name resolution is only supported in Adaptive Server 15.5 and later.</p> <p>You must ensure that deferred name resolution is supported in the replicate Adaptive Server before you enable deferred name resolution support in Replication Server.</p> <p>After you execute <code>deferred_name_resolution</code> with <code>alter connection</code> or <code>alter logical connection</code>, suspend and resume the connection.</p> <p>Default: off</p>
disk_affinity	<p>Specifies an allocation hint for assigning the next partition. Enter the logical name of the partition to which the next segment should be allocated when the current partition is full.</p> <p>Default: off</p>
dist_stop_unsupported_cmd	<p>When <code>dist_stop_unsupported_cmd</code> is on, DIST suspends itself if a command is not supported by downstream Replication Server. If it is off, DIST ignores the unsupported command.</p> <p>Regardless of <code>dist_stop_unsupported_cmd</code> parameter's setting, Replication Server always logs an error message when it sees the first instance of a command that cannot be sent over to a lower-version Replication Server.</p> <p>Default: off</p>
db_packet_size	<p>The maximum size of a network packet. During database communication, the network packet value must be within the range accepted by the database.</p> <p>Default: 512-byte network packet for all Adaptive Server databases Maximum: 16,384 bytes</p>
dist_sqt_max_cache_size	<p>The maximum Stable Queue Transaction (SQT) cache size for the inbound queue. The default, 0, means the current setting of the <code>sqt_max_cache_size</code> parameter is used as the maximum cache size for the connection.</p> <p>Default: 0</p> <p>For 32-bit Replication Server:</p> <ul style="list-style-type: none"> • Minimum – 0 • Maximum – 2147483647 <p>For 64-bit Replication Server:</p> <ul style="list-style-type: none"> • Minimum – 0 • Maximum – 2251799813685247

database_param	Description and value
dsi_alt_writetext	<p>Controls how large-object updates are sent to the replicate database. The values are:</p> <ul style="list-style-type: none">• dcany – generates a writetext command that includes primary key columns. This setting prevents full table scans when populating non-ASE replicate databases using DirectConnect Anywhere™ as an interface.• off – generates an Adaptive Server writetext command that includes a text pointer. <p>Default: off</p> <hr/> <p>Note If you are using ExpressConnect to connect non-ASE replicate databases, then you are not required to configure the dsi_alt_writetext database parameter.</p> <hr/>
dsi_bulk_copy	<p>Turns the bulk copy-in feature on or off for a connection. If dynamic_sql and dsi_bulk_copy are both on, DSI applies bulk copy-in. Dynamic SQL is used if bulk copy-in is not used. Sybase recommends that you turn dsi_bulk_copy on to improve performance if you have large batches of inserts.</p> <p>Default: off.</p> <hr/>

database_param	Description and value
dsi_bulk_threshold	<p>The number of consecutive insert commands in a transaction that, when reached, triggers Replication Server to use bulk copy-in. When Stable Queue Transaction (SQT) encounters a large batch of insert commands, it retains in memory the number of insert commands specified to decide whether to apply bulk copy-in. Because these commands are held in memory, Sybase suggests that you do not configure this value much higher than the configuration value for dsi_large_xact_size.</p> <p>Replication Server uses dsi_bulk_threshold for real-time loading (RTL) replication to Sybase IQ and high volume adaptive replication (HVAR) to Adaptive Server. If the number of commands for an insert, delete, or update operation on one table is less than the number you specify after compilation, RTL and HVAR use language instead of bulk interface.</p> <p>Minimum: 1</p> <hr/> <p>Note Do not set to '1' when you enable RTL or HVAR as this detrimental to performance.</p> <hr/> <p>Default: 20</p> <p>Configuration level: Server, database</p> <p>For setting, use configure replication server for server-level or alter connection for database-level.</p> <hr/> <p>Note You must set dsi_compile_enable to 'on' to use dsi_bulk_threshold for RTL or HVAR.</p> <hr/>
dsi_charset_convert	<p>The specification for handling character-set conversion on data and identifiers between the primary Replication Server and the replicate Replication Server. This parameter applies to all data and identifiers to be applied at the DSI in question. The values are:</p> <ul style="list-style-type: none"> on – convert from the primary Replication Server character set to the replicate Replication Server character set; if character sets are incompatible, shut down the DSI with an error. allow – convert where character sets are compatible; apply any unconverted updates to the database, as well. off – do not attempt conversion. This option is useful if you have different but compatible character sets and do not want any conversion to take place. During subscription materialization, a setting of "off" behaves as if it were "allow." <p>Default: on</p> <hr/>
dsi_cmd_batch_size	<p>The maximum number of bytes that Replication Server places into a command batch.</p> <p>Default: 8192 bytes</p> <hr/>

database_param	Description and value
dsi_cmd_prefetch	<p>Allows DSI to pre-build the next batch of commands while waiting for the response from data server, and therefore improves DSI efficiency. If you also tune your data server to enhance performance, it is likely that you will gain an additional performance increase when you use this feature.</p> <p>Default: on</p> <p>When you set dsi_compile_enable to 'on', Replication Server ignores what you set for dsi_cmd_prefetch.</p> <p>License: Separately licensed under the Advanced Services Option. See "Replication Server – Advanced Services Option," in Chapter 4, "Performance Tuning" in the <i>Replication Server Administration Guide Volume 2</i>.</p>
dsi_cmd_separator	<p>The character that separates commands in a command batch.</p> <p>Default: newline (\n)</p> <hr/> <p>Note You must update this parameter in an interactive mode, not by executing a DDL-generated script, or any other script. You cannot reset dsi_cmd_separator by running a script.</p>
dsi_command_convert	<p>Specifies how to convert a replicate command when you set dsi_compile_enable to 'on'. A combination of these operations specifies the type of conversion:</p> <ul style="list-style-type: none"> • d – delete • i – insert • u – update • t – truncate • none – no operation <p>Combinations of operations for dsi_command_convert include i2none, u2none, d2none, i2di, t2none, and u2di.</p> <p>You must type the number "2". The operation before conversion precedes the "2" and the operations after conversion are after the "2". For example:</p> <ul style="list-style-type: none"> • d2none – do not replicate the delete command. • i2di,u2di – convert both insert and update to delete followed by insert, which is equivalent to an autocorrection. • t2none – do not replicate truncate table command. <p>Default: none.</p> <p>You can also configure this parameter at the table-level.</p> <p>For setting, use alter connection for database-level, or alter connection with the for replicate table named clause for table-level configuration.</p>

database_param	Description and value
dsi_commit_check_locks_intrvl	<p>The number of milliseconds (ms) the DSI executor thread waits between executions of the <code>rs_dsi_check_thread_lock</code> function string. Used with parallel DSI.</p> <p>Default: 1000ms (1 second)</p> <p>Minimum: 0</p> <p>Maximum: 86,400,000 ms (24 hours)</p>
dsi_commit_check_locks_log	<p>The number of times the DSI executor thread executes the <code>rs_dsi_check_thread_lock</code> function string before logging a warning message. Used with parallel DSI.</p> <p>Default: 200</p> <p>Minimum: 1</p> <p>Maximum: 1,000,000</p>
dsi_commit_check_locks_max	<p>The maximum number of times a DSI executor thread checks whether it is blocking other transactions in the replicate database before rolling back its transaction and retrying it. Used with parallel DSI.</p> <p>Default: 400</p> <p>Minimum: 1</p> <p>Maximum: 1,000,000</p>
dsi_commit_control	<p>Specifies whether commit control processing is handled internally by Replication Server using internal tables (on) or externally using the <code>rs_threads</code> system table (off).</p> <p>Default: on</p>

database_param	Description and value
dsi_compile_enable	<p>Set to 'on' to enable RTL or HVAR at the server-level, database-level, or table-level.</p> <p>Default:</p> <ul style="list-style-type: none"> off – server and database-level. Replication Server uses continuous log order row by row change replication. on – table-level <p>For setting, use configure replication server for server-level, alter connection for database-level, or alter connection with the for replicate table named clause for table-level configuration.</p> <p>Set dsi_compile_enable to 'off' for an affected table if replicating new row changes causes problems, such as when there is a trigger on the table which requires all the operations on that table to be replicated in log order, and therefore compilation is not allowed.</p> <hr/> <p>Note Set dsi_compile_enable to 'on' at the server or database-level before you set dsi_compile_enable to 'off' at the table-level.</p> <hr/> <p>When you set dsi_compile_enable to 'on', Replication Server ignores what you set for replicate_minimal_columns and dsi_cmd_prefetch.</p> <p>After you execute dsi_compile_enable at the server, database, or table-level, suspend and resume the connection.</p>
dsi_compile_max_cmds	<p>Specifies, in number of commands, the maximum size of a group of transactions. When HVAR or RTL reaches the maximum group size for the current group that it is compiling, HVAR or RTL starts a new group.</p> <p>If there is no more data to read, and even if the group does not reach the maximum number of commands, HVAR or RTL completes grouping the current set of transactions into the current group.</p> <p>Minimum: 100</p> <p>Default: 100,000</p> <p>You can configure the parameter at the server or database levels</p> <p>For setting, use configure replication server for server-level or alter connection for database-level.</p> <hr/> <p>Note You must set dsi_compile_enable to 'on' to use dsi_compile_max_cmds.</p>

database_param	Description and value
dsi_connector_type	<p>Specifies the database driver technology used for implementing the connector. This parameter along with dsi_dataserver_make is used to identify the connector that is associated with the connection. If you are replicating to ASE or IQ, set this parameter value to <i>ctlib</i> or if replicating to Oracle, set the value to <i>oci</i>.</p> <p>Default: <i>ctlib</i>.</p> <p>Valid values: <i>ctlib</i>, <i>oci</i>.</p>
dsi_dataserver_make	<p>Specifies the data server type that contains the replicate database that you want to connect to.</p> <p>Possible values are: ASE, IQ, and ORA.</p> <p>Use dsi_dataserver_make and dsi_connector_type to identify the connector that is associated with the connection.</p> <p>Set to IQ to replicate to Sybase IQ. Set to ASE to replicate to Adaptive Server, and ORA to replicate to Oracle.</p> <p>You can configure dsi_dataserver_make at the database level.</p> <p>If you do not specify this parameter, Replication Server deduces the data server type from the function-string class name of the database connection.</p> <p>If the functions-string class is customized, Replication Server cannot deduce the data server type and defaults to 'ASE'.</p>
dsi_exec_request_sproc	<p>Turns on or off request stored procedures at the DSI of the primary Replication Server.</p> <p>Default: on</p>
dsi_fadeout_time	<p>The number of seconds of idle time before a DSI connection is closed. A value of "-1" indicates that a connection will not close.</p> <p>Default: 600 seconds</p>
dsi_ignore_underscore_name	<p>When the transaction partitioning rule is set to "name," specifies whether or not Replication Server ignores transaction names that begin with an underscore. Values are "on" and "off."</p> <p>Default: on</p>

database_param	Description and value
dsi_isolation_level	<p>Specifies the isolation level for transactions. The ANSI standard and Adaptive Server supported values are:</p> <ul style="list-style-type: none">• 0 – ensures that data written by one transaction represents the actual data.• 1 – prevents dirty reads and ensures that data written by one transaction represents the actual data.• 2 – prevents nonrepeatable reads and dirty reads, and ensures that data written by one transaction represents the actual data.• 3 – prevents phantom rows, nonrepeatable reads, and dirty reads, and ensures that data written by one transaction represents the actual data. <hr/> <p>Note Data servers supporting other isolation levels are supported as well through the use of the <code>rs_set_isolation_level</code> function string. Replication Server supports all values for replicate data servers.</p> <hr/> <p>The default value is the current transaction isolation level for the target data server.</p>
dsi_keep_triggers	<p>Specifies whether triggers should fire for replicated transactions in the database.</p> <p>Set off to cause Replication Server to set triggers off in the Adaptive Server database, so that triggers do not fire when transactions are executed on the connection.</p> <p>Set on for all databases except standby databases.</p> <p>Default: on (except standby databases)</p>
dsi_large_xact_size	<p>The number of commands allowed in a transaction before the transaction is considered to be large.</p> <p>Minimum: 4</p> <p>Default: 100</p>
dsi_max_cmds_to_log	<p>The number of commands to write into the exceptions log for a transaction.</p> <p>Default: -1 (all commands)</p>
dsi_max_xacts_in_group	<p>Specifies the maximum number of transactions in a group. Larger numbers may improve data latency at the replicate database. Range of values: 1 – 1000.</p> <p>Default: 20</p>
dsi_max_text_to_log	<p>The number of bytes to write into the exceptions log for each <code>rs_writetext</code> function in a failed transaction. Change this parameter to prevent transactions with large text, unitext, image or rawobject columns from filling the RSSD or its log.</p> <p>Default: -1 (all text, unitext, image, or rawobject columns)</p>

database_param	Description and value
dsi_non_blocking_commit	<p>The number of minutes that Replication Server saves a message after a commit. A 0 value means that non-blocking commit is disabled.</p> <hr/> <p>Note You cannot use this parameter with alter connection to configure an active database connection in a standby environment.</p> <hr/> <p>Default: 0 Maximum: 60</p>
dsi_num_large_xact_threads	<p>The number of parallel DSI threads to be reserved for use with large transactions. The maximum value is one less than the value of dsi_num_threads.</p> <p>Default: 0</p>
dsi_num_threads	<p>The number of parallel DSI threads to be used. The maximum value is 255.</p> <p>Default: 1</p>
dsi_partitioning_rule	<p>Specifies the partitioning rules (one or more) the DSI uses to partition transactions among available parallel DSI threads. Values are origin, ignore_origin, origin_sessid, time, user, name, and none. See the <i>Replication Server Administration Guide Volume 2</i> for detailed information.</p> <p>Default: none</p>
dsi_quoted_identifier	<p>Enables or disables quoted identifier support in the Data Server Interface (DSI).</p> <p>Default: off</p>
dsi_replication	<p>Specifies whether or not transactions applied by the DSI are marked in the transaction log as being replicated.</p> <p>When dsi_replication is set to “off,” the DSI executes set replication off in the Adaptive Server database, preventing Adaptive Server from adding replication information to log records for transactions that the DSI executes. Since these transactions are executed by the maintenance user and, therefore, not usually replicated further (except if there is a standby database), setting this parameter to “off” avoids writing unnecessary information into the transaction log.</p> <p>dsi_replication must be set to “on” for the active database in a warm standby application for a replicate database, and for applications that use the replicated consolidated replicate application model.</p> <p>Default: on (“off” for standby database in a warm standby application)</p>

database_param	Description and value
dsi_replication_ddl	<p>Supports bidirectional replication by specifying whether or not transactions are to be replicated back to the original database.</p> <p>When dsi_replication_ddl is set to on, DSI sends set replication off to the replicate database, which instructs it to mark the succeeding DDL transactions available in the system log not to be replicated. Therefore, these DDL transactions are not replicated back to the original database, which enables DDL transaction replication in bidirectional MSA replication environment.</p> <p>Default: off</p>
dsi_rs_ticket_report	<p>Determines whether to call function string rs_ticket_report or not. rs_ticket_report function string is invoked when dsi_rs_ticket_report is set to on.</p> <p>Default: on</p>

database_param	Description and value
dsi_serialization_method	<p>Specifies the method used to determine when a transaction can start, while still maintaining consistency. In all cases, commit order is preserved.</p> <p>These methods are ordered from most to least amount of parallelism. Greater parallelism can lead to more contention between parallel transactions as they are applied to the replicate database. To reduce contention, use the dsi_partition_rule option.</p> <ul style="list-style-type: none">• no_wait – specifies that a transaction can start as soon as it is ready—without regard to the state of other transactions.• wait_for_start – specifies that a transaction can start as soon as the transaction scheduled to commit immediately before it has started.• wait_for_commit – specifies that a transaction cannot start until the transaction scheduled to commit immediately preceding it is ready to commit.• wait_after_commit – specifies that a transaction cannot start until the transaction scheduled to commit immediately preceding it has committed completely. <hr/> <p>Note You can only set dsi_serialization_method to no_wait if dsi_commit_control is set to “on”.</p> <hr/> <p>These options are retained only for backward compatibility with older versions of Replication Server:</p> <ul style="list-style-type: none">• none – same as wait_for_start.• single_transaction_per_origin – same as wait_for_start with dsi_partitioning_rule set to origin. <hr/> <p>Note The isolation_level_3 value is no longer supported as a serialization method but it is the same as setting dsi_serialization_method to wait_for_start and dsi_isolation_level to 3.</p> <hr/> <p>Default: wait_for_commit</p>

database_param	Description and value
dsi_sqt_max_cache_size	<p>Maximum SQT (Stable Queue Transaction interface) cache size for the outbound queue, in bytes.</p> <p>The default, “0,” means that the current setting of <code>sqt_max_cache_size</code> is used as the maximum cache size for the connection.</p> <p>Default: 0</p> <p>For 32-bit Replication Server:</p> <ul style="list-style-type: none">• Minimum – 0• Maximum – 2147483647 <p>For 64-bit Replication Server:</p> <ul style="list-style-type: none">• Minimum – 0• Maximum – 2251799813685247
dsi_text_convert_multiplier	<p>Changes the length of text or untext datatype columns at the replicate site. Use <code>dsi_text_convert_multiplier</code> when text or untext datatype columns must expand or contract due to character set conversion. Replication Server multiplies the length of primary text or untext data by the value of <code>dsi_text_convert_multiplier</code> to determine the length of text or untext data at the replicate site. Its type is float.</p> <ul style="list-style-type: none">• If the character set conversion involves expanding text or untext datatype columns, set <code>dsi_text_convert_multiplier</code> equal to or greater than 1.0.• If the character set conversion involves contracting text or untext datatype columns, set <code>dsi_text_convert_multiplier</code> equal to or less than 1.0. <p>Default: 1</p>
dsi_timer	<p>Use the <code>dsi_timer</code> configuration parameter to specify a delay between the time transactions commit at the primary database and the time transactions commit at the standby or replicate database. Replication Server processes transactions in the outbound queue in commit order after the period of delay is over.</p> <p>After you execute <code>dsi_timer</code> with <code>alter connection</code> or <code>alter logical connection</code>, suspend and resume the connection.</p> <p>Specify the delay in the hh:mm format.</p> <ul style="list-style-type: none">• Maximum: 24 hours.• Default: 00:00, which means there is no delay. <hr/> <p>Note Replication Server does not support time zone differences between the RepAgent or Replication Agent at the primary database and the Replication Server with the DSI connection where you want to execute <code>dsi_timer</code>.</p>

database_param	Description and value
dsi_xact_group_size	<p>The maximum number of bytes, including stable queue overhead, to place into one grouped transaction. A grouped transaction is multiple transactions that the DSI applies as a single transaction. A value of -1 means no grouping. Sybase recommends that you set dsi_xact_group_size to the maximum value and use dsi_max_xacts_in_group to control the number of transactions in a group.</p> <hr/> <p>Note Obsolete for Replication Server version 15.0 and later. Retained for compatibility with older Replication Servers.</p> <hr/> <p>Maximum: 2,147,483,647 Default: 65,536 bytes</p>
dump_load	<p>Set to “on” at replicate sites only to enable coordinated dump. See the <i>Replication Server Administration Guide Volume 2</i> for details.</p> <p>Default: off</p>
dynamic_sql	<p>Turns dynamic SQL feature on or off for a connection. Other dynamic SQL related configuration parameters will take effect only if this parameter is set to on.</p> <hr/> <p>Note If dynamic_sql and dsi_bulk_copy are both on, DSI applies bulk copy-in. Dynamic SQL is used if bulk copy-in is not used.</p> <hr/> <p>Default: off</p>
dynamic_sql_cache_management	<p>Manages the dynamic SQL cache for a connection.</p> <p>Values:</p> <ul style="list-style-type: none"> • mru – specifies that once dynamic_sql_cache_size is reached, the old dynamic SQL prepared statements are deallocated to give room for new statements. • fixed – specifies that once the dynamic_sql_cache_size is reached, allocation for new dynamic SQL statements stops. <p>Default: “fixed”</p>
dynamic_sql_cache_size	<p>Allows Replication Server to estimate how many database objects can use dynamic SQL for a connection. You can use dynamic_sql_cache_size to limit resource demand on a data server.</p> <p>Default: 100 Minimum: 1 Maximum: 65,535</p>

database_param	Description and value
exec_cmds_per_timeslice	<p>Specifies the number of LTL commands an LTI or RepAgent executor thread can process before yielding the CPU. By increasing this value, you allow the RepAgent executor thread to control CPU resources for longer periods of time, which may improve throughput from RepAgent to Replication Server.</p> <p>Set this parameter at the connection level using alter connection.</p> <p>See “Controlling the number of commands the RepAgent executor can process,” in Chapter 4, “Performance Tuning” in the <i>Replication Server Administration Guide Volume 2</i>.</p> <p>Default: 2,147,483,647</p> <p>Minimum: 1</p> <p>Maximum: 2,147,483,647</p>
exec_nrm_request_limit	<p>Specifies the amount of memory available for messages from a primary database waiting to be normalized.</p> <p>Set nrm_thread to ‘on’ with configure replication server before you use exec_nrm_request_limit.</p> <p>Minimum: 16,384 bytes</p> <p>Maximum: 2,147,483,647 bytes</p> <p>Default for:</p> <ul style="list-style-type: none">• 32-bit – 1,048,576 bytes (1MB)• 64-bit – 8,388,608 bytes (8MB) <p>After you change the configuration for exec_nrm_request_limit, suspend and resume the Replication Agent.</p> <p>License: Separately licensed under the Advanced Services Option. See “Replication Server – Advanced Services Option,” in Chapter 4, “Performance Tuning” in the <i>Replication Server Administration Guide Volume 2</i>.</p>
exec_sqm_write_request_limit	<p>Specifies the amount of memory available for messages waiting to be written to an inbound queue.</p> <p>Default: 1MB</p> <p>Minimum: 16KB</p> <p>Maximum: 2GB</p>

database_param	Description and value
md_sqm_write_request_limit	<p>Specifies the amount of memory available to the Distributor for messages waiting to be written to the outbound queue.</p> <hr/> <p>Note In Replication Server 12.1, md_sqm_write_request_limit replaces md_source_memory_pool. md_source_memory_pool is retained for compatibility with older Replication Servers.</p> <hr/> <p>Default: 1MB Minimum: 16KB Maximum: 2GB</p>
parallel_dsi	<p>Provides a shorthand method for configuring parallel DSI threads.</p> <p>A setting of “on” configures these values:</p> <ul style="list-style-type: none"> • dsi_num_threads to 5 • dsi_num_large_xact_threads to 2 • dsi_serialization_method to “wait_for_commit” • dsi_sqt_max_cache_size to 1 million bytes (on 32-bit platform) and 20 million bytes (on 64-bit platform). <p>A setting of “off” configures these parallel DSI values to their defaults.</p> <p>You can set this parameter to “on” and then set individual parallel DSI configuration parameters to fine-tune your configuration.</p> <p>Default: off</p>
rep_as_standby	<p>When the database is marked with sp_reptostandby and rep_as_standby is on, tables with a table replication definition not covered by a database replication definition are replicated. To replicate the tables, set:</p> <ul style="list-style-type: none"> • rep_as_standby to on • send_maint_xacts_to_replicate to false • send_warm_standby_xacts to true <p>Default: off</p>

database_param	Description and value
replicate_minimal_columns	<p>Specifies whether Replication Server should send all replication definition columns for all transactions, or only those needed to perform update or delete operations at the replicate database.</p> <p>Values are On and Off.</p> <p>Replication Server uses this connection-level parameter when a replication definition does not contain the replicate minimal columns clause, or if there is no replication definition at all.</p> <p>Otherwise, Replication Server ignores the value of this parameter.</p> <p>Default: On</p> <p>You can use admin config to display replicate_minimal_columns configuration information.</p> <p>When you set dsi_compile_enable to 'on', Replication Server ignores what you set for replicate_minimal_columns.</p> <p>See “Using replicate_minimal_columns with dynamic SQL.” in Chapter 4, “Performance Tuning” in the <i>Replication Server Administration Guide Volume 2</i>.</p>
save_interval	<p>The number of minutes that the Replication Server saves messages after they have been successfully passed to the destination data server. See the <i>Replication Server Administration Guide Volume 2</i> for details.</p> <p>Default: 0 minutes</p>
sub_sqm_write_request_limit	<p>Specifies the memory available to the subscription materialization or dematerialization thread for messages waiting to be written to the outbound queue.</p> <p>Default: 1MB</p> <p>Minimum: 16KB</p> <p>Maximum: 2GB</p>
use_batch_markers	<p>Controls the processing of function strings rs_batch_start and rs_batch_end. If use_batch_markers is set to on, the rs_batch_start function string is prepended to each batch of commands and the rs_batch_end function string is appended to each batch of commands.</p> <p>Set use_batch_markers to on only for replicate data servers that require additional SQL to be sent at the beginning or end of a batch of commands that is not contained in the rs_begin function string.</p> <p>Default: off</p>

security_param

A parameter that affects network-based security for connections. See Table 3-28 on page 313 for a list of parameters and a description of values.

`set security_services to 'default'`

Resets all network-based security features for the connection to match the global settings of your Replication Server.

new_ds and *new_db*

Name of the new data server and database for the connection.

Note The *new_ds* and *new_db* parameters can have the same values that you have defined for *data_server* and *database* parameters.

`trace`

Allows ExpressConnect tracing at the DSI level.

trace_value

A character string consisting of these components:

- *module* – Specifies the module type. Valid value is *econn*.
- *condition* – Specifies if a trace option is set to *on* or *off*.

The syntax for *trace_value* is '*module, condition,[on | off]*'.

Note The *trace* parameter in the *alter connection* command allows empty string. For example:

```
alter connection to data_server.database
set trace to ''
```

An empty string disables ExpressConnect tracing values after the connection or when the Replication Server is restarted.

Examples

Example 1 Changes the function-string class for the *pubs2* database in the *TOKYO_DS* data server to *sql_derived_class*:

```
suspend connection to TOKYO_DS.pubs2

alter connection to TOKYO_DS.pubs2b
set function string class to sql_derived_class

resume connection to TOKYO_DS.pubs2
```

Example 2 Changes the number of LTL commands the LTI or RepAgent Executor thread can process before it must yield the CPU to other threads:

```
suspend connection to TOKYO_DS.pubs2
alter connection to TOKYO_DS.pubs2b
```

```
set exec_cmds_per_timeslice to '10'  
resume connection to TOKYO_DS.pubs2
```

Usage

- Use suspend connection to suspend activity on the connection before altering it.
- Execute alter connection at the Replication Server where the connection was created.
- Before you use log transfer off to stop data transfer from a primary database, be sure there are no replication definitions defined for data in the database.
- To change the route to a Replication Server, use alter route.
- Use set function string class [to] *function_class* to activate class-level translations for non-Sybase data servers.
- You can set connection parameters using the alter connection parameter.
- Execute alter connection at the Replication Server where the connection was created.

Database connection parameters

- Use alter connection to change the configuration parameters of a DSI or a database connection. To change a DSI configuration value, suspend the connection to the DSI, change the value, and then resume the connection to the DSI. This procedure causes the new value to take effect.
- Replication Server configuration parameters are stored in the rs_config system table. Some parameters can be modified by updating rows in the table. See the *Replication Server Administration Guide Volume 1* for more information.
- See the *Replication Server Administration Guide Volume 2* for more information about configuring parallel DSI threads.
- Use assign action to enable retry of transactions that fail due to specific data server errors.
- Before you change the function-string class, make sure that the class and all the required function strings exist for the new class.
- Before you change the error class, make sure the new class exists.
- Change the character for data servers that require a command separator to recognize the end of a command.

If you have specified a different separator character and want to change it back to a newline character, enter the *alter connection* command as follows:

```
alter connection to data_server.database
set to '<Return>'
```

where you press the Return key, and no other characters, between the two single-quote characters.

The *dsi_bulk_copy* parameter

When *dsi_bulk_copy* is on, SQT counts the number of consecutive insert statements on the same table that a transaction contains. If this number reaches the *dsi_bulk_threshold*, DSI:

- 1 Bulk-copies the data to Adaptive Server until DSI reaches a command that is not insert or that belongs to a different replicate table.
- 2 Continues with the rest of the commands in the transaction.

Adaptive Server sends the result of bulk copy-in at the end of the bulk operation, when it is successful, or at the point of failure.

Note The DSI implementation of bulk copy-in supports multistatement transactions, allowing DSI to perform bulk copy-in even if a transaction contains commands that are not part of the bulk copy.

The *dsi_partitioning_rule* parameter

You can specify more than one partitioning rule at a time. Separate values with a comma, but no spaces. For example:

```
alter connection to data_server.database
set dsi_partitioning_rule to 'origin,time'
```

The *dataserver and database name* parameter

Using *dataserver* and *database name* parameter you can switch the connection from using one connector to using another connector. For example, if you replicating to Oracle using the ASE/CT-Lib connector and DirectConnect for Oracle and you want to switch your connection to use the Oracle/OCI connector, you may be required to use a new data server and database name. Because the name given to the DirectConnect/Oracle in the Sybase interfaces file may not be the same as the Oracle data server name in the Oracle TNS Names file. To change:

- 1 Suspend the connection.

- 2 Alter the connection setting dsi_dataserver_make to *ora* and dsi_connector_type to *oci*.
- 3 Alter the connection setting dataserver and database name to new_ds and new_db

where:

- *new_ds* – name of the data server in the Oracle *tnsnames.ora* file
- *new_db* – name of the database

Note The *new_ds* and *new_db* parameters can have the same values that you have defined for *data_server* and *database* parameters.

- 4 Resume the connection.

The *dump_load* parameter

Before setting *dump_load* to “on,” create function strings for the *rs_dumpdb* and *rs_dumptran* functions. Replication Server does not generate function strings for these functions in the system-provided classes or in derived classes that inherit from these classes.

The *save_interval* configuration parameter

Set *save_interval* to save transactions in the DSI queue that can be used to resynchronize a database after it has been restored from backups. Setting a *save_interval* is also useful when you set up a warm standby of a database that holds replicate data or receives replicated functions. You can use *sysadmin restore_dsi_saved_segments* to restore backlogged transactions.

Network-based security parameters

- Both ends of a connection must use compatible Security Control Layer (SCL) drivers with the same security mechanisms and security features. The data server must support *set proxy* or an equivalent command.

It is the replication system Administrator’s responsibility to choose and set security features for each server. Replication Server does not query the security features of remote servers before attempting to establish a connection. Connections fail if security features at both ends of the connection are not compatible.

- *alter connection* modifies network-based security settings for an outgoing connection from Replication Server to a target data server. It overrides default security parameters set with *configure replication server*.

- If `unified_login` is set to “required,” *only* the replication system Administrator with “sa” permission can log in to the Replication Server without a credential. If the security mechanism should fail, the replication system Administrator can log in to Replication Server with a password and disable `unified_login`.
- A Replication Server can have more than one security mechanism; each supported mechanism is listed in the `libtcl.cfg` file under SECURITY.
- Message encryption is a costly process with severe performance penalties. In most instances, it may be wise to set `msg_confidentiality` “required” only for certain connections. Alternatively, choose a less costly security feature, such as `msg_integrity`.

Using *alter connection* to change maintenance passwords

- You can change the maintenance user password of any DSI connection using the `alter connection` command:

```
alter connection to data_server.database
set password to password
```
- If your Replication Server is using ERSSD and the `data_server.database` match the ERSSD names, using `alter connection` and `set password` updates the `rs_maintusers` table, issues `sp_password` at ERSSD, and updates the configuration file line `RSSD_maint_pw`.

Permissions

`alter connection` requires “sa” permission.

See also

`admin show_connections`, `admin who`, `create connection`, `configure replication server`, `create error class`, `create function string class`, `drop connection`, `resume connection`, `set proxy`, `suspend connection`

alter connector

Description	Changes the attributes of a database connector.
Syntax	<pre>alter connector <i>dataserver_make.connector_type</i> set <i>option</i> [to] <i>value</i></pre>
Parameters	<p><i>dataserver_make</i></p> <p>Indicates the database server.</p> <p><i>connector_type</i></p> <p>Indicates the connector technology used for the connector implementation.</p> <p><i>option</i></p> <p>Provides you with choices for various trace options for a connector.</p> <p>The supported options are:</p> <ul style="list-style-type: none">• <code>trace</code>• <code>trace_logpath</code> <p><i>value</i></p> <p>A character string containing a new value for the option.</p> <p>If you are using the trace option, the syntax for <i>value</i> parameter is <i>trace_value</i>, which takes the form as “<i>module, condition[on/off]</i>”, where:</p> <ul style="list-style-type: none">• <i>module</i> – Specifies the module type. Valid value is <i>econn</i>.• <i>condition</i> – Specifies if a trace option is set to <i>on</i> or <i>off</i>. <p>See “alter connection” on page 123</p> <p>If you are using <code>trace_logpath</code> option, the syntax for the <i>value</i> parameter is <i>full_path_name</i>.</p>
Examples	<p>Example 1 Configures all DSI instances to use the ASE/CT-Lib connector with <code>general_1</code> trace condition enabled:</p> <pre>alter connector "ase"."ctlib" set trace to "econn,general_1,on"</pre> <p>Example 2 In this example, the <i>option</i> parameter is set to <code>trace_logpath</code> and all the trace messages produced by the ASE/CT-Lib connector are written to the connector-specific trace file in addition to the Replication Server log file:</p> <pre>alter connector "ase"."ctlib" set trace_logpath to "/sybase/sybase_rep/log/"</pre> <p>In general, the log file name consists of these parts:</p> <ul style="list-style-type: none">• <i>ec</i>

- *dataserver_make*
- *connector_type*
- *.log*

The *dataserver_make* and *connector_type* are variables. The values will depend on the type of database being used and the associated connector technology. For example, the connector-specific log file created for ASE/CT-Lib is *ecasectlib.log*.

Example 3 To turn off trace messages being written to connector-specific trace file, alter the *trace_logpath* configuration setting:

```
alter connector "ase"."ctlib"  
set trace_logpath to "
```

Usage

- Execute *alter connector* at the Replication Server where the connection was created.
- Execute *alter connector* to turn on traces for all connections using the specified connector.

alter database replication definition

Description Changes an existing database replication definition.

Syntax

```
alter database replication definition db_repdef
    with primary at srv.db
    {[not] replicate DDL | [not] replicate setname setcont |
    [not] replicate [in {SQLDML | DML_options} table_list]}
    [with dsi_suspended]

setname ::= {tables | functions | transactions | system procedures}
setcont ::= [in ([owner1.] name1 [, [owner2.] name2 [, ...]])]
```

Note The term functions in *setname* refers to user-defined stored procedures or user-defined functions.

Parameters

db_repdef
Name of the database replication definition.

server_name.db
Name of the primary server/database combination. For example:
TOKYO.dbase.

[not] replicate DDL
Tells Replication Server whether or not to send DDL to subscribing databases. If “replicate DDL” is not included, or the clause includes “not,” DDL is not sent to the replicate database.

[not] replicate *setname setcont*
Specifies whether or not to send objects stated in the *setname* category to the replicate database. The *setname* category can have a maximum of one clause each for tables, functions, transactions, and system procedures.

If you omit the system procedures *setname* or include the not option, Replication Server does not replicate the system procedures.

If you omit tables, functions, or transactions *setname* or include the not option, Replication Server replicates all objects of the *setname* category.

[not] replicate [in {SQLDML | *DML_options*} *table_list*]
Informs Replication Server whether or not to replicate SQL statements to tables defined in *table_list*.

SQLDML

Specifies these Data Manipulation Language (DML) operations:

- U – update
- D – delete
- I – insert select
- S – select into

DML_options

Any combination of these DML operations:

- U – update
- D – delete
- I – insert select
- S – select into

When the database replication mode is set to any combination of UDIS the RepAgent sends both individual log records and the information needed by Replication Server to build the SQL statement.

owner

An owner of a table or a user who executes a transaction. Replication Server does not process owner information for functions or system procedures.

You can replace *owner* with a space surrounded by single quotes or with an asterisk.

- A space (' ') – indicates no owner.
- An asterisk (*) – indicates all owners. Thus, for example, *.publisher means all tables named publisher, regardless of owner.

name

The name of a table, function, transaction, or system procedure.

You can replace *name* with a space surrounded by single quotes or with an asterisk.

- A space (' ') – indicates no name. For example, maintuser.' ' means all unnamed maintenance user transactions.
- An asterisk (*) – indicates all names. Thus, for example, robert.* means all tables (or transactions) owned by robert.

with dsi_suspended

Tells the replicate Replication Server to suspend the replicate DSI. Can be used to signal need to resynchronize databases.

Examples

Example 1 Changes the database replication definition rep_1C to filter out table2. The replicate DSI will be suspended:

```
alter database replication definition rep_1C
with primary at PDS.pdb
not replicate tables in (table2)
with dsi_suspended
```

Example 2 Applies update and delete statements for tables tb1 and tb2:

```
alter database replication definition dbrepdef
with primary at ds1.pdb1
replicate 'UD' in (tb1,tb2)
go
```

Usage

- When alter database replication definition is executed, Replication Server writes an rs_marker to the inbound queue. alter database replication definition does not take affect until the marker reaches the DIST, which gives the DIST time to incorporate the changes in the Database Subscription Resolution Engine (DSRE).
- Altering a database replication definition may desynchronize the primary and replication databases. See the *Replication Server Administration Guide Volume 1* for instructions for resynchronizing databases.

SQL statement replication

- If you do not specify a filter in your replication definition, the default is the not replicate clause. Apply alter database replication definition to change the SQLDML filters. You can either specify one or multiple SQLDML filters in a replicate clause.
- For more information about SQL statement replication see create database replication definition.

See also

create database replication definition, drop database replication definition

alter error class

Description

Changes an existing error class by copying error actions from another error class.

Syntax	<pre>alter [replication server] error class <i>error_class</i> set template to <i>template_error_class</i></pre>
Parameters	<p>replication server</p> <p>Indicates that the error class is a Replication Server error class and not a data server error class.</p> <p><i>error_class</i></p> <p>The error class to modify.</p> <p>set template to <i>template_error_class</i></p> <p>Use this clause to update an error class based on another error class. <code>alter error class</code> copies the error actions from the template error class to the existing error class.</p>
Examples	<p>Example 1 Changes <code>my_error_class</code> using <code>rs_sqlserver_error_class</code> as the basis:</p> <pre>alter error class my_error_class set template to rs_sqlserver_error_class</pre> <p>Example 2 Changes the <code>my_rs_err_class</code> Replication Server error class based on <code>rs_repserver_error_class</code>, which is the default Replication Server error class:</p> <pre>alter replication server error class my_rs_err_class set template to rs_repserver_error_class</pre>
Usage	<ul style="list-style-type: none">• Use the <code>alter error class</code> command and another error class as a template to alter error classes. <code>alter error class</code> copies error actions from the template error class to the error class you want to alter, and overwrites error actions which have the same error code.• The <code>rs_sqlserver_error_class</code> is the default error class provided for Adaptive Server databases while the <code>rs_repserver_error_class</code> is the default error class provided for Replication Server. Initially, these two error classes do not have a primary site. You must create these error classes at a primary site before you can change the default error actions.• You can assign non-Adaptive Server error classes to specific connections on non-Adaptive Server replication databases using the <code>create connection</code> and <code>alter connection</code> commands.• When Replication Server establishes a connection to a non-ASE replicate server, Replication Server verifies if the option to return native error codes from the non-ASE replicate server is enabled for the connection. If the option is not enabled, Replication Server logs a warning message that the connection works but error action mapping may not be correct.

See “ReturnNativeError,” in the Replication Server Options documentation to set the option in the Enterprise Connect™ Data Access (ECDA) Option for ODBC for your replicate server.

- For a list of non-Adaptive Server error classes, see Table 3-24 on page 238. For more information about non-Adaptive Server replication error classes, see the *Replication Server Administration Guide Volume 2*.

See also

assign action, create error class, drop error class

alter function

Description	Adds parameters to a user-defined function.
Syntax	<pre>alter function <i>table_rep_def.function_name</i> add parameters <i>@param_name datatype</i> [, <i>@param_name datatype</i>]...</pre>
Parameters	<p><i>table_rep_def</i> The name of the replication definition upon which the user-defined function operates.</p> <p><i>function_name</i> The name of the user-defined function to be altered.</p> <p><i>@param_name</i> The name of a parameter to be added to the user-defined function's parameter list. The parameter name must conform to the rules for identifiers and must be preceded by an @ sign.</p> <p><i>datatype</i> The datatype of the parameter. See "Datatypes" on page 21 for a list of the datatypes and their syntax. The parameter cannot be text, unitext, raw object, or image.</p>
Examples	<pre>alter function publishers_rep.upd_publishers add parameters @state char(2)</pre> <p>Adds an integer parameter named state to the upd_publishers function for the publishers_rep replication definition.</p>
Usage	<ul style="list-style-type: none"> • Before executing alter function, quiesce the replication system. You can use Replication Server Manager or the procedure described in the <i>Replication Server Troubleshooting Guide</i> to quiesce the system. • A user-defined function can have up to 255 parameters. • Altering functions during updates can cause unpredictable results. The affected data should be quiescent before you alter the function. • After altering a user-defined function, you may also have to alter function strings that use the new parameters. • When you alter a user-defined function for a replication definition, it is altered for all replication definitions of the primary table. • Do not use alter function for replicated functions. Use alter function rep def instead. alter function is used only for the asynchronous stored procedures described in Chapter 6, "RSSD Stored Procedures."

Permissions	alter function requires “create object” permission.
See also	admin quiesce_check, alter function string, create function, create function string, drop function, drop function string

alter function replication definition

Description Changes an existing function replication definition created by the create function replication definition command.

Note Support for create function replication definition and alter function replication definition are scheduled to be discontinued. Sybase suggests that you use these commands instead:

- create applied function replication definition and alter applied function replication definition
 - create request function replication definition and alter request function replication definition
-

Syntax

```
alter function replication definition function_rep_def
{
  deliver as 'proc_name' |
  add @param_name datatype [, @param_name datatype]... |
  add searchable parameters @param_name[, @param_name]... |
  send standby {all | replication definition}
  parameters
}
```

Parameters

function_rep_def
The name of the function replication definition to be altered.

deliver as
Specifies the name of the stored procedure to execute at the database where you are delivering the replicated function. *proc_name* is a character string of up to 200 characters. If you do not use this optional clause, the function is delivered as a stored procedure with the same name as the function replication definition.

add
Specifies additional parameters and their datatypes for the function replication definition.

@param_name
The name of a parameter to be added to the list of replicated parameters or searchable parameters. Each parameter name must begin with a @ character.

datatype
The datatype of the parameter you are adding to a parameter list. See “Datatypes” on page 21 for a list of supported datatypes and their syntax. Adaptive Server stored procedures and function replication definitions may not contain parameters with the text, unitext, and image datatypes.

add searchable parameters

Specifies additional parameters that can be used in the where clauses of the define subscription or define subscription command.

send standby

In a warm standby application, specifies whether to send all parameters in the function (send standby all parameters) or just those specified in the replication definition (send standby replication definition parameters) to a standby database. The default is send standby all parameters.

Examples

Example 1 Adds three parameters to the titles_frep function replication definition: a varchar parameter named @notes, a datetime parameter named @pubdate, and a bit parameter named @contract:

```
alter function replication definition titles_frep
add @notes varchar(200), @pubdate datetime,
@contract bit
```

Example 2 Adds the @type and @pubdate parameters to the list of searchable parameters in the titles_frep function replication definition:

```
alter function replication definition titles_frep
add searchable parameters @type, @pubdate
```

Example 3 Changes the titles_frep function replication definition to be delivered as the newtitles stored procedure at the destination database, typically the primary database (used for request function delivery):

```
alter function replication definition titles_frep
deliver as 'newtitles'
```

Usage

- alter function replication definition changes a function replication definition by adding replicated parameters, adding searchable parameters, specifying whether to send all parameters to the warm standby, or specifying a different name for the stored procedure to execute in the destination database.
- The name, parameters, and datatypes you specify for a function replication definition you are altering must match the stored procedure you are replicating. You can specify only those parameters you are interested in replicating.
- You must execute alter function replication definition at the Replication Server that manages the primary database (where you created the function replication definition).

- A parameter name must not appear more than once in any clause.
- If you are adding parameters, coordinate alter function replication definition with distributions for the function replication definition. Follow the steps in “Procedure to alter a function replication definition” on page 157 to avoid errors.
- You can use the optional `deliver as` clause to specify the name of the stored procedure to execute at the destination database where you are delivering the replicated function. Typically, you use this option in request function delivery. For more information, see create function replication definition.

See the *Replication Server Administration Guide Volume 1* for more information on alter function replication definition.

Procedure to alter a function replication definition

❖ **Altering a function replication definition**

- 1 Quiesce the replication system using Sybase Central’s Replication Manager plug-in or the procedure described in the *Replication Server Troubleshooting Guide*.

Ideally, you should first quiesce primary updates and ensure that all primary updates have been processed by the replication system. If you are unable to do that, then old updates in the primary log will not have values for new parameters, and the replication system will use nulls instead. You may need to take this into account when altering function strings in step 4 below.

- 2 Alter the stored procedure at the primary and the replicate sites.
- 3 Alter the function replication definition. Wait for the modified function replication definition to arrive at the replicate sites.
- 4 If necessary, alter any function strings pertaining to the function replication definition. Wait for the modified function strings to arrive at the replicate sites.
- 5 If necessary, modify subscriptions on the function replication definition at replicate sites. To modify a subscription, drop it and re-create it using drop subscription and create subscription (with no materialization option).

Altering a replication definition does not affect current subscriptions. If new parameters are added to the function replication definition, they are replicated with any new updates for all existing subscriptions.

- 6 Resume updates to the data at the primary database.

Permissions

alter function replication definition requires “create object” permission.

See also

alter function string, create function replication definition, drop function replication definition

alter function string

Description	Replaces an existing function string.
Syntax	<pre>alter function string [<i>replication_definition</i>.]<i>function</i>[:<i>function_string</i>] for <i>function_class</i> [scan '<i>input_template</i>'] [output {language '<i>lang_output_template</i>' rpc 'execute procedure [@<i>param_name</i>=]{<i>constant</i> ?<i>variable</i>!mod?} [, [@<i>param_name</i>=]{<i>constant</i> ?<i>variable</i>!mod?}]}... ' writetext [use primary log with log no log] none}]</pre>
Usage	<ul style="list-style-type: none"> alter function string is the same as create function string, except that it executes drop function string first. The function string is dropped and re-created in a single transaction to prevent errors that would result from missing function strings. Alter function strings for functions with class scope at the primary site for the function string class. See create function string class for more information about the primary site for a function-string class. Alter function strings for functions with replication definition scope, including user-defined functions, at the site where the replication definition was created. Each replication definition has its own set of function strings. For rs_select, rs_select_with_lock, rs_datarow_for_writetext, rs_get_textptr, rs_textptr_init, and rs_writetext function strings, Replication Server uses the <i>function_string</i> name to determine which string to alter. If a <i>function_string</i> name was provided when the function string was created, you must specify it with alter function string so that the function string to be altered can be found. See create function string for more information about alter function string, including descriptions of keywords and options. To restore the default function string for a function, omit the output clause.
Permissions	alter function string requires “create object” permission.
See also	alter connection, create connection, create function, create function string, create function string class, define subscription, drop function string

alter function string class

Description	Alters a function-string class, specifying whether it should be a base class or a derived class.
Syntax	<pre>alter function string class <i>function_class</i> set parent to {<i>parent_class</i> null}</pre>
Parameters	<p><i>function_class</i></p> <p>The name of an existing function-string class to be altered.</p> <p>set parent to</p> <p>Designates an existing class as a parent for the class you are altering; or, with the null keyword, designates that the class should be a base class.</p> <p><i>parent_class</i></p> <p>The name of an existing function-string class you designate as the parent class for a new derived class. <code>rs_sqlserver_function_class</code> may not be used as a parent class.</p> <p>null</p> <p>Specifies that the class should be a base class.</p>
Examples	<p>Example 1 Specifies that <code>sqlserver2_function_class</code> should become a derived class, inheriting function strings from the parent class <code>rs_default_function_class</code>:</p> <pre>alter function string class sqlserver2_function_class set parent to rs_default_function_class</pre> <p>Example 2 Specifies that the derived function-string class named <code>rpc_xact</code> should be a base class:</p> <pre>alter function string class rpc_xact set parent to null</pre>
Usage	<ul style="list-style-type: none"> • Use <code>alter function string class</code> to change a derived function-string class to a base class, to change the parent class of a derived class, or to change a base class to a derived class. • The primary site for a derived class is the same as its parent class. Alter derived classes at the primary site of the parent class. However, if the parent class is a system-provided class, <code>rs_default_function_class</code> or <code>rs_db2_function_class</code>, the primary site for the derived class is the Replication Server where you created the derived class. • See <code>create function string class</code> for more information about <code>alter function string class</code>.

- For more information about function-string classes, function strings, and functions, see the *Replication Server Administration Guide Volume 2*.
- Replication Server distributes the altered function-string class to qualifying sites through the replication system. The changes do not appear immediately at all such sites because of normal replication system lag time.

Permissions

alter function string class requires “sa” permission.

See also

alter connection, create connection, create function, create function string, create function string class, drop function string class

alter logical connection

Description	Disables or enables the Distributor thread for a logical connection, changes attributes of a logical connection, and enables or disables replication of truncate table to the standby database.
Syntax	<pre>alter logical connection to logical_ds.logical_db { set distribution {on off} set logical_database_param to 'value'}</pre>
Parameters	<p><i>logical_ds</i> The data server name for the logical connection.</p> <p><i>logical_db</i> The database name for the logical connection.</p> <p>distribution on Enables the Distributor thread for the logical connection.</p> <p>distribution off Disables the Distributor thread for the logical connection.</p> <p><i>logical_database_param</i> The name of a configuration parameter that affects logical connections. Table 3-16 describes the parameters you can set with alter logical connection.</p> <p><i>value</i> A setting for a configuration parameter that matches the parameter. <i>value</i> is a character string.</p>

Table 3-16: Configuration parameters affecting logical connections

logical_database_param	value
dist_stop_unsupported_cmd	<p>Use dist_stop_unsupported_cmd to set DIST to suspend itself or to continue running when it encounters commands not supported by downstream Replication Server. When dist_stop_unsupported_cmd is on, DIST suspends itself if a command is not supported by downstream Replication Server. If it is off, DIST ignores the unsupported command.</p> <p>Regardless of dist_stop_unsupported_cmd parameter’s setting, Replication Server always logs an error message when it sees the first instance of a higher version command that cannot be sent over to a lower version Replication Server.</p> <p>Default: off</p>
materialization_save_interval	<p>Materialization queue save interval. This parameter is only used for standby databases in a warm standby application.</p> <p>Default: “strict” for standby databases</p>

logical_database_param	value
replicate_minimal_columns	<p>Specifies whether Replication Server should send all replication definition columns for all transactions or only those needed to perform update or delete operations at the standby database. Values are “on” and “off.”</p> <p>Replication Server uses this value in standby situations only when a replication definition does not contain a <code>send standby</code> option with any parameter, or if there is no replication definition at all.</p> <p>Otherwise, Replication Server uses the value of the “replicate minimal columns” or “replicate all columns” parameter in the replication definition.</p> <p>Default: on</p> <p>When you set <code>dsi_compile_enable</code> to ‘on’, Replication Server ignores what you set for <code>replicate_minimal_columns</code>.</p>
save_interval	<p>The number of minutes that the Replication Server saves messages after they have been successfully passed to the destination data server. See the <i>Replication Server Administration Guide Volume 2</i> for details.</p> <p>Default: 0 minutes</p>
send_standby_repdef_cols	<p>Specifies which columns Replication Server should send to the standby database for a logical connection. Overrides “send standby” options in the replication definition that tell Replication Server which table columns to send to the standby database. Values are:</p> <ul style="list-style-type: none"> • on – send only the table columns that appear in the matching replication definition. Ignore the “send standby” option in the replication definition. • off – send all table columns to the standby. Ignore the “send standby” option in the replication definition. • check_repdef – send all table columns to the standby based on “send standby” option. <p>Default: check_repdef</p>
send_truncate_table	<p>Specifies whether to enable or disable replication of truncate table to standby database. Values are:</p> <ul style="list-style-type: none"> • on – enables replication of truncate table to standby database. This is the default. • off – disables replication of truncate table to standby database.

logical_database_param	value
ws_sqldml_replication	<p>Specifies whether to replicate SQL statements to warm standby data servers. Values are:</p> <ul style="list-style-type: none">• on – replicates SQL statements. The default statements replicated are update, delete, insert select, and select into.• off – ignores all SQL statements. <hr/> <p>Note ws_sqldml_replication has lower precedence than the table replication definition for SQL replication. If your table replication definition contains send standby clause for a table, this clause determines whether or not to replicate the DML statements, except select into, regardless of the ws_sqldml_replication parameter setting.</p> <hr/>

Examples

Example 1 Disables the distributor thread for the LDS.pubs2 logical connection:

```
alter logical connection to LDS.pubs2
set distribution off
```

Example 2 Changes the save interval for the LDS.pubs2 logical connection to “0,” allowing messages in the DSI queue for the logical connection to be deleted:

```
alter logical connection to LDS.pubs2
set save_interval to '0'
```

Example 3 Enables the replication of truncate table to the standby database:

```
alter logical connection to LDS.pubs2
set send_truncate_table to 'on'
```

Usage

- To copy truncate table to a warm standby database, set the send_truncate_table option to “on.”
- Set the send_truncate_table option to “on” only when both the active and warm standby databases are at Adaptive Server version 11.5 or later.
- If you specify the send_truncate_table to on clause, Replication Server copies the execution of truncate table to the warm standby database for all tables marked for replication.
- Use the alter logical connection command to disable the Distributor thread after you set up a warm standby application. When you add a database to the replication system, Replication Server creates a Distributor thread to process subscriptions for the data.

- Use the `set distribution off` clause to disable the Distributor thread for a logical connection. Use this option when you have set up a warm standby for a database but there are no subscriptions for the data in the database, and if the database is not a source of replicated stored procedure execution. Such a logical database may be a warm standby application that does not involve normal replication, or it may be a logical replicate database.
- Use `set distribution on` to start the Distributor thread for a logical connection after you disable it with `set distribution off`. Do this to create replication definitions and subscriptions for the data in the logical database, or to initiate replicated stored procedures in the logical database.
- You can suspend or resume a Distributor thread for a physical or logical database connection using the `suspend distributor` and `resume distributor` commands.
- See the *Replication Server Administration Guide Volume 1 and Volume 2* for more information about setting up and managing warm standby applications.
- You can set parameters that affect all logical connections originating at the current Replication Server with the `configure replication server` command.
- The `save_interval` parameter for a logical connection is set to 'strict,' by default, when the logical connection is created. This ensures that messages are not deleted from DSI queues before they are applied to the standby database.

If the standby database is not available for a long period of time, Replication Server's queues may fill. To avoid this, change `save_interval` from 'strict' to "0" (minutes). This allows Replication Server to delete the queues.

Warning! The `save_interval` parameter affects only the DSI queue. The `materialization_save_interval` parameter affects only currently existing materialization queues. They should *only* be reset under serious conditions caused by a lack of stable queue space. Resetting it (from 'strict' to a given number of minutes) may lead to message loss at the standby database. Replication Server cannot detect this type of loss; you must verify the integrity of the standby database yourself.

- The `materialization_save_interval` parameter for a logical connection is set to 'strict,' by default, when the logical connection is created. This ensures that messages are not deleted from materialization queues before they are applied to the standby database.

If the standby database is not available for a long period of time, Replication Server's queues may fill. To avoid this, change `materialization_save_interval` from 'strict' to "0" (minutes). This allows Replication Server to delete the queues.

See also `admin logical_status`, `configure replication server`, `create logical connection`, `resume distributor`, `suspend distributor`

alter partition

Description	Changes the size of a partition.
Syntax	<code>alter partition <i>logical_name</i> [expand [size =size]]</code>
Parameters	<p><i>logical_name</i></p> <p>A name for the partition. The name must conform to the rules for identifiers. The name is also used in the <code>drop partition</code> and <code>create partition</code> commands.</p> <p><code>expand</code></p> <p>Specifies that the partition is to increase in size.</p> <p><code>size</code></p> <p>Specifies the number of megabytes the partition is to increase. The default value is 2MB.</p>
Examples	<p>Example 1 This example increases the size of logical partition P1 by 50MB:</p> <pre>alter partition P1 expand size = 50</pre> <p>Example 2 This example increases the size of logical partition P2 by 2MB:</p> <pre>alter partition P2</pre>
Usage	<ul style="list-style-type: none">• <code>alter partition</code> allows users to expand a currently used partition to a larger size. This function is useful when Replication Server needs more disk space and there is still space available in the same disk of the existing partition.• In case of insufficient physical disk space, <code>alter partition</code> aborts and an error message displays. The allocated space for the partition is the same as before the command was applied.• The maximum size that can be allocated to a partition is 1TB, which is approximately 1,000,000MB.
Permissions	Only the "sa" user can execute <code>alter partition</code> .

See also `admin disk_space`, `create partition`, `drop partition`

alter queue

Description	Specifies the behavior of the stable queue that encounters a large message that is greater than 16K bytes. Applicable only when the Replication Server version is 12.5 or later and the Replication Server site version is 12.1 or earlier.
Syntax	<pre>alter queue, <i>q_number</i>, <i>q_type</i>, set sqm_xact_with_large_msg [to] {skip shutdown} set sqm_cache_enable to "on off" set sqm_page_size to "<i>numblocks</i>" set sqm_cache_size to "<i>numpages</i>"</pre>
Parameters	<p><i>q_number</i> The queue number of the stable queue.</p> <p><i>q_type</i> The queue type of the stable queue. Values are "0" for outbound queues and "1" for inbound queues.</p> <p>sqm_xact_with_large_msg {skip shutdown} Specifies whether the SQM should skip the message or shut down, when a message larger than 16K bytes is encountered.</p> <p>sqm_cache_enable to "on" "off" Enables or disables caching for the stable queue. Queue-level caching overrides server-level caching that is set using configure replication server. The default value of sqm_cache_enable is "on".</p> <p>sqm_page_size Sets the page size of the stable queue. Setting the page size at the queue level overrides server-level page size that is set using configure replication server. The default value of sqm_page_size is 4.</p> <p>"numblocks" Specifies the number of 16K blocks in a page. Configuring the page size also sets the I/O size of Replication Server. For example, setting page size to 4 instructs Replication Server to write to the stable queue in 64K chunks. <i>numblocks</i> accepts values from 1 to 64.</p> <p>sqm_cache_size Sets the cache size of the stable queue. Setting the cache size at the queue level overrides server-level cache size that is set using configure replication server. The default value of sqm_cache_size is 16.</p> <p>"numpages" Specifies the number of pages in the cache. The range is 1 to 512 pages.</p>
Examples	Shuts down queue number 2 if a large message is passed to the queue:

```
alter queue, 2, 0, set sqm_xact_with_large_msg to  
shutdown
```

Usage

- If you make changes to the `sqm_cache_enable`, `sqm_page_size`, and `sqm_cache_size` parameters restart the server for the changes to take effect.
- `alter queue` fails if the site version is 12.5 or later.

Permissions

`alter queue` requires "sa" permission.

See also

`alter route`, `resume queue`, `resume route`

alter replication definition

Description	Changes an existing replication definition.
Syntax	<pre>alter replication definition <i>replication_definition</i> {with replicate table named [<i>table_owner</i>.]<i>'table_name'</i> add <i>column_name</i> [as <i>replicate_column_name</i>] [<i>datatype</i> [null not null]] [map to <i>published_datatype</i>] [quoted],... alter columns with <i>column_name</i> [as <i>replicate_column_name</i>] [quoted not quoted],... alter columns with <i>column_name</i> <i>datatype</i> [null not null] [map to <i>published_datatype</i>],... references {[<i>table_owner</i>.]<i>table_name</i> [(<i>column_name</i>) null]} alter columns <i>column_name</i> {quoted not quoted} add primary key <i>column_name</i> [, <i>column_name</i>]... drop primary key <i>column_name</i> [, <i>column_name</i>]... add searchable columns <i>column_name</i> [, <i>column_name</i>]... drop searchable columns <i>column_name</i> [, <i>column_name</i>]... drop <i>column_name</i> [, <i>column_name</i>] ... send standby [off {all replication definition} columns] replicate {minimal all} columns replicate {SQLDML ['off'] '<i>options</i>'} replicate_if_changed <i>column_name</i> [, <i>column_name</i>]... always_replicate <i>column_name</i> [, <i>column_name</i>]... {with without} dynamic sql alter replicate table name {quoted not quoted}} [with DSI_suspended]</pre>
Parameters	<p><i>replication_definition</i></p> <p>The name of the replication definition to change.</p> <p>with replicate table named</p> <p>Specifies the name of the table at the replicate database. <i>table_name</i> is a character string of up to 200 characters. <i>table_owner</i> is an optional qualifier for the table name, representing the table owner. Data server operations may fail if actual table owners do not correspond to what you specify in the replication definition.</p> <p>add columns <i>column_name</i></p> <p>Specifies additional columns and their datatypes for the replication definition. <i>column_name</i> is the name of a column to be added to the replicated columns list. The column name must be unique for a replication definition.</p> <p>Also add columns <i>declared_column_name</i>. See “Using column-level datatype translations” on page 178.</p>

as *replicate_column_name*

For columns you are adding to the replication definition, specifies a column name in a replicate table into which data from the primary column will be replicated. *replicate_column_name* is the name of a column in a replicate table that corresponds to the specified column in the primary table. Use this clause when the replicate and primary columns have different names.

datatype

The datatype of the column you are adding to a replication definition column list or the datatype of an existing column you are altering. See “Datatypes” on page 21 for a list of supported datatypes and their syntax.

If a column is listed in an existing replication definition for a primary table, subsequent replication definitions for the same primary table must specify the same datatype.

Use *as declared_datatype* if you are specifying a column-level datatype translation for the column. A declared datatype must be a native Replication Server datatype or a datatype definition for the primary datatype.

null or not null

Applies only to text, unitext, image, and rawobject columns. Specifies whether a null value is allowed in the replicate table. The default is not null, meaning that the replicate table does not accept null values.

The null status for each text, unitext, image, and rawobject column must match for all replication definitions for the same primary table, and must match the settings in the actual tables. Specifying the null status is optional if an existing replication definition of the same primary table has text, unitext, image, or rawobject columns.

quoted | not quoted

Specifies whether a table or column name is a quoted identifier.

alter columns *column_name*

Specifies columns and their datatypes to alter in the replication definition. *column_name* is the name of a column to be changed. The column name must be unique for a replication definition.

Use alter columns *declared_column_name* when specifying a column-level datatype translation.

map to *published_datatype*

Specifies the datatype of a column after a column-level datatype translation. *published_datatype* must be a Replication Server native datatype or a datatype definition for the published datatype.

references *table owner.table name column name*

Specifies the table name of the table with referential constraints at the primary database that you want to add or change as a referencing table. Use the null option to drop a reference. *table_name* is a character string of up to 200 characters. *table_owner* is optional, and represents the table owner. *column name* is optional. Data server operations may fail if the actual table owners do not correspond to what you specify in the replication definition. See “Handling tables that have referential constraints” on page 311 for more information on usage.

add/drop primary key

Used to add or remove columns from the primary keys column list.

Replication Server depends on primary keys to find the correct rows at the replicate or standby table. To drop all primary key columns, first alter the corresponding replication definition to add the new primary keys, then drop the old primary key columns in the table. If all primary keys are missing, the DSI will shut down. See create replication definition for additional information on primary keys.

add searchable columns *column_name*

Specifies additional columns that can be used in where clauses of the create subscription or define subscription command. *column_name* is the name of a column to add to the searchable columns list. The same column name must not appear more than once in each clause.

You cannot specify text, unitext, image, rawobject, rawobject in row or encrypted columns as searchable columns.

drop searchable columns *column_name*

Specifies columns to remove from the searchable column list. You can remove columns from the searchable column list only if they are not used in subscription or article where clauses.

drop *column_name*

Specifies columns to remove.

send standby

Specifies how to use the replication definition in replicating into a standby database in a warm standby application. See “Replicating into a standby database” on page 178 for details on using this clause and its options.

replicate minimal columns

Sends (to replicate Replication Servers) only those columns needed to perform update or delete operations at replicate databases. To replicate all columns, use replicate all columns.

`replicate SQLDML ['off']`

Turns on or off the SQL statement replication of the DML option specified.

`replicate 'options'`

Replicates any combination of these DML operations:

- U – update
- D – delete
- I – insert select

`replicate_if_changed`

Specifies text, unitext, image, or rawobject columns to be added to the `replicate_if_changed` column list. When multiple replication definitions exist for the same primary table, using this clause to change one replication definition changes all replication definitions of the same primary table.

`always_replicate`

Specifies text, image, or rawobject columns to be added to the `always_replicate` column list. When multiple replication definitions exist for the same primary table, using this clause to change one replication definition changes all replication definitions of the same primary table.

`with dynamic sql`

Specifies that DSI applies dynamic SQL to the table if the command qualifies and enough cache space is available. This is the default.

See the *Replication Server Administration Guide Volume 2* for the conditions a command must meet to qualify for dynamic SQL.

`without dynamic sql`

Specifies that DSI must not use dynamic SQL commands.

with DSI_suspended

Allows you to suspend the standby DSI, if there is one, and each of the subscribing replicate DSI threads. Replication Server suspends the DSI thread in the standby or replicate database after Replication Server applies all the data for the old replication definition version to the standby or replicate database.

After Replication Server suspends a DSI thread, you can make changes to the target schema, and to any customized function strings. When you resume the DSI thread, Replication Server replicates the primary updates using the altered replication definition.

You do not need to use with DSI_suspended if:

- There is no subscription to the replication definition.
- You do not need to change customized function strings.
- You do not need to change the replicate or standby database schema.

Note If there is a subscription from a replicate Replication Server with a site version earlier than 1550, the replicate DSI threads for that Replication Server are not suspended.

Examples

Example 1 Adds state as a searchable column to the authors_rep replication definition:

```
alter replication definition authors_rep
add searchable columns state
```

Example 2 Changes the titles_rep replication definition to specify that only the minimum number of columns will be sent for delete and update operations:

```
alter replication definition titles_rep
replicate minimal columns
```

Example 3 Changes the titles_rep replication definition to specify that the replication definition can be subscribed to by a replicate table called copy_titles owned by the user “joe”:

```
alter replication definition titles_rep
with replicate table named joe.'copy_titles'
```

Example 4 Changes the pubs_rep replication definition to specify that the primary column pub_name will replicate into the replicate column pub_name_set:

```
alter replication definition pubs_rep
```

```
alter columns with pub_name as pub_name_set
```

Example 5 Introduces a column-level translation that causes hire_date column values to be translated from rs_db2_date (primary) format to the native datatype smalldatetime (replicate) format:

```
alter replication definition employee_repdef
alter columns with hire_date as rs_db2_date
map to smalldatetime
```

Example 6 Marks the table named foo as a quoted identifier:

```
alter replication definition repdef
alter replicate table name "foo" quoted
```

Example 7 Removes the quoted identifier marking from the column foo_coll:

```
alter replication definition repdef
with replicate table named "foo"
alter columns "foo_coll" not quoted
```

Example 8 Instructs Replication Server to suspend the target DSI after primary data that exists before you execute alter replication definition is replicated to the target database:

```
alter replication definition pubs_rep
alter columns with pub_name as pub_name_set
with DSI_suspended
```

Example 9 Drops the address, city, state, and zip columns from the “authors” replication definition:

```
alter replication definition authors
drop address, city, state, zip
```

Usage

- Use the alter replication definition command to change a replication definition by:
 - Adding or dropping primary keys
 - Changing the name of a target replicate table
 - Changing the names of target replicate columns
 - Adding columns and indicating the names of corresponding target replicate columns
 - Adding or dropping searchable columns
 - Changing replication definition usage by warm standby applications
 - Changing column datatypes

- Changing between replicating all or minimal columns
- Changing replication status for text, unitext, image, or rawobject columns
- Introducing or removing a column-level datatype translation
- Including or excluding the table in the dynamic SQL application at DSI
- Execute alter replication definition at the primary site for the replication definition.
- For a database replication definition to replicate encrypted columns without using a table level replication definition, you must define the encryption key for the encrypted columns with `INIT_VECTOR NULL` and `PAD NULL`.
- In a mixed-version environment, where the primary Replication Server has a version later than that of the replicate Replication Server, you cannot change a replication definition that is supported and subscribed to by the replicate Replication Server if the replicate Replication Server cannot support the modification. However, if the replicate Replication Server supports but does not subscribe to the replication definition, the replication definition is modified and is dropped from the replicate Replication Server.
- See “Replicating SQL statements” on page 310 for more information about replicating SQL statements.
- See create replication definition for more information about the options in the alter replication definition command.

Adding columns

- If you add columns, coordinate alter replication definition with distributions for the replication definition. To avoid errors, follow the steps in “Procedure to alter a replication definition” on page 179.
- If a column you are adding to a replication definition contains an identity column, the maintenance user must be the owner of the table (or must be “dbo” or aliased to “dbo”) at the replicate database in order to use the Transact-SQL `identity_insert` option. A primary table can contain only one identity column.
- If the column you are adding to a replication definition contains a timestamp column, the maintenance user must be the owner of the table (or must be “dbo” or aliased to “dbo”) at the replicate database. A primary table can contain only one timestamp column.

Dropping columns

- If there is a subscription from a replicate Replication Server with a site version earlier than 1550, the primary Replication Server rejects the alter replication definition request to drop a column.

Note If you alter a replication definition to drop a column, you may need to reset autocorrection or dynamic SQL settings at replicate Replication Servers with site versions earlier than 1550.

- If there are multiple replication definitions for a primary table, alter replication definition drops only the columns from the replication definition you specify in *repdef_name* in the command line.
- The drop parameter drops a column or columns from a table replication definition. If a column is part of the primary key or searchable columns, drop drops the column from the primary key list or searchable column list. Replication Server rejects an alter replication definition request to drop a column if the column is:
 - The only column
 - The only primary key column for the replication definition
 - In the where clause of a subscription or article
 - Before a searchable column which is specified in the where clause of an article or subscription.

Altering column datatypes

- You cannot change the column datatype if it is used in a subscription or article where clause.
- You cannot change the *rs_address* datatype.
- You can change the column datatype to a text, untext, image, rawobject, or rawobject in row datatype only if it is not a primary key or searchable column.
- To change the published datatype of a column, you must specify both the declared datatype and the map to option.
- If there are more than one replication definition for a primary table, declared datatype and nullability of a column should be consistent across all replication definitions of the table.
- See the *Replication Server Administration Guide Volume 1*, which describes how to change datatypes.

- Changes between null and not null can only be used for text, untext, image and rawobject columns.

Using column-level datatype translations

- To effect column-level datatype translations, you must first set up and install the heterogeneous datatype support (HDS) objects as described in the *Replication Server Configuration Guide* for your platform.
- You cannot use text, untext, image, or rawobject datatypes as a base datatype or a datatype definition or as a source or target of either a column-level or class-level translation.
- *declared_datatype* depends on the datatype of the value delivered to Replication Server:
 - If the Replication Agent delivers a base Replication Server datatype, *declared_datatype* is the base Replication Server datatype.
 - If the Replication Agent delivers any other datatype, *declared_datatype* must be the datatype definition for the original datatype in the primary database.
- *published_datatype* is the datatype of the value after a column-level translation, but before any class-level translation. *published_datatype* must be a Replication Server native datatype or a datatype definition for the datatype in another database.
- Columns declared in multiple replication definitions must use the same *declared_datatype* in each replication definition. *published_datatype* can differ.

Replicating all or minimal columns

- When you use replicate minimal option for a replication definition, data is sent to replicate Replication Servers for the minimum number of columns needed for delete or update operations. Specify replicate all columns to replicate all columns. See create replication definition for additional information about this feature.

Replicating into a standby database

- Replication Server does not require replication definitions to maintain a standby database in a warm standby application. Using replication definitions may improve performance in replicating into the standby database. You can create a replication definition just for this purpose for each table in the logical database.

- Use `send standby` with any option other than `off` to use this replication definition to replicate transactions for this table to the standby database. The replication definition's primary key columns and `replicate minimal columns` setting are used to replicate into the standby database. The options for this method include:
 - Use `send standby` or `send standby all columns` to replicate all primary table columns into the standby database.
 - Use `send standby replication definition columns` to replicate only the replication definition's columns into the standby database.
- Use `send standby off` to indicate that no single replication definition for this table should be used in replicating into the standby database. All the columns in the table are replicated into the standby database, and the union of all primary key columns in all replication definitions for the table is used in replicating into the standby database. The `replicate_minimal_columns` setting of the logical connection determines whether to send minimal columns or all columns for update and delete. See `alter logical connection`.

If no replication definition exists for a table, all the columns in the table are replicated into the standby database and Replication Server constructs the primary key. In this case, `replicate_minimal_columns` is on.

Handling tables that have referential constraints

You can use a replication definition to specify tables that have referential constraints, such as a foreign key and other check constraints, so that Replication Server is aware of these tables when you enable RTL or HVAR. See “High Volume Adaptive Replication to Adaptive Server,” in Chapter 4, “Performance Tuning” in the *Replication Server Administration Guide Volume 2* and Chapter 13, “Replication into Sybase IQ” in the *Replication Server Heterogeneous Replication Guide*.

Procedure to alter a replication definition

When you request changes to replication definitions, Replication Server coordinates the propagation of replication definition changes and data replication automatically. You can request replication definition changes directly at the primary Replication Server, or at the primary database using the `alter replication definition`, `alter applied replication definition`, or `alter request function replication definition` commands, while making changes to the database schema.

When the primary database log does not contain data for the replication definition being changed, you can issue the replication definition request directly at the primary Replication Server. Otherwise, it is always safe to issue the replication definition requests at the primary database, using the `rs_send_repserver_cmd` stored procedure.

If the database does not support `rs_send_repserver_cmd`, you need to wait until the primary database log does not have any data rows for the schema that you are changing, and then execute the alter replication definition request at the primary Replication Server.

See “Replication definition change request process,” in Chapter 9, “Managing Replicated Tables” in the *Replication Server Administration Guide Volume 1*.

Permissions

alter replication definition requires “create object” permission.

See also

`admin verify_repserver_cmd`, `alter function string`, `create replication definition`, `drop replication definition`, `setrs_set_quoted_identifier`, `rs_send_repserver_cmd`, `rs_helprepversion`

alter request function replication definition

Description

Changes the function replication definition created by the `create request function replication definition` command.

Syntax

```
alter request function replication definition repdef_name
    {with replicate function named 'func_name' |
    add @param_name datatype[, @param_name datatype]... |
    add searchable parameters @param_name[, @param_name]... |
    send standby {all | replication definition} parameters}
    [with DSI_suspended]
```

Parameters

repdef_name

The name of the request function replication definition to change.

with replicate function named '*func_name*'

Specifies the name of the stored procedure to execute at the replicate database. The replicate function name of this replication definition must be different from its primary function name. *func_name* is a character string with a maximum length of 255 characters.

add

Specifies additional parameters and their datatypes for the function replication definition.

@param_name

The name of the parameter that you want to add to the list of replicated or searchable parameters. Each parameter name must begin with the @ character.

datatype

The datatype of the parameter you want to add to a parameter list. See “Datatypes” on page 21 for a list of the datatypes and their syntax. Adaptive Server stored procedures and function replication definitions cannot contain parameters with the text, unitext, rawobject, and image datatypes.

add searchable parameters

Specifies additional parameters that you can use in the where clause of the create subscription or define subscription command.

send standby

In a warm standby application, specifies whether to send all parameters in the function (send standby all parameters) or only those specified in the replication definition (send standby replication definition parameters), to a standby database. The default is send standby all parameters.

with DSI_suspended

Allows you to suspend the standby DSI, if there is one, and each of the subscribing replicate DSI threads. Replication Server suspends the DSI thread in the standby or replicate database after Replication Server applies all the data for the old replication definition version to the standby or replicate database.

After Replication Server suspends a DSI thread, you can make changes to target stored procedures, and to any customized function strings. When you resume the DSI thread, Replication Server replicates the primary updates using the altered replication definition.

You do not need to use with DSI_suspended if:

- There is no subscription to the replication definition.
- You do not need to change customized function strings.
- You do not need to change the replicate or standby database stored procedure.

Note If there is a subscription from a replicate Replication Server with a site version earlier than 1550, the replicate DSI threads for that Replication Server are not suspended.

Examples

Example 1 Adds @notes, @pubdate, and @contract parameters to the titles_frep function replication definition:

```
alter request function replication definition
    titles_frep
add @notes varchar(200), @pubdate datetime,
    @contract bit
```

Example 2 Adds the @type and @pubdate parameters to the list of searchable parameters in the titles_frep function replication definition:

```
alter request function replication definition
    titles_frep
add searchable parameters @type, @pubdate
```

Example 3 Changes the titles_frep function replication definition to be replicated as the newtitles stored procedure at the replicate database, and instructs Replication Server to suspend the target DSI after primary data that exists before you execute alter request replication definition is replicated to the replicate database:

```
alter request function replication definition titles_frep
with replicate function named 'newtitles'
with DSI suspended
```

Usage

- Use alter request function replication definition to change an existing request function replication definition. You can add replicated and searchable parameters, select which parameters to send to the warm standby, and specify a different name for the stored procedure to execute at the replicate database.
- alter request function replication definition can alter only the replication definition created with the create request function replication definition command.
- When you change a function replication definition, the name, parameters, and datatypes that you specify for the function replication definition must match the stored procedure that you are replicating. Only the parameters specified in the function replication definition are replicated.
- Multiple function replication definitions for the same stored procedure must have the same parameter list. If you add a new parameter, the new parameter is automatically added to all the function replication definitions created for that stored procedure.
- You must execute the alter request function replication definition command at the primary Replication Server where you created the function replication definition.

- A parameter name cannot appear more than once in any clause.
- When adding parameters, you *must* instruct Replication Server to coordinate alter request function replication definition with distributions for the function replication definition. In addition, you must instruct Replication Server to coordinate changes to stored procedures and replicatim definitions.

See “Replication definition change request process,” in Chapter 9, “Managing Replicated Tables” in the *Replication Server Administration Guide Volume 1* to alter replication definitions.

- Use the with replicate function named clause to specify the stored procedure name you want to execute at the replicate database. See create request function replication definition.

For more information about altering a request function replication definition, see the *Replication Server Administration Guide Volume 1*.

Permissions

alter request function replication definition requires “create object” permission.

See also

alter function string, alter applied function replication definition, create applied function replication definition, create request function replication definition, drop function replication definition, rs_send_repserver_cmd, rs_helprepversion

alter route

Description	Changes the attributes of a route from the current Replication Server to a remote Replication Server.
Syntax	<pre>alter route to <i>dest_replication_server</i> { set next site [to] <i>thru_replication_server</i> set username [to] '<i>user</i>' set password [to] '<i>passwd</i>' set password [to] '<i>passwd</i>' set <i>route_param</i> [to] '<i>value</i>' set <i>security_param</i> [to] '<i>value</i>' set security_services [to] 'default'}</pre>
Parameters	<p><i>dest_replication_server</i> The name of the destination Replication Server whose route you are altering.</p> <p><i>thru_replication_server</i> The name of an intermediate Replication Server through which messages for the destination Replication Server will be passed.</p> <p><i>user</i> The login name to use for the route.</p> <p><i>passwd</i> The password to use with the login name.</p> <p><i>route_param</i> A parameter that affects routes. Refer to Table 3-17 for a list of parameters and values.</p> <p><i>value</i> A setting for <i>route_param</i>. It is a character string.</p>

Table 3-17: Configuration parameters affecting routes

route_param	value
disk_affinity	Specifies an allocation hint for assigning the next partition. Enter the logical name of the partition to which the next segment should be allocated when the current partition is full. Default: off
rsi_batch_size	The number of bytes sent to another Replication Server before a truncation point is requested. Default: 256KB Minimum: 1KB Maximum: 128MB
rsi_fadeout_time	The number of seconds of idle time before Replication Server closes a connection with a destination Replication Server. Default: -1 (specifies that Replication Server will not close the connection)

route_param	value
rsi_packet_size	Packet size, in bytes, for communications with other Replication Servers. The range is 1024 to 16384 bytes. Default: 2048 bytes
rsi_sync_interval	The number of seconds between RSI synchronization inquiry messages. The Replication Server uses these messages to synchronize the RSI outbound queue with destination Replication Servers. Values must be greater than 0. Default: 60 seconds
rsi_xact_with_large_msg	Specifies route behavior if a large message is encountered. This parameter is applicable only to direct routes where the site version at the replicate site is 12.1 or earlier. Values are “skip” and “shutdown.” Default: shutdown
save_interval	The number of minutes that the Replication Server saves messages after they have been successfully passed to the destination Replication Server. See the <i>Replication Server Administration Guide Volume 2</i> for details. Default: 0 minutes

security_param

specifies the name of a security parameter. For a list and description of security parameters that can be set with `alter route`, refer to Table 3-28 on page 313.

`set security_services [to] 'default'`

resets all network-based security features for the connection to match the global settings of your Replication Server.

Examples

Example 1 In examples 1 and 2, direct routes exist from the Tokyo Replication Server (TOKYO_RS) to the San Francisco Replication Server (SF_RS) and to the Sydney Replication Server (SYDNEY_RS). The following commands change one direct route into an indirect route, so that TOKYO_RS passes messages destined for SYDNEY_RS through SF_RS.

Entered at SF_RS, this command creates a direct route to SYDNEY_RS that will be used by the new indirect route:

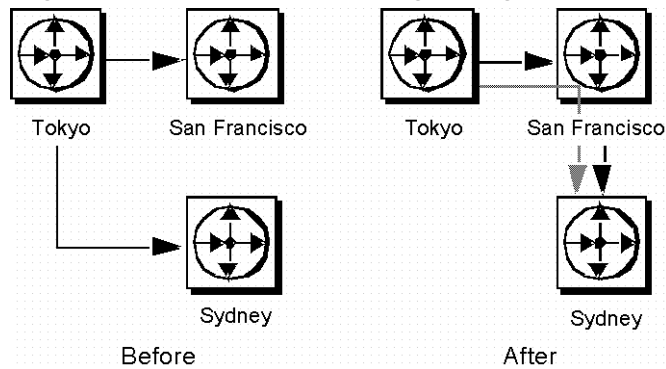
```
create route to SYDNEY_RS
set username SYDNEY_rsi_user
set password SYDNEY_rsi_passwd
```

Example 2 Entered at TOKYO_RS, this command changes the direct route from TOKYO_RS to SYDNEY_RS to an indirect route, specifying SF_RS as an intermediate Replication Server:

```
alter route to SYDNEY_RS
set next site SF_RS
```

Figure 3-1 shows the routes before and after changing the routing scheme.

Figure 3-1: Before and after altering routing in examples 1 and 2



Examples 3 and 4 change the routing so that TOKYO_RS sends messages directly to SYDNEY_RS again, instead of passing them through SF_RS.

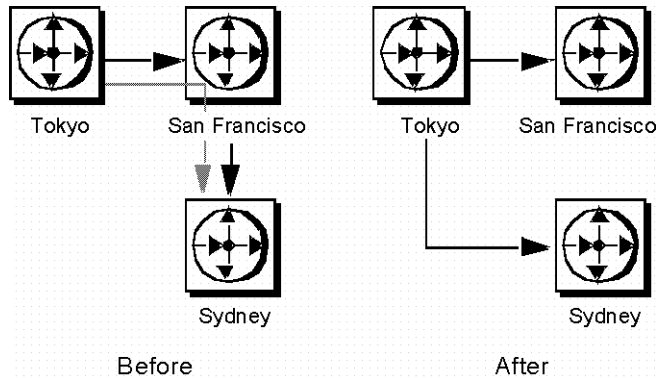
Example 3 Entered at TOKYO_RS, this command changes the route from TOKYO_RS to SYDNEY_RS from an indirect route to a direct route:

```
alter route to SYDNEY_RS
set username SYDNEY_rsi
set password SYDNEY_rsi_passwd
```

Example 4 Entered at SF_RS, this command removes the direct route from SF_RS to SYDNEY_RS:

```
drop route to SYDNEY_RS
```

Together, the commands in examples 3 and 4 cancel the effects of examples 1 and 2. Figure 3-2 shows the routes after the second set of commands is entered.

Figure 3-2: After altering routing

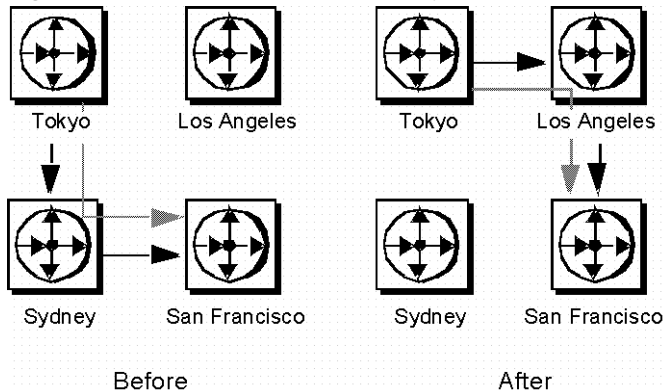
In example 5, direct routes exist from TOKYO_RS to SYDNEY_RS and from SYDNEY_RS to SF_RS, and an indirect route exists from TOKYO_RS to SF_RS, through SYDNEY_RS. This example changes this routing scheme so that TOKYO_RS passes messages destined for SF_RS through a different Replication Server, LA_RS in Los Angeles.

Example 5 Entered at TOKYO_RS, this command changes the intermediate Replication Server for the indirect route to LA_RS instead of SYDNEY_RS.

```
alter route to SF_RS
set next site LA_RS
```

Before the route can be altered, direct routes must have been created from TOKYO_RS to LA_RS and from LA_RS to SF_RS.

Figure 3-3 shows the routes before and after the necessary commands have been entered. (Direct routes to and from SYDNEY_DS are not shown because you may have dropped them.)

Figure 3-3: Before and after necessary commands

Example 6 Entered at TOKYO_RS, this command changes the password for the direct route from TOKYO_RS to LA_RS. The new password is “LApass.”

```
alter route to LA_RS
set password LApass
```

Before you change the password for the direct route, you must suspend the route using `suspend route`.

Example 7 Sets the security service to DCE for the route to LA_RS:

```
suspend route to LA_RS

alter route to LA_RS
set security_mechanism to 'dce'

resume route to LA_RS
```

Usage

- Use `alter route` to change:
 - A direct route to an indirect route.
 - An indirect route to a direct route.
 - The next intermediate site in an existing route.
 - The password for the RSI user for an existing direct route.
 - A route configuration parameter.
 - A network-based security parameter.

For an overview of routes, see the *Replication Server Administration Guide Volume 1*.

- Execute `alter route` at the Replication Server that is the source for a direct route.
- Use `set next site thru_replication_server` when you are changing a direct route into an indirect route, or when you are changing the intermediate site in an indirect route.
- If you are changing a direct route to an indirect route, you must first create direct routes from the source site to the intermediate site, and from the intermediate site to the destination site. Do this with `create route`.
- If you are changing the intermediate site in an indirect route, you must first create direct routes from the new intermediate site to the destination site, and from the new intermediate site to the destination site. Do this with `create route`.
- An indirect route may have one or more intermediate Replication Servers. For example, an indirect route from A_RS to D_RS may pass through intermediate sites B_RS and C_RS.
- To change an indirect route to a direct route, use `alter route` without the `set next site` clause, specifying the login name and password to use at the destination Replication Server. For example, an indirect route from A_RS->B_RS->C_RS changes to a direct route A_RS->C_RS.
- To exchange one intermediate site for the next intermediate site, execute `alter route` with the `set next site` clause. For example, an indirect route A_RS->B_RS->C_RS->D_RS changes to A_RS->C_RS->D_RS.
- You can set route parameters using the `configure route` or `alter route` parameter.
- Use `suspend route` to suspend activity on the route before altering it.

set password and set username

- Use `set username user` and `set password passwd` only when you are changing an indirect route to a direct route. You cannot change the user name or password for indirect routes; attempting to do so changes the indirect route to a direct route.
- Use `set password passwd` only when you are changing the password for a direct route. Before you change the password for a direct route, use `suspend route`.

Route parameters

- Setting a save interval allows the system to tolerate partition or stable queue failures at the destination Replication Server. Backlogged messages are sent to the destination Replication Server during recovery with the rebuild queues command.

See the *Replication Server Administration Guide Volume 2* for detailed information about the save interval and stable queue recovery.

- Sybase recommends that you leave the `rsi_batch_size`, `rsi_fadeout_time`, `rsi_packet_size`, and `rsi_sync_interval` parameters at their default values to optimize performance.
- You must suspend the connection before altering a route parameter with `alter route`. After executing the `alter route` command, you must resume the route for the change to take effect.

Network-based security parameters

- Both ends of a route must use compatible Security Control Layer (SCL) drivers with the same security mechanisms and security features. It is the replication system Administrator's responsibility to choose and set security features for each server. The Replication Server does not query the security features of remote servers before attempting to establish a connection. Connections will fail if security features at both ends of the route are not compatible.
- `alter route` alters network-based security settings for an outgoing connection from Replication Server to a target Replication Server. Security parameters set by `alter route` override default values set by configure replication server.
- If `unified_login` is set to "required," *only* the "sa" user can log in to the Replication Server without a credential. If the security mechanism should fail, the "sa" user can then log in to Replication Server with a password and disable `unified_login`.
- A Replication Server can have more than one security mechanism; each supported mechanism is listed in the *libtcl.cfg* file under SECURITY.
- Message encryption is a costly process with severe performance penalties. In most instances, it is wise to set `msg_confidentiality` "on" only for certain connections. Alternatively, choose a less costly security feature, such as `msg_integrity`.

- You must suspend the connection before altering a security parameter with `alter route`. After you execute `alter route`, resume the route for the change to take effect.

Procedure to alter a route

Note If you are changing a configuration parameter, you only need to suspend the route before executing `alter route`.

- 1 Quiesce the replication system. For more detailed information, refer to the *Replication Server Troubleshooting Guide*.
- 2 Suspend log transfer with `suspend log transfer` at each Replication Server that manages a database with a RepAgent.
- 3 Execute the `alter route` command at the source Replication Server. You may alter as many routes as necessary.
- 4 Resume RepAgent connections to each RSSD and user database using `resume log transfer`.

See the *Replication Server Administration Guide Volume 1* for complete procedures for altering routes.

Permissions

`alter route` requires "sa" permission.

See also

`admin quiesce_check`, `admin quiesce_force_rsi`, `alter connection`, `alter logical connection`, `alter queue`, `configure connection`, `create logical connection`, `create replication definition`, `configure replication server`, `drop logical connection`, `create connection`, `create route`, `drop connection`, `drop route`, `resume log transfer`, `set proxy`, `suspend log transfer`, `suspend route`

alter schedule

Description	Enables or disables a schedule that executes commands.
Syntax	alter schedule <i>sched_name</i> set [on off]
Parameters	<i>sched_name</i> The name of the schedule to alter. set [on off] Enables or disables a schedule. By default, a schedule is on after you create it.
Examples	To disable schedule1 enter: <pre>alter schedule schedule1 set off</pre>
Usage	Enables or disables a schedule in Replication Server.
Permissions	alter schedule requires “sa” permission.
See also	alter connection, drop schedule, configure replication server, create schedule

alter user

Description	Changes a user's password.
Syntax	<pre>alter user <i>user</i> set password {<i>new_passwd</i> null} [verify password <i>old_passwd</i>]</pre>
Parameters	<p><i>user</i></p> <p>The login name whose password you want to modify.</p> <p><i>new_passwd</i></p> <p>The new password. It can be up to 30 characters long and include letters, numerals, and symbols. Case is significant. If the password contains spaces, enclose the password in quotation marks. When you create or alter a user login name, you must specify a password or "null." A null password lets a user log in immediately without being prompted for a password string.</p> <p>verify password <i>old_passwd</i></p> <p>Enter your current password. Users who do not have "sa" permission must enter this clause.</p>
Examples	<p>The user with login name "louise" has changed her own password from "EnnuI" to "somNIfic":</p> <pre>alter user louise set password somNIfic verify password EnnuI</pre>
Usage	<ul style="list-style-type: none">• If your Replication Server uses ERSSD, you can change the ERSSD primary user password using the alter user command:<pre>alter user <i>user</i> set password <i>new_passwd</i></pre>If this user name matches the ERSSD primary user name, ERSSD updates the rs_users table, issues sp_password at ERSSD to change the password, and updates the configuration file line RSSD_primary_pw.• Users with "sa" permission can omit the verify password clause. Other users must provide this clause in order to change their own passwords.
Permissions	alter user requires "sa" permission when altering another user's password.
See also	create user, drop user

assign action

Description	Assigns Replication Server error-handling actions to data server or Replication Server errors received by the DSI thread.
Syntax	<pre>assign action {ignore warn retry_log log retry_stop stop_replication} for <i>error_class</i> to <i>server_error1</i> [, <i>server_error2</i>]...</pre>
Parameters	<p>ignore Instructs Replication Server to ignore the error and continue processing. ignore should be used when the data server error code indicates a successful execution or an inconsequential warning.</p> <p>warn Instructs Replication Server to display a warning message in its log file without rolling back the transaction or interrupting execution.</p> <p>retry_log Instructs Replication Server to roll back the transaction and retry it. The number of retry attempts is set with alter connection. If the error continues after retrying, Replication Server writes the transaction in the exceptions log and executes the next transaction.</p> <p>log Instructs Replication Server to roll back the current transaction, log it in the exceptions log, and then execute the next transaction.</p> <p>retry_stop Instructs Replication Server to roll back the transaction and retry it. The number of retry attempts is set with the alter connection. If the error continues after retrying, Replication Server suspends replication for the database.</p> <p>stop_replication Instructs Replication Server to roll back the current transaction and suspend replication for the database. This action is equivalent to using suspend connection.</p> <p><i>error_class</i> The error class name for which the action is being assigned.</p> <p><i>server_error</i> A data server or Replication Server error number.</p>
Examples	Example 1 Instructs Replication Server to ignore data server errors 5701 and 5703:

```
assign action ignore
for pubs2_db_err_class
to 5701, 5703
```

Example 2 Warns if Replication Server encounters row count errors, which is indicated by error number 5186:

```
assign action warn
for rs_repserver_error_class to 5186
```

If there is a row count error, this error message displays:

```
DSI_SQLDML_ROW_COUNT_INVALID 5186
Row count mismatch for the SQL Statement Replication
command executed on 'mydataserver.mydatabase'. The
command impacted 10 rows but it should impact 15 rows."
```

Usage

- Use assign action to tell Replication Server how to handle errors returned by data servers. This command overrides any action previously assigned to a data server error.
- Execute assign action at the primary site where the create error class was executed.
- Assign actions for an error class before you create any distributions that use the error class. Assigning actions for an active distribution can lead to unpredictable results.
- If a data server error has no action assigned, the default action stop_replication is taken. For Replication Server errors, the default action taken depends on the type of error that occurred. See Table 3-18 on page 197 for a list of supported Replication Server errors and the default actions for these errors.
- Be sure to assign error actions that are appropriate for the error condition. For example, if you assign the ignore action to an error returned by the data server when a begin transaction command fails, the subsequent commit or rollback command may generate an unexpected error.
- Data servers return errors to Replication Server through the Client/Server Interfaces error-handling mechanism. Warnings and error messages are written to the Replication Server log file.
- Replication Server distributes error actions to qualifying sites through the replication system. The changes do not appear immediately because of normal replication system lag time.

Error actions with multiple errors

- When an operation results in multiple errors, Replication Server chooses the most severe action to perform for the set of errors. For example, if one error indicates that a transaction has been rolled back and is assigned the `retry_log` action, and another error indicates that the transaction log is full and is assigned the `stop_replication` action, a transaction that returns both errors causes Replication Server to perform the `stop_replication` action. The severity of the error actions, from least severe to most severe, are as follows:
 1. `ignore`
 2. `warn`
 3. `retry_log`
 4. `log`
 5. `retry_stop`
 6. `stop_replication`

Error actions for `rs_sqlserver_error_class`

- Predefined error actions for Adaptive Servers are provided with the `rs_sqlserver_error_class` error class.
- To assign different error actions in the `rs_sqlserver_error_class`, you must first choose a primary site for the error class. Log into the Replication Server at that site and create the error class using `create error class`.

Error actions for `rs_repserver_error_class`

- Predefined error actions for Replication Server are provided with the `rs_repserver_error_class` error class.
- To assign different error actions to the `rs_repserver_error_class`, you must first choose a primary site for the error class. Log in to the Replication Server at the primary site and create the error class using `create replication server error class`.
- Table 3-18 lists the valid Replication Server errors and their default error actions.

Table 3-18: Updates to Replication Server error class error numbers

server_error	Error message	Default error action	Description
5185	Row count mismatch for the command executed on <code>'dataserver.database'</code> . The command impacted <code>x</code> rows but it should impact <code>y</code> rows.	stop_replication	This message appears if the affected number of rows is different from the expected number of rows, after a command that is not part of SQL Statement Replication, or a stored procedure, or a row change with autocorrection enabled is sent to the data server.
5186	Row count mismatch for the command executed on <code>'dataserver.database'</code> . The command impacted <code>x</code> rows but it should impact <code>y</code> rows.	stop_replication	Row count verification error for SQL statement replication if the affected number of rows is different from what is expected.
5187	Row count mismatch for the autocorrection delete command executed on <code>'dataserver.database'</code> . The command deleted <code>x</code> rows but it should delete <code>y</code> rows.	stop_replication	This message appears if the affected number of rows is different from the expected number of rows, after a delete command is sent to the data server, and if autocorrection is enabled.
5193	You cannot enable autocorrection if SQL Statement Replication is enabled. Either enable SQL Statement Replication only or disable SQL StatementReplication before you enable autocorrection.	stop_replication	Cannot enable autocorrection if SQL statement replication is enabled. Either enable SQL statement replication only or disable SQL statement replication before you enable autocorrection
5203	Row count mismatch on <code>'dataserver.database'</code> . The delete command generated by <code>dsi_command_convert</code> deleted <code>x</code> rows, whereas it should delete <code>y</code> rows.	stop_replication	This message appears if the number of rows deleted is different from the expected number of rows to be deleted.

- For information about `rs_repserver_error_class` see Table 3-24 on page 238.

Displaying error actions

- The `rs_helperror` stored procedure displays the Replication Server error actions mapped to a given data server error number.

Permissions

assign action requires "sa" permission.

See also

alter error class, configure connection, create connection, create error class, drop error class, `rs_helperror`, suspend connection

check publication

Description	Finds the status of a publication and the number of articles the publication contains.
Syntax	<code>check publication <i>pub_name</i> with primary at <i>data_server.database</i></code>
Parameters	<i>pub_name</i> The name of the publication to check. <i>with primary at data_server.database</i> Specifies the location of the primary data. If the primary database is part of a warm standby application, <i>data_server.database</i> is the name of the logical data server and database.
Examples	Checks the status of the publication <code>pubs2_pub</code> , where the primary database is <code>TOKYO_DS.pubs2</code> : <pre>check publication pubs2_pub with primary at TOKYO_DS.pubs2</pre>
Usage	<ul style="list-style-type: none">• Use <code>check publication</code> to find the status of a publication and the number of articles the publication contains. See the <i>Replication Server Administration Guide Volume 1</i> for more information about publications.• Execute <code>check publication</code> at the Replication Server that manages the replicate database or at the Replication Server that manages the primary database.• If you execute <code>check publication</code> at the replicate Replication Server, the publication is checked at the primary Replication Server using the current user name and password. You must have the same login name and password at the primary Replication Server to display current information about the publication.• To check subscription status, use <code>check subscription</code>. See <code>check subscription</code> for more information. <p>Messages returned by <i>check publication</i></p> <ul style="list-style-type: none">• When you execute <code>check publication</code> at a primary or replicate Replication Server, it returns one of these messages: <pre>Publication <i>pub_name</i> for primary database <i>data_server.database</i> is valid. The number of articles in the publication is <i>number_articles</i>.</pre>

Publication *pub_name* for primary database *data_server.database* is invalid. The number of articles in the publication is *number_articles*.

- When you execute `check publication` at a replicate Replication Server, it returns this message if it cannot contact the primary Replication Server:

Failed to get publication information from primary.

Permissions

Any user may execute this command. A user who enters this command at a replicate Replication Server must have the same login name and password in the primary Replication Server.

See also

`check subscription`, `create publication`, `validate publication`

check subscription

Description	Finds the materialization status of a subscription to a replication definition or a publication.
Syntax	<pre>check subscription <i>sub_name</i> for {<i>table_rep_def</i> <i>function_rep_def</i> [publication <i>pub_name</i> database replication definition <i>db_repdef</i>] with primary at <i>data_server.database</i>} with replicate at <i>data_server.database</i></pre>
Parameters	<p><i>sub_name</i> The name of the subscription to check.</p> <p>for <i>table_rep_def</i> Specifies the name of the table replication definition the subscription is for.</p> <p>for <i>function_rep_def</i> Specifies the name of the function replication definition the subscription is for.</p> <p>for publication <i>pub_name</i> Specifies the name of the publication the subscription is for.</p> <p>database replication definition <i>db_repdef</i> Specifies the name of the database replication definition the subscription is for.</p> <p>with primary at <i>data_server.database</i> Specifies the location of the primary data. If the primary database is part of a warm standby application, <i>data_server.database</i> is the name of the logical data server and database. Include this clause only for a subscription for a publication.</p> <p>with replicate at <i>data_server.database</i> Specifies the location of the replicate data. If the replicate database is part of a warm standby application, <i>data_server.database</i> is the name of the logical data server and database.</p>
Examples	<p>Example 1 Checks the status of the subscription <code>titles_sub</code> for the replication definition <code>titles_rep</code>, where the replicate database is <code>SYDNEY_DS.pubs2</code>:</p> <pre>check subscription titles_sub for titles_rep with replicate at SYDNEY_DS.pubs2</pre>

Example 2 Checks the status of the subscription pubs2_sub for the publication pubs2_pub, where the primary database is TOKYO_DS.pubs2 and the replicate database is SYDNEY_DS.pubs2:

```
check subscription pubs2_sub
  for publication pubs2_pub
  with primary at TOKYO_DS.pubs2
  with replicate at SYDNEY_DS.pubs2
```

Usage

- Use check subscription to find the status of a subscription during subscription materialization or dematerialization, or during the process of refreshing a publication subscription. The subscription can be to a table replication definition, function replication definition, or publication.

See the *Replication Server Administration Guide Volume 1* for more information about subscriptions.

- Execute check subscription at the Replication Server that manages the database where the replicate data is to be stored or the Replication Server that manages the primary database.

The results of check subscription differ depending on where the command is executed. If the Replication Server manages both the primary and replicate database, check subscription returns two status messages.

- To check publication status, use check publication. See check publication for more information.
- Refer to the *Replication Server Troubleshooting Guide* for detailed information about monitoring subscriptions using check subscription.

Messages returned by *check subscription*

- When you execute check subscription at a replicate Replication Server, it returns one of these messages.

In a warm standby application, there may be two lines of output showing the status at the active and at the standby replicate database.

INVALID	sub_name doesn't exist
REMOVING	REMOVING subscription sub_name from system tables at the Replicate.
DEMATERIALIZING	Subscription sub_name is DEMATERIALIZING at the Replicate.
VALID	Subscription sub_name is VALID at the Replicate.
VALIDATING	Subscription sub_name is VALIDATING at the Replicate.
MATERIALIZED	Subscription sub_name has been MATERIALIZED at the Replicate.
ACTIVE	Subscription sub_name is ACTIVE at the Replicate.

ACTIVATING	Subscription <i>sub_name</i> is ACTIVATING at the Replicate.
ACTIVATING	Subscription <i>sub_name</i> is ACTIVATING at the Standby of the Replicate.
QCOMPLETE and ACTIVE	Subscription <i>sub_name</i> is ACTIVE at the Replicate and Materialization Queue has been completed.
QCOMPLETE	Materialization Queue for Subscription <i>sub_name</i> has been completed.
ACTIVE and not QCOMPLETE	Subscription <i>sub_name</i> is ACTIVE at the Replicate, but Materialization Queue for it has not been completed.
DEFINED	Subscription <i>sub_name</i> has been defined at the Replicate.

- In addition to the above messages, executing check subscription at a replicate Replication Server may return one of these messages:

ERROR	Subscription <i>sub_name</i> has experienced an unrecoverable error during Materialization or Dematerialization. Please consult the error log for more details.
PENDING	Other subscriptions are being created or dropped for the same replication definition/database. Subscription <i>sub_name</i> will be processed when previous requests are completed.
RECOVERING	Subscription <i>sub_name</i> has experienced a recoverable error during Materialization or Dematerialization. It will be recovered by Subscription Daemon (dSub).

- When you execute check subscription at a primary Replication Server, it returns one of these messages:

INVALID	<i>subscription_name</i> doesn't exist
DEMATERIALIZING	Subscription <i>sub_name</i> is DEMATERIALIZING at the PRIMARY.
VALID	Subscription <i>sub_name</i> is VALID at the PRIMARY.
ACTIVE	Subscription <i>sub_name</i> is ACTIVE at the PRIMARY.
ACTIVATING	Subscription <i>sub_name</i> is ACTIVATING at the PRIMARY.
DEFINED	Subscription <i>sub_name</i> has been defined at the PRIMARY.

Permissions Any user may execute this command.

See also activate subscription, check publication, create subscription, define subscription, drop subscription, validate subscription

configure connection

Description Changes the attributes of a database connection.

Note configure connection is identical in behavior to the alter connection command.

Syntax For syntax information, see alter connection.

Usage For usage information, see alter connection.

configure logical connection

Description Changes attributes of a logical connection.

Note configure logical connection is identical to the alter logical connection command.

Syntax For syntax information, see alter logical connection.

Usage For usage information, see alter logical connection.

configure replication server

Description	Sets characteristics of the Replication Server, including network-based security. Configures ERSSD.
Syntax	<pre>configure replication server { set repserver_param to 'value' set route_param to 'value' set database_param to 'value' set logical_database_param to 'value' set security_param to 'value' set id_security_param to 'value' set security_services [to] 'default'</pre>
Parameters	<p><i>repserver_param</i> The name of a parameter that affects the Replication Server. Refer to Table 3-19 and Table 3-23 for a description of parameters and values.</p> <p><i>value</i> A setting for a configuration parameter.</p>

Table 3-19: Replication Server configuration parameters

repserver_param	value
block_size to 'value' with shutdown	<p>Specifies the maximum queue block size. The queue block size is the number of bytes in a contiguous block of memory used by stable queue structures.</p> <p>Range of values allowed: 16KB, 32KB, 64KB, 128KB, or 256KB</p> <p>Default: 16KB</p> <hr/> <p>Note When you execute the command to change the block size, Replication Server shuts down. You must include the “with shutdown” clause after specifying the block size. You should change this parameter only with the configure replication server command. Doing otherwise corrupts the queues.</p> <hr/> <p>License: Separately licensed under the Advanced Services Option. See “Replication Server – Advanced Services Option,” in Chapter 4, “Performance Tuning” in the <i>Replication Server Administration Guide Volume 2</i> for instructions.</p>
cm_fadeout_time	<p>The number of seconds of idle time before Replication Server closes a connection with the RSSD. A value of -1 specifies that a connection will never be closed.</p> <p>Default: 300 seconds</p> <p>Minimum: 1 second</p> <p>Maximum: 2,147,483,648 seconds</p>

repserver_param	value
cm_max_connections	<p>The maximum number of outgoing connections available to the connection manager. The value must be greater than 0.</p> <p>Default: 64</p>
current_rssd_version	<p>The Replication Server version supported by this RSSD. The Replication Server checks this value at startup.</p> <p>Note Do not change the value for this parameter. This value should only be modified by the <code>rs_init</code> program when you upgrade or downgrade.</p> <p>Default: N/A</p>
deferred_queue_size	<p>The maximum size of an Open Server deferred queue. If Open Server limits are exceeded, increase the maximum size. The value of <code>deferred_queue_size</code> must be greater than 0.</p> <p>Note You must restart the Replication Server for any changes to this parameter to take effect.</p> <p>Default: 2048 on Linux and HPIA32 1024 on other platforms</p>
dist_direct_cache_read	<p>Enables the distributor (DIST) thread to read SQL statements from the Stable Queue Thread (SQT) cache directly. This reduces contention between the inbound and outbound queues, and leads to improved Replication Server performance.</p> <p>Default: on</p> <p>License: Separately licensed under the Advanced Services Option. See “Replication Server – Advanced Services Option,” in Chapter 4, “Performance Tuning” in the <i>Replication Server Administration Guide Volume 2</i>.</p>
ha_failover	<p>Enables or disables Sybase Failover support for new database connections from the Replication Server to Adaptive Servers. Values are:</p> <ul style="list-style-type: none"> • on - Failover is enabled • off - Failover is disabled <p>Default: off</p>
id_server	<p>The name of the ID Server for this Replication Server.</p> <p>Note Do not change the value of this parameter. <code>id_server</code> is set when you run <code>rs_init</code> and should only be modified by the <code>rs_init</code> program when you upgrade or downgrade Replication Server.</p> <p>Default: N/A</p>
init_sqm_write_delay	<p>Write delay for the Stable Queue Manager if queue is being read.</p> <p>Default: 100 milliseconds</p>

repserver_param	value
init_sqm_write_max_delay	<p>The maximum write delay for the Stable Queue Manager if the queue is not being read.</p> <p>Default: 1000 milliseconds</p>
mem_reduce_malloc	<p>Enable to allocate memory in larger chunks, which reduces the number of memory allocations and leads to improved Replication Server performance.</p> <p>Default: off</p> <p>License: Separately licensed under the Advanced Services Option. See “Replication Server – Advanced Services Option,” in Chapter 4, “Performance Tuning” in the <i>Replication Server Administration Guide Volume 2</i>.</p>
memory_limit	<p>The maximum total memory the Replication Server can use, in megabytes. Values for several other configuration parameters are directly related to the amount of memory available from the memory pool indicated by memory_limit. These include md_sqm_write_request_limit, queue_dump_buffer_size, sqt_max_cache_size, sre_reserve, and sts_cachesize.</p> <p>Default: 2,047MB</p> <p>For 32-bit Replication Server:</p> <ul style="list-style-type: none">• Minimum – 0• Maximum – 2,047MB <p>For 64-bit Replication Server:</p> <ul style="list-style-type: none">• Minimum – 0• Maximum – 2,147,483,647 <p>If additional memory allocation exceeds the value for memory_limit that you specify, Replication Server enlarges memory_limit by ten percent each time memory_limit is exceeded. However, the value of memory_limit is restored to the value you set after Replication Server restarts.</p> <p>If the value you set is larger than 2,047MB, downgrading resets the value to 2,047MB to protect against overflow.</p>
minimum_rssd_version	<p>The minimum version of the Replication Server that can use this RSSD. When the current_rssd_version is greater than the version of the Replication Server, this value is checked when the Replication Server is started.</p> <p>Note Do not change the value for this parameter. This value should only be modified by the rs_init program when you upgrade or downgrade.</p> <p>Default: N/A</p>

repserver_param	value
nrm_thread	<p>Enables the NRM thread which Replication Server can use to normalize and pack Log Transfer Language (LTL) commands in parallel with parsing by the RepAgent Executor thread. Parallel processing by the NRM thread reduces the response time of the RepAgent executor thread. The NRM thread is a thread split from RepAgent executor thread.</p> <p>Use the configure replication server command to set nrm_thread to on before you use exec_nrm_request_limit.</p> <p>Default: on</p> <p>License: Separately licensed under the Advanced Services Option. See “Replication Server – Advanced Services Option,” in Chapter 4, “Performance Tuning” in the <i>Replication Server Administration Guide Volume 2</i>.</p>
num_client_connections	<p>The maximum number of incoming client connections allowed. If Open Server limits are exceeded, increase the maximum number. The value must be greater than or equal to 30.</p> <p>Default: 30</p>
num_concurrent_subs	<p>The maximum number of concurrent subscription materialization/dematerialization requests allowed. (Limit applies to atomic and non-atomic materialization only; does not apply to bulk materialization.) Requests over the maximum are fulfilled after other requests have been fulfilled. The minimum value is 1.</p> <p>Default: 10</p>
num_msgqueues	<p>The maximum number of Open Server message queues allowed. If Open Server limits are exceeded, increase the maximum number. The value must be greater than the num_threads setting.</p> <p>Default: 178</p>
num_msgs	<p>The maximum number of Open Server message queue messages allowed. If Open Server limits are exceeded, increase the maximum number.</p> <p>Default: 45,568</p>
num_mutexes	<p>The maximum number of Open Server mutexes allowed. If Open Server limits are exceeded, increase the maximum number. The value must be greater than the num_threads setting.</p> <p>Default: 128</p>
num_stable_queues	<p>The maximum number of stable queues allowed (HP9000 only). Each stable queue uses 32,768 bytes of shared memory. The minimum number of stable queues allowed is 32.</p> <p>Each standby database connection uses an additional 16,384 bytes of shared memory. Every two standby database connections count as one additional stable queue.</p> <p>Default: 32</p>

repserver_param	value
num_threads	<p>The maximum number of Open Server threads allowed. If Open Server limits are exceeded, increase the maximum number. The value must be greater than or equal to 20.</p> <p>Default: 50</p>
oserver	<p>The name of the current Replication Server.</p> <hr/> <p>Note Do not change the value for this parameter. You specified the current Replication Server name when you installed it using rs_init.</p> <hr/> <p>Default: N/A</p>
password_encryption	<p>Indicates if password encryption is enabled/disabled:</p> <ul style="list-style-type: none"> • 1 – encryption enabled • 0 – encryption disabled <p>Default: 0</p>
prev_min_rssd_version	<p>Following an rs_init installation upgrade, this value contains the previous value of minimum_rssd_version.</p> <hr/> <p>Note Do not change the value for this parameter. This value should be modified only by rs_init when you upgrade or downgrade.</p> <hr/> <p>Default: N/A</p>
prev_rssd_version	<p>Following an rs_init installation upgrade, this value contains the previous value of current_rssd_version.</p> <hr/> <p>Note Do not change the value for this parameter. This value should be modified only by rs_init when you upgrade or downgrade.</p> <hr/> <p>Default: N/A</p>
queue_dump_buffer_size	<p>The maximum command length, in bytes, used by the sysadmin dump_queue command. Commands larger than the specified length are truncated. The range is 1000 to 32,768.</p> <p>Default: 1000 bytes</p>
rec_daemon_sleep_time	<p>Specifies the sleep time for the recovery daemon, which handles “strict” save interval messages in warm standby applications and certain other operations.</p> <p>Default: 2 minutes</p>
rssd_error_class	<p>Error class for the RSSD.</p> <p>Default: rs_sqlserver_error_class</p>

repserver_param	value
send_enc_password	<p>Ensures that all Replication Server client connections are made with encrypted passwords—except for the first connection to the RSSD. Values are “on” and “off.”</p> <p>See the <i>Replication Server Administration Guide Volume 1</i> for more information.</p> <p>Default: off</p>
send_timestamp_to_standby	<p>Specifies whether to send timestamp columns to the replicate database when there is no replication definition. Values are on and off.</p> <p>See the <i>Replication Server Administration Guide Volume 1</i> for more information.</p> <p>Default: off</p>
smp_enable	<p>Enables symmetric multiprocessing (SMP). Specifies whether Replication Server threads should be scheduled internally by Replication Server or externally by the operating system. When Replication Server threads are scheduled internally, Replication Server is restricted to one machine processor, regardless of how many may be available. Values are “on” and “off.”</p> <p>Default: on</p>
sqm_cache_enable	<p>Sets server-wide stable queue caching. Values are on and off.</p> <p>Default: on</p>
sqm_cache_size	<p>Sets server-wide stable queue cache size. Enclose the number of pages in single quotes or double quotes. Range is 1 to 512.</p> <p>Default: 16</p>
sqm_page_size	<p>Sets server-wide stable queue page size in blocks per page. Enclose page sizes in single quotes or double quotes. For example, setting page size to 4 instructs Replication Server to write to the stable queue in 64K chunks.</p> <p>Configuring the page size also sets the I/O size of Replication Server. The range is 1 to 64.</p> <p>Default: 4</p>
sqm_recover_segs	<p>Specifies the number of stable queue segments Replication Server allocates before updating the RSSD with recovery QID information.</p> <p>See “Specifying the number of stable queue segments allocated,” in Chapter 4, “Performance Tuning” in the <i>Replication Server Administration Guide Volume 2</i>.</p> <p>Sybase recommends that you increase the value of sqm_recover_segs to improve performance.</p> <p>Default: 1</p> <p>Minimum: 1</p> <p>Maximum: 2,147,483,648</p>

repserver_param	value
sqm_seg_prealloc	Enables or disables segment preallocation. sqm_seg_prealloc is static and requires a server restart. Values are on and off. Default: on
sqm_warning_thr1	Percent of partition segments (stable queue space) to generate a first warning. The range is 1 to 100. Default: 75
sqm_warning_thr2	Percent of partition segments used to generate a second warning. The range is 1 to 100. Default: 90
sqm_warning_thr_ind	Percent of total partition space that a single stable queue uses to generate a warning. The range is 51 to 100. Default: 70
sqm_write_flush	Specifies whether data written to memory buffers is flushed to the disk before the write operation completes. Values are: <ul style="list-style-type: none"> on – data written to the memory buffer is flushed to the disk. off – data written to the memory buffer is not flushed to the disk. dio – enables direct I/O and allows Replication Server to read and write to the disk without file system buffering. Available only in Solaris (SPARC) and Linux. Default: on
sqt_init_read_delay	The length of time an SQT thread sleeps while waiting for an SQM read before checking to see if it has been given new instructions in its command queue. With each expiration, if the command queue is empty, SQT doubles its sleep time up to the value set for sqt_max_read_delay. Default: 1 ms Minimum: 0 ms Maximum: 86,400,000 ms (24 hours)
sqt_max_cache_size	Maximum SQT (Stable Queue Transaction) interface cache memory, in bytes. For 32-bit Replication Server: <ul style="list-style-type: none"> Default – 1,048,576 Minimum – 0 Maximum – 2,147,483,647 For 64-bit Replication Server: <ul style="list-style-type: none"> Default – 20,971,520 Minimum – 0 Maximum – 2,251,799,813,685,247 If the value you set is larger than 2,147,483,647 bytes, downgrading resets the value to 2,147,483,647 bytes to protect against overflow.

repserver_param	value
sqt_max_read_delay	<p>The maximum length of time an SQT thread sleeps while waiting for an SQM read before checking to see if it has been given new instructions in its command queue.</p> <p>Default: 1 ms Minimum: 0 ms Maximum: 86,400,000 ms (24 hours)</p>
sre_reserve	<p>The amount of additional space to allocate for new subscriptions. For example, 100 (100%) means double the current space. The range is 0 to 500.</p> <p>To update the sre_reserve parameter for a replication definition, insert into or update the rs_config system table directly.</p> <p>Default: 0</p>
stats_reset_rssd	<p>Indicates whether RSSD truncates previous sampling data or overwrites it with new information.</p> <p>Values: on – overwrite old sampling data with new information. off – keep previous sampling data.</p> <p>Default: on</p>
stats_sampling	<p>Enables sampling counters.</p> <p>Default: off</p>
stats_show_zero_counters	<p>Specifies whether the admin stats command reports counters with zero observations for a specified sample period.</p> <p>The values are:</p> <ul style="list-style-type: none"> • on – counters with zero observations are reported. • off – counters with zero observations are not reported. <p>Default: off</p>
sts_cachesize	<p>The total number of rows cached for each cached RSSD system table. Increasing this number to the number of active replication definitions prevents Replication Server from executing expensive table lookups.</p> <p>Default: 100</p>
sts_full_cache_system_table_name	<p>Specifies an RSSD system table that is to be fully cached. Fully cached tables do not require access to the RSSD for simple select statements.</p> <p>Default: rs_columns, rs_functions, rs_objects, and rs_repobjs are fully cached. Sybase recommends that you cache these tables to improve performance.</p> <p>See “Caching system tables,” in Chapter 4, “Performance Tuning” in the <i>Replication Server Administration Guide Volume 2</i> for a list of RSSD tables that can be fully cached.</p>
sub_daemon_sleep_time	<p>Number of seconds the subscription daemon sleeps before waking up to recover subscriptions. The range is 1 to 31,536,000.</p> <p>Default: 120 seconds</p>

repserver_param	value
varchar_truncation	Enables truncation of varchar columns at the primary or replicate Replication Server. Set varchar_truncation at the replicate Replication Server when a character set conversion takes place at both Replication Servers. Default: off

route_param

Affects routes. See Table 3-17 on page 184 for a list and description of route parameters. configure replication server sets parameter values for all routes that originate at the source Replication Server.

database_param

Affects connections. See Table 3-15 on page 124 for a list and description of connection parameters. configure replication server sets parameter values for all connections that originate at the source Replication Server.

logical_database_param

Affects logical connections. See Table 3-17 on page 184 for a list and description of parameters. configure replication server sets parameter values for all logical connections that originate at the source Replication Server

security_param

Affects network-based security. See Table 3-20 on page 214 for a list and description of parameters.

Table 3-20: Parameters affecting network-based security

security_param	value
msg_confidentiality	Indicates whether Replication Server sends and receives encrypted data. If set to “required,” outgoing data is encrypted. If set to “not required,” Replication Server accepts incoming data that is encrypted or not encrypted. Default: not_required
msg_integrity	Indicates whether data is checked for tampering. Default: not_required
msg_origin_check	Indicates whether the source of data should be verified. Default: not_required
msg_replay_detection	Indicates whether data should be checked to make sure it has not been intercepted and resent. Default: not_required
msg_sequence_check	Indicates whether data should be checked to make sure it was received in the order sent. Default: not_required

security_param	value
mutual_auth	Indicates whether the remote server must provide proof of identify before a connection is established. Default: not_required
security_mechanism	The name of the third-party security mechanism enabled for the pathway. Default: first mechanism listed in the SECURITY section of <i>libtcl.cfg</i>
send_enc_password	Ensures that all Replication Server client connections are made with encrypted passwords—except for the first connection to the RSSD. Values are “on” and “off.” Default: off
unified_login	Indicates how Replication Server seeks to log in to remote data servers and accepts incoming logins. The values are: <ul style="list-style-type: none"> • “required” – always seeks to log in to remote server with a credential. • “not_required” – always seeks to log in to remote server with a password. Default: not_required
use_security_services	Tells Replication Server whether to use security services. If use_security_services is “off,” no security features take effect. Note This parameter can only be set by configure replication server.
use_ssl	Indicates whether Replication Server is enabled for session-based SSL security. The values are: <ul style="list-style-type: none"> • “on” – Replication Server is enabled for SSL. • “off” – Replication Server is not enabled for SSL. Default: off

id_security_param

Affects network-based security for the ID Server. See Table 3-21 on page 215 for a list and description of these parameters.

Table 3-21: Security parameters for connecting to the ID Server

security_param	value
id_msg_confidentiality	Indicates whether Replication Server sends and receives encrypted data packets. If set to “required,” outgoing data is encrypted. If set to “not required,” Replication Server accepts incoming data that is encrypted or not encrypted. Default: not required
id_msg_integrity	Indicates whether data packets are checked for tampering. Default: not required
id_msg_origin_check	Indicates whether the source of data packets should be verified. Default: not required

security_param	value
id_msg_replay_detection	Indicates whether data packets should be checked to make sure they have not been intercepted and resent. Default: not required
id_msg_sequence_check	Indicates whether data packets should be checked to make sure they are received in the order sent. Default: not required
id_mutual_auth	Requires the ID Server to provide proof of identify before Replication Server establishes a connection. Default: not required
id_security_mech	Specifies the name of the supported security mechanism. Supported security mechanisms are listed under SECURITY in the <i>libtcl.cfg</i> file. If no name is specified, Replication Server uses the default mechanism. Default: the first mechanism in the list
id_unified_login	Indicates how Replication Server seeks to connect to ID Server. The values are: required – always seeks to log in to ID Server with a credential. not required – always seeks to log in to ID Server with a password. Note Only the “sa” user can log in to Replication Server without a credential if unified_login is “required.” If the security mechanism should fail, the “sa” user can log in and disable unified_login. Default: not required

set security_services [to] 'default'

Resets all network-based security features for the connection to match the global settings of your Replication Server. It does not reset the use_security_services feature.

If Replication Server supports more than one security mechanism, set security_services [to] 'default' also sets the security mechanism to the default, the first mechanism listed in the SECURITY section of the *libtcl.cfg* file.

Examples

Example 1 Sets Replication Server to send data in encrypted format:

```
configure replication server
set id_msg_confidentiality to 'required'
```

Example 2 Sets all security features to match the global settings:

```
configure replication server
set security_services to 'default'
```

Example 3 Changes the rsi_save_interval parameter to two minutes for all routes originating at the current Replication Server:

```
suspend route to each_dest_replication_server

configure replication server
set rsi_save_interval to '2'

resume route to each_dest_replication_server
```

Example 4 Sets the queue block size to 64.

```
configure replication server
set block_size to '64' with shutdown
```

Usage

- Each parameter has two values: the configured value and the run value. Replication Server uses the configured value when it restarts. The run value is the value the Replication Server is using currently. When you start Replication Server the values are equal.
- Configured values are stored in the rs_config system table in the RSSD. For a description of the table, see rs_config in Chapter 8, “Replication Server System Tables.”
- Replication Server shuts down automatically when you set the queue block size with the “set block_size to ‘block_size’ with shutdown” Replication Server parameter. The new block size takes effect after you restart Replication Server. See “Increasing queue block size,” in Chapter 4, “Performance Tuning” in the *Replication Server Administration Guide Volume 2*.
- varchar_truncation enables truncation of varchar columns at the primary or replicate Replication Server. When incoming varchar data exceeds the column length specified in the replication definition, the following occurs:

Table 3-22: varchar_truncation

	varchar_truncation set at primary Replication Server	varchar_truncation set at replicate Replication Server
varchar_truncation set to “on”	Replication Server truncates incoming data to the length specified in the replication definition.	Replication Server truncates incoming data to the length specified in the replication definition.
varchar_truncation set to “off”	RepAgent prints a message in the Replication Server log, and Replication Server ignores rows that exceed the column length specified in the replication definition.	Replication Server prints a message in the Replication Server log, and the DSI shuts down.

- Use ha_failover to enable Sybase failover support. In the event of an ASE server failover, all connections from Replication Server to ASE will fail. Replication Server will retry connections. Setting ha_failover to on will allow the new connections to failover to the new ASE server.

- Use ERSSD configuration parameters to configure backup time, directory location and RepAgent name.

Table 3-23: ERSSD configuration parameters

ERSSD configuration parameter	Value	Default
erssd_backup_start_time	Time the backup starts. Specified as: “hh:mm AM” or “hh:mm PM”, using a 12-hour clock, or “hh:mm” using a 24-hour clock.	Default: 01:00 AM
erssd_backup_start_date	Date the backup begins. Specified as “MM/DD/YYYY”.	Default: current date
erssd_backup_interval	Interval between backups of database and log. Specified as “nn hours” or “nn minutes” or “nn seconds”.	Default: 24 hours
erssd_backup_dir	Location of stored backup files. Should be a full directory path. Configuring this path causes immediate backup.	Default: Same directory as the transaction log mirror; initial value specified in rs_init.
erssd_ra	Configures Replication Agent name, in order to create a route from the current site to another Replication Server. This server name must exist in the interfaces name.	<i>erssd_name_ra</i>

Replication Server parameters

- Replication Server parameters specify default values that affect the local Replication Server.
- Replication Server parameters are static. You must restart Replication Server for them to take effect.

Route parameters

- Route parameters specify default values for all routes that originate at the source Replication Server.
- You can override default values specified using configure replication server by using alter route to set values for individual routes.
- You must suspend all routes originating at the current Replication Server before executing the configure replication server command. After you have changed the parameter, you must resume all routes for the change to take effect.

Database parameters

- Database parameters specify default values for all connections that originate at the source Replication Server.
- You can override default values specified using `configure replication server` by using `alter connection` to set values for an individual connection.
- You must suspend all connections originating at the current Replication Server before executing `configure replication server`. After you change the parameter, resume all connections for the change to take effect.

Logical database parameters

- Logical database parameters specify default values for logical connections that originate at the source Replication Server.
- You can override default values specified using `configure replication server` by using `configure logical connection` to set values for a specific logical connection.
- Logical database parameters are dynamic. They take effect immediately.

Network-based security parameters

- With the exception of `use_security_services` and `use_ssl`, security parameters configured with `configure replication server` are dynamic; they take effect immediately.
- `use_security_services` and `use_ssl` are static. If you change their values, you must restart Replication Server for the change to take effect.
- Default network-based security parameters set with `configure replication server` specify values for all incoming and outgoing pathways related to the current Replication Server.
- You can override default security settings specified using `configure replication server` by using `alter route` or `alter connection` to reset security values for individual *outgoing* pathways.
- If `unified_login` is set to “required,” *only* the “sa” user can log in to the Replication Server without a credential. If the security mechanism should go down, the “sa” user can log in to Replication Server with a password and disable `unified_login`.
- A Replication Server can support more than one security mechanism. Each supported mechanism is listed in the *libtcl.cfg* file under SECURITY.

- Both ends of a route must use compatible Security Control Layer (SCL) drivers with the same security mechanisms and security settings. It is the replication system Administrator's responsibility to choose and set security features for each server. Replication Server does not query the security features of remote servers before it attempts to establish a connection. Network connections fail if security features at both ends of the pathway are not compatible.
- Message encryption is a costly process with severe performance penalties. In most instances, it is wise to set `msg_confidentiality` to "required" only for certain pathways. Alternatively, choose a less costly feature, such as `msg_integrity`, to ensure security.

Permissions

configure replication server requires "sa" permission.

See also

admin security_property, admin security_setting, alter connection, alter route, configure connection, configure route, create connection, create route, set proxy

configure route

Description Changes the attributes of a route from the current Replication Server to a remote Replication Server.

Note `configure route` is identical to the `alter route` command.

Syntax See `alter route` for syntax information.

Usage See `alter route` for usage information.

connect

Description Transforms the Replication Server into a gateway to its RSSD, ID server, to a remote Replication Server, or to a remote data server.

Syntax `connect [to] [rssd | idserver | srv_name | ds_name.db_name]`

Parameters *rssd*
Turns Replication Server into a gateway to its RSSD. Allows the gateway to use *RSSD_primary_user* and *RSSD_primary_pw* entries in its configuration file. *rssd* is the default connect to option.

idserver
Turns Replication Server into a gateway to its ID server, providing that the Replication Server itself is not the ID server. Allows the gateway to use *ID_user* and *ID_pw* entries in the configuration file.

srv_name
The name of the remote Replication Server you want the gateway to connect to. Gateway uses RSI to log into the remote server, and requires a direct route to the remote server.

ds_name.db_name

The name of the remote data server and database that you want the gateway to connect to. The Replication Server gateway uses the maintenance user to log into the remote data server. This allows you to perform tasks that maintenance users of the designated database are permitted to do. However, you cannot access the other databases defined in the data server you connected to.

Replication Server gateway can directly connect to Adaptive Server, and to Sybase® IQ data servers that do not require Enterprise Connect Data Access (ECDA). For other data servers, Replication Server gateway has to connect to use the ECDA to connect the Replication Server and the remote data server.

Examples

Example 1 Creates a gateway connection to the RSSD

ost_relinuxvm_01.emb from the Replication Server ost_relinuxvm_02 by logging into ost_relinuxvm_02 and issuing the connection to command:

```
isql -Usa -P -S ost_relinuxvm_02
1> connect to
2> go
```

Gateway connection to 'ost_relinuxvm_01.emb' is created.

The show server command verifies the connection:

```
1> show server
2> go
```

ost_relinuxvm_01.emb

Example 2 Connects to Replication Server ost_relinuxvm_03 from the Replication Server ost_relinuxvm_02:

```
isql -Usa -P -S ost_relinuxvm_02
1> connect to ost_relinuxvm_03
2> go
```

The show server command verifies the connection:

```
1> show server
2> go
ost_relinuxvm_03
```

Example 3 Creates a gateway connection to the Adaptive Server
ost_replinuxvm_01.pdb:

```
isql -Usa -P -S ost_replinuxvm_02
1> connect to ost_replinuxvm_01.pdb1
2> go
Gateway connection to 'ost_replinuxvm_01.pdb1' is
created.

1> select db_name()
2> go
-----
pdb1
(1 row affected)
```

Usage

- Issuing the connect command requires an sa role for the first log in to Replication Server.
- Issuing the connect command without specifying an option creates a gateway connection to the RSSD.
- When acting as a gateway, Replication Server uses your RSSD primary user name and password to log in to RSSD, your ID server user name and password to log in to ID Server, and your Remote Server Identification (RSI) to log in to a remote Replication Server. You do not need to supply this information more than once, when you access Replication Server itself.
- Cascaded connections created in the gateway are kept in a connection stack, with the Replication Server that issued the first connect command placed at the bottom of the stack.
- Replication Server cannot directly connect to itself. However, you can work around this by using a cascading connection.
- When using Replication Server Gateway, the client and the server must use the same locale set because Replication Server cannot perform character set conversion.

Permissions

Transforming the Replication Server into a gateway requires “sa” permission.

See also

disconnect, show connection, show server

create applied function replication definition

Description	Creates an applied function replication definition and a user-defined function for a stored procedure that is to be replicated. The applied function is applied at the replicate database by the maintenance user.
Syntax	<pre>create applied function replication definition <i>repdef_name</i> with primary at <i>dataserver.database</i> [with all functions named '<i>func_name</i>' [[with primary function named '<i>func_name</i>'] [with replicate function named '<i>func_name</i>']] ([<i>@param_name datatype</i> [, <i>@param_name datatype</i>]...]) [searchable parameters (<i>@param_name</i> [, <i>@param_name</i>]...)] [send standby {all replication definition} parameters]</pre>
Parameters	<p><i>repdef_name</i> The applied function replication definition name. The name must conform to the rules for identifiers.</p> <p>with primary at Specifies the primary data server and the primary database.</p> <p><i>dataserver</i> The name of the data server containing the primary data. If the primary database is part of a warm standby application, <i>dataserver</i> is the logical data server name.</p> <p><i>database</i> The name of the database containing the primary data. If the primary database is part of a warm standby application, <i>database</i> is the logical database name.</p> <p>with all functions named Specifies the stored procedure name at the primary and replicate databases.</p> <p>'<i>func_name</i>' The function name. <i>func_name</i> is a character string with a maximum length of 255 characters.</p> <p>with primary function named Specifies the stored procedure name at the primary database. with primary function named allows you to specify a name for the primary function that is different from the replication definition name. If you do not specify a primary function name, Replication Server uses the replication definition name as the name of the primary function.</p>

with replicate function named

Specifies the name of the stored procedure to execute at the replicate database. If you do not specify a replicate function name, Replication Server uses the replication definition name as the name of the replicate function.

@param_name

A parameter name from the function. A parameter name cannot appear more than once in the clause in which it appears. You are not required to include parameters and their datatypes, but you *must* include a pair of parentheses, whether or not you include any parameters.

datatype

The datatype of a parameter in the function. See “Datatypes” on page 21 for a list of the datatypes and their syntax. Adaptive Server stored procedures and function replication definitions cannot contain parameters with the text, unitext, rawobject, and image datatypes.

searchable parameters

Specifies a list of parameters that can be used in where clauses of define subscription, create subscription, or create article. You *must* include a pair of parentheses if you include the searchable parameters clause.

send standby

In a warm standby application, specifies whether to send to a standby database, all the parameters in the function (send standby all parameters) or only those specified in the replication definition (send standby replication definition parameters). The default is send standby all parameters.

Examples

Example 1 Creates an applied function replication definition named titles_frep for a function of the same name. The primary data is in the pubs2 database of the LDS data server:

```
create applied function replication definition titles_frep
with primary at LDS.pubs2
(@title_id varchar(6), @title varchar(80), @type char(12), @pub_id char(4),
 @price money, @advance money, @total_sales int)
searchable parameters (@title_id, @title)
```

Example 2 Creates an applied function replication definition named titles_frep for a function of the same name. The stored procedure is named upd_titles in the replicate database:

```
create applied function replication definition titles_frep
with primary at LDS.pubs2
with replicate function named 'upd_titles'
(@title_id varchar(6), @title varchar(80), @type char(12), @pub_id char(4),
 @price money, @advance money, @total_sales int)
```

```
searchable parameters (@title_id, @title)
```

Example 3 Creates an applied function replication definition named `titles_frep` for a function named `upd_titles_prim`. The stored procedure is named `upd_titles_prim` in the primary database and `upd_titles` in the replicate database:

```
create applied function replication definition titles_frep
with primary at LDS.pubs2
with primary function named 'upd_titles_prim'
with replicate function named 'upd_titles'
(@title_id varchar(6), @title varchar(80), @type char(12), @pub_id char(4),
 @price money, @advance money, @total_sales int)
searchable parameters (@title_id, @title)
```

Usage

- Use `create applied function replication definition` to describe a stored procedure that you want to replicate. The difference between the applied function replication definition and the request function replication definition is that the function replicated through applied function replication definition is executed at the replicate site by the maintenance user while the function replicated through request function replication definition is executed at the replicate site by the same user who executes the primary function at the primary site. For an overview of replicated stored procedures, see the *Replication Server Administration Guide Volume 1*.
- When you create an applied function replication definition for a primary function, make sure that the function *does not* already have an existing function replication definition that satisfies both these conditions:
 - Was created using the `create function replication definition` command
 - The function replication definition is used for the request function replication without subscription in Replication Server 15.0.1 and earlier version

If these conditions are both true, the existing request function replication definition is disabled. See the *Replication Server Administration Guide Volume 2* for more information about applied function replication definition in Replication Server 15.0.1 and earlier.

- Execute `create applied function replication definition` at the Replication Server that manages the database where the primary data is stored.
- Before executing `create applied function replication definition`, be sure that:
 - The function replication definition name is unique in the replication system. Replication Server cannot always enforce this requirement when you use `create applied function replication definition`.

- A connection exists between the Replication Server and the primary database. See `create connection`.

You can also create connections using `rs_init`. For more information, see the *Replication Server Installation Guide* and the *Replication Server Configuration Guide* for your platform.

- The name, parameters, and datatypes you specify for the function replication definition must match those of the stored procedure involved. Only the parameters specified in the function replication definition are replicated.
- Unlike replicated stored procedures associated with table replication definitions, stored procedures associated with function replication definitions are not required to update a table. This allows you to replicate transactions that are not associated with replicated data.

For more information about stored procedures, see Chapter 6, “RSSD Stored Procedures.” For more information about the two types of replicated stored procedure, see `sp_setrepproc` on page 571.

- Replication Server distributes the new function replication definition to qualifying sites through the replication system. The changes do not appear immediately at all qualifying sites because of normal replication system lag time.

User-defined functions and function strings

- When you create an applied function replication definition, Replication Server automatically creates a corresponding user-defined function. Similarly, in `rs_sqlserver_function_class`, Replication Server automatically creates a default function string for the user-defined function.
- You can customize the function string in `rs_sqlserver_function_class` and in user-defined function-string classes using `create function string`.
- For each user-defined base function-string class in which the user-defined function is used, and for each derived class that inherits from the base function-string class, use `create function string` to create a function string. The function string should invoke a stored procedure or RPC, with language appropriate for the replicate data server.
- For an overview of function-string classes, function strings, and functions, see the *Replication Server Administration Guide Volume 2*.

with primary at clause

Use the *with primary at* clause to specify the primary data server and database. The primary database is the database that contains the invoked stored procedure.

with replicate function named clause

Use the *with replicate function named* clause to specify the name of the stored procedure to execute at the replicate database. If you do not use *with replicate function named* when you create or alter the function replication definition, the function is delivered as a stored procedure with the same name as the function replication definition. In a warm standby database, the stored procedure has the same name as in the active database and *with replicate function named* is ignored.

A round-trip replication enables a database to send a data change request to another database and to replicate the data change back to the requesting database. See the *Replication Server Administration Guide Volume 1* for more information about how to set up a round-trip replication with both applied and request function replication definitions.

Applied function replication definitions for HDS parameters

Although you cannot create function replication definitions that change the datatype of a parameter's value, you can use HDS datatype definitions to declare parameters of applied function replication definitions. The declared parameters are subjected to class-level translations.

See the *Replication Server Administration Guide Volume 1* for more information about HDS.

Altering function replication definitions

- Use *alter applied function replication definition* to add parameters or searchable parameters to an existing applied function replication definition. You can also specify a different replicate name for the function.
- To remove or rename parameters in function replication definition, drop all subscriptions to the function replication definition. After dropping the subscriptions, drop the function replication definition and re-create it.

Subscribing to function replication definitions

To subscribe to an applied function replication definition, use *create subscription* with the *without materialization* clause, or use *define subscription* and the other commands involving bulk materialization.

Function replication definitions and table replication definitions

- When replicating stored procedures using applied functions, create table replication definitions and subscriptions for the tables that are affected by the replicated stored procedures. This ensures that normal transactions and stored procedure executions that affect the tables are replicated. However, if a DML is inside a stored procedure that is marked as replicated, the DML is not replicated. In this case, subscribe to the stored procedure even if you have already subscribed to the table.
- If you plan to use a function replication definition and a table replication definition for the same table, you can materialize the table data with the subscription for the table replication definition. You can then create the subscription for the function replication definition using `create subscription` with the `without materialization` clause.

Creating multiple replication definitions

- You can create multiple applied function replication definitions for the same primary function, and customize each one, so that it can be subscribed to by a different replicate function. See the *Replication Server Administration Guide Volume 1* for details.
- Different applied function replication definitions created for the same primary function *must* use the same parameter with same name and the same datatype.
- If an applied function replication definition specifies different names for the replication definition and the primary function, only Replication Server version 15.1 or later can subscribe to it.
- The same primary function can have applied function replication definitions or request function replication definitions, but not both. The function replication definition created with the `create function replication` command is considered as an applied function at the primary Replication Server where the function replication definition is created.
- In a warm standby database, the stored procedure has the same name as the active database, and the `with replicate function` clause is ignored. If one of the applied function replication definitions is created with the `send standby replication definition parameters` clause, the parameters specified in the function replication definition are delivered to the standby database. Otherwise, all of the parameters in the primary function are delivered.

- In an MSA environment, if a function replication definition for a primary function created with the send standby clause does not exist, the function delivered to the replicate database has the same name as the primary function with all the primary function's parameters. Otherwise, the function delivered to the replicate database takes the name specified in the with replicate function named clause of the function replication definition, and includes parameters specified in the same function replication definition.

Permissions

create applied function replication definition requires “create object” permission.

See also

alter function string, alter applied function replication definition, alter request function replication definition, create connection, create function string, create request function replication definition, define subscription, drop function replication definition, sp_setrepproc, rs_send_repserver_cmd

create article

Description Creates an article for a table or function replication definition and specifies the publication that is to contain the article.

Syntax

```
create article article_name
    for pub_name
with primary at data_server.database
with replication definition {table_rep_def | function_rep_def}
    [where {column_name | @param_name}
        {< | > | >= | <= | = | &} value
    [and {column_name | @param_name}
        {< | > | >= | <= | = | &} value]...
    [or where {column_name | @param_name}
        {< | > | >= | <= | = | &} value
    [and {column_name | @param_name}
        {< | > | >= | <= | = | &} value]...]
```

Parameters

article_name
A name for the article. It must conform to the rules for identifiers and be unique within the publication.

for *pub_name*
The name of the publication that contains the article.

with primary at *data_server.database*
Specifies the location of the primary data. If the primary database is part of a warm standby application, *data_server.database* is the name of the logical data server and database.

with replication definition *table_rep_def*
Specifies the name of the table replication definition the article is for.

with replication definition *function_rep_def*
Specifies the name of the function replication definition the article is for.

where

Sets criteria for the column or parameter values to be replicated via a subscription to the publication that contains this article. If no where clause is included, all rows or parameters are replicated.

A where clause is composed of one or more simple comparisons, where a searchable column or searchable parameter is compared to a literal value with one of the following relational operators: <, >, <=, >=, =, or &. (The & operator is supported *only* for rs_address columns or parameters.) You can join comparisons with the keyword and.

Column or parameter names used in a where clause must also be included in the searchable columns list of the table replication definition or the searchable parameters list of the function replication definition.

You can include multiple where clauses in an article, separated with the keyword or.

The maximum size of a where clause in an article is 255 characters.

column_name

A column name from the primary table, for an article that contains a table replication definition.

@param_name

A parameter name from a replicated stored procedure, for an article that contains a function replication definition.

value

A value for a specified column or parameter. See “Datatypes” on page 21 for entry formats for values for different datatypes.

Column or parameter names used in the expression must be included in the searchable columns or searchable parameters list of the replication definition.

Examples

Example 1 Creates an article called titles_art for the publication pubs2_pub, based on the replication definition titles_rep:

```
create article titles_art
for publication pubs2_pub
with primary at TOKYO_DS.pubs2
with replication definition titles_rep
```

Example 2 Creates an article called titles_art for the publication pubs2_pub, as in the previous example. This command includes a where clause that replicates only the rows for popular computing books, for which the type column is set to “popular_comp”:

```
create article titles_art
  for publication pubs2_pub
  with primary at TOKYO_DS.pubs2
  with replication definition titles_rep
  where type = 'popular_comp'
```

Example 3 Creates an article called `titles_art` for the publication `pubs2_pub`, as in the previous examples. This command includes two `where` clauses that together replicate the rows for both popular computing books and traditional cookbooks:

```
create article titles_art
  for publication pubs2_pub
  with primary at TOKYO_DS.pubs2
  with replication definition titles_rep
  where type = 'popular_comp'
  or where type = 'trad_cook'
```

Usage

- Use `create article` to specify a replication definition for which you want to replicate data using a specified publication. Optional `where` clauses help determine which data is replicated.
- Execute `create article` at the Replication Server that manages the database where the primary data is stored.
- Using `create article` automatically invalidates the publication the article is for. You cannot create new subscriptions until you validate the publication. You cannot replicate data for the new articles until you refresh the subscription.
- For more information about working with replication definitions, articles, and publications, see the *Replication Server Administration Guide Volume I*.

For more information about subscribing to publications, refer to Chapter 10, “Managing Subscriptions,” in the same book.

- Replication Server distributes information about a publication and its articles to a replicate site only when you create or refresh a subscription for the publication.

Requirements for using *create article*

- Before executing `create article`, make sure that:
 - The publication for which you are creating the article already exists.
 - The replication definition for the article already exists.

Adding articles to a new publication

- After you create a publication, you use `create article` to create articles and assign them to the publication. An article specifies a table replication definition or function replication definition and a parent publication. Optionally, it may also include `where` clauses according to the needs of the subscribing replicate site.

A publication must contain at least one article before it can be validated and before you can create subscriptions for it. See `create publication` for more information.

Articles and subscriptions

- When you create a subscription for a publication, Replication Server creates an internal subscription for each of its articles.
- Including multiple `where` clauses for an article, separated by the `or` keyword, allows you to work around the Replication Server restriction that allows only one `where` clause per subscription. A publication subscription cannot include a `where` clause—use `where` clauses in the articles instead.

Adding articles to a publication with a subscription

- If you add a new article to an existing publication, or drop an article from the publication, the publication is invalidated. Although replication for existing articles continues unaffected, in order to begin replication for the new articles you must:
 - Validate the publication when you finish making changes to the publication, then
 - Refresh the publication subscription.

See `create subscription` and `define subscription` for more information on the two methods of refreshing publication subscriptions. See also `validate publication`.

Permissions

`create article` requires “create object” permission.

See also

`check publication`, `create applied function replication definition`, `create publication`, `create replication definition`, `create request function replication definition`, `create subscription`, `define subscription`, `drop article`, `drop publication`, `validate publication`

create connection

Description	Adds a database to the replication system and sets configuration parameters for the connection. To create a connection for an Adaptive Server database, use Sybase Central or <code>rs_init</code> . To create a connection for a non-Adaptive Server database, see <code>create connection using profile</code> .
Syntax	<pre>create connection to <i>data_server.database</i> set error class [to] <i>error_class</i> set function string class [to] <i>function_class</i> set username [to] <i>user</i> [set password [to] <i>passwd</i>] [set replication server error class [to] <i>rs_error_class</i>] [set <i>database_param</i> [to] 'value' [set <i>database_param</i> [to] 'value']...] [set <i>security_param</i> [to] 'value' [set <i>security_param</i> [to] 'value']...] [with {log transfer on, dsi_suspended}] [as active for <i>logical_ds.logical_db</i> as standby for <i>logical_ds.logical_db</i> [use dump marker]]</pre>
Parameters	<p><i>data_server</i> The data server that holds the database to be added to the replication system.</p> <p><i>database</i> The database to be added to the replication system.</p> <p><i>error_class</i> The error class that is to handle errors for the database.</p> <p><i>function_class</i> The function string class to be used for operations in the database.</p> <p><i>user</i> The login name of the Replication Server maintenance user for the database. Replication Server uses this login name to maintain replicated data. You must specify a user name if network-based security is not enabled.</p> <p><i>passwd</i> The password for the maintenance user login name. You <i>must</i> specify a password unless a network-based security mechanism is enabled.</p> <p><i>rs_error_class</i> The error class that handles Replication Server errors for a database. The default is <code>rs_repserver_error_class</code>.</p> <p><i>database_param</i> A parameter that affects database connections from the Replication Server. Parameters and values are described in Table 3-15 on page 124.</p>

value

A character string that contains a value for the option.

security_param

A parameter that affects network-based security. See Table 3-28 on page 313 for a list and description of security parameters that you can set with `create connection`.

`log transfer on`

Indicates that the connection may be a primary data source or the source of replicated functions. When the clause is present, Replication Server creates an inbound queue and is prepared to accept a RepAgent connection for the database. If you omit this option, the connection cannot accept input from a RepAgent.

`dsi_suspended`

Starts the connection with the DSI thread suspended. You can resume the DSI later. This option is useful if you are connecting to a non-Sybase data server that does not support Replication Server connections.

`as active for`

Indicates that the connection is a physical connection to the active database for a logical connection.

`as standby for`

Indicates that the connection is a physical connection to the standby database for a logical connection.

`logical_ds`

The data server name for the logical connection.

`logical_db`

The database name for the logical connection.

`use dump marker`

Tells Replication Server to apply transactions to a standby database after it receives the first dump marker after the enable replication marker in the transaction stream from the active database. Without this option, Replication Server applies transactions it receives after the enable replication marker.

Note If you are using the cross platform dump and load XPDL feature in an MSA replication, do not use the `use dump marker` clause for materialization.

Examples

Example 1 Creates a connection for the pubs2 database in the SYDNEY_DS data server. Replication Server will use the ansi_error error class to handle errors for the database. It will use the function strings in the sqlserver_derived_class function string class for data manipulation operations. The connection will use the pubs2_maint login name with the password pubs2_maint_ps to log into the pubs2 database:

```
create connection to SYDNEY_DS.pubs2
set error class ansi_error
set function string class sqlserver_derived_class
set username pubs2_maint
set password pubs2_maint_pw
```

Example 2 Creates a connection similar to the first example. However, in this example, the tokyo_rs_error Replication Server error class handles the Replication Server errors for the connection and the with log transfer clause is specified. This allows the connection to accept input from a RepAgent. The connection is with a database that contains primary data or that will be a source of replicated functions:

```
create connection to TOKYO_DS.pubs2
set error class ansi_error
set function string class sqlserver_derived_class
set username pubs2_maint
set password pubs2_maint_pw
set replication server error class tokyo_rs_error
with log transfer on
```

Usage

- Use create connection to add a database to the replication system. Normally, you use this command to add connections to non-Sybase databases. To create a standard connection with an Adaptive Server database, use Sybase Central or rs_init.
- To create a connection that uses heterogeneous datatype support (HDS) to translate datatypes from the primary to the replicate database, you can also use scripts provided by Sybase that both create the connection and install HDS. See the *Replication Server Configuration Guide* for your platform for instructions.
- Execute create connection at the Replication Server that manages the database.
- Replication Server distributes database connection information to qualifying sites through out the replication system. The changes do not appear immediately at all sites because of normal replication system lag time.

- You must specify an error class, even if you use the default error class: `rs_sqlserver_error_class`.
- You do not have to specify a Replication Server error class unless it is a new Replication Server error class. The default Replication Server error class is `rs_repserver_error_class`.
- Only one connection is allowed per database. This is enforced by the ID Server, which registers each database in its `rs_idnames` system table. The ID Server must be available when you create a connection for a database.
- Use set function string class [to] *function_class* to activate class-level translations for non-Sybase data servers.

Database connection parameters

- Replication Server configuration parameters are stored in the `rs_config` system table. See the *Replication Server Administration Guide Volume 1* for more information about the database connection parameters in the `rs_config` system table.
- See the *Replication Server Administration Guide Volume 2* for more information about configuring parallel DSI threads.
- Use assign action to enable retry of transactions that fail due to specific data server errors.

The *dump_load* configuration parameter

- Before setting *dump_load* to “on,” create function strings for the `rs_dumpdb` and `rs_dumptran` functions. Replication Server does not generate function strings for these functions in the system-provided classes or in derived classes that inherit from these classes.

The *save_interval* configuration parameter

- Set *save_interval* to save transactions in the DSI queue that can be used to resynchronize a database after it has been restored from backups. Setting a save interval is also useful when you set up a warm standby of a database that holds replicate data or receives replicated functions. You can use `sysadmin restore_dsi_saved_segments` to restore backlogged transactions.

Error classes and function classes

- Table 3-24 shows the error and function classes that Replication Server provides for Replication Server and database connections.

Table 3-24: Error and function classes

Class name	Description
<code>rs_repserver_error_class</code>	Error action assignments for Replication Server.

Class name	Description
rs_sqlserver_error_class	Error action assignments for Adaptive Server databases.
rs_sqlserver_function_class	Function-string class for Adaptive Server databases. Cannot participate in function string inheritance. Replication Server generates function strings automatically.
rs_default_function_class	Function-string class for Adaptive Server databases. You cannot modify function strings. You can specify this class as a parent class, but not as a derived class. Replication Server generates function strings automatically.
rs_db2_error_class	Error class for DB2 databases.
rs_db2_function_class	Function-string class for DB2 databases. You cannot modify function strings. You can specify this class as a parent class, but not as a derived class. Replication Server generates function strings automatically.
rs_iq_error_class	Error class for Sybase IQ databases.
rs_iq_function_class	Function-string class for Sybase IQ Oracle databases. You cannot modify function strings. You can specify this class as a parent class, but its derived classes cannot inherit any class-level translations from the parent class. Replication Server generates function strings automatically.
rs_mssql_error_class	Error class for Microsoft SQL Server databases.
rs_ms_function_class	Function-string class for Microsoft SQL Server databases. You cannot modify function strings. You can specify this class as a parent class, but its derived classes cannot inherit any class-level translations from the parent class. Replication Server generates function strings automatically.
rs_oracle_error_class	Error class for Oracle databases.
rs_oracle_function_class	Function-string class for Oracle databases. You cannot modify function strings. You can specify this class as a parent class, but its derived classes cannot inherit any class-level translations from the parent class. Replication Server generates function strings automatically.
rs_udb_error_class	Error class for UDB databases.
rs_udb_function_class	Function-string class for UDB databases. You cannot modify function strings. You can specify this class as a parent class, but its derived classes cannot inherit any class-level translations from the parent class. Replication Server generates function strings automatically.

Note The `rs_dumpdb` and `rs_dumptran` system functions are not initially defined, even for function-string classes in which Replication Server generates default function strings. If you intend to use coordinated dumps, you must create function strings for these functions. Note also that you cannot perform coordinated dumps on a standby database. See the *Replication Server Administration Guide Volume 2* for more information about using function strings. For more information about the `rs_dumpdb` and `rs_dumptran` functions, see Chapter 4, “Replication Server System Functions.”

User name and password

- You specify the maintenance user login name and password when creating the connection. The maintenance user login name must be granted all necessary permissions to maintain replicated data in the database.

Note When two sites in a replication system have the same database name, the maintenance user login names must be different. The default login name, created by Sybase Central or rs_init is *DB_name_maint*. When setting up the system, change one of the login names so each are unique.

Warm standby applications

- To create a logical connection for a warm standby application, use create logical connection.
- In a warm standby application, the connections for the active database and the standby database must have log transfer on.
- The function-string class for a database in a warm standby application is used only when the database is the active database. Replication Server uses rs_default_function_class for the standby database.

Changing connection attributes

- Use alter connection to change the attributes of a connection.
- If the password of the maintenance user has been changed, use alter connection to enter the new password.

Network-based security parameters

- Both ends of a connection must use compatible Security Control Layer (SCL) drivers with the same security mechanisms and security features. The remote server must support the set proxy or equivalent command. It is the replication system Administrator's responsibility to choose and set security features for each server. The Replication Server does not query the security features of remote servers before attempting to establish a connection. Connections fail if security features at both ends of the connection are not compatible.
- create connection specifies security settings for an outgoing connection from Replication Server to a target data server. Security features set by create connection override those set by configure replication server.

- If `unified_login` is set to “required,” *only* the replication system Administrator with “sa” permission can log in to the Replication Server without a credential. If the security mechanism should fail, the replication system Administrator can log in to Replication Server with a password and disable `unified_login`.
- A Replication Server can have more than one security mechanism; each supported mechanism is listed in the *libtcl.cfg* file under SECURITY.
- Message encryption is a costly process with severe performance penalties. In most instances, it is wise to set `msg_confidentiality` to “required” only for certain connections. Alternatively, choose a less costly security feature, such as `msg_integrity`.

Permissions

create connection requires “sa” permission.

See also

`admin show_connection_profiles`, `alter connection`, `create connection using profile`, `configure connection`, `configure connection`, `create error class`, `create function string class`, `create logical connection`, `alter route`, `drop connection`, `resume connection`, `rs_classes`, `rs_profdetail`, `rs_profile`, `rs_systext`, `suspend connection`

create connection using profile

Description

`create connection using profile` clause uses predefined information to configure the connection between Replication Server and a non-Adaptive Server database, and, if needed, to modify the RSSD and the named *data_server.database*. To create a connection to Adaptive Server, see `create connection`.

Syntax

```
create connection to data_server.database
using profile connection_profile[:version]
set username [to] user
[other_create_connection_options]
[display_only]
```

Parameters

data_server

The data server that holds the database to be added to the replication system.

database

The database to be added to the replication system.

connection_profile

Indicates the connection profile that you want to use to configure a connection, modify the RSSD, and build replicate database objects.

version

Specifies the connection profile version to use.

user

The login name of the Replication Server maintenance user for the database. Replication Server uses this login name to maintain replicated data. You must specify a user name if network-based security is not enabled.

other_create_connection_options

Use the other create connection options to set connection options not specified in the profile, such as setting your password, or to override options specified in the profile, such as specifying a custom function string class to override the function string class provided in Replication Server. For a complete list of the other create connection options, see *create connection*.

display_only

Use *display_only* with the *using profile* clause to display the commands that will be executed and the names of the servers upon which the commands will be executed. See the client and Replication Server logs for the result of using *display_only*.

Examples

Example 1 Creates a connection to an Oracle replicate database:

```
create connection to oracle.instance
using profile rs_ase_to_oracle;standard
set username to ora_maint
set password to ora_maint_pwd
```

Example 2 Creates a connection to a Microsoft SQL Server replicate database that is also a primary database. In this example, the command replaces any error class setting provided by the connection profile with the *my_msss_error_class* error class:

```
create connection to msss_server.msss_db
using profile rs_ase_to_msss
set username to msss_maint
set password to msss_maint_pwd
set error class to my_msss_error_class
with log transfer on
```

Example 3 Creates a connection to a DB2 replicate database using a specific version of the profile—*v9_1*. In this example, the command overrides the command batch size provided by the connection profile with a new value—16384:

```
create connection to db2.subsys
using profile rs_ase_to_db2;v9_1
set username to db2_maint
```

```
set password to db2_maint_pwd
set dsi_cmd_batch_size to '16384'
```

Example 4 Use the `display_only` option to show the commands that will be executed if you use a particular profile. The commands and the command output display on your screen and are also written to the Replication Server log:

```
create connection to oracle.instance
using profile rs_ase_to_oracle;standard
set username to ora_maint
set password to ora_maint_pwd
display_only

go
```

Display only using Connection Profile `rs_ase_to_oracle;standard`.

Command(s) intended for: `prs01`

```
create connection to oracle.instance
  set error class to rs_oracle_error_class
  set function string class to rs_oracle_function_class
  set username to ora_maint
  set password to *****
  set batch to off
```

Command(s) intended for 'edsprs01.edbprs01':

```
delete from rs_translation where classid = 0x0000000001000007 and
                               source_dtid = 0x000000000000000c
```

Command(s) intended for 'edsprs01.edbprs01':

```
insert rs_translation (prsid, classid, type, source_dtid, target_dtid,
                       target_length, target_status, rowtype)
values (0, 0x0000000001000007, 'D', 0x000000000000000c, 0x0000000000010200,
       19, 0, 0)
```

Command(s) intended for 'edsprs01.edbprs01':

```
delete from rs_translation where classid = 0x0000000001000007 and
                               source_dtid = 0x000000000000000d
```

Command(s) intended for 'edsprs01.edbprs01':

```
insert rs_translation (prsid, classid, type, source_dtid, target_dtid,
                       target_length, target_status, rowtype)
values (0, 0x0000000001000007, 'D', 0x000000000000000d, 0x0000000000010200,
       19, 0, 0)
```

Command(s) intended for 'edsprs01.edbprs01':

```
delete from rs_translation where classid = 0x0000000001000007 and
```

```
source_dtid = 0x0000000000000001

Command(s) intended for 'edsprs01.edbprs01':
insert rs_translation (prsid, classid, type, source_dtid, target_dtid,
                      target_length, target_status, rowtype)
values (0, 0x0000000001000007, 'D', 0x0000000000000001, 0x0000000000010202,
       0, 0, 0)
Command(s) intended for 'edsprs01.edbprs01':
delete from rs_translation where classid = 0x0000000001000007 and
                      source_dtid = 0x0000000000000013

Command(s) intended for 'edsprs01.edbprs01':
insert rs_translation (prsid, classid, type, source_dtid, target_dtid,
                      target_length, target_status, rowtype)
values (0, 0x0000000001000007, 'D', 0x0000000000000013, 0x0000000000010202,
       0, 0, 0)

Command(s) intended for 'edsprs01.edbprs01':
delete from rs_translation where classid = 0x0000000001000007 and
                      source_dtid = 0x000000000000000E

Command(s) intended for 'edsprs01.edbprs01':
insert rs_translation (prsid, classid, type, source_dtid, target_dtid,
                      target_length, target_status, rowtype)
values (0, 0x0000000001000007, 'D', 0x000000000000000E, 0x0000000000010205,
       136, 0, 0)

Command(s) intended for 'edsprs01.edbprs01':
delete from rs_translation where classid = 0x0000000001000007 and
                      source_dtid = 0x000000000000000F

Command(s) intended for 'edsprs01.edbprs01':
insert rs_translation (prsid, classid, type, source_dtid, target_dtid,
                      target_length, target_status, rowtype)
values (0, 0x0000000001000007, 'D', 0x000000000000000f, 0x0000000000010205,
       136, 0, 0)

Command(s) intended for 'edsprs01.edbprs01':
delete from rs_translation where classid = 0x0000000001000007 and
                      source_dtid = 0x000000000000001b

Command(s) intended for 'edsprs01.edbprs01':
insert rs_translation (prsid, classid, type, source_dtid, target_dtid,
                      target_length, target_status, rowtype)
values (0, 0x0000000001000007, 'D', 0x000000000000001b, 0x0000000000010201,
       9, 0, 0)
```

```
Command(s) intended for 'edsprs01.edbprs01':
delete from rs_translation where classid = 0x0000000001000007 and
                                source_dtid = 0x000000000000001c
```

```
Command(s) intended for 'edsprs01.edbprs01':
insert rs_translation (prsid, classid, type, source_dtid, target_dtid,
                        target_length, target_status, rowtype)
values (0, 0x0000000001000007, 'D', 0x000000000000001c, 0x0000000000010200,
        19, 0, 0)
```

```
Command(s) intended for 'oracle.instance':
drop table rs_info
```

```
Command(s) intended for 'oracle.instance':
commit
```

```
Command(s) intended for 'oracle.instance':
create table rs_info (rskey varchar2 (20), rsval varchar2 (20))
```

```
Command(s) intended for 'oracle.instance':
commit
```

```
Command(s) intended for 'oracle.instance':
insert into rs_info values ('charset_name', 'iso_1')
```

```
Command(s) intended for 'oracle.instance':
insert into rs_info values ('sortorder_name', 'bin_iso_1')
```

```
Command(s) intended for 'oracle.instance':
commit
```

```
Command(s) intended for 'oracle.instance':
drop public synonym rs_lastcommit
```

```
Command(s) intended for 'oracle.instance':
commit
```

```
Command(s) intended for 'oracle.instance':
drop table rs_lastcommit
```

```
Command(s) intended for 'oracle.instance':
commit
```

```
Command(s) intended for 'oracle.instance':
create table rs_lastcommit(origin number(8),origin_qid char(72),
```

```
        secondary_qid char(72),origin_time  date,
        dest_commit_time date)
```

```
Command(s) intended for 'oracle.instance':
commit
```

```
Command(s) intended for 'oracle.instance':
grant all on rs_lastcommit to public
```

```
Command(s) intended for 'oracle.instance':
commit
```

```
Command(s) intended for 'oracle.instance':
create public synonym rs_lastcommit for rs_lastcommit
```

```
Command(s) intended for 'oracle.instance':
commit
```

```
Command(s) intended for 'oracle.instance':
CREATE OR REPLACE PROCEDURE
    RS_UPDATE_SEQUENCE(SequenceName VARCHAR2, SequenceValue NUMBER,
                        Increment NUMBER)
AS CurrentID NUMBER; LastID NUMBER; SeqCursor INTEGER; SQLStmt
    VARCHAR2(1024);
Result NUMBER;
BEGIN
SQLStmt := 'SELECT ' || SequenceName || '.NEXTVAL FROM DUAL';
SeqCursor := DBMS_SQL.OPEN_CURSOR;
DBMS_SQL.PARSE(SeqCursor,SQLStmt,DBMS_SQL.NATIVE);
DBMS_SQL.DEFINE_COLUMN(SeqCursor, 1, LastID);
Result := DBMS_SQL.EXECUTE_AND_FETCH(SeqCursor);
DBMS_SQL.COLUMN_VALUE(SeqCursor,1,CurrentID);
LOOP
    IF ( Increment < 0 ) THEN EXIT WHEN CurrentID <= SequenceValue;
        EXIT WHEN CurrentID > LastID;
    ELSE EXIT WHEN CurrentID >= SequenceValue;
        EXIT WHEN CurrentID < LastID;
    END IF;
    LastID := CurrentID;
    Result := DBMS_SQL.EXECUTE_AND_FETCH(SeqCursor);
    DBMS_SQL.COLUMN_VALUE(SeqCursor,1,CurrentID);
END
LOOP;
    DBMS_SQL.CLOSE_CURSOR(SeqCursor);
END;
```

```
Command(s) intended for 'oracle.instance':  
grant all on RS_UPDATE_SEQUENCE to public
```

```
Command(s) intended for 'oracle.instance':  
DROP sequence rs_ticket_seq
```

```
Command(s) intended for 'oracle.instance':  
CREATE sequence rs_ticket_seq
```

```
Command(s) intended for 'oracle.instance':  
Drop table rs_ticket_history
```

```
Command(s) intended for 'oracle.instance':  
CREATE TABLE rs_ticket_history(cnt numeric(8,0), h1 varchar(10,  
    h2 varchar(10), h3 varchar(10), h4 varchar(50), pdb varchar(30),  
    prs varchar(30), rrs varchar(30), rdb varchar(30), pdb_t date,  
    exec_t date, dist_t date, rsi_t date, dsi_t date,  
    rdb_t date default current_date, exec_b int, rsi_b int, dsi_tnx int,  
    dsi_cmd int, ticket varchar(1024))
```

```
Command(s) intended for 'oracle.instance':  
create unique index rs_ticket_idx on rs_ticket_history(cnt)
```

```
Command(s) intended for 'oracle.instance':  
create or replace trigger rs_ticket_tri  
    before insert on rs_ticket_history  
    for each row  
    begin  
        if :new.cnt is null then  
            select rs_ticket_seq.nextval into :new.cnt from dual;  
        end if;  
    end rs_ticket_tri;
```

```
Command(s) intended for 'oracle.instance':  
grant all on rs_ticket_history to public
```

```
Command(s) intended for 'oracle.instance':  
commit
```

Usage

- Connection profiles specify the function-string class and the error class. Connection profiles can also specify other connection options such as whether commands should be batched and what command separator to use. In addition to connection settings, connection profiles can specify class-level translations to install in the RSSD and objects, such as the `rs_lastcommit` table, to be created in the replicate database.

- When you create a connection using a connection profile, the system table services (STS) caches are refreshed so that you do not need to restart Replication Server.
- Always specify the set username clause right after the using profile clause.

See also

admin show_connection_profiles, create connection

create database replication definition

Description Creates a replication definition for replicating a database or a database object.

Syntax

```
create database replication definition db_repdef
    with primary at server_name.db
    [[not] replicate DDL]
    [[not] replicate setname setcont]
    [[not] replicate setname setcont]
    [[not] replicate setname setcont]
    [[not] replicate setname setcont]
    [[not] replicate {SQLDML | DML_options} [in table_list]]

setname ::= {tables | functions | transactions | system procedures}
setcont ::= [[in] ([owner1.]name1[, [owner2.]name2 [, ... ]]])]
```

Note The term functions in *setname* refers to user-defined stored procedures or user-defined functions.

Parameters

db_repdef
Name of the database replication definition.

server_name.db
Name of the primary server/database combination. For example:
TOKYO.dbase.

[[not] replicate DDL
Tells Replication Server whether or not to send DDL to subscribing databases. If “replicate DDL” is not included, or the clause includes “not,” DDL is not sent to the replicate database.

[not] replicate *setname setcont*

Specifies whether or not to send objects stated in the *setname* category to the replicate database. The *setname* category can have a maximum of one clause for tables, one clause for functions, one clause for transactions, and one clause for system procedures.

If you omit the system procedures *setname* or include the not option, Replication Server does not replicate the system procedures.

If you omit tables, functions, or transactions *setname* or include the not option, Replication Server replicates all objects of the *setname* category.

[not] replicate {SQLDML | *DML_options*} [in *table_list*]

Informs Replication Server whether or not to replicate SQL statements to tables defined in *table_list*.

SQLDML

These DML operations:

- U – update
- D – delete
- I – insert select
- S – select into

DML_options

Any combination of these DML operations:

- U – update
- D – delete
- I – insert select
- S – select into

When the database replication mode is set to any combination of UDIS the RepAgent sends both the individual log records and the information needed by Replication Server to build the SQL statement.

owner

An owner of a table or a user who executes a transaction. Replication Server does not process owner information for functions or system procedures.

You can replace *owner* with a space surrounded by single quotes or with an asterisk.

- A space (' ') – indicates no owner.
- An asterisk (*) – indicates all owners. Thus, for example, *.publisher means all tables named publisher, regardless of owner.

name

The name of a table, function, transaction, or system procedure.

You can replace *name* with a space surrounded by single quotes or with an asterisk.

- A space (' ') – indicates no name. For example, maintuser.' ' means all unnamed maintenance user transactions.
- An asterisk (*) – indicates all names. Thus, for example, robert.* means all tables (or transactions) owned by robert.

Examples

Example 1 Creates a database replication definition rep_1B. This database replication definition specifies that only tables employee and employee_address are replicated:

```
create database replication definition rep_1B
with primary at PDS.pdb
replicate tables in (employee, employee_address)
```

Example 2 Creates a database replication definition rep_2. In this example, the database my_db is replicated, DDL is replicated, but system procedures are not replicated:

```
create database replication definition rep_2
with primary at dsA.my_db
replicate DDL
not replicate system procedures
```

Example 3 Replicates insert, update, delete, and select into commands from all the tables in the pdb1 database. All transactions and functions are replicated but DDL and system procedures are not:

```
create database replication definition rep_3
with primary at ds3.pdb1
replicate SQLDML
```

This example has the same result as the preceding example:

```
create database replication definition rep_3
with primary at ds3.pdb1
replicate 'UDSI'
```

Example 4 Filters out the select into statement for all tables. The second clause, not replicate 'U' in (T), filters out updates on table T:

```
create database replication definition dbrepdef
with primary at ds1.pdb1
not replicate 'S'
not replicate 'U' in (T)

go
```

Example 5 Enables update and delete statements on all tables using the replicate 'UD' clause:

```
create database replication definition dbrepdef_UD
with primary at ds2.pdb1
replicate 'UD'

go
```

Example 6 You can use multiple clauses to specify a table multiple times in the same definition. However, you can use each of U, D, I, and S only once per definition:

```
create database replication definition dbrepdef
with primary at ds2.pdb1
replicate tables in (tb1,tb2)
replicate 'U' in (tb1)
replicate 'I' in (tb1,tb2)

go
```

Example 7 A replication definition that replicates all user stored procedures, system procedures, and DML for all the tables in the database except for the table T. For the table T, the replication definition replicates all commands except for the delete command:

```
create database replication definition repdef_7
with primary at ds3.pdb1
replicate functions
replicate system procedures
replicate 'IUS' /* replicate 'IUS' DML for all tables, including */
/* table 'T' */
not replicate 'D' in (T) /* not replicate 'D' DML for table T, but */
/* replicate 'D' for all other tables */
```

Usage

- create database replication definition lets you replicate all, all with some exceptions, or only some of the tables, functions, transactions, and system procedures from the primary database.

- Use create database replication definition alone or in conjunction with table and function replication definitions.
- With only a database replication definition, that is, without table or function replication definitions, Replication Server cannot transform data. However, it can perform minimal column replication. This data replication behavior is similar to that of a default warm standby.

For a database replication definition to replicate encrypted columns without using a table level replication definition, you must define the encryption key for the encrypted column with INIT_VECTOR NULL and PAD NULL. If a table in the database includes encrypted columns where the encryption key was created with random padding (the default) or initialization vectors, a table level replication definition is required to ensure database consistency.

- Database replication definitions are global objects. They are replicated to all Replication Servers that have a route from the defining Replication Server.
- Database replication definitions do not affect request function replication.
- Table and function filters are not implemented if table and function subscriptions exist.
- Replication Server does not process owner information for functions and system procedures.

Owner information

- Replication Server always uses owner information provided in the database replication definition.
- Replication Server does not use owner information provided in a table replication definition if the table is marked with sp_reptostandby.
- Replication Server only uses owner information provided in a table replication definition if the table is marked by sp_setreptable with the owner_on clause.

SQL statement replication

- To replicate SQL statements in an MSA environment, you must include the replicate SQLDML clause in your replication definition.
- You can use multiple replicate clauses in a create database replication definition. However, for an alter database replication definition, you can use only one clause.

- If you do not specify a filter in your replication definition, the default is the not replicate clause. Apply alter database replication definition to change the SQLDML filters. You can either specify one or multiple SQLDML filters in a replicate clause.
- If a table replication definition with send standby clause is defined for a table, the SQL replication settings of the table replication definition overrides the settings defined in the database replication definition for that table.

See also

alter database replication definition, drop database replication definition

create error class

Description	Creates an error class.
Syntax	<code>create [replication server] error class <i>error_class</i></code> <code>[set template to <i>template_error_class</i>]</code>
Parameters	<p><code>replication server</code> Indicates that the new error class is a Replication Server error class and not a data server error class.</p> <p><code>error_class</code> The name for the new error class. The name must be unique in the replication system and must conform to the rules for identifiers.</p>

Note A Replication Server error class and a data server error class cannot share the same name.

`set template to template_error_class`

Use this clause to create an error class based on another error class. `create error class` copies the error actions from the template error class to the new error class.

Examples **Example 1** This example creates a new error class named `pubs2_db_err_class`:

```
create error class pubs2_db_err_class
```

Creates the `my_error_class` error class based on `rs_oracle_error_class`:

```
create error class my_error_class set template to  
rs_oracle_error_class
```

Example 2 Creates a new Replication Server error class named `pubs2_rs_err_class`:

```
create replication server error class  
pubs2_rs_err_class
```

Example 3 Creates the `my_rs_err_class` Replication Server error class based on `rs_repserver_error_class`, which is the default Replication Server error class:

```
create replication server error class my_rs_err_class  
set template to rs_repserver_error_class
```

Usage

- Use `create error class` to create an error class. An error class is a name used to group error action assignments for a database.
- This command has the following requirements:

- Routes must exist from the Replication Server where an error class is created to the Replication Servers managing data servers that are to use the error class.
- The `rs_sqlserver_error_class` is the default error class provided for Adaptive Server databases while the `rs_repserver_error_class` is the default error class provided for Replication Server. Initially, these two error classes do not have a primary site. You must create these error classes at a primary site before you can change the default error actions.
- After using create error class, use the `rs_init_erroractions` stored procedure to initialize the error class.
- Associate an error class with a database using `create connection` or `alter connection`. Each database can have one error class. An error class can be associated with multiple databases.
- Replication Server distributes the new error class to qualifying sites through out the replication system. The changes do not appear immediately at all such sites because of normal replication system lag time.

Assigning error actions

- Use `assign action` to change the Replication Server response to specific data server errors. Actions are assigned at the Replication Server where the error class is created.

Dropping error classes

- Use `drop error class` to remove an error class and any actions associated with it.

Non-Adaptive Server error classes

- You can assign non-Adaptive Server error classes to specific connections on non-Adaptive Server replication databases using the `create connection` and `alter connection` commands.
- When Replication Server establishes a connection to a non-ASE replicate server, Replication Server verifies if the option to return native error codes from the non-ASE replicate server is enabled for the connection. If the option is not enabled, Replication Server logs a warning message that the connection works but error action mapping may not be correct.

See “ReturnNativeError,” in the Replication Server Options documentation to set the option in the Enterprise Connect™ Data Access (ECDA) Option for ODBC for your replicate server.

- For a list of non-Adaptive Server error classes, see Table 3-24 on page 238. For more information about non-Adaptive Server replication error classes, see the *Replication Server Administration Guide Volume 1*.

Permissions

create error class requires “sa” permission.

See also

alter connection, alter error class, assign action, create connection, drop error class, move primary, rs_init_erroractions

create function

Description Creates a user-defined function.

Note When you create a function replication definition, a user-defined function is automatically created. For more information, see *create applied function replication definition* and *create request function replication definition*.

If your application uses asynchronous procedure delivery associated with table replication definitions, you may need to create user-defined functions. For more information, see the *Replication Server Administration Guide Volume 2*.

Syntax `create function replication_definition.function
([@param_name datatype [, @param_name datatype]...])`

Parameters *replication_definition*
The name of the replication definition the function is for. You can create only one user-defined function for all the replication definitions for the same table. If there are multiple replication definitions for the same table, you can specify the name of any one of them. However, each replication definition has its own function string for the user-defined function.

function
The name of the function. The name must be unique for the replication definition and must conform to the rules for identifiers. The names of the system functions listed in Chapter 4, “Replication Server System Functions,” and all function names that begin with “rs_”, are reserved.

@param_name
The name of an argument to the user-defined function. Each parameter name must be preceded by an @ sign and must conform to the rules for identifiers. The value of each parameter is supplied when the function is executed.

datatype
The datatype of the parameter. Some datatypes require a length, in parentheses, after the datatype name. See “Datatypes” on page 21 for a description of the datatypes and their syntax. The datatype cannot be text, unitext, rawobject, or image.

Examples Creates a user-defined function named `newpublishers`, with four parameters, for the `publishers_rep` replication definition:

```
create function publishers_rep.newpublishers
  (@pub_id char(4), @pub_name varchar(40),
   @city varchar(20), @state char(2))
```

Usage

- Use `create function` to create user-defined functions.

- Execute create function on the Replication Server where the replication definition was created.
- User-defined functions may be used in asynchronous procedure delivery. See the *Replication Server Administration Guide Volume 2* for more information about asynchronous procedures.
- You must include the parentheses () surrounding the listed parameters, even when you are defining functions with no parameters.
- For each of the three system-provided function-string classes in which the user-defined function will be used, and for each derived class that inherits from these classes, Replication Server generates a default function string for the user-defined function.
- You can customize the function string in rs_sqlserver_function_class and in user-created function-string classes using create function string.
- For each user-created base function-string class in which the user-defined function will be used, and for each derived class that inherits from such a class, you must create a function string, using create function string. The function string should invoke a stored procedure or RPC, with language appropriate for the replicate data server.
- For an overview of function-string classes, function strings, and functions, see the *Replication Server Administration Guide Volume 2*.
- Replication Server distributes the new user-defined function to qualifying sites through the replication system. The changes do not appear immediately at all such sites because of normal replication system lag time.
- When you create a user-defined function for a replication definition, it is created for all replication definitions in the primary table.

Permissions

create function requires “create object” permission.

See also

create applied function replication definition, create function string, create request function replication definition, drop function

create function replication definition

Description Creates a function replication definition and user-defined function for a stored procedure that is to be replicated.

Note create function replication definition and alter function replication definition are deprecated commands. Sybase suggests that you use these instead:

- create applied function replication definition and alter applied function replication definition.
 - create request function replication definition and alter request function replication definition.
-

Syntax

```
create function replication definition
    function_rep_def
with primary at data_server.database
[deliver as 'proc_name']
    ([@param_name datatype [, @param_name datatype]...])
    [searchable parameters (@param_name
    [, @param_name]...)]
    [send standby {all | replication definition}
    parameters]
```

Parameters

function_rep_def
A name for the function replication definition. It must conform to the rules for identifiers.

with primary at
Specifies the data server and database containing the primary data.

data_server
The name of the data server containing the primary data. If the primary database is part of a warm standby application, *data_server* is the logical data server name.

database
The name of the database containing the primary data. If the primary database is part of a warm standby application, *database* is the logical database name.

deliver as
Specifies the name of the stored procedure to execute at the database where you are delivering the replicated function. *proc_name* is a character string of up to 200 characters. If you do not use this clause, the function is delivered as a stored procedure with the same name as the function replication definition.

@param_name

A parameter name from the function. A parameter name must not appear more than once in each clause in which it appears. You are not required to include parameters and their datatypes, but you *must* include the parentheses () for this clause, whether or not you include any parameters.

datatype

The datatype of a parameter in the function. See “Datatypes” on page 21 for a list of the datatypes and their syntax. Adaptive Server stored procedures and function replication definitions cannot contain parameters with the text, unitext, rawobject, and image datatypes.

searchable parameters

Specifies a list of parameters that can be used in where clauses of define subscription, create subscription, or create article. You *must* include the parentheses () if you include this clause.

send standby

In a warm standby application, specifies whether to send all parameters in the function (send standby all parameters) or just those specified in the replication definition (send standby replication definition parameters) to a standby database. The default is send standby all parameters.

Examples

Example 1 Creates a function replication definition named titles_frep for a function and stored procedure of the same name. The primary data is in the pubs2 database in the LDS data server. Use a function replication definition like this for an applied function:

```
create function replication definition titles_frep
with primary at LDS.pubs2
(@title_id varchar(6), @title varchar(80),
 @type char(12), @pub_id char(4),
 @price money, @advance money,
 @total_sales int)
searchable parameters (@title_id, @title)
```

Example 2 Creates a function replication definition named titles_frep for a function and stored procedure of the same name, as in the previous example. In this case, the stored procedure to be invoked in the destination database is named upd_titles. Use a function replication definition like this for a request function:

```
create function replication definition titles_frep
with primary at LDS.pubs2
deliver as 'upd_titles'
(@title_id varchar(6), @title varchar(80),
 @type char(12), @pub_id char(4),
```

```
@price money, @advance money,  
@total_sales int)  
searchable parameters (@title_id, @title)
```

Usage

- Use `create function replication definition` to describe a stored procedure that is to be replicated. For an overview of replicated stored procedures, see the *Replication Server Administration Guide Volume 1*.
- Execute `create function replication definition` at the Replication Server that manages the database where the primary data is stored.
- You can create only one function replication definition per replicated stored procedure.
- Before executing this command, be sure that:
 - The function replication definition name is unique in the replication system. Replication Server cannot always enforce this requirement when you use `create function replication definition`.
 - A connection exists from the Replication Server to the database where the primary data is stored. See `create connection` for more information. You can also create connections using `rs_init`. Refer to the Replication Server installation and configuration guides for your platform.
 - The name, parameters, and datatypes you specify for the function replication definition match those of the stored procedure involved. You can specify only those parameters you are interested in replicating.
- Unlike replicated stored procedures associated with table replication definitions, stored procedures associated with function replication definitions are not required to update a table. This allows you to replicate transactions that are not associated with replicated data. For more information about stored procedures, see Chapter 6, “RSSD Stored Procedures.”

See `sp_setrepproc` on page 571 for more information on the two types of replicated stored procedures.

- Replication Server distributes the new function replication definition to qualifying sites through the replication system. The changes do not appear immediately at all such sites because of normal replication system lag time.

User-defined functions and function strings

- When you create a function replication definition, Replication Server automatically creates a corresponding user-defined function.
- For the system-provided function-string classes in which the user-defined function associated with this function replication definition will be used, and for each derived class that inherits from these classes, Replication Server generates a default function string for the user-defined function.
- You can customize the function string in `rs_sqlserver_function_class` and in user-created function-string classes using `create function string`.
- For each user-created base function-string class in which the user-defined function will be used, and for each derived class that inherits from such a class, you must create a function string, using `create function string`. The function string should invoke a stored procedure or RPC, with language appropriate for the replicate data server.
- For an overview of function-string classes, function strings, and functions, see the *Replication Server Administration Guide Volume 2*.

The *with primary at* clause

- Use the *with primary at* clause to specify the data server and database containing the primary data. This is not necessarily the database that contains the invoked stored procedure.

For applied functions (primary-to-replicate function replication) and request functions (replicate-to-primary function replication), create the function replication definition at the Replication Server managing the primary data, and specify the primary database using the *with primary at* clause.

The *deliver as* clause

- Use the optional *deliver as* clause to specify the name of the stored procedure to execute at the destination database where you are delivering the replicated function. If you do not use this clause when you create or alter the function replication definition, the function is delivered as a stored procedure with the same name as the function replication definition.

In a warm standby database the stored procedure has the same name as in the active database so the *deliver as* clause is ignored.

Typically, you would use the *deliver as* clause for request function delivery; that is, when a function is replicated from a replicate Replication Server to a primary Replication Server. This way, the name of the replicated function is not the same as the stored procedure that is executed.

Use this method with “round-trip” stored procedure replication, where the primary Replication Server that is the destination for the request function executes an applied function, to which the originating replicate Replication Server in turn subscribes.

See the *Replication Server Administration Guide Volume 1* for more information.

Function replication definitions for HDS parameters

- Although you cannot create function replication definitions that alter the datatype of a parameters value, you can use HDS datatype definitions to declare parameters for applied function replication definitions. Such parameters are then subject to class-level translations. See the *Replication Server Administration Guide Volume 1* for more information about HDS.
- Replication Server does not perform translations on parameter values for request functions. Note, however, that during function-string mapping Replication Server uses the delimiters defined for the parameter values of their declared datatype to generate SQL.

Altering function replication definitions

- Use alter function replication definition to add parameters or searchable parameters to an existing function replication definition. You can also specify a new stored procedure name to use when delivering the replicated function at the destination database.
- If you need to remove or rename parameters in function replication definition, you must drop all subscriptions to the function replication definition (applied functions only). Then drop the function replication definition and re-create it.

Subscribing to function replication definitions

- In order to subscribe to a function replication definition, use create subscription with the without materialization clause, or use define subscription and the other commands involving bulk materialization.

Function replication definitions and table replication definitions

- In replicating stored procedures through applied functions, it is advisable to create table replication definitions and subscriptions for the same tables that the replicated stored procedures will affect. By doing this, you can ensure that any normal transactions that affect the tables will be replicated as well as the stored procedure executions.

DML inside stored procedures marked as replicated is not replicated through table replication and you must subscribe to the stored procedure even if you have subscribed to the table.

- If you plan to use both kinds of replication definition for the same table, you can materialize the table data with the subscription for the table replication definition. Then you can create the subscription for the function replication definition using `create subscription` with the `without materialization` clause.

Permissions

`create function replication definition` requires “create object” permission.

See also

`alter function replication definition`, `alter function string`, `create connection`, `create function string`, `define subscription`, `drop function replication definition`, `sp_setrepproc`

create function string

Description	Adds a function string to a function-string class. Replication Server uses function strings to generate instructions for data servers.
Syntax	<pre>create function string [<i>replication_definition</i>.]<i>function</i>[:<i>function_string</i>] for <i>function_class</i> [with overwrite] [scan '<i>input_template</i>'] [output {language '<i>lang_output_template</i>' rpc 'execute procedure [<i>@param_name</i>=]{<i>constant</i> ? <i>variable</i>!mod?} [, [<i>@param_name</i>=] {<i>constant</i> ? <i>variable</i>!mod?}}...' writetext [use primary log with log no log] none}]</pre>
Parameters	<p><i>replication_definition</i></p> <p>The name of the replication definition the function operates on. Use only for functions with replication definition scope.</p> <p>Functions have either function-string-class scope or replication definition scope. Functions that direct transaction control have function-string class scope. User-defined functions, and functions that modify data, have replication definition scope.</p> <p><i>function</i></p> <p>The name of the function. Names for system functions must be provided as documented in Chapter 4, “Replication Server System Functions.” Names for user-defined functions must match an existing user-defined function.</p> <p><i>function_string</i></p> <p>A function string name is required when customizing <code>rs_get_textptr</code>, <code>rs_textptr_init</code>, and <code>rs_writetext</code> functions, and optional for others. For <code>rs_get_textptr</code>, <code>rs_textptr_init</code>, and <code>rs_writetext</code>, a function string is needed for each text, unitext, or image column in the replication definition. The function string name supplied <i>must</i> be:</p> <ul style="list-style-type: none"> • The text, unitext, or image column name for the replication definition. • Able to conform to the rules for identifiers. • Unique in the scope of the function. <p>Replication Server also uses the function string name in the generation of error messages.</p> <p><i>function_class</i></p> <p>The function-string class the new function string belongs to.</p>

with overwrite

If the function string already exists, this option drops and re-creates the function string, as though you used `alter function string` instead.

scan

Precedes an input template.

input_template

A character string, enclosed in single quote characters, that Replication Server scans to associate an `rs_select` or `rs_select_with_lock` function string with the where clause in a create subscription command. An input template string is written as a SQL select statement, with user-defined variables instead of the literal values in the subscription's where clause.

output

Precedes an output template.

language

Tells Replication Server to submit the output template commands to the data server using the Client/Server Interfaces language interface.

lang_output_template

A character string, enclosed in single quote characters, that contains instructions for a data server. A language output template string may contain embedded variables, which Replication Server replaces with run-time values before it sends the string to the data server.

rpc

An output template that tells Replication Server to use the Client/Server Interfaces remote procedure call (RPC) interface. Replication Server interprets the string and constructs a remote procedure call to send to the data server.

These keywords and options appear in RPC output templates:

procedure – the name of the remote procedure to execute. It could be an Adaptive Server stored procedure, a procedure processed by an Open Server gateway RPC handler, or a registered procedure in an Open Server gateway. Refer to the *Open Server Server-Library/C Reference Manual* for information about processing RPCs in a gateway program.

@param_name – the name of an argument to the procedure, as defined by the procedure. If the *@param_name = value* form is used, parameters can be supplied in any order. If parameter names are omitted, parameter values must be supplied in the order defined in the remote procedure.

constant – a literal value with the datatype of the parameter it is assigned to.

?variable!mod? – *variable* is the placeholder for a run-time value. It can be a column name, the name of a system-defined variable, the name of a parameter in a user-defined function, or the name of a variable defined in an input template. The variable must refer to a value with the same datatype as the parameter it is assigned to. For a list of system-defined variables, see “System-defined variables” on page 275.

The *mod* portion of a variable name identifies the type of data the variable represents. The variable modifier is required for all variables and must be one of the following:

Table 3-25: Function string variable modifiers

Modifier	Description
new, new_raw	A reference to the new value of a column in a row you are inserting or updating
old, old_raw	A reference to the existing value of a column in a row you are updating or deleting
user, user_raw	A reference to a variable that is defined in the input template of an <i>rs_select</i> or <i>rs_select_with_lock</i> function string
sys, sys_raw	A reference to a system-defined variable
param, param_raw	A reference to a function parameter

Modifier	Description
text_status	<p>A reference to the text_status value for text, unitext, or image data. Possible values are:</p> <ul style="list-style-type: none">• 0x000 – Text field contains NULL value, and the text pointer has not been initialized.• 0x0002 – Text pointer is initialized.• 0x0004 – Real text data will follow.• 0x0008 – No text data will follow because the text data is not replicated.• 0x0010 – The text data is not replicated but it contains NULL values.

Note Function strings for user-defined functions may not use the *new* or *old* modifiers.

writetext

instructs Replication Server to use the Client-Library™ function `ct_send_data` to update a text, unitext, or image column value. This option applies only to the `rs_writetext` function.

The following options appear in `writetext` output templates to specify the logging behavior of the text, unitext, or image column in the replicate database:

`use primary log` – logs the data in the replicate database, if the logging option was specified in the primary database.

`with log` – logs the data in the replicate database transaction log.

`no log` – does not log the data in the replicate database transaction log.

none

applies to all functions, and provides the flexibility to identify which function-strings Replication Server can avoid executing on replicate databases:

- For `rs_writetext` functions – instructs Replication Server not to replicate a text, unitext, or image column value.
- For non-`rs_writetext` functions – instructs Replication Server not to execute commands on the replicate database.

Examples

Example 1 Creates a function string for the `rs_begin` function:

```
create function string rs_begin
for sqlserver2_function_class
output language
'begin transaction'
```

Example 2 Creates a function string for the `rs_commit` function that contains two commands separated with a semicolon. The function string executes an Adaptive Server stored procedure that updates the `rs_lastcommit` system table and then commits the transaction:

```
create function string rs_commit
for sqlserver2_function_class
output language
'execute sqlrs_update_lastcommit
  @origin = ?rs_origin!sys?,
  @origin_qid = ?rs_origin_qid!sys?,
  @secondary_qid = ?rs_secondary_qid!sys?;
commit transaction'
```

Example 3 Examples 3 and 4 create a replication definition for the titles table and an `rs_insert` function string for the `sqlserver2_function_class`. The function string inserts data into the `titles_rs` table instead of into the titles table in the replicate database:

```
create replication definition titles_rep
with primary at LDS.pubs2
(title_id varchar(6), title varchar(80),
 type char(12), pub_id char(4), advance money,
 total_sales int, notes varchar(200),
 pubdate datetime, contract bit, price money)
primary key (title_id)
searchable columns (price)
```

Example 4 Examples 3 and 4 create a replication definition for the titles table and an `rs_insert` function string for the `sqlserver2_function_class`. The function string inserts data into the `titles_rs` table instead of into the titles table in the replicate database:

```
create function string titles_rep.rs_insert
for sqlserver2_function_class
output language
'insert titles_rs values (?title_id!new?,
  ?title!new?, ?type!new?, ?pub_id!new?,
  ?advance!new?, ?total_sales!new?, ?notes!new?,
  ?pubdate!new?, ?contract!new?, ?price!new?)'
```

Example 5 Examples 5 and 6 create a user-defined function `update_titles` and a corresponding function string for the `sqlserver2_function_class`. The function string executes an Adaptive Server stored procedure named `update_titles`:

```
create function titles_rep.update_titles
(@title_id varchar(6), title varchar(80),
 @price money)
```

Example 6 Examples 5 and 6 create a user-defined function `update_titles` and a corresponding function string for the `sqlserver2_function_class`. The function string executes an Adaptive Server stored procedure named `update_titles`:

```
create function string titles_rep.update_titles
for sqlserver2_function_class
output rpc
'execute update_titles
  @title_id = ?title_id!param?,
  @title = ?title!param?,
  @price = ?price!param?'
```

Example 7 The `rs_select` function string in example 7 is used to materialize subscriptions that request rows with a specified value in the `title_id` column. Similar to example 8, the input templates given by the scan clauses differentiate the two function strings:

```
create function string
  titles_rep.rs_select;title_id_select
for sqlserver2_function_class
scan 'select * from titles
     where title_id = ?title_id!user?'
output language
'select * from titles
  where title_id = ?title_id!user?'
```

Example 8 The `rs_select` function string in example 8 is an example of an RPC function string. It is used to materialize subscriptions that request rows where the value of the price column falls within a given range:

```
create function string
  titles_rep.rs_select;price_range_select
for sqlserver2_function_class
scan 'select * from titles
     where price > ?price_min!user?
     and price < ?price_max!user?'
output rpc
'execute titles_price_select
  ?price_min!user?, ?price_max!user?'
```

Usage

- Use `create function string` to add a function string to a function-string class. Function strings contain the database-specific instructions needed by Replication Server to convert a function to a command for a database.
- For an overview of functions, function strings, and function-string classes, see the *Replication Server Administration Guide Volume 2*.

- Create or alter function strings for functions with class scope at the primary site for the function-string class. See *create function string class* for more information about the primary site for a function-string class.
- Create or alter function strings for functions with replication definition scope, including user-defined functions, at the site where the replication definition was created. Each replication definition has its own set of function strings.
- Replication Server distributes the new function string to qualifying sites through the replication system. The changes do not appear immediately at all such sites because of normal replication system lag time.
- Some function strings are generated dynamically; they are not stored in the RSSD.

Function strings and function-string classes

- For each of the system-provided function-string classes in which a function will be used, and for each derived class that inherits from these classes, Replication Server generates a default function string for the function. This is true for both system functions and user-defined functions. (Default function strings for the `rs_dumpdb` and `rs_dumptran` functions are not provided. You only need to create them if you are using coordinated dumps.
- Customize the function string in `rs_sqlserver_function_class` using *alter function string*. Customize the function string in user-created function-string classes using *create function string*.
- For each user-created base function-string class in which the function will be used, and for each derived class in which you want to override the inherited function string, you must create a function string, using *create function string*.
- Omitting the output clause instructs Replication Server to generate a function string in the same way that it generates function strings for the `rs_sqlserver_function_class` or `rs_default_function_class` function-string classes.
- The default function string for a user-defined function is an invocation of a stored procedure where the name is the function name and the parameters are the function parameters. The stored procedure is executed as a language command, not as an RPC.

Function strings and *replicate minimal columns*

- If you have specified replicate minimal columns for a replication definition, you cannot normally create non-default function strings for the `rs_update`, `rs_delete`, `rs_get_textptr`, `rs_textptr_init`, or `rs_datarow_for_writetext` system functions.

However, you can create non-default function strings for the `rs_update` and `rs_delete` functions if you use the `rs_default_fs` system variable within the function string. This variable represents the default function-string behavior. You can add additional commands to extend the function-string behavior.

- See create replication definition for more information about the replicate minimal columns option.

Input and output templates

- Depending on the function, function strings can have input and output templates. Replication Servers substitute variable values into the templates and pass the result to data servers for processing.
- Input and output templates have the following requirements:
 - They are limited to 64K. The result of substituting run-time values for embedded variables in function-string input or templates must not exceed 64K.
 - Input templates and language or RPC output templates are delimited with two single quote characters (').
 - Variable names in input templates and output templates are delimited with question marks (?).
 - A variable name and its modifier are separated with an exclamation mark (!).
- When creating function strings:
 - Use two consecutive single quote characters (") to represent one literal single quote character within or enclosing data of character or date/time datatypes, as shown for "Berkeley" in the following character string:

```
'insert authors
(city, au_id, au_lname, au_fname)
values ('Berkeley', ?au_id!new?,
?au_lname!new?,
?au_fname!new?) '
```

- Use two consecutive question marks (??) to represent one single question mark within data of character datatypes.
 - Use two consecutive semi-colons (;;) to represent one single semi-colon within data of character datatypes.
- Input templates
- Input templates are used only with the `rs_select` and `rs_select_with_lock` functions, which are used during non-bulk subscription materialization and with purge subscription dematerialization. Replication Server matches the subscription's `where` clause with an input template to find the function string to use.
 - Input templates have the following requirements:
 - They contain only user-defined variables, whose values come from the constants in the `where` clause. The user-defined variables can also be referenced in the function string's output template.
 - If the *input_template* is omitted, it can match any `select` command. This allows you to create a default function string that is executed when no other function string in the function-string class has an *input_template* matching the `select` command.
- Output templates
- Output templates determine the format of the command sent to a replicate data server. Most output templates can use one of these formats: language RPC, or none. An output template for an `rs_writetext` function string can use the RPC format or the additional formats `writetext` or `none`. For a description of these formats, see the *Replication Server Administration Guide Volume 2*.
 - When Replication Server maps function string output templates to data server commands, it formats the variables using the format expected by Adaptive Server. It modifies datatypes for modifiers that do not end in *_raw* (the modifiers that are normally used), as follows:
 - Adds an extra single quote character to single quote characters appearing in character and date/time values to escape the special meaning of the single quote character.
 - Adds single quote characters around character and date/time values, if they are missing.
 - Adds the appropriate monetary symbol (the dollar sign in U.S. English) to values of money datatypes.
 - Adds the "0x" prefix to values of binary datatypes.

- Adds a combination of a backslash (\) and newline character between existing instances of a backslash and newline character in character values. Adaptive Server treats a backslash followed by a newline as a continuation character, and therefore deletes the added pair of characters, leaving the original characters intact.

Replication Server does not modify datatypes in these ways for modifiers that end in `_raw`.

The following table summarizes how Replication Server formats each datatype for the modifiers that do not end in `_raw`:

Table 3-26: Formatting for function string variables

Datatype	Formatting of literals
bigint,int, smallint, tinyint, rs_address	Integer number
unsigned bigint, unsigned int, unsigned smallint, unsigned tinyint	Unsigned Integer number
decimal, numeric, identity	Exact decimal number
float, real	Decimal number
char, varchar	Enclosed in single quote character Adds single quote character to any instance of a single quote character Pads instances of backslash + newline characters
unichar, univarchar	Unicode
money, smallmoney	Adds the appropriate money symbol (dollar sign for U.S. English)
date, time, datetime, smalldatetime	Enclosed in single quote characters Adds single quote character to any instance of a single quote character
binary, timestamp, varbinary	Prefixed with 0x
bit	1 or 0

- Output templates have the following requirements:
 - The result of substituting run-time values for embedded variables in function-string output templates must not exceed 64K.

- You can put several commands in a language function-string output template, separating them with semicolons (;). If the database is configured to allow command batches, which is the default, Replication Server replaces the semicolons with that connection's DSI command separator character before sending the function string in a single batch to the data server. The separator character is defined in the `dsi_cmd_separator` option of the `alter connection` command.

To represent a semicolon that should not be interpreted as a command separator, use two consecutive semicolons (;).

If the connection to the database is not configured to allow batches, Replication Server sends the commands in the function string to the data server one at a time. To enable or disable batching for a database, use `alter connection`.

System-defined variables

The following table lists system-defined variables that can be used in function-string output templates. Use the `sys` or `sys_raw` modifier for these variables:

Table 3-27: Replication Server system-defined variables

System variable	Datatype	Description
<code>rs_default_fs</code>	text	The default generated function-string text for the function
<code>rs_deliver_as_name</code>	varchar(200)	For execution of a replicated function, name of the procedure to be invoked at the destination
<code>rs_destination_db</code>	varchar(30)	Name of the database where a transaction was sent
<code>rs_destination_ds</code>	varchar(30)	Name of the data server where a transaction was sent
<code>rs_destination_ldb</code>	varchar(30)	Name of the logical database where a transaction was sent
<code>rs_destination_lds</code>	varchar(30)	Name of the logical data server where a transaction was sent
<code>rs_destination_ptype</code>	char(1)	Physical connection type ("A" for active or "S" for standby) for the database where a transaction was sent
<code>rs_destination_user</code>	varchar(30)	User who will execute the transaction at the destination
<code>rs_dump_dbname</code>	varchar(30)	Name of the database where a database or transaction dump originated
<code>rs_dump_label</code>	varchar(30)	Label information for a database or transaction dump. For Adaptive Server, this variable holds a datetime value that is the time the dump originated.
<code>rs_dump_status</code>	int(4)	Dump status indicator: <ul style="list-style-type: none"> 0 – indicates that the dump transaction command does not contain the parameter with <code>standby_access</code> 1 – indicates that the dump transaction command contains the parameter with <code>standby_access</code>
<code>rs_dump_timestamp</code>	varbinary(16)	Timestamp of a database or transaction dump
<code>rs_lorigin</code>	int(4)	ID of the originating logical database for a transaction

System variable	Datatype	Description
<i>rs_isolation_level</i>	varchar(30)	Transaction isolation level of a database connection.
<i>rs_origin</i>	int(4)	ID of the originating database for a transaction
<i>rs_origin_begin_time</i>	datetime	The time that a command was applied at the origin Note If you execute <code>select getdate()</code> while ASE is still processing user database recovery, the returned value of <code>select getdate()</code> may be different from the value of <i>rs_origin_begin_time</i> .
<i>rs_origin_commit_time</i>	datetime	The time that a transaction was committed at the origin Note If you execute <code>select getdate()</code> while ASE is still processing user database recovery, the returned value of <code>select getdate()</code> may be different from the value of <i>rs_origin_begin_time</i> .
<i>rs_origin_db</i>	varchar(30)	Name of the origin database
<i>rs_origin_ds</i>	varchar(30)	Name of the origin data server
<i>rs_origin_ldb</i>	varchar(30)	Name of the logical database for a warm standby application
<i>rs_origin_lds</i>	varchar(30)	Name of the logical data server for a warm standby application
<i>rs_origin_qid</i>	varbinary(36)	Origin queue ID of the first command in a transaction
<i>rs_origin_user</i>	varchar(30)	User who executed the transaction at the origin
<i>rs_origin_xact_id</i>	binary(120)	The system-assigned unique ID of a transaction
<i>rs_origin_xact_name</i>	varchar(30)	User-assigned name of the transaction at origin
<i>rs_secondary_qid</i>	varbinary(36)	Queue ID of a transaction in a subscription materialization or dematerialization queue
<i>rs_last_text_chunk</i>	int(4)	If the value is 0, this is not the last chunk of text data. If the value is 1, this is the last chunk of text data.
<i>rs_writetext_log</i>	int(4)	If the value is 0, <i>rs_writetext</i> has not finished logging text, unitext, and image data at the primary database transaction log. If the value is 1, <i>rs_writetext</i> has finished logging text, unitext, and image data at the primary database transaction log.

rs_origin_commit_time
system variable

If you are not using parallel DSI to process large transactions before their commit has been read from the DSI queue, the value of the *rs_origin_commit_time* system variable contains the time when the last transaction in the transaction group committed at the primary site.

If you are using parallel DSI to process large transactions before their commit has been read from the DSI queue, when the DSI threads start processing one of these transactions, the value of the *rs_origin_commit_time* system variable is set to the value of the *rs_origin_begin_time* system variable. When the commit statement for the transaction is read, the value of *rs_origin_commit_time* is set to the actual commit time. Therefore, when the configuration parameter *dsi_num_large_xact_threads* is set to a value greater than zero, the value for *rs_origin_commit_time* is not reliable for any system function other than *rs_commit*.

System variables and NULL values

- The following system variables may have NULL values:
 - *rs_origin_ds*
 - *rs_origin_db*
 - *rs_origin_user*
 - *rs_origin_xact_name*
 - *rs_destination_db*
 - *rs_destination_user*
 - *rs_dump_dbname*
 - *rs_dump_label*

When a system variable has no value, Replication Server maps the word “NULL” into function-string templates. This may cause syntax errors in some generated statements. For example, the following command would be generated if *rs_origin_xact_name* has a null value:

```
begin transaction NULL
```

To prevent this error, create a function string with an output template like the following:

```
'begin transaction t_?rs_origin_xact_name!sys_raw?'
```

If the *rs_origin_xact_name* system variable is null, the transaction name will be “t_NULL”.

Replacing function strings

- To replace a function string, use *alter function string* or *create function string with overwrite*. Either approach executes *drop function string* and *create function string* in a single transaction, preventing errors that could result from temporarily missing function strings.

Permissions	create function string requires “create object” permission.
See also	alter function string, configure connection, create connection, create function string class, create subscription, define subscription, drop function string, alter function string

create function string class

Description	Creates a function-string class.
Syntax	create function string class <i>function_class</i> [set parent to <i>parent_class</i>]
Parameters	<p><i>function_class</i></p> <p>The name of the function-string class to create. It must conform to the rules for identifiers. Function-string class names have a global name space, so they must be unique in the replication system.</p> <p>set parent to</p> <p>Designates a parent class for a new derived class.</p> <p><i>parent_class</i></p> <p>The name of an existing function-string class you designate as the parent class for a new derived class. <code>rs_sqlserver_function_class</code> cannot be used as a parent class.</p>
Examples	<p>Example 1 Creates a derived function-string class named <code>sqlserver_derived_class</code> that will inherit function strings from the system-provided class <code>rs_default_function_class</code>:</p> <pre>create function string class sqlserver_derived_class set parent to rs_default_function_class</pre> <p>Example 2 Creates a function-string class named <code>sqlserver2_function_class</code>. This class will be a base class, and will not inherit function strings. You can, however, specify this class as a parent class for a derived class:</p> <pre>create function string class sqlserver2_function_class</pre>
Usage	<ul style="list-style-type: none"> • Use <code>create function string class</code> to create a function-string class. Function-string classes group function strings for a database. The function-string class, with its member function strings, is associated with a database. This association is made with the <code>create connection</code> or <code>alter connection</code> command. • The Replication Server to which <code>create function string class</code> is sent becomes the primary Replication Server for the newly-created function-string class. • Create a new derived class using the <code>set parent to</code> clause to specify a parent class from which the new class is to inherit function strings. Omit this clause to create a new base class, which does not inherit function strings from a parent class.

- For an overview of function-string classes, function strings, and functions, see the *Replication Server Administration Guide Volume 2*.
- Before you execute this command, make sure that the name for the new function-string class is unique in the replication system. Replication Server does not detect all name conflicts.
- Replication Server distributes the new function-string class to qualifying sites through the replication system. The changes do not appear immediately at all such sites because of normal replication system lag time.
- To modify function strings in the class `rs_sqlserver_function_class`, you must first select a Replication Server to be the primary site for the class. Then execute `create function string class` for `rs_sqlserver_function_class` at that site.
- The Replication Server that serves as the primary site for any function-string class must have routes to all other Replication Servers where the class will be used.
- The primary site for a derived class is the same as the primary site of its parent class. You must create a derived class at the primary site of its parent class. However, if the parent class is a system-provided class, `rs_default_function_class` or `rs_db2_function_class`, the primary site of the derived class is the Replication Server where you created the derived class.

System-provided function-string classes

- Replication Server provides three function-string classes that you can use:
 - `rs_sqlserver_function_class` – default generated Adaptive Server function strings are provided for this class. The default function strings in `rs_sqlserver_function_class` are identical to those in `rs_default_function_class`. This class is assigned by default to Adaptive Server databases you add to the replication system using `rs_init`. You can customize function strings for this class. `rs_sqlserver_function_class` cannot be used as a parent class or a derived class.
 - `rs_default_function_class` – default generated Adaptive Server function strings are provided for this class. The default function strings in `rs_sqlserver_function_class` are identical to those in `rs_default_function_class`. You cannot customize function strings for this class. This class can be used as a parent class but cannot become a derived class.

- `rs_db2_function_class` – default generated DB2-specific function strings are provided for this class. Although this class is a derived class of `rs_default_function_class`, with customizations for DB2, you cannot customize function strings for this class. `rs_db2_function_class` can be used as a parent class but cannot be made a derived class.

Benefits of function-string inheritance

- Using derived classes that inherit from system-provided classes `rs_default_function_class` or `rs_db2_function_class`, either directly or indirectly, allows you to customize only the function strings you want to customize and inherit all others, even for new table or function replication definitions.

If you use classes that do not inherit from system-provided classes, you must create all function strings yourself, either in parent or derived classes, and add new function strings whenever you create a new table or function replication definition.

- After an upgrade to a future release of Replication Server, derived classes that inherit from the system-provided classes `rs_default_function_class` or `rs_db2_function_class`, either directly or indirectly, will inherit function-string definitions for any new system functions.

Adding function strings to a function-string class

- After you create a function-string class that does not inherit function strings from a parent class, add function strings for the system functions that have function-string-class scope. Then add function strings for system functions and user-defined functions that have replication definition scope and will be replicated to databases that use the new function-string class.
- To create or customize function strings in a function-string class, use `create function string`. You cannot create function strings in the classes `rs_default_function_class` or `rs_db2_function_class`.

Permissions

`create function string class` requires “sa” permission.

See also

`alter connection`, `alter function string class`, `create connection`, `create function`, `create function string class`, `move primary`

create logical connection

Description	Creates a logical connection. Replication Server uses logical connections to manage warm standby applications.
Syntax	<pre>create logical connection to <i>data_server.database</i> [set <i>logical_database_param</i> [to] '<i>value</i>' [set <i>logical_database_param</i> [to] '<i>value</i>'...]</pre>
Parameters	<p><i>data_server</i></p> <p>The name of a data server. The data server does not have to be a real data server.</p> <p><i>database</i></p> <p>The name of a database. The database does not have to be a real database.</p> <p><i>logical_database_param</i></p> <p>The name of the configuration parameter that affects logical connections. Table 3-16 describes the parameters that you can set with create logical connection.</p>
Examples	<p>Example 1 Creates a logical connection called LDS.logical_pubs2:</p> <pre>create logical connection to LDS.logical_pubs2</pre> <p>Example 2 Creates a logical connection for an existing connection. For example, you would enter this if the database TOKYO_DS.pubs2 already exists and will serve as the active database in the warm standby application:</p> <pre>create logical connection to TOKYO_DS.pubs2</pre>
Usage	<ul style="list-style-type: none">• create logical connection creates a logical connection to be used with warm standby applications. See the <i>Replication Server Administration Guide Volume 2</i> for information about setting up and managing warm standby applications.• The logical connection is for a symbolic <i>data_server.database</i> specification. The data server and database do not have to be real; Replication Server maps them to the current active database.• If you are creating a logical connection for an existing connection, <i>data_server.database</i> must refer to the data server and database names of the existing connection. Otherwise, it is recommended that the logical name be different from the active and standby database names.• Replication definitions and subscriptions use the logical connection name.• After you create the logical connection, use rs_init to add the physical active and standby databases for the logical connection.

Permissions	create logical connection requires “sa” permission.
See also	alter logical connection, configure connection, configure logical connection, drop connection, drop logical connection, switch active

create partition

Description	Makes a partition available to Replication Server. A partition can be a disk partition or an operating system file.
Syntax	<code>create partition <i>logical_name</i> on '<i>physical_name</i>' with size <i>size</i> [starting at <i>vstart</i>]</code>
Parameters	<p><i>logical_name</i></p> <p>A name for the partition. The name must conform to the rules for identifiers. The name is also used in the drop partition and alter partition commands.</p> <p><i>physical_name</i></p> <p>The full specification of the partition. This name must be enclosed in single quotation marks.</p> <p><i>size</i></p> <p>The size, in megabytes, of the partition. Maximum size possible is 1TB.</p> <p>starting at <i>vstart</i></p> <p>Specifies the number of megabytes (<i>vstart</i>) to offset from the beginning of the partition.</p>
Examples	<p>Example 1 Adds a 20MB partition named P1 on the device named <code>/dev/rsd0a</code>:</p> <pre>create partition P1 on '/dev/rsd0a' with size 20</pre> <p>Example 2 Adds a 20MB partition named P1 on the device named <code>/dev/rsd0a</code>. Since an offset of 1MB is specified, however, the total usable partition space available to Replication Server is 19MB:</p> <pre>create partition P1 on '/dev/rsd0a' with size 20 starting at 1</pre>
Usage	<ul style="list-style-type: none">• Replication Server uses partitions for stable message queues. The message queues hold data until it is sent to its destination.• Increasing the available disk space in partitions allows Replication Server to support more routes and database connections and to continue to queue messages during longer failures.• The maximum size for a partition is 1TB, which is approximately 1,000,000MB.• Disk partitions must not be mounted for use by the operating system and should not be used for any other purpose, such as for swap space or an Adaptive Server disk device.

- Allocate the entire partition to Replication Server. If you allocate part of a partition for Replication Server, you cannot use the remainder for any other purpose. If you use the starting at *vstart* clause, the partition space available to Replication Server is what is left after you subtract the offset size from the total partition size.
- The starting at *vstart* clause makes space available at the beginning of the partition for disk mirroring information.
- You can use operating system files for partitions. However, the operating system buffers file I/O, so you may not be able to recover stable queues completely following a failure. Therefore, you should only use files for partitions in a test environment—unless your operating system does not support physical disk partitions.
- If you use an operating system file, you must create it before executing `create partition`. On UNIX platforms, you can create the file with the `touch` command. The file can be zero bytes in length; the `create partition` command extends the file to the specified *size*.
- The “sybase” user should own the disk partition or operating system file and must have read and write permissions on it. Users other than “sybase” should not have write or read permission on the partition.

Permissions

`create partition` requires “sa” permission.

See also

`admin disk_space`, `drop partition`, `alter partition`

create publication

Description	Creates a publication for tables or stored procedures that are to be replicated as a group to one or more subscribing replicate databases.
Syntax	<pre>create publication <i>pub_name</i> with primary at <i>data_server.database</i></pre>
Parameters	<p><i>pub_name</i></p> <p>A name for the publication. It must conform to the rules for identifiers and be unique for the specified primary data server and database.</p> <p>with primary at <i>data_server.database</i></p> <p>Specifies the location of the primary data. If the primary database is part of a warm standby application, <i>data_server.database</i> is the name of the logical data server and database.</p>
Examples	<p>Creates a publication called pubs2_pub that you can use to replicate data for multiple tables and stored procedures in the pubs2 database.</p> <pre>create publication pubs2_pub with primary at TOKYO_DS.pubs2</pre>
Usage	<ul style="list-style-type: none">• Use create publication to create a publication, an object that makes it easy to set up replication for multiple tables or stored procedures in a database. You create a publication, add articles, which specify replication definitions, and then create a single subscription for the publication.• Execute create publication at the Replication Server that manages the database where the primary data is stored.• For more information about working with replication definitions, articles, and publications, see the <i>Replication Server Administration Guide Volume I</i>. <p>For more information about subscribing to publications, refer to Chapter 10, “Managing Subscriptions,” in the same book.</p> <ul style="list-style-type: none">• Replication Server distributes information about a new publication to a replicate site only when you create or refresh a subscription for the publication. <p>Requirements for using <i>create publication</i></p> <ul style="list-style-type: none">• Before executing create publication, make sure that:<ul style="list-style-type: none">• The publication name you enter is unique for the primary data server and database.• A connection exists from the Replication Server to the database where the primary tables or stored procedures are stored.

Preparing publications for subscription

- After you create a publication, you use `create article` to create articles and assign them to the publication. An article specifies a table replication definition or function replication definition and includes optional `where` clauses according to the needs of the subscribing replicate site. See `create article` for more information.
- Because a replicate table cannot subscribe to two or more replication definitions for the same primary object, a publication cannot contain two or more articles for different replication definitions for the same primary table and the same replicate table.
- When all of the articles have been assigned, you must validate the publication using `validate publication` before a replicate site can subscribe to it. Validating a publication verifies that the publication contains at least one article and marks the publication ready for subscription. See `validate publication` for more information.
- To check publication status, use `check publication`. This command displays the number of articles the publication contains and indicates if the publication is valid. See `check publication` for more information.

Subscribing to publications

- When a publication is valid, you can create a subscription for the publication in order to begin replication to a replicate database. All forms of subscription materialization are supported. See `create subscription` or `define subscription` for more information.
- When you create a publication subscription, Replication Server creates a separate underlying subscription for each article that the publication contains. Each article subscription uses the name of the parent publication subscription.
- A subscription to a publication cannot include a `where` clause. Instead, you can customize replication to replicate sites by including one or more `where` clauses in each article the publication contains.

Articles for table replication definitions

- If a publication contains articles for table replication definitions only, you can use `create subscription` to subscribe to the publication using atomic or non-atomic materialization. See `create subscription` for more information.
- You can also use bulk materialization for the publication subscription:
 - When data already exists at the replicate database, use `create subscription` with the `without materialization` clause.

- When you must manually transfer subscription data, use `define subscription` and the other bulk materialization commands. See `define subscription` for more information.

Articles for function replication definitions

- If a publication contains articles for function replication definitions only, use bulk materialization for the publication subscription:
 - When data already exists at the replicate database, use `create subscription` with the `without materialization` clause. See `create subscription` for more information.
 - When you must manually transfer subscription data, use `define subscription`, `activate subscription`, and `validate subscription` to subscribe to the publication using bulk materialization. See `define subscription` for more information.

Articles for both table and function replication definitions

- If a publication contains articles for both table replication definitions and function replication definitions, you can use the same subscription command even though each type of replication definition requires a different materialization method.

In order to create the subscription, first transfer data to the replicate database for component subscriptions that require bulk materialization, such as those for function replication definitions. Then use `create subscription` to subscribe to the publication:

- Subscriptions for articles for table replication definitions are materialized using atomic or non-atomic materialization—unless you use the `without materialization` clause.
- Subscriptions for articles for function replication definitions are materialized without materialization.

In cases where the stored procedure for a function replication definition operates on a table for which there is also a table replication definition, no separate data transfer is necessary.

Refreshing publication subscriptions

- If you add a new article to an existing publication, or drop an article from the publication, the publication is invalidated. Although replication for existing articles continues unaffected, in order to begin replication for any new articles or create new publication subscriptions you must:
 - Validate the publication when you have completed making changes to the publication, then

- Refresh the publication subscription.
- In order to refresh a publication subscription for atomic or non-atomic materialization:
 - Re-create the subscription using `create subscription`. See `create subscription` for more information.
- In order to refresh a publication subscription for bulk materialization:
 - When data already exists at the replicate database, use `create subscription` with the `without materialization` clause.
 - Re-create the subscription using `define subscription`, `activate subscription`, and `validate subscription` and transfer subscription data manually as necessary. See `define subscription` for more information.

Dropping subscriptions, articles, and publications

- You can drop a subscription to a publication and, optionally, purge the subscription data for the component subscriptions to articles for table replication definitions. See `drop subscription`.
- If there is no subscription, you can drop an article that a publication contains and, optionally, drop the associated replication definition if it is not used elsewhere. After you drop an article, the publication is invalid. See `drop article`.
- You can drop a publication if there are no subscriptions for the publication. When you drop a publication, its articles are also dropped. Optionally, you can also drop all of the replication definitions for the publication's articles, if they are not used elsewhere. See `drop publication` for more information.

Publications in warm standby applications

- In a warm standby application, replication definitions used in replicating to the standby database may also be specified by articles included in publications.

Permissions

`create publication` requires “create object” permission.

See also

`check publication`, `create article`, `create applied function replication definition`, `create replication definition`, `create request function replication definition`, `create subscription`, `define subscription`, `drop article`, `drop publication`, `drop subscription`, `validate subscription`

create request function replication definition

Description	Creates a request function replication definition and a user-defined function for a stored procedure that is to be replicated. The request function is applied at the replicate database by the same user who executes the stored procedure at the primary database.
Syntax	<pre>create request function replication definition <i>repdef_name</i> with primary at <i>dataserver.database</i> with primary function named '<i>func_name</i>' with replicate function named '<i>func_name</i>' ([<i>@param_name datatype</i> [, <i>@param_name datatype</i>]...]) [searchable parameters (<i>@param_name</i> [, <i>@param_name</i>]...)] [send standby {all replication definition} parameters]</pre>
Parameters	<p><i>repdef_name</i> The function replication definition name. The name must conform to the rules for identifiers.</p> <p>with primary at Specifies the data server and database that contains the primary data.</p> <p><i>dataserver</i> The name of the data server that contains the primary data. If the primary database is part of a warm standby application, <i>dataserver</i> is the logical data server name.</p> <p><i>database</i> The name of the database that contains the primary data. If the primary database is part of a warm standby application, <i>database</i> is the logical database name.</p> <p>with primary function named Specifies the stored procedure name at the primary database. If you do not specify a primary function name, Replication Server uses the replication definition name as the name of the primary function. The primary function name must be different from the replicate function name specified in the with replicate function named clause.</p> <p>'<i>func_name</i>' The name of the function, with a maximum length of 255 characters.</p>

with replicate function named

Specifies the name of the stored procedure to execute at the replicate database. If you do not specify a replicate function name, Replication Server uses the replication definition name as the name of the replicate function. The replicate function name must be different from the primary function name specified in the with primary function named clause.

Note The primary stored procedure refers to the stored procedure originally invoked by the client, while the replicate stored procedure refers to the stored procedure replicated from the primary database and invoked by the replicate Replication Server.

This request function behavior is different from the request function behavior in Replication Server 15.0.1 and earlier. For more information about the behavior of request function in Replication Server 15.0.1 and earlier versions, see the *Replication Server Administration Guide Volume 2*.

@param_name

A parameter name from the function. A parameter name cannot appear more than once in each clause in which it appears. You are not required to include parameters and their datatypes, but you *must* include a pair of parentheses, whether or not you include any parameters.

datatype

The datatype of a parameter in the function. See “Datatypes” on page 21 for a list of the datatypes and their syntax. Adaptive Server stored procedures and function replication definitions cannot contain parameters with the text, unitext, rawobject, and image datatypes.

searchable parameters

Specifies a list of parameters that can be used in where clauses of define subscription, create subscription, or create article. You *must* include the parentheses () if you include this clause.

send standby

In a warm standby application, specifies whether to send to the standby database, all the parameters in the function (send standby all parameters) or just those specified in the replication definition (send standby replication definition parameters). The default is send standby all parameters.

Examples

Creates a request function replication definition named titles_frep for a function named upd_titles_prim. The stored procedure to be invoked in the destination database is named upd_titles:

```
create request function replication definition titles_frep
```

```
with primary at LDS.pubs2
with primary function named 'upd_titles_prim'
with replicate function named 'upd_titles'
(@title_id varchar(6), @title varchar(80), @type char(12), @pub_id char(4),
 @price money, @advance money, @total_sales int)
searchable parameters (@title_id, @title)
```

Usage

- Use create request function replication definition to describe a stored procedure that you want to replicate. The difference between the applied function replication definition and the request function replication definition is that the function replicated through an applied function replication definition is executed at the replicate site by the maintenance user while the function replicated through a request function replication definition is executed at the replicate site by the same user who executes the primary function at the primary site. For an overview of replicated stored procedures, see the *Replication Server Administration Guide Volume 1*.
- When you create a request function replication definition for a primary function, make sure that the function *does not* already have an existing function replication definition that satisfies both these conditions:
 - Was created using the create function replication definition command
 - the function replication definition is used for the request function replication without subscription in Replication Server 15.0.1 and earlier version

If these conditions are true, the existing request function replication definition is disabled. See the *Replication Server Administration Guide Volume 2* for more information about request function replication definition in Replication Server 15.0.1 and earlier.

- Execute the create request function replication definition command at the Replication Server that manages the database where the primary stored procedure is stored.
- Before executing create request function replication definition, be sure that:
 - The function replication definition name is unique in the replication system. Replication Server cannot always enforce this requirement when you use create request function replication definition.

- A connection exists from the Replication Server to the database where the primary data is stored. See `create connection`. You can also create connections using `rs_init`; see the *Replication Server Installation Guide* and the *Replication Server Configuration Guide* for your platform.
- The name, parameters, and datatypes you specify for the function replication definition must match those of the stored procedure involved. Only the parameters specified in the function replication definition are replicated.
- Replication Server distributes the new function replication definition to qualifying sites through the replication system. The changes do not appear immediately at all such sites because of normal replication system lag time.

User-defined functions and function strings

- When you create a request function replication definition, Replication Server automatically creates a corresponding user-defined function. Similarly, in `rs_sqlserver_function_class`, Replication Server automatically creates a default function string for the user-defined function.
- You can customize the function string in `rs_sqlserver_function_class` and in user-defined function-string classes using `create function string`.
- For each user-created base function-string class in which the user-defined function will be used, and for each derived class that inherits from such a class, you must create a function string, using `create function string`. The function string should invoke a stored procedure or RPC, with language appropriate for the replicate data server.
- For an overview of function-string classes, function strings, and functions, see the *Replication Server Administration Guide Volume 2*.

with primary at clause

Use the `with primary at` clause to specify the primary data server and database. The primary database is the database that contains the invoked primary stored procedure.

with replicate function named clause

Use the *with replicate function named* clause to specify the name of the stored procedure you want to execute at the destination database where you are delivering the replicated function. If you do not use *with replicate function named* when you create or alter the function replication definition, the function is delivered as a stored procedure with the same name as the function replication definition. In a warm standby database, the stored procedure has the same name as in the active database so *with replicate function named* is ignored.

A round-trip replication enables a database to send a data change request to another database and to replicate the data change back to the requesting database. See the *Replication Server Administration Guide Volume 1* for more information about how to set up a round-trip replication with both applied and request function replication definitions.

Request function replication definitions for HDS parameters

Although you cannot create function replication definitions that alter the datatype of a parameter's value, you can use HDS datatype definitions to declare parameters for request function replication definitions. The declared parameters are subjected to class-level translations.

See the *Replication Server Administration Guide Volume 1* for more information about HDS.

Altering function replication definitions

- Use *alter request function replication definition* to add parameters or searchable parameters to an existing request function replication definition. You can also specify a different replicate name for the function.
- To remove or rename parameters in function replication definition, drop all subscriptions to the function replication definition. After dropping the subscriptions, drop the function replication definition and re-create it.

Subscribing to function replication definitions

To subscribe to a request function replication definition, use *create subscription* with the *without materialization* clause, or use *define subscription* and the other commands involving bulk materialization.

Creating multiple replication definitions

- You can create multiple request function replication definitions for the same primary function, and customize each one so that it can be subscribed to by a different replicate function. See the *Replication Server Administration Guide Volume 1* for details.

- Different request function replication definitions created for the same primary function *must* use the same parameter with same name and the same datatype.
- A request function replication definition can only be subscribed to Replication Servers version 15.1.
- The same primary function can have applied function replication definitions or request function replication definitions, but not both. The function replication definition created with the create function replication definition command is considered as an applied function at the primary Replication Server where the function is created.
- In a warm standby database, the stored procedure has the same name as the active database, and the with replicate function named clause is ignored. If one of the request function replication definition is created with the send standby replication definition parameters clause, the parameters specified in the function replication definition are delivered to the standby database. Otherwise, all of the parameters in the primary function are delivered.
- In an MSA environment, if there is no function replication definition for a primary function created with the send standby clause, the function delivered to the replicate database has the same name as the primary function with all the primary function's parameters. Otherwise, the function delivered to the replicate database has the name specified in the with replicate function named clause of that function replication definition, and with parameters specified in the same function replication definition.

Permissions

create request function replication definition requires “create object” permission.

See also

alter applied function replication definition, alter function string, alter request function replication definition, create applied function replication definition, create connection, create function string, define subscription, drop function replication definition, sp_setrepproc, rs_send_repserver_cmd

create replication definition

Description	Creates a replication definition for a table that is to be replicated.
Syntax	<pre>create replication definition <i>replication_definition</i> with primary at <i>data_server.database</i> [with all tables named [<i>table_owner</i>.] '<i>table_name</i>' [quoted] [with primary table named [<i>table_owner</i>.] '<i>table_name</i>' with replicate table named [<i>table_owner</i>.] '<i>table_name</i>' [quoted]] (<i>column_name</i> [as <i>replicate_column_name</i>] [<i>datatype</i> [null not null] [map to <i>published_datatype</i>]] [quoted] [, <i>column_name</i> [as <i>replicate_column_name</i>] [<i>datatype</i> [null not null] computed] [map to <i>published_datatype</i>]] [quoted]...) [references [<i>table_owner</i>.]'<i>table_name</i>' [(<i>column_name</i>)]] primary key (<i>column_name</i> [, <i>column_name</i>]...) [searchable columns (<i>column_name</i> [, <i>column_name</i>]...)] [send standby [{all replication definition} columns]] [replicate {minimal all} columns] [replicate {SQLDML ['off'] '<i>options</i>'}] [replicate_if_changed (<i>column_name</i> [, <i>column_name</i>]...)] [always_replicate (<i>column_name</i> [, <i>column_name</i>]...)] [with dynamic sql without dynamic sql]</pre>
Parameters	<p><i>replication_definition</i></p> <p>The replication definition, which must conform to the rules for identifiers. The replication definition name is assumed to be the name of both the primary and replicate tables, unless you specify the table names.</p> <p>with primary at <i>data_server.database</i></p> <p>Specifies the location of the primary data. If the primary database is part of a warm standby application, <i>data_server.database</i> is the name of the logical data server and database.</p> <p>with all tables named</p> <p>Specifies the table name at both the primary and replicate databases. <i>table_name</i> is a character string of up to 200 characters. <i>table_owner</i> is optional, and represents the table owner. Data server operations may fail if the actual table owners do not correspond to what you specify in the replication definition.</p> <p>quoted</p> <p>Use the quoted parameter to specify that the table or column name being created is a quoted identifier.</p>

with primary table named

Specifies the table name at the primary database. *table_name* is a character string of up to 200 characters. *table_owner* is optional and represents the table owner. Data server operations may fail if the actual table owners do not correspond to what you specify in the replication definition.

If you specify the primary table name but do not also specify the replicate table name, the replication definition name is assumed to be the name of the replicate table.

with replicate table named

Specifies the name of the table at the replicate database. *table_name* is a character string of up to 200 characters. *table_owner* is optional and represents the table owner. Data server operations may fail if the actual table owners do not correspond to what you specify in the replication definition.

If you specify the replicate table name but do not also specify the primary table name, the replication definition name is assumed to be the name of the primary table.

column_name

A column name from the primary table. You cannot use a column name more than once in each clause.

Each column and datatypes must be enclosed in parentheses ().

as *replicate_column_name*

Specifies a column name in a replicate table into which data from the primary column will be copied. Use this clause when the source and destination columns have different names.

datatype

The datatype of the column in the primary table. See “Datatypes” on page 21 for a list of the datatypes and syntax.

Use as *declared_datatype* if you are specifying a column-level datatype translation. A declared datatype must be a native Replication Server datatype or a datatype definition for the primary datatype.

For different replication definitions created against the same table, the column datatypes must be the same, however the published datatypes may be different. See the *Replication Server Administration Guide Volume 1* for more information.

Specifying the datatype is optional if a replication definition created against the same table already has this column.

null or not null

Applies only to text, unitext, image, or rawobject columns. Specifies whether a null value is allowed in the replicate table. The default is not null, meaning that the replicate table does not accept null values.

The null status for each text, unitext, image, and rawobject column must match for all replication definitions for the same primary table, and must match the settings in the actual tables. Specifying the null status is optional if an existing replication definition of the same primary table has text, unitext, image, and rawobject columns.

You cannot change this setting for a column once it is included in a replication definition for the table. To change the value, you must drop and re-create all replication definitions that include the column.

map to *published_datatype*

Specifies the datatype of a column after a column-level datatype translation, but before any class-level translation and before presentation to the replicate database.

references *table owner.table name column name*

Specifies the table name of the table with referential constraints at the primary database. *table_name* is a character string of up to 200 characters. *table_owner* is optional, and represents the table owner. *column name* is optional. Data server operations may fail if the actual table owners do not correspond to what you specify in the replication definition.

primary key *column_name*

Specifies the columns that form the primary key for the table. You cannot use a column name more than once in each clause.

You cannot include text, unitext, image, rawobject, rawobject in row, or rs_address columns as part of the primary key.

searchable columns *column_name*

Specifies the columns that can be used in where clauses of create subscription, define subscription, or create article. You cannot use a column name more than once in each clause.

You cannot specify text, unitext, image, rawobject, rawobject in row or encrypted columns as searchable columns.

send standby

Specifies how to use the replication definition in replicating into a standby database in a warm standby application. See “Replication definitions and warm standby applications” on page 307 for details on using this clause and its options.

replicate minimal columns or replicate all columns

Sends all replication definition columns for every transaction or only those needed to perform update or delete operations at replicate databases. The default is to replicate all columns.

Note If your replication definition has the `[replicate {minimal | all} columns]` clause, the `[replicate {minimal | all} columns]` clause must always precede the `[replicate {SQLDML ['off'] | 'options'}]` clause.

replicate SQLDML ['off']

Turns on or off the SQL replication of the DML operation specified.

replicate 'options'

Replicates any combination of these DML operations:

- U – update
- D – delete
- I – insert select

replicate_if_changed

Replicate text, unitext, image, or rawobject columns only when their column data changes.

always_replicate

Always replicate text, unitext, image, and rawobject columns.

with dynamic sql

Specifies that DSI applies dynamic SQL to the table if the command qualifies and enough cache space is available. This is the default.

See the *Replication Server Administration Guide Volume 2* for the conditions a command must meet to qualify for dynamic SQL.

without dynamic sql

Specifies that DSI must not use dynamic SQL commands.

Examples

Example 1 Creates a replication definition named `authors_rep` for the `authors` table. The primary copy of the `authors` table is in the `pubs2` database in the LDS data server. All copies of the table are also named `authors`. Only the minimum number of columns will be replicated for delete and update operations:

```
create replication definition authors_rep
  with primary at LDS.pubs2
  with all tables named 'authors'
    (au_id varchar(11), au_lname varchar(40),
```

```
    au_fname varchar(20), phone char(12),  
    address varchar(12), city varchar(20),  
    state char(2), country varchar(12), postalcode  
    char(10))  
primary key (au_id)  
searchable columns (au_id, au_lname)  
replicate minimal columns
```

Example 2 Creates a replication definition called `blurbs_rep` for the `blurbs` table owned by “emily” in the `pubs2` database. Data in the `copy` column, which uses the text datatype and accepts null values, will be replicated when the column data changes:

```
create replication definition blurbs_rep  
with primary at TOKYO_DS.pubs2  
with all tables named emily.'blurbs'  
    (au_id char(12), copy text null)  
primary key (au_id)  
replicate_if_changed (copy)
```

Example 3 Where at least one replication definition already exists for the primary table `publishers` in the `pubs2` database, this command creates an additional replication definition called `pubs_copy_rep`. This replication definition can be subscribed to by replicate tables that are named `pubs_copy` and for which “joe” is the owner. Subscriptions may fail for replicate tables that are also named `pubs_copy` but for which “joe” is not the owner:

```
create replication definition pubs_copy_rep  
with primary at TOKYO_DS.pubs2  
with primary table named 'publishers'  
with replicate table named joe.'pubs_copy'  
    (pub_id, pub_name as pub_name_set)  
primary key (pub_id)
```

Data for the `pub_name` column in the primary table will replicate into the `pub_name_set` column in the replicate table, which must share the same datatype. You do not need to specify the datatype for a column in an existing replication definition. In this example, the `city` and `state` columns from the primary table are not required for the replicate table `pubs_copy`, and are thus excluded from this replication definition.

Example 4 Creates a replication definition that replicates all modified columns of the authors table to the standby database. This definition also replicates to the MSA, however, only the modified values of `au_id` and `au_lname` columns are replicated. `au_id` is the key used to update and delete from the authors table:

```
create replication definition authors_rep
```

```
with primary at LDS.pubs2
with all tables named 'authors'
    (au_id varchar(11), au_lname varchar(40))
primary key (au_id)
send standby
replicate minimal columns
```

Example 5 Creates a table foo where column foo_col1 is a quoted identifier:

```
create replication definition repdef
with primary at primaryDS.primaryDB
with all tables named "foo"
    ("foo_col1" int quoted, "foo_col2" int)
primary key ("foo_col1")
```

Example 6 Creates a table replication definition that replicates update and delete statements:

```
create replication definition repdef1
with primary at ds3.pdb1
with all tables named 'tb1'
    (id_col int, str_col char(40))
primary key (id_col)
replicate all columns
replicate 'UD'

go
```

Usage

- Execute this command at the Replication Server that manages the database where the primary version of the table is stored.
- Use `rs_helprep` to determine which replication definitions are available to Replication Server version 12.0 and earlier. For more information, see `rs_helprep` on page 612.
- For an overview of defining and maintaining replicated tables, and for information about working with replication definitions, articles, and publications, see the *Replication Server Administration Guide Volume 1*.
- Before executing the create replication definition command, be sure that:
 - The replication definition name you enter is unique among all replication definitions (table or function) in the replication system. Replication Server cannot always enforce this requirement when you enter create replication definition.

- A connection exists from the Replication Server to the database where the primary table is stored. See `create connection` for more information. You can also create database connections using `rs_init`. See the Replication Server installation and configuration guides for your platform.
- If you use more than one version of Replication Server (for example, version 12.0 and version 11.0.x) and you create multiple replication definitions for the same primary table, review any mixed-version issues for your replication system (for example, if column names are different for the same table in both versions). See “Creating multiple replication definitions” on page 302 for details.
- Replication Server distributes the new replication definition to qualifying sites through the replication system. The changes do not appear immediately at all sites because of normal replication system lag time.

Replication status

- The replication status for text, unitext, image, and rawobject columns must be the same in the Adaptive Server database and in the replication definition.
- Use `alter replication definition` to change replication status.
- The replication status must be consistent for all of the replication definitions created against the same primary table.
 - If you change the replication status using `alter replication definition`, the replication status for other replication definitions against the same primary table also changes.
 - You do not have to specify replication status if the column is already listed in another replication definition for the same primary table.

Creating multiple replication definitions

- You can create multiple replication definitions for the same primary table and customize each one so it can be subscribed to by a replicate table whose characteristics are different from those of the primary table and other replicate tables.

In addition to describing the primary table, each replication definition can specify, for example, a smaller number of columns, different column names, or a different table name for a replicate table. Replicate tables that match the specified characteristics can subscribe to the replication definition. You can also use multiple replication definitions even when replicate and primary tables match.

This feature also allows you to create one replication definition for normal replication and another one for standby if the database requirements are different. See the *Replication Server Administration Guide Volume 1* for details.

- A replicate table can subscribe to only one replication definition per primary table, although it can subscribe to the same replication definition more than once.
- Different replication definitions created for the same primary table must use the same column datatype and the same null status for text, unitext, and image columns.
- If a table is replicated to standby or MSA connection using internal replication definition and dynamic SQL is enabled for the connection, any new replication definition for the table should define the column order consistent with the column order in the primary database. Otherwise, it may invalidate the existing prepared statements and may require the standby or MSA connection to be restarted.

Functions and function strings

- Replication Server creates `rs_insert`, `rs_delete`, `rs_update`, `rs_truncate`, `rs_select`, and `rs_select_with_lock` functions for the replication definition. If the replication definition contains text, unitext, image, or rawobject data, Replication Server also creates `rs_datarow_for_writetext`, `rs_get_textptr`, `rs_textptr_init`, and `rs_writetext` functions.
- Replication Server generates default function strings for these functions for the system-provided function-string classes and for derived classes that inherit from these classes. Some function strings may be generated dynamically, so they never exist in the RSSD. For other function-string classes, you must create all the function strings.
- For each function-string class, each replication definition for the same table has its own set of function strings for the system functions.
- When you create, drop, or alter a user-defined function, it is created, dropped, or altered for all the replication definitions for the same primary table.
- Although different replication definitions for the same primary table share the same user-defined functions, each user-defined function has its own function string. You create user-defined functions using `create function` when you replicate stored procedures using the method associated with table replication definitions.

Specifying columns and datatypes

- When you specify the columns and datatypes you want to replicate, observe these guidelines:
 - Columns that have user-defined datatypes must be defined in the replication definition with the underlying base datatypes.
 - The replication status (that is, `replicate_if_changed`, `always_replicate`) of a text, `unitext`, image, or `rawobject` column must be the same for all replication definitions on the primary table. If you change a text, `unitext`, image, or `rawobject` column's replication status using `alter replication definition`, the replication status for that column also changes for other replication definitions for the same primary table.

You do not have to specify the replication status of a text, `unitext`, image, or `rawobject` column that is part of a replication definition for the same table.

- Omit length and precision from numeric datatype declarations. Replication Server processes numeric datatype values without affecting precision.

Note If you use the `map to` option to translate a larger `varchar` to a `varchar` with a smaller number of characters per column, make sure that any data you replicate does not exceed the character length of the column you replicate to.

For instance, you can map a `varchar(100)` to a `varchar(25)` column, as long as the item you replicate does not exceed the limit of `varchar(25)`. If it does, an error message appears.

- If a replication definition column list contains an `IDENTITY` column and the replicate table is in Adaptive Server, the maintenance user must be the owner of the table (or must be “dbo” or aliased to “dbo”) at the replicate database in order to use the Transact-SQL `identity_insert` option.

A primary table (with one or multiple replication definitions) can contain only one `IDENTITY` column. However, you may use the `map to` option to publish multiple columns as the identity datatype with one or multiple replication definitions.

- If a replication definition column list contains a `timestamp` column and the replicate table is in Adaptive Server, the maintenance user must be the owner of the table (or must be “dbo” or aliased to “dbo”) at the replicate database.

A primary table with one or multiple replication definitions can contain only one timestamp column. However, you may use the map to option to publish multiple columns as the timestamp datatype with one or multiple replication definitions.

- The `rs_address` datatype allows a unique subscription resolution technique. Bitmaps of the `rs_address` datatype (based on the underlying `int` datatype) are compared with a bitmask in a subscription's where clause to determine whether a row should be replicated. To use this subscription resolution method, you must first create tables that use columns of the `int` datatype. In creating a replication definition, include these columns in the column list, but declare the datatype to be `rs_address` instead of `int`.

See `create subscription` for more information. Also, see the *Replication Server Administration Guide Volume 1* for more information about using the `rs_address` datatype.

Specifying columns and datatypes for column-level translations

- You cannot use `text`, `untext`, `image`, or `rawobject` datatypes as a base datatype or a datatype definition or as a source or target of either a column-level or class-level translation.
- *declared_datatype* depends on the datatype of the value delivered to Replication Server:
 - If the Replication Agent delivers a native Replication Server datatype, *declared_datatype* is the native datatype.
 - If the Replication Agent delivers any other datatype, *declared_datatype* must be the datatype definition for the original datatype in the primary database.
- *published_datatype* is the datatype of the value after a column-level translation, but before any class-level translation. *published_datatype* must be a Replication Server native datatype or a datatype definition for the datatype in the target database.
- Columns declared in multiple replication definitions must use the same *declared_datatype* in each replication definition. The *published_datatype* can differ.

Using the *replicate minimal columns* option

- Using the *replicate minimal columns* option can improve DSI performance, reduce message overhead, and reduce queue size. It can also help to avoid application problems caused by triggers that are set for columns that are not actually changed.

For details on how this option works, see the *Replication Server Administration Guide Volume 2*.

- The following requirements apply to replicating minimal columns:
 - Normally, *replicate minimal columns* can be used only with replication definitions that use the default function strings for the *rs_update* and *rs_delete* functions. If you specify *replicate minimal columns*, you can create non-default *rs_update* and *rs_delete* function strings for the replication definition using the *rs_default_fs* system variable within the function string. See *create function string* for details.
 - You cannot use autocorrection with the *replicate minimal columns* option. If you specify *set autocorrection on* before you set *replicate minimal columns*, an informational message is logged for each delete or update operation. If you first specify *replicate minimal columns*, you cannot specify *set autocorrection on* for the replication definition.
 - If you have specified *replicate minimal columns* for a replication definition, you cannot create a subscription for it using non-atomic materialization (*create subscription* command, without *holdlock* option), or use the bulk materialization option that simulates non-atomic materialization. See the *Replication Server Administration Guide Volume 2* for more information.

Replicating *text*, *unitext*, *image*, or *rawobject* datatypes

- The primary key of the replication definition must include the column or columns that uniquely identify a single row in the table.

- The `always_replicate` and `replicate_if_changed` clauses let you specify the replication status for text, unitext, image, and rawobject columns. You can also set this status in the Adaptive Server system procedures `sp_setreptable` and/or `sp_setrepcol`, or `sp_reptostandby`. The replication status must be the same in the Adaptive Server system procedures and in the replication definitions of a primary table. If there are inconsistencies, the RepAgent can shut down. See the *Replication Server Administration Guide Volume 1* for information on setting status and resolving inconsistencies if they occur. See “Replication definitions and warm standby applications” on page 307 for information about replicating text, unitext, image, and rawobject data into warm standby applications.
- You must specify the replication definition’s replication status as `always_replicate` when you mark a table with `sp_setreptable` only, because the `sp_setreptable` default replication status is `always_replicate`. You can change a table’s replication status to `replicate_if_changed` by changing the table’s replication definition replication status to `replicate_if_changed` and marking every column in the table with the `sp_setrepcol` replication status set to `replicate_if_changed`.
- The following requirements apply to replicating text, unitext, image, or rawobject datatypes:
 - If a text, unitext, image, or rawobject column appears in the `replicate_if_changed` column list, attempting to enable autocorrection for the replication definition will cause an error. Autocorrection requires that all text, unitext, image, and rawobject columns appear in the `always_replicate` list for the replication definition.
 - If a text, unitext, image, or rawobject column with `replicate_if_changed` status was not changed in an update operation at the primary table and the update causes the row to migrate into a subscription, the inserted row at the replicate table will be missing the text, unitext, image, or rawobject data. Replication Server displays a warning message in the error log when the row migrates into the subscription and the text, unitext, image, or rawobject data is missing. In this case, run `rs_subcmp` to reconcile the data in the replicate and primary tables.

Replication definitions and warm standby applications

- Replication Server does not require replication definitions to maintain a standby database in a warm standby application. Using replication definitions may improve performance in replicating into the standby database. You can create a replication definition just for this purpose for each table in the logical database.

- Use `send standby` with any option to use the replication definition to replicate transactions for the table to the standby database. The replication definition's primary key columns and `replicate minimal columns` setting are used to replicate into the standby database. The options for this method include:
 - Use `send standby` or `send standby all columns` to replicate all columns in the table to the standby database.
 - Use `send standby replication definition columns` to replicate only the replication definition's columns to the standby database.
- Use `send standby off` in `alter replication definition` to indicate that you don't want any single replication definition for this table to be used in replicating into the standby database.

When none of a primary table's replication definitions are marked as used by the standby, all columns are replicated into the standby database, the union of all primary keys for all replication definitions for the table is used for the primary key, and minimal columns are replicated. The `replicate_minimal_columns` setting for the logical connection determines whether to send minimal columns or all columns for update and delete. See `alter logical connection` and `alter replication definition` for details.

- See the *Replication Server Administration Guide Volume 2* for more information about the performance optimizations gained by using replication definitions for replicating into the standby database.
- In a primary table with multiple replication definitions, if a replication definition is already marked as used by the standby, another replication definition created or altered with `send standby` unmarks the first one.
- You must specify the replication definition's replication status as `replicate_if_changed` when you mark a database with `sp_reptostandby only`, because the `sp_reptostandby` default replication status is `replicate_if_changed`. You cannot change the replication status of text, unitext, image, and rawobject columns when the database is marked with `sp_reptostandby only`.
- When you mark a database with `sp_reptostandby` and a table in that database with `sp_setreptable`, you must specify the replication status for the replication definition as `always_replicate`—because the default replication status is `always_replicate`. You can change a table's replication status to `replicate_if_changed` by changing the table's replication definition replication status to `replicate_if_changed` and marking every column in the table with the `sp_setrepcol` replication status set to `replicate_if_changed`.

Altering replication definitions

- Use alter replication definition to add more columns or more searchable columns and to make other changes to the settings for an existing replication definition. See alter replication definition for details.
- If you need to remove or rename primary columns in an existing replication definition, you must drop all subscriptions to the replication definition, drop the replication definition and re-create it, and re-create the subscriptions.

Replicating stored procedures

- To enable replication of stored procedures, use create applied function replication definition or create request function replication definition. For an overview of replicating stored procedures, see the *Replication Server Administration Guide Volume 1*.

Replicating computed columns

- create replication definition supports the replication of materialized computed columns. Materialized computed columns need to be defined using its base datatype in the replication definition.
- Materialized computed column is a computed column whose value is stored in the table page same as regular columns. It is re-evaluated upon each insert or update on its base column. It is not re-evaluated in a query.
- There is another type of computed column called virtual or non-materialized computed column. The value of this computed column is not stored in the table or an index. It is only evaluated when it is referenced in a query and no action is taken upon insert or update operation.

Replication of virtual computed columns is not supported and this should not be included in the replication definition.

For more information on replicating computed columns, see *Replication Server Administration Guide Volume 1*.

Using quoted identifiers

- When you use the quoted parameter to mark an identifier, and the dsi_quoted_identifier is set to on for a replicate server that subscribed to the replication definition, that replicate server receives the marked identifier as a quoted identifier. If the dsi_quoted_identifier is off, the markings are ignored and the replicate server does not receive quoted identifiers.

- When replicating to a warm standby database and to replication definition subscribers, and the primary table name is marked as quoted but the replicate table name is not, or vice-versa, Replications Server sends both the primary table name and the replicate table name as quoted.
- An embedded double quote character in identifiers is not supported.
- Data servers such as Adaptive Server, SQL Anywhere, Microsoft SQL Server, Universal Database (UDB), and Oracle handle quoted identifiers differently in terms of length, special characters, and reserved words supported. In a heterogeneous environment, you must ensure that the quoted identifiers being replicated are valid on both the primary and replicate data servers.
- For replication of a quoted identifier to succeed, the primary Replication Server and the Replication Server that connects to the replicate data server version must be 15.2 and later. However, intermediate Replication Servers in a route can be of lower versions.

Replicating SQL statements

- A table replication definition with the `send standby` clause can specify a replicate 'I' statement. You can replicate an insert select statement as a SQL replication statement only in warm standby or MSA environments. A table replication definition without a `send standby` clause cannot replicate the insert select statement.
- By default, warm standby applications do not replicate the DML commands that support SQL statement replication. To use SQL replication, you can:
 - Create table replication definitions using `replicate SQLDML` and `send standby` clauses.
 - Set the `WS_SQLDML_REPLICATION` parameter to on. The default value is `UDIS`. However, `WS_SQLDML_REPLICATION` has a lower precedence than the table replication definition for SQL replication. If your table replication definition contains `send standby` clause for a table, the clause determines whether or not to replicate the DML statements, regardless of the `WS_SQLDML_REPLICATION` parameter setting.
- SQL statement replication cannot perform autocorrection. If Data Server Interface (DSI) encounters a DML command for SQL statement replication and auto-correction is on, DSI is suspended and stops replication by default. Use the `assign action` command with error number 5193 to specify how Replication Server handles this error.

Replication Server does not replicate SQLDML until the table level subscription is validated.

- SQL statement replication is not supported when:
 - A replicate database has a different table schema than the primary database.
 - Replication Server must perform data or schema transformation.
 - Subscriptions include where clauses.
 - Updates include one or more text or image columns.

Handling tables that have referential constraints

For both alter replication definition and create replication definition with the reference clause, Replication Server:

- Treats the reference clause as a column property. Each column can reference only one table.
- Does not process the column name you provide in the *column_name* parameter within the reference clause.
- Does not allow referential constraints with cyclical references. For example, the original referenced table cannot have a referential constraint to the original referencing table.

During replication processing, RTL loads:

- Inserts to the referenced tables before the referencing table you specify in the replication definition.
- Deletes to the referenced tables after the table you specify in the replication definition.

In some cases, updates to both tables fail because of conflicts. To prevent RTL from retrying replication processing, and thus decreasing performance, you can:

- Stop replication updates by setting *dsi_command_convert* to “u2di,” which converts updates to deletes and inserts.
- Turn off *dsi_compile_enable* to avoid compiling the affected tables.

RTL cannot compile and thus marks out tables with customized function-strings, and tables that have referential constraints to an existing table that it cannot compile. By marking out these tables, RTL optimizes replication processing by avoiding transaction retries due to referential constraint errors.

Permissions

create replication definition requires “create object” permission.

See also

alter function string, alter replication definition, configure logical connection, create connection, create applied function replication definition, create request function replication definition, create function string, create subscription, drop replication definition, rs_set_quoted_identifier, set, sp_setrepcol, sp_setreptable, rs_send_repserver_cmd

create route

Description	Designates the route to use for a connection from the current Replication Server to a remote Replication Server.
Syntax	<pre>create route to <i>dest_replication_server</i> { set next site [to] <i>thru_replication_server</i> [set username [to] <i>user</i>] [set password [to] <i>passwd</i>] [set <i>route_param</i> to 'value' [set <i>route_param</i> to 'value'...]] [set <i>security_param</i> to 'value' [set <i>security_param</i> to 'value'...]] }</pre>
Parameters	<p><i>dest_replication_server</i> The destination Replication Server.</p> <p><i>thru_replication_server</i> The intermediate Replication Server through which to pass messages for the destination Replication Server. Specify this when creating an indirect route.</p> <p><i>user</i> The Replication Server login name to use to log in to the destination Replication Server. This is the login name used by the RSI user thread. If no user name is entered, Replication Server uses the principal user name entered with the -S flag when Replication Server was started.</p> <p><i>passwd</i> The password to use with the login name. If no password is entered, Replication uses a null value.</p> <p><i>route_param</i> a parameter that affects routes. See Table 3-17 on page 184 for a list of parameters and values.</p> <p><i>value</i> a character string containing a value for a parameter.</p> <p><i>security_param</i> Specifies the name of a security parameter. Refer to Table 3-28 on page 313 for a list and description of security parameters that can be set with create route.</p>

Table 3-28: Parameters affecting network-based security

security_param	value
msg_confidentiality	<p>Indicates whether Replication Server sends and receives encrypted data. If set to “required,” outgoing data is encrypted. If set to “not required,” Replication Server accepts incoming data that is encrypted or not encrypted.</p> <p>Default: not_required</p>

security_param	value
msg_integrity	Indicates whether data is checked for tampering. Default: not_required
msg_origin_check	Indicates whether the source of data should be verified. Default: not_required
msg_replay_detection	Indicates whether data should be checked to make sure it has not been read or intercepted. Default: not_required
msg_sequence_check	Indicates whether data should be checked for interception. Default: not_required
mutual_auth	Requires remote server to provide proof of identify before a connection is established. Default: not_required
security_mechanism	The name of the third-party security mechanism enabled for the pathway. Default: first mechanism listed in the SECURITY section of <i>libtcl.cfg</i>
unified_login	Indicates how Replication Server seeks to log in to remote data servers and accepts incoming logins. The values are: <ul style="list-style-type: none">• required – always seeks to log in to remote server with a credential.• not_required – always seeks to log in to remote server with a password. Default: not_required
use_security_services	Tells Replication Server whether to use security services. If use_security_services is “off,” no security features take effect.
Note This parameter can only be set by configure replication server.	

Examples

Example 1 Entered at the TOKYO_RS Replication Server, this command creates a direct route from TOKYO_RS to the SYDNEY_RS Replication Server. TOKYO_RS can log in to SYDNEY_RS over this route, using the login name “sydney_rsi” with the password “sydney_rsi_ps.”

```
create route to SYDNEY_RS
set username sydney_rsi
set password sydney_rsi_ps
```

Example 2 Entered at TOKYO_RS, this command creates an indirect route from TOKYO_RS to SYDNEY_RS, through the intermediate Replication Server, MANILA_RS. Direct routes must already exist from TOKYO_RS to MANILA_RS and from MANILA_RS to SYDNEY_RS:

```
create route to SYDNEY_RS
set next site MANILA_RS
```

Example 3 This command creates a direct route similar to that in the first example. However, if network-based security is enabled, TOKYO_RS must log in to SYDNEY_RS with a credential:

```
create route to SYDNEY_RS
set unified_login 'required'
```

Usage

- Use `create route` to create a direct or indirect route from the current Replication Server to a remote Replication Server.
- Before creating a route, you should have determined your overall routing scheme. See the *Replication Server Administration Guide Volume 1* for information on creating and managing routes.
- Replication Server does not support routing schemes where routes diverge from the same source Replication Server, then converge at the same intermediate or destination Replication Server.
- Replication Server distributes information about the new route to qualifying sites through the replication system. The changes do not appear immediately at all such sites because of normal replication system lag time.
- If Replication Server is configured with Embedded RSSD (ERSSD), you can create a route as long as both Replication Servers are 15.0 or higher. If the route being created is the first route originating from the current site, log transfer will be started and a Replication Agent will be started automatically:

To change the Replication Agent's name, enter:

```
configure replication server
set erssid_ra to 'value'
```

Direct routes

- Specify an RSI user name and password and omit the next site clause from `create route` to set up a direct route from the current Replication Server to the destination Replication Server.
- Before you create a direct route, create login names and passwords in the destination Replication Server. You can use `rs_init` to set these up; the default user name is “*RS_name_rsi*” and the default password is “*RS_name_rsi_ps*.”

If a route is created with a user and password that do not exist at the destination Replication Server, add or change the user and password at that destination.

Indirect routes

- Include the next site clause in `create route` to set up an indirect route for Replication Server messages. For example, messages originating in New York and destined for all European sites can be routed through a London site, along an indirect route. Using indirect routes decreases the volume of messages passed through a portion of the route.
- Before you create an indirect route, you must first create a direct route from the source Replication Server to the intermediate Replication Server, and from the intermediate Replication Server to the destination Replication Server.
- A route can have any number of intermediate Replication Servers. However, because each additional intermediate Replication Server increases the lag time between the primary and replicate sites, you should limit the number of intermediate sites.

Routes and RSSD tables

- The RSI user name and password you specify when you create a direct route is added to the `rs_users` system table in the RSSD of the destination Replication Server. The user name and password are also added to the `rs_maintusers` system table in the RSSD of the source Replication Server.
- When you create a route, the source Replication Server sends the destination Replication Server the login name and password of the source RSSD's primary user. The destination Replication Server uses this login to create subscriptions to some of the RSSD system tables at the source Replication Server. This primary user login name is usually named "*source_RSSD_name_prim*," and is stored in the `rs_users` system table at the destination Replication Server.

Network-based security parameters

- Both ends of a route must use compatible Security Control Layer (SCL) drivers with the same security mechanisms and security features. It is the replication system Administrator's responsibility to choose and set security features for each server. The Replication Server does not query the security features of remote servers before it attempts to establish a connection.
- `create route` specifies network-based security settings that affect how the current Replication Server logs in to the target Replication Server and how secure message transmission is accomplished.

- If `unified_login` is set to “required,” *only* the “sa” user can log in to the Replication Server without a credential. If the security mechanism should fail, the “sa” user can then log in to Replication Server with a password and disable `unified_login`.
- A Replication Server can have more than one security mechanism; each supported mechanism is listed in the *libtcl.cfg* file under SECURITY.
- Message encryption is a costly process with severe performance penalties. In most instances, it may be wise to set `msg_confidentiality` “on” only for certain routes. Alternatively, choose a less costly security feature, such as `msg_integrity`.

Permissions

create route requires “sa” permission.

See also

alter connection, alter route, configure replication server, create connection, drop connection, drop route

create schedule

Description	Creates a schedule to execute shell commands at a time you specify.
Syntax	create schedule <i>sched_name</i> as ' <i>sched_time</i> ' [set {on off}] for exec ' <i>command</i> '
Parameters	<i>sched_name</i> The name of the schedule you provide, which: <ul style="list-style-type: none">• Must conform to the rules for identifiers• Must be unique• Can be 1 – 30 bytes long <i>sched_time</i> The time and day to execute ' <i>command</i> '. Provide the day and time in the restricted UNIX cron style with a single space separating the time and date parameters:

[*mm*] [*HH*] [*DOM*] [*MON*] [*DOW*]

The time and date parameters are:

Parameter	Description	Values
<i>mm</i>	Minutes past the hour	0 – 59. Use “*” to include all legal values.
<i>HH</i>	Hour in 24-hour notation	0 – 23. Use “*” to include all legal values.
<i>DOM</i>	Day of the month	1 – 31. Use “*” to include all days of the month.
<i>MON</i>	Month of the year	1 – 12. Use “*” to include all months of the year.
<i>DOW</i>	Day of the week	0 – 6 with 0=Sunday. Use “*” to include all days of the week.

- Use an asterisk “*” to specify all valid values. For example, “17 20 * * *” represents a daily schedule at 8:17 p.m.
- Use a comma “,” to separate values that are not part of a range. For example, “17 20 1,15 * *” represents 8:17 p.m. on the 1st and 15th of every month, where 1 and 15 are the values for the *DOM* parameter.
- Use a hyphen “-” between two values to specify a range of values, inclusive of the two values. For example, “17 20 * * 1-5” represents 8:17 p.m. from Monday to Friday where “1-5” are the range of values for the *DOW* parameter.
- For the *DOM*, *MON*, or *DOW* parameters, you can specify the day using both the *DOM* and *DOW* parameters. Replication Server follows all schedules you specify in the string. For example, “0 12 16 * 1” represents 12:00 p.m. every Monday and 12:00 p.m. on the 16th of every month.

set {on | off}

Enables or disables the schedule when you create it. By default, the schedule is on.

command

The shell command, such as scripts, executables, or batch files to execute at the specified schedule. Enclose in single quotes.

Shell commands:

- Must be in `$$SYBASE/$SYBASE_REP/sched` for UNIX or `%SYBASE%\%SYBASE_REP%\sched` for Windows
- Can include parameters delimited with a space within the shell command.
- In Windows, create schedule executes the command specified in the last parameter within the shell command or batch file. You must also include stdout to a file in the create schedule command line.

In shell command names, you:

- Can use only ASCII alphanumeric characters, such as A – Z, a – z, and 0 – 9
- Can use the “.”, “-”, and “_” characters
- Cannot use the “\” and “/” characters
- Must include the *.bat* suffix if you are executing on Windows. For example, the name should be *suspend_conn.bat* on Windows and *suspend_conn.sh* on UNIX.

Examples

Example 1 Create “schedule1” in Windows that suspends the connection to the pubs2 database in the SYDNEY_DS dataserver at 12:00 p.m. every Monday and 12:00 p.m. on the 16th of every month:

- 1 Create a text file, such as *sql.txt* that contains the actual Replication Server command line that you want to schedule. For example, *sql.txt* can contain

```
suspend connection to SYDNEY_DS.pubs2
go
```

- 2 Create a batch file, such as *suspend_conn.bat* in Windows that contains isql and relevant parameters to run the command line in *sql.txt*. For example, *suspend_conn.bat* can contain:

```
%SYBASE%\OCS-15_0\bin\isql.exe -Usa -P -S SYDNEY_DS -i
%SYBASE%\sql.ini -i %SYBASE%\REP-15_5\sched\sql.txt
```

- 3 Create the schedule, “schedule1”:

```
create schedule schedule1 as '0 12 16 * 1' for exec
'test.bat > c:\temp\test.out'
go
```

Example 2 Create “schedule2” to execute the *suspend_conn.sh* script in UNIX that suspends the connection to the pubs2 database in the SYDNEY_DS dataserver every day at 8:17p.m.:

```
create schedule schedule2 as '17 20 * * *' for exec
'suspend_conn.sh'
```

Example 3 Create “schedule2” to execute the *resume_conn.sh* script that resumes the connection to the pubs2 database in the SYDNEY_DS dataserver every day at 7:15 AM:

```
create schedule schedule2 as '15 7 * * *' for exec
'resume_conn.sh'
```

Usage

- Schedule the time you want to perform replication tasks – to report, for example, on a specific state of the replicate database while the replicate database is frozen and not receiving data from the primary database.
- You can schedule replication to happen only during specific night hours, so that the processing of the next day does not change the replicate database, and reporting can occur on the data of the day before, that is frozen in the replicate database. You can do this by creating schedules to suspend and resume connections to the replicate database at specific times of the day.

Permissions

create schedule requires “sa” permission.

See also

admin schedule, alter schedule, drop schedule

create subscription

Description	Creates and initializes a subscription and materializes subscription data. The subscription may be for a database replication definition, table replication definition, function replication definition, or publication.
Syntax	<pre>create subscription <i>sub_name</i> for {<i>table_repdef</i> <i>func_repdef</i> {publication <i>pub</i> database replication definition <i>db_repdef</i>} with primary at <i>server_name.db</i>} with replicate at <i>data_server.database</i> [where {<i>column_name</i> @<i>param_name</i>} {< > >= <= = &} <i>value</i> [and {<i>column_name</i> @<i>param_name</i>} {< > >= <= = &} <i>value</i>]...] [without holdlock incrementally without materialization] [subscribe to truncate table] [for new articles]</pre>
Parameters	<p><i>sub_name</i> The name of the subscription, which must conform to the rules for naming identifiers. The subscription name must be unique for the replication definition, where applicable, and for the replicate data server and database.</p> <p>for <i>table_rep_def</i> Specifies the table replication definition the subscription is for.</p> <p>for <i>function_rep_def</i> Specifies the name of the function replication definition the subscription is for.</p> <p>for publication <i>pub_name</i> Specifies the publication the subscription is for.</p> <p>for database replication definition <i>db_repdef</i> Specifies the database replication definition the subscription is for.</p> <p>with primary at <i>data_server.database</i> Include this clause for a subscription for a publication or a database replication definition. Specifies the location of the primary data. If the primary database is part of a warm standby application that uses logical connections, <i>data_server.database</i> is the name of the logical data server and database.</p> <p>with replicate at <i>data_server.database</i> Specifies the location of the replicate data. If the replicate database is part of a warm standby application, <i>data_server.database</i> is the name of the logical data server and database.</p>

where

Sets criteria for the column or parameter values that are to be replicated via the subscription. If you omit the where clause, all rows or parameters are replicated.

You can include a where clause in a subscription for a table or function replication definition. You cannot include a where clause in a database or publication subscription.

A where clause is composed of one or more simple comparisons, in which a searchable column or searchable parameter from the replication definition is compared to a literal value using one of these relational operators: <, >, <=, >=, =, or &. (The & operator is supported *only* for rs_address columns or parameters.) You can join comparisons using the keyword and.

Column or parameter names used in the expression must be included in the searchable columns list of the table replication definition or the searchable parameters list of the function replication definition.

Java columns cannot be evaluated in subscription expressions. Thus, you cannot include a Java column of type rawobject or rawobject in row in a where clause.

The maximum size of a where clause in a subscription is 255 characters.

Note You cannot convert binaries with less than seven bytes into integers. Workarounds include using zeros to pad binary values up to eight bytes, or using integer values instead of binary values.

column_name

A column name from the primary table, for a subscription to a table replication definition.

@param_name

A parameter name from a replicated stored procedure, for a subscription to a function replication definition.

value

A value for a specified column or parameter. See “Datatypes” on page 21 for entry formats for values for different datatypes.

Column or parameter names used in the expression must be included in the searchable columns or searchable parameters list of the replication definition.

without holdlock

Selects data from the primary database without a holdlock, for non-atomic materialization. The rows are applied at the replicate database in increments of 1000-row inserts per transaction. See “Nonatomic materialization” on page 330 for more information.

incrementally

Initializes the subscription and apply subscription data in increments of 1000-row inserts per transaction. A holdlock is used on the primary database, for atomic materialization.

without materialization

Does not materialize data for the subscription. Use this option when there is no activity at the primary database and the data already exists in the replicate database. Or, use this option when you have suspended activity in the primary database and manually transferred the data to the replicate database. Database subscriptions must include this option.

subscribe to truncate table

For a subscription to a table replication definition, a database replication definition, or to a publication, enables replication of the truncate table command to the subscribing replicate database.

You must set this option the same as it is set for any existing subscriptions that replicate data into the same replicate table for a particular database. Otherwise, the new subscription is rejected.

for new articles

Refreshes an existing subscription. Instructs Replication Server to check the subscription against the publication and then to create subscriptions against unsubscribed articles.

Examples

Example 1 Creates a subscription named titles_sub. It specifies that rows from the titles table with columns of the type “business” are to be replicated in the titles table in the pubs2 database of the data server named SYDNEY_DS:

```
create subscription titles_sub
for titles_rep
with replicate at SYDNEY_DS.pubs2
where type = 'business'
```

Example 2 Creates a subscription named titles_sub that includes rows from the titles table with prices that are greater than or equal to \$10.00:

```
create subscription titles_sub
for titles_rep
with replicate at SYDNEY_DS.pubs2
```

```
where price >= $10.00
```

Example 3 Creates a subscription named `myproc_sub` for the function replication definition `myproc_rep`. In order to use this command to create a subscription for a function replication definition, data must already exist at the replicate database, and you must use the `without materialization` clause:

```
create subscription myproc_sub
for myproc_rep
with replicate at SYDNEY_DS.pubs2
without materialization
```

Example 4 Creates a subscription named `pubs2_sub` for the publication `pubs2_pub`:

```
create subscription pubs2_sub
for publication pubs2_pub
with primary at TOKYO_DS.pubs2
with replicate at SYDNEY_DS.pubs2
```

Example 5 Creates a database subscription `pubs2_sub` for the database replication definition `pubs2_rep`:

```
create subscription pubs2_sub
for database replication definition pubs2_rep
with primary at NEWYORK_DS.pubs2
with replicate at TOKYO_DS.pubs2
without materialization
subscribe to truncate table
```

Usage

- To subscribe to a function or database replication definition, use `create subscription` with the `without materialization` clause, or use `define subscription` and the other bulk materialization commands.
- Execute `create subscription` at the Replication Server of the database where the replicated data will be stored.
- See the *Replication Server Administration Guide Volume 1*, for more information about subscriptions and the role they play in replication.
- If you need to change which replication definition a subscription is for, you must drop the subscription and re-create it, specifying the name of the replication definition to which you want to subscribe.

- You can create multiple replication definitions for the same primary table or database. You cannot subscribe to more than one replication definition for the same replicate table or database, although you can subscribe to the same replication definition more than once.
- If you want to materialize text, unitext, image, or rawobject data, you can use automatic materialization *only* if the size of your data row is less than 32K. Otherwise, you must use bulk materialization.

Subscribing to database replication definitions

- When you create a database subscription, you cannot use the where clause to limit data subscription. All data is subscribed.
- With database subscriptions, you can use only the no materialization or bulk materialization methods. Use define subscription to use dump and load or other bulk materialization method. Use create subscription to use the no materialization method.
- You cannot subscribe to more than one database replication definition from the same origin.
- If your replicate Replication Server is at lower version than your primary Replication Server, you cannot create a database subscription at the replicate Replication Server for a primary database controlled by the primary Replication Server.
- To successfully create a table replication definition for a primary database that is subscribed by a database subscription, the replicate Replication Server must be at the same or higher version as the table replication definition.

Subscribing to publications

- When a publication is valid, you can create a subscription for the publication in order to begin replication to a replicate database. All forms of subscription materialization are supported.
- When you create a publication subscription, Replication Server creates a separate underlying subscription for each article that the publication contains. Each article subscription uses the name of the parent publication subscription.
 - When you use atomic or non-atomic materialization, article subscriptions are materialized one at a time in the order that the articles were added to the publication.

- When you use `create subscription` with the `without materialization` clause, all article subscriptions are activated and validated at the same time.
- A subscription to a publication cannot include a `where` clause. Instead, you can customize replication to replicate sites by including one or more `where` clauses in each article the publication contains.

Specifying columns subject to HDS translations

- When you create a subscription that includes a `where` clause, make sure that the value in the `where` clause comparison is in the declared datatype format.
- Subscriptions that specify columns subject to class- or column-level translations in the `where` clause cannot be dematerialized automatically. You must use either the bulk or the no-materialization method.

Replicating *truncate table*

- When you create the first subscription, you can either include or not include the `subscribe to truncate table` option. Each subsequent subscription that replicates into the same table must follow the example of the first subscription. Otherwise, the subscription is rejected when you try to create it.
- You can change the current “subscribe to truncate table” status of a particular replicate table by executing `sysadmin apply_truncate_table`.

Requirements for executing *create subscription*

- In addition to the permissions listed below, make sure that these requirements are met before you execute `create subscription`.

For a subscription to a table replication definition:

- A replication definition exists for the primary table you are replicating, and the table is marked for replication with `sp_setreptable`.
- If you subscribe to tables marked using `sp_reptostandby`, you must configure the primary database connection using the `rep_as_standby` configuration parameter and configure RepAgent using `send_warm_standby_exacts`.
- Tables referenced in the replication definition exist in both the primary and the replicate database. Each table has the columns and datatypes defined in the replication definition.

This table is visible to the user creating the subscription and to the user maintaining it. The easiest way to achieve this is to have the Database Owner create the table.

For a subscription to a function replication definition:

- A replication definition exists for the stored procedure you are replicating, and the stored procedure is marked for replication with `sp_setrepproc`.
- Stored procedures referenced in the function replication definition exist in both the primary and replicate database. Each stored procedure has the parameters and datatypes defined in the function replication definition.

For a subscription to a publication:

- A publication exists that contains articles for the primary tables or stored procedure you are replicating. The articles specify replication definitions that meet the requirements described above.
- The publication is valid.
- These requirements apply when you create subscriptions in warm standby applications:
 - If the destination database is part of a warm standby application, the table must exist in both the active and standby databases. Both tables must be marked for replication using `sp_setreptable` or `sp_reptostandby`.
 - For a logical primary database, you cannot create a subscription while Replication Server is in the process of adding a standby database.
- If a primary Adaptive Server database contains a replicated table and another table that has the same name, the owner of the second (unreplicated) table cannot create a subscription to the replicated table without using custom `rs_select` or `rs_select_with_lock` function strings. For example:
 - If there is a replication definition for a primary table named `db.dbo.table1`, and
 - Database user “jane” owns a table named `db.jane.table1`, then
 - Jane cannot create a subscription to the replication definition for `db.dbo.table1` using the default function strings.

Requirements for
warm standby
applications

Requirements for
tables with the same
name

Atomic materialization

- The default method for materializing subscriptions with this command is atomic materialization. Atomic materialization locks the primary table and copies subscription data through the network in a single atomic operation.
- During atomic materialization, no rows appear at the replicate database until the select transaction has been completed in the primary database. If the subscription specifies a large number of rows, the select transaction can run for a long time, causing a delay at the replicate site.
- If you plan to use the atomic method of subscription materialization:
 - You or the Database Owner must own the primary table, or you must use user-defined function strings for select operations at the primary database.
 - The Database Owner or the maintenance user must own the replicate table, or you must use user-defined function strings for select operations at the replicate database. If the owner of the replicate table is different from the owner of the primary table, you must create a unique function string by using a distinct function-string class.

Requirements for
using atomic
materialization

Using the *without holdlock* or *incrementally* option

- The *without holdlock* or *incrementally* options are alternatives to the default atomic method of subscription materialization. When you specify these options, Replication Server applies the rows in batches, so that data appears at the replicate database a batch at a time.

As a result, during materialization, queries at the replicate database may return incomplete data for the subscription. This temporary condition ends when check subscription indicates the subscription is valid.

The *incrementally*
option

- The *incrementally* option is a variation of atomic materialization. Use this option for large subscriptions to avoid a long-running transaction at the replicate database. The subscription data is not applied atomically at the replicate database, so the data is available; however, it is incomplete until materialization has completed and the subscription is validated.
- When *incrementally* is used, the select is performed with a holdlock to maintain serial consistency with the primary database. The replicate table passes through states that occurred previously at the primary database.

In all cases, replicate data is consistent with the primary database by the time materialization completes and check subscription indicates that the subscription is valid.

Nonatomic materialization

- The without holdlock option uses non-atomic materialization. When this option is specified, materialization rows are selected from the primary database without a holdlock. This can introduce inconsistency if rows are updated at the primary database after the select. To correct inconsistencies, use set autocorrection on when using without holdlock.
- When data already exists at the replicate database, you can use atomic or non-atomic materialization instead of bulk materialization.

Requirements for using nonatomic materialization

- If you plan to use non-atomic method of subscription materialization:
 - Do not use without holdlock if you update data by distributing applied functions from the primary database or if you update the data with commutative functions. For example, if a stored procedure updates a row by incrementing the previous value of a column, the value may be incorrect when materialization has completed.
 - If the replicate minimal columns option is set for a replication definition, you cannot use without holdlock to create new subscriptions.
 - For non-atomic subscriptions, if a non-atomic subscription is materializing when switch active executes, it is marked “SUSPECT.”

Note If you are using create subscription with either atomic or non-atomic materialization methods and you have quoted identifiers in your replication definition, then you must alter your primary connection to allow the use of quoted identifiers.

No materialization

The without materialization clause specifies the no-materialization method. It provides an convenient way to create a subscription when the subscription data already exists at the replicate database.

Requirements for no materialization

- The subscription data must already exist at the replicate database.
- The primary and replicate database must be in sync.
- Activity must be stopped at the primary database so that there are no further updates in the Replication Server stable queue.

Using the *rs_address* datatype

- You can subscribe to replication definitions whose columns or parameters use the special datatype *rs_address*. This datatype allows a unique subscription resolution method, whereby bitmaps of the *rs_address* datatype (based on the underlying *int* datatype) are compared with a bitmask in a subscription's *where* clause. The bitmap comparison tells the primary Replication Server whether or not a replicate site should receive the data in each row.
- For *rs_address* columns or parameters *only*, the bitmap comparison operator *&* is supported in the *where* clause, as follows:

```
where rs_address_column1 & bitmask
[and rs_address_column2 & bitmask]
[and other_search_conditions]
```

- Replication Server does not replicate a row if the only changed columns are *rs_address* columns, unless the changed bits indicate that the row should be inserted or deleted at the replicate database.

Because of this filtering, *rs_address* columns in replicate databases may not be identical to the corresponding columns at the primary database. This optimizes applications that use *rs_address* columns to specify the destination replicate databases.

How the *rs_address* datatype works

- Each bit in an *rs_address* column field may represent a category of data, such as inventory or billing. In a subscription bitmask, you set the corresponding bit to “on” (1), for each category of data you want to replicate to the subscribing site.

For example, users at a warehouse site who are interested in inventory data would set the inventory bit to “on” in a subscription bitmap. If the same warehouse users are not interested in billing data, they would set that bit to “off” (0). When a bit is set to “on” in both a subscription bitmask and an *rs_address* column, the row containing the bit is replicated.

32-bit limitation of underlying *int* datatype for *rs_address*

- Due to the 32-bit limitation of the underlying *int* datatype, you may need to construct primary tables with more than one *rs_address* column. The *and* keyword allows you to create a single subscription to perform bitmap comparisons on more than one *rs_address* column.

However, to subscribe to a row when one or more bits are set in either of two or more *rs_address* columns, you must create separate subscriptions.

Using 32-bit hexadecimal numbers for `rs_address`

- You can also specify search conditions for non-`rs_address` columns using the `and` keyword and the comparison operators (other than `&`) described in the command syntax. If you use `and` to specify search conditions, subscription data may not be replicated or may migrate out of a subscription, even if `rs_address` bitmap comparisons would otherwise replicate a row.
- You can compare `rs_address` columns to 32-bit integer values or 32-bit hexadecimal numbers in the `where` clause. If you use hexadecimal numbers, pad each number with zeros, as necessary, to create an 8-digit hexadecimal value.

Warning! Be very cautious about comparing `rs_address` columns to hexadecimal numbers in the `where` clause of a subscription. Hexadecimal values are treated as binary strings by Adaptive Server and Replication Server. Binary strings are converted to integers by copying bytes. The resulting bit pattern may represent different integer values on different platforms.

For example, `0x0000100` represents 65,536 on platforms that consider byte 0 most significant, and represents 256 on platforms that consider byte 0 least significant. Because of these byte-ordering differences, bitmap subscriptions involving hexadecimal numbers may not work in a multi-platform replication system.

- See “Datatypes” on page 21 for more information about the `rs_address` and `int` datatypes. Also, see the *Replication Server Administration Guide Volume 1*.
- Refer to the *Adaptive Server Enterprise Reference Manual* and the *Open Client and Open Server Common Libraries Reference Manual* for more information about conversion between datatypes.

Monitoring a subscription

- When Replication Server materializes a subscription, it logs in to the primary data server, using the subscription creator’s login name, and selects the rows from the primary table. Use `check subscription` to monitor the progress of the materialization.
- `create subscription` returns a prompt before the data materialization is complete. Materialization is complete when `check subscription` reports “VALID” at the replicate Replication Server.

Permissions

To execute `create subscription`, you must have the following login names and permissions:

- The same login name and password at the replicate Replication Server, primary Replication Server, and primary Adaptive Server database.
- “create object” or “sa” permission at the replicate Replication Server where you enter this command.
- “create object”, “primary subscribe”, or “sa” permission at the primary Replication Server.
- select permission on the primary table in the primary Adaptive Server database.
- execute permission on the rs_marker stored procedure in the primary Adaptive Server database.
- The replicate database maintenance user must have select, insert, update, and delete permissions on the replicate table, and execute permissions for functions used in replication.

See also

alter applied function replication definition, alter database replication definition, alter request function replication definition, check subscription, create article, create database replication definition, create applied function replication definition, create function string, create publication, create replication definition, create request function replication definition, define subscription, drop subscription, set, sysadmin apply_truncate_table

create user

Description	Adds a new user login name to a Replication Server.
Syntax	<pre>create user <i>user</i> set password {<i>passwd</i> null}</pre>
Parameters	<p><i>user</i></p> <p>The new user's Replication Server login name. Login names must conform to the rules for identifiers.</p> <p><i>passwd</i></p> <p>The user's password. It can be up to 30 characters long and can include letters, numerals, and symbols. Case is significant. If the password contains spaces, enclose the password in quotation marks. When you create or alter a user login name, you must specify a password or "null." A null password lets a user log in without being prompted for a password.</p>
Examples	<p>Creates a new user login name "louise" with the password "EnnuI":</p> <pre>create user louise set password EnnuI</pre>
Usage	<ul style="list-style-type: none">• create user creates a new login name for a user.• Users can change their own passwords with the alter user command.• Case is significant for user login names and passwords.
Permissions	create user requires "sa" permission.
See also	alter user, drop user, grant, revoke

define subscription

Description	Adds a subscription to the Replication Server system tables, but does not materialize or activate the subscription. The subscription may be for a database replication definition, a table replication definition, a function replication definition, or for a publication. This command begins the process of bulk subscription materialization, or the process of refreshing a publication subscription.
Syntax	<pre>define subscription <i>sub_name</i> for {<i>table_rep_def</i> <i>function_rep_def</i> publication <i>pub_name</i> database replication definition <i>db_repdef</i> with primary at <i>data_server.database</i>} with replicate at <i>data_server.database</i> [where {<i>column_name</i> @<i>param_name</i>} {< > >= <= = &} <i>value</i> [and {<i>column_name</i> @<i>param_name</i>} {< > >= <= = &} <i>value</i>]...] [subscribe to truncate table] [for new articles] [use dump marker]</pre>
Parameters	<p><i>sub_name</i> The name of the subscription, which must conform to the rules for naming identifiers. The subscription name must be unique for the replication definition, where applicable, and for the replicate data server and database.</p> <p>for <i>table_rep_def</i> Specifies the table replication definition the subscription is for.</p> <p>for <i>function_rep_def</i> Specifies the name of the function replication definition the subscription is for.</p> <p>for publication <i>pub_name</i> Specifies the publication the subscription is for.</p> <p>for database replication definition <i>db_repdef</i> Specifies the database replication definition the subscription is for.</p> <p>with primary at <i>data_server.database</i> Specifies the location of the primary data. If the primary database is part of a warm standby application, <i>data_server.database</i> is the name of the logical data server and database. Include this clause only for a subscription for a publication.</p>

with replicate at *data_server.database*

Specifies the location of the replicate data. If the replicate database is part of a warm standby application, *data_server.database* is the name of the logical data server and database.

where

Sets criteria for the column or parameter values that are to be replicated via the subscription. If you omit the where clause, all rows or parameters are replicated.

You can include a where clause in a subscription for a table or function replication definition. You cannot include a where clause in a publication subscription.

A where clause is composed of one or more simple comparisons, in which a searchable column or searchable parameter from the replication definition is compared to a literal value using one of these relational operators: <, >, <=, >=, =, or &. (The & operator is supported *only* for rs_address columns or parameters.) You can join comparisons with the keyword and.

Column or parameter names used in the expression must be included in the searchable columns list of the table replication definition or in the searchable parameters list of the function replication definition.

Java columns cannot be evaluated in subscription expressions. Thus, you cannot include a Java column of type rawobject or rawobject in row in a where clause.

column_name

A column name from the primary table, for a subscription to a table replication definition.

@param_name

A parameter name from a replicated stored procedure, for a subscription to a function replication definition.

value

A value for a specified column or parameter. See “Datatypes” on page 21 for entry formats for values for different datatypes.

subscribe to truncate table

For a subscription to a table replication definition or to a publication, enables replication of the truncate table command to the subscribing replicate database.

You must set this option the same as it is set for any existing subscriptions that replicate data into the same replicate table. Otherwise, the new subscription will be rejected.

for new articles

Refreshes an existing subscription. Instructs Replication Server to check the subscription against the publication and then to create subscriptions against unsubscribed articles.

use dump marker

Tells Replication Server to apply transactions to a replicate database. use dump marker activates and validates the database subscription automatically. Without this option, users must activate and validate the database subscription manually.

Note Use dump marker one at a time as you cannot define multiple database subscriptions with dump marker. You also need to place a dump database command between each subscription command. If you are using the cross platform dump and load XPDL feature in an MSA replication, avoid using the use dump marker clause for materialization.

Examples

Example 1 Creates a subscription named titles_sub. It specifies that rows from the titles table with columns of the type “business” are to be replicated in the titles table in the pubs2 database of the data server named SYDNEY_DS:

```
define subscription titles_sub
  for titles_rep
    with replicate at SYDNEY_DS.pubs2
    where type = 'business'
```

Example 2 Creates a subscription named titles_sub that includes rows from the titles table with prices that are greater than or equal to \$10.00:

```
define subscription titles_sub
  for titles_rep
    with replicate at SYDNEY_DS.pubs2
    where price >= $10.00
```

Example 3 Creates a subscription named `myproc_sub` for the function replication definition `myproc_rep`:

```
define subscription myproc_sub
  for myproc_rep
    with replicate at SYDNEY_DS.pubs2
```

Example 4 Creates a subscription named `pubs2_sub` for the publication `pubs2_pub`:

```
define subscription pubs2_sub
  for publication pubs2_pub
    with primary at TOKYO_DS.pubs2
    with replicate at SYDNEY_DS.pubs2
```

Example 5 Creates a subscription `pubs2_sub` for the database replication definition `pubs2_rep`:

```
define subscription pubs2_sub
  for database replication definition pubs2_rep
    with primary at NEWYORK_DS.pubs2
    with replicate at TOKYO_DS.pubs2
    subscribe to truncate table
    use dump marker
```

Refer to the *Replication Server Design Guide* for examples of creating subscriptions for a complete replication system.

Usage

- Use `define subscription` to create a subscription manually using bulk materialization. With bulk materialization, subscription creation and materialization is performed in discrete steps so that you can load the initial data from media rather than sending it from the primary database through the WAN.
- If you have added any new articles to a publication with an existing subscription, you must refresh the publication subscription in order to create new subscriptions for these articles.
- Activate the subscription using `activate subscription` and validate the subscription using `validate subscription`.
- Although you can create multiple replication definitions for the same primary table, you cannot subscribe to more than one replication definition for the same replicate table. However, you can subscribe to the same replication definition more than once.

Subscribing to publications

- You can create a subscription for a valid publication to begin replication to a replicate database. All forms of subscription materialization are supported.
- Use `define subscription` to create new article subscriptions in the publication subscription. Then use `activate subscription`, manually load the subscription data for the new article subscriptions, and use `validate subscription` to validate the publication subscription.
- When you create a publication subscription, Replication Server creates a separate underlying subscription for each article that the publication contains. Each article subscription uses the name of the parent publication subscription.
- When you activate and validate a publication subscription, all of its article subscriptions are activated and validated at the same time.
- A subscription to a publication cannot include a `where` clause. Instead, you can customize replication to replicate sites by including one or more `where` clauses in each article the publication contains.

Subscribing to database replication definitions

- When you create a database subscription, you cannot use the `where` clause to limit data subscription. All data is subscribed.
- With database subscriptions, you can use only the no materialization or bulk materialization methods. Use `define subscription` to use `dump` and `load` or other bulk materialization method. Use `create subscription` to use the no materialization method.
- You cannot subscribe to more than one database replication definition from the same origin.

Replicating *truncate table*

- When you create the first subscription for a table, you can either include or not include the `subscribe to truncate table` option. Each subsequent subscription that copies information into the same table must follow the example of the first subscription. Otherwise, it will be rejected when you try to create it.
- You can view or change the current “subscribe to truncate table” status of a particular replicate table by executing `sysadmin apply_truncate_status`.

Working with the *rs_address* datatype

See create subscription for information about working with columns or parameters that use the *rs_address* datatype.

Requirements for executing *define subscription*

- In addition to the permissions listed below, make sure these requirements are met before you execute this command.
- For a subscription to a table replication definition:
 - A replication definition exists for the primary table you are replicating, and the table is marked for replication with *sp_setreptable*.
 - Tables referenced in the replication definition exist in both the primary and the replicate database. Each table has the columns and datatypes defined in the replication definition.

This table is also visible to the user creating the subscription and the user maintaining it. The easiest way to achieve this is to have the Database Owner create the table.

For a subscription to a function replication definition:

- A replication definition exists for the stored procedure you are replicating, and the stored procedure is marked for replication with *sp_setrepproc*.
- Stored procedures referenced in the function replication definition exist in both the primary and replicate database. Each table has the parameters and datatypes defined in the function replication definition.

For a subscription to a publication:

- A publication exists that contains articles for the primary tables or stored procedure you are replicating. The articles specify replication definitions that meet the requirements described above.
- The publication is valid.

Creating subscriptions using *define subscription*

- You can use *define subscription* to subscribe to a table replication definition, a function replication definition, or a publication.
 - For a subscription to a table replication definition, enter *define subscription* at the Replication Server that manages the database where the replicate data is to be stored.

- For a subscription to a function replication definition, enter `define subscription` at the Replication Server that manages the database where the destination stored procedure is to be executed via applied function delivery.
- For a subscription to a publication, enter `define subscription` at the Replication Server that manages the database where the replicate data is to be stored or where destination stored procedures are to be executed.
- A table subscription maintains a replicate copy of a table, or selected rows from a table, in a database. Changes made to the primary version are also applied to the copy.
- A function subscription replicates user-defined function invocations associated with a function replication definition. A replicated function typically includes parameters and modifies data, but it need not involve replicated data.
- A publication subscription involves underlying subscriptions for the articles the publication contains, which replicate table or user-defined function invocations depending on the replication definitions in the article.
- See the *Replication Server Administration Guide Volume 1* for more information about subscriptions and the role they play in replication.

Alternative command to create subscriptions

- Use `create subscription` to create, materialize, activate, and validate, in a single step, a subscription for a table replication definition, function definition replication, or publication.

Permissions

To execute `define subscription`, you must have the following login names and permissions:

- The same login name and password at the replicate Replication Server, primary Replication Server, and primary database.
- “create object” or “sa” permission at the replicate Replication Server where you enter this command.”
- “create object”, “primary subscribe”, or “sa” permission at the primary Replication Server.

See also

`alter applied function replication definition`, `alter request function replication definition`, `activate subscription`, `check subscription`, `create article`, `create function replication definition`, `create publication`, `create applied function replication definition`, `create request function replication definition`, `create subscription`, `drop subscription`, `sysadmin apply_truncate_table`, `validate subscription`

disconnect

Description	Terminates connection to a server.
Syntax	{disconnect disc} [all]
Examples	<p>Creates a connection from ost_replinuxvm_02 to from ost_replinuxvm_03, and then ost_replinuxvm_02 disconnects from ost_replinuxvm_03:</p> <pre>isql -Usa -P -S ost_replinuxvm_02 1> connect to ost_replinuxvm_03 2> go Gateway connection to 'ost_replinuxvm_03' is created. 1> disc 2> go Gateway connection to 'ost_replinuxvm_03' is dropped.</pre>
Usage	<ul style="list-style-type: none">• disconnect exits the connection stack one at a time. To exit from all the connections, use disconnect all.• The disconnect command behaves differently in Replication Server 15.1 and earlier. In these versions, a disconnect command terminates the gateway mode, and returns the working server status to the Replication Server that issued the first connect command. When your connection stack includes Replication Server versions 15.2, and 15.1 or earlier, and you issue a disconnect command, the show connection and show server commands may not display the expected output.
Permissions	Any user may execute this command.
See also	connect, show connection, show server

drop article

Description	Drops an article and optionally drops its replication definition.
Syntax	<pre>drop article <i>article_name</i> for <i>pub_name</i> with primary at <i>data_server.database</i> [drop_repdef]</pre>
Parameters	<p><i>article_name</i> The name of the article to drop.</p> <p>for <i>pub_name</i> Specifies the name of the publication the article is for.</p> <p>with primary at <i>data_server.database</i> Specifies the location of the primary data. If the primary database is part of a warm standby application, <i>data_server.database</i> is the name of the logical data server and database.</p> <p>drop_repdef An optional keyword that causes the replication definition the article is for to be dropped—if it is not used elsewhere.</p>
Examples	<p>Example 1 Drops the article named <code>titles_art</code> for the publication <code>pubs2_pub</code> in the <code>TOKYO_DS.pubs2</code> database:</p> <pre>drop article titles_art for pubs2_pub with primary at TOKYO_DS.pubs2</pre> <p>Example 2 Drops the article named <code>titles_art</code> for the publication <code>pubs2_pub</code> in the <code>TOKYO_DS.pubs2</code> database. This command also drops the replication definition the article is for, if it is not used elsewhere:</p> <pre>drop article titles_art for pubs2_pub with primary at TOKYO_DS.pubs2 drop_repdef</pre>
Usage	<ul style="list-style-type: none">• Use <code>drop article</code> to remove an article from a publication. Execute <code>drop article</code> at the Replication Server that manages the database where the primary data is stored.• You can drop an article if there are no subscriptions for the article. Drop subscriptions first, as necessary.

- Optionally, you can also drop the replication definition for the article, if it is not part of any other article and has no subscriptions.
- A dropped article is removed at the replicate site only when create/define subscription is executed there.

Dropping articles from a publication with a subscription

- If you drop an article from an existing publication, the publication is invalidated. You must drop all existing article subscriptions using drop subscription for article before the article can be dropped. To create new publication subscriptions you must:
 - Validate the publication when you have completed making changes to the publication, then

See create subscription and define subscription for more information on the two methods of refreshing publication subscriptions.

Permissions

drop article requires “create object” permission.

See also

check subscription, create article, create publication, create subscription, define subscription, drop function replication definition, drop publication, drop replication definition, drop subscription

drop connection

Description	Removes a database from the replication system.
Syntax	drop connection to <i>data_server.database</i>
Parameters	<i>data_server</i> The name of the data server with the database to be removed from the replication system. <i>database</i> The name of the database whose connection is to be dropped.
Examples	Drops the connection to the pubs2 database in the SYDNEY_DS data server: <pre>drop connection to SYDNEY_DS.pubs2</pre>
Usage	<ul style="list-style-type: none">• Use drop connection to remove database connection information from the Replication Server system tables. This command does not remove replicated data from any database in the system.• Before you drop a connection:<ul style="list-style-type: none">• Drop any subscriptions that replicate data to the database.• If the connection is to a primary database, drop any replication definitions for tables in the database.• Before you re-create a connection to a database with the same name, you may need to use sysadmin dropdb.• Replication Server distributes information about the dropped database connection to qualifying sites through the replication system. The changes do not appear immediately at all such sites because of normal replication system lag time.
Permissions	drop connection requires “sa” permission.
See also	alter connection, create connection, resume connection, suspend connection, sysadmin dropdb

drop database replication definition

Description	Deletes an existing database replication definition.
Syntax	drop database replication definition <i>db_repdef</i> with primary at <i>server_name.db</i>

Parameters	<p><i>db_repdef</i> Name of the database replication definition.</p> <p><i>server_name.db</i> Name of the primary server/database combination. For example: TOKYO.dbase.</p>
Examples	<p>Deletes the database replication definition dbrep1:</p> <pre>drop database replication definition dbrep1 with primary at PDS.my_db</pre>
Usage	<p>drop database replication definition succeeds only if there is no database subscription to the named database replication definition.</p>
See also	<p>alter database replication definition, create database replication definition</p>

drop error class

Description	Drops an error class and any actions associated with it.
Syntax	<code>drop [replication server] error class <i>error_class</i></code>
Parameters	<p><code>replication server</code> Indicates that the error class is a Replication Server error class and not a data server error class.</p> <p><i>error_class</i> The name of the error class to drop.</p>
Examples	<p>Example 1 Drops the <code>pubs2_db_err_class</code> error class from the Replication Server. Also drops any error actions that were assigned for the <code>pubs2_db_err_class</code> error class:</p> <pre>drop error class pubs2_db_err_class</pre> <p>Example 2 Drops the <code>sydney_rs_err_class</code> Replication Server error class from the Replication Server. Also drops any error actions that were assigned for the <code>sydney_rs_err_class</code> error class:</p> <pre>drop replication server error class sydney_rs_err_class</pre>
Usage	<ul style="list-style-type: none">• Use the drop error class command to remove an error class. When an error class is dropped, all actions assigned for it are also dropped.• You execute drop error class at the Replication Server where the error class was created.• You cannot drop:<ul style="list-style-type: none">• The <code>rs_sqlserver_error_class</code> error class.• The <code>rs_repserver_error_class</code> error class.• An error class that is in use with a database• To change the primary site for an error class, use the move primary of error class command.• Replication Server distributes information about the dropped class to qualifying sites through the replication system. The changes do not appear immediately at all such sites because of normal replication system latency.
Permissions	drop error class requires “sa” permission.
See also	assign action, alter error class, create connection, create error class, drop connection, move primary

drop function

Description	Drops a user-defined function and its function strings.
Syntax	<code>drop function [<i>replication_definition</i>.]<i>function</i></code>
Parameters	<p><i>replication_definition</i></p> <p>The name of the replication definition the function was created for.</p> <p><i>function</i></p> <p>The name of the function to drop.</p>
Examples	<p>Drops the <code>upd_publishers</code> user-defined function for the <code>publishers_rep</code> replication definition. Also drops any function strings defined for the function:</p> <pre>drop function publishers_rep.upd_publishers</pre>
Usage	<ul style="list-style-type: none">• Use drop function to remove a function name and any function strings that have been created for it.• Execute drop function at the Replication Server where the replication definition was created.• You cannot drop system functions. For more information about system functions, see Chapter 4, “Replication Server System Functions.”• Replication Server distributes information about the dropped user-defined function to qualifying sites through the replication system. The changes do not appear immediately at all such sites because of normal replication system lag time.• When you drop a user-defined function for a replication definition, it is dropped for all replication definitions in the primary table.• Do not execute drop function for replicated functions. Use drop function rep def instead.
Permissions	drop function requires “create object” permission.
See also	create function, drop function string, move primary

drop function replication definition

Description	Drops a function replication definition and its user-defined function.
Syntax	drop function replication definition <i>function_rep_def</i>
Parameters	<i>function_rep_def</i> The name of the function replication definition to drop.
Examples	Drops the function replication definition named <code>titles_frep</code> and its user-defined function and function string: <pre>drop function replication definition titles_frep</pre>
Usage	<ul style="list-style-type: none">• Use drop function replication definition to remove a function replication definition.• Before you can drop a function replication definition, you must drop all subscriptions for it.• Execute drop function replication definition at the primary Replication Server for the function replication definition.• After you drop the stored procedure defined by this function replication definition, execute <code>sp_setrepproc</code> in the database, setting the procedure's replicate status to 'false'. This stops the RepAgent from transferring log entries to the Replication Server.• Replication Server distributes information about the dropped function replication definition to qualifying sites through the replication system. The changes do not appear immediately at all such sites because of normal replication system lag time.
Permissions	drop function replication definition requires “create object” permission.
See also	alter applied function replication definition, alter request function replication definition, check subscription, create applied function replication definition, create request function replication definition, create subscription, define subscription, drop subscription

drop function string

Description	Drops a function string for a function-string class.
Syntax	<code>drop function string</code> <code>[<i>replication_definition</i>.]<i>function</i>[:<i>function_string</i> all]</code> <code>for <i>function_class</i></code>
Parameters	<p><i>replication_definition</i></p> <p>The name of the table or function replication definition the function operates on.</p> <p><i>function</i></p> <p>The name of the function the function string was created for.</p> <p><i>function_string</i></p> <p>The name of the function string to drop. The default function string name is the same as the function name.</p> <p>all</p> <p>Causes Replication Server to drop all function strings for a function. Although only the <code>rs_select</code>, <code>rs_select_with_lock</code>, <code>rs_datarow_for_writetext</code>, <code>rs_get_textptr</code>, <code>rs_textptr_init</code>, and <code>rs_writetext</code> functions can have multiple function strings, this option can be used as shorthand for the <i>function_string</i> name.</p> <p><i>function_class</i></p> <p>The name of the function-string class from which the function string will be dropped.</p>

Examples

Example 1 Drops the function strings for the `rs_insert` function for the `publishers_rep` replication definition in the derived class `sqlserver_derived_class`. The `rs_insert` function string will now be inherited from the parent class:

```
drop function string
publishers_rep.rs_insert
for sqlserver_derived_class
```

Example 2 Drops the function string for the `upd_publishers` user-defined function for the `publishers_rep` replication definition in the `sqlserver2_function_class` function-string class:

```
drop function string
publishers_rep.upd_publishers
for sqlserver2_function_class
```

Example 3 Drops all function strings for the `rs_select_with_lock` function for the `publishers_rep` replication definition in the class `sqlserver2_func_class`:

```
drop function string
publishers_rep.rs_select_with_lock;all
for sqlserver2_func_class
```

Usage

- To replace an existing function string with a new one, use either `alter function string` or `create function with overwrite`.

Warning! If a transaction occurs between the time a function string is dropped and the time it is re-created, Replication Server detects the function string as missing and fails the transaction.

- Dropping a function drops corresponding function strings from all function-string classes.
- Dropping a customized function string from a derived function-string class causes that class to inherit the function-string from its parent class.
- Dropping a customized function string from `rs_sqlserver_function_class` causes Replication Server to delete the customized and default function string. To revert the customized function string to the default function string for a function in the `rs_sqlserver_function_class`, use `alter function string` and omit the output clause.
- Replication Server distributes information about the dropped function string to qualifying sites through the replication system. The changes do not appear immediately at all such sites because of normal replication system lag time.

Permissions

`drop function string` requires “create object” permission.

See also

`alter function string`, `create function`, `create function string`, `create function string class`, `drop function`

drop function string class

Description	Drops a function-string class.
Syntax	drop function string class <i>function_class</i>
Parameters	<i>function_class</i> The name of the function-string class to drop.
Examples	<p>Example 1 Drops the derived function-string class <code>sqlserver_derived_class</code> and all of its customized function strings:</p> <pre>drop function string class sqlserver_derived_class</pre> <p>Example 2 Drops the function-string class <code>sqlserver2_function_class</code> and its function strings:</p> <pre>drop function string class sqlserver2_function_class</pre>
Usage	<ul style="list-style-type: none">• Use drop function string class to remove a function-string class. function-string classes group all function strings for a database.• Dropping a function-string class also drops all of the associated function strings and removes all references to the class.• A function-string class that is still in use on a database connection cannot be dropped.• You cannot drop any of the three system-provided classes, <code>rs_sqlserver_function_class</code>, <code>rs_default_function_class</code>, or <code>rs_db2_function_class</code>.• You cannot drop any function-string class that is a parent class for an derived class.
Permissions	drop function string class requires “sa” permission.
See also	create function string class, drop function, drop function string

drop logical connection

Description	Drops a logical connection. Logical connections are used to manage warm standby applications.
Syntax	drop logical connection to <i>data_server.database</i>
Parameters	<i>data_server</i> The logical data server specified in the create logical connection command. <i>database</i> The name of the database specified in the create logical connection command.
Examples	Drops the logical connection for a data server named LDS and a database named <i>pubs2</i> : <pre>drop logical connection to LDS.pubs2</pre>
Usage	<ul style="list-style-type: none">• Use this command to drop a logical connection when you are dismantling a warm standby application.• Before you can drop the logical connection, you must drop the connection to the standby database.
Permissions	drop logical connection requires “sa” permission.
See also	create connection, create logical connection, drop connection, switch active

drop partition

Description	Removes a disk partition from the Replication Server.
Syntax	drop partition <i>logical_name</i>
Parameters	<i>logical_name</i> The name assigned to a partition created with create partition.
Examples	Drops the partition named P1 from the Replication Server: <pre>drop partition P1</pre>
Usage	<ul style="list-style-type: none">Use drop partition to remove a disk partition. This command first marks the partition as “pending drop.” Once it is marked, no new data is written on the partition. After all of the data stored on the partition has been successfully delivered, the partition is dropped. <hr/> <p>Note If not all the data stored on the partition is ready to drop, drop partition can create confusing behavior. For example, when a partition queue contains a segment that is filled only partially, the queue cannot drop until the segment is filled. Since the partition is designated “pending drop,” the segment cannot fill, and the command fails to drop the partition.</p> <hr/> <ul style="list-style-type: none">For a complete discussion of recovering from failed partitions, see the <i>Replication Server Administration Guide Volume 2</i>.
Permissions	drop partition requires “sa” permission.
See also	admin disk_space, alter partition, create partition

drop publication

Description	Drops a publication and all of its articles, and optionally drops the replication definitions for the articles.
Syntax	<pre>drop publication <i>pub_name</i> with primary at <i>data_server.database</i> [drop_repdef]</pre>
Parameters	<p><i>pub_name</i> The name of the publication to drop.</p> <p>with primary at <i>data_server.database</i> Specifies the location of the primary data. If the primary database is part of a warm standby application, <i>data_server.database</i> is the name of the logical data server and database.</p> <p>drop_repdef An optional keyword that causes the replication definitions for the publication's articles to be dropped—if it is not used elsewhere.</p>
Examples	<p>Example 1 Drops the publication named pubs2_pub for the primary database TOKYO_DS.pubs2:</p> <pre>drop publication pubs2_pub with primary at TOKYO_DS.pubs2</pre> <p>Example 2 Drops the publication named pubs2_pub for the primary database TOKYO_DS.pubs2. This command also drops all the replication definitions for the publication's articles, for replication definitions that are not used elsewhere:</p> <pre>drop publication pubs2_pub with primary at TOKYO_DS.pubs2 drop_repdef</pre>
Usage	<ul style="list-style-type: none"> • Use drop publication to remove a publication. Execute drop publication at the Replication Server that manages the database where the primary data is stored. • You can drop a publication if there are no subscriptions for the publication. Drop subscriptions first, as necessary. • When you drop a publication, its articles are also dropped. Optionally, you can also drop all of the replication definitions for the publication's articles, if they are not part of any other article and have no subscriptions.

- A dropped publication is removed from a replicate site when `define/create subscription` or `check publication` is executed there for the publication.

Permissions

`drop publication` requires “create object” permission.

See also

`check publication`, `create publication`, `drop article`, `drop function replication definition`, `drop replication definition`, `drop subscription`

drop replication definition

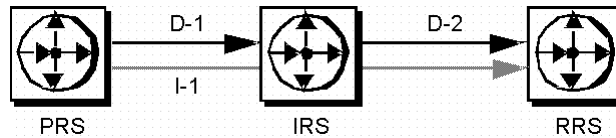
Description	Drops a replication definition and its functions.
Syntax	drop replication definition <i>replication_definition</i>
Parameters	<i>replication_definition</i> The name of the replication definition to drop.
Examples	Drops the replication definition named publishers_rep and any function strings that exist for it: <pre>drop replication definition publishers_rep</pre>
Usage	<ul style="list-style-type: none">• Use drop replication definition to remove a replication definition. Before a replication definition can be dropped, all subscriptions for it must be dropped.• Execute drop replication definition at the primary Replication Server for the replication definition.• If the dropped replication definition is the last replication definition for a primary table stored in an Adaptive Server, then, execute sp_setreplicate in the database after the replication definition is dropped. Set the table's replicate status to false to stop the Adaptive Server from logging special replication records for the table.• If you use more than one version of Replication Server (for example, Replication Server version 11.5 and version 11.0.x) and create multiple replication definitions for the same primary table, the first replication definition created, which has the same primary and replicate table names, the same primary and replicate column names, and does not include table owner name, is marked and propagated to Replication Servers of version 11.0.x or earlier. When a replication definition that was propagated to a Replication Server of version 11.0.x or earlier is dropped, the oldest replication definition (if there is one) compatible with 11.0.x is propagated to 11.0.x or earlier sites. See create replication definition for more information about working with replication definitions in a mixed-version environment. <ul style="list-style-type: none">• Replication Server distributes information about the dropped replication definition to qualifying sites through the replication system. The changes do not appear immediately at all sites because of normal replication system lag time.
Permissions	drop replication definition requires “create object” permission.

See also

alter replication definition, check subscription, create replication definition, create subscription, define subscription, drop article, drop publication, drop subscription, rs_send_repserver_cmd

drop route

Description	Closes the route to another Replication Server.
Syntax	<code>drop route to <i>dest_replication_server</i> [with nowait]</code>
Parameters	<i>dest_replication_server</i> The name of the Replication Server whose route is to be dropped. <i>with nowait</i> Instructs Replication Server to close the route, even if it cannot communicate with the destination Replication Server. Use with <i>nowait</i> only as a last resort. This clause forces Replication Server to drop a route that has subscriptions or is used by an indirect route. Additional steps are usually required to remove the invalid references from the RSSDs of the affected Replication Servers.
Examples	Drops the route from the site where the command is entered to the SYDNEY_RS Replication Server: <code>drop route to SYDNEY_RS</code>
Usage	<ul style="list-style-type: none">• <code>drop route</code> closes the route from the Replication Server where it is entered to the specified Replication Server.• Before dropping a route, you must:<ul style="list-style-type: none">• At the destination Replication Server, drop all subscriptions for primary data in databases managed by the source Replication Server.• Drop any indirect routes that use the route. <p>For example, in Figure 3-4, route I-1 is an indirect route from the primary Replication Server (PRS) to the replicate Replication Server (RRS) via the intermediate Replication Server (IRS). It uses direct routes D-1 and D-2.</p>

Figure 3-4: Example of direct and indirect routes

Before you can drop direct route D-2, you must drop all subscriptions at the replicate Replication Server for replication definitions at the primary or intermediate Replication Server, then drop indirect route I-1.

Warning! Use the `with nowait` clause only if you will never use the destination Replication Server again or if you must drop the route from the source Replication Server while the destination Replication Server is unavailable. Avoid the `with nowait` clause whenever possible so that the destination Replication Server can be updated correctly.

- After dropping a route using `with nowait`, you can use `sysadmin purge_route_at_replicate` at the (former) destination site to remove subscriptions and route information from the system tables at the destination.
- If the Replication Server from which the route is to be dropped is an intermediate site for another Replication Server, the route cannot be dropped. See the *Replication Server Administration Guide Volume 1* for more information.
- For Replication Servers with ERSSD, if the route being dropped is the last route originating from this source, then:
 - ERSSD Replication Agent is shut down
 - Log transfer is turned off from the ERSSD at the end of dropping the route

Permissions

drop route requires “sa” permission.

See also

alter route, create connection, create route, `sysadmin purge_route_at_replicate`

drop schedule

Description	Drops a schedule that executes commands.
Syntax	drop schedule <i>sched_name</i>
Parameters	<i>sched_name</i> The name of the schedule to drop.
Examples	To delete schedule1 enter: <pre>drop schedule 'schedule1'</pre>
Usage	Deletes a schedule from Replication Server.
Permissions	drop schedule requires “sa” permission.
See also	admin schedule, alter schedule, create schedule

drop subscription

Description	Drops a subscription to a database replication definition, table replication definition, function replication definition, article, or publication.
Syntax	<pre>drop subscription <i>sub_name</i> for {<i>table_rep_def</i> <i>function_rep_def</i> {article <i>article_name</i> in <i>pub_name</i> publication <i>pub_name</i> database replication definition <i>db_repdef</i> with primary at <i>data_server.database</i>} with replicate at <i>data_server.database</i> [without purge [with suspension [at active replicate only]] [incrementally] with purge]</pre>
Parameters	<p><i>sub_name</i> The name of the subscription to drop. If you are dropping a subscription for an article within a publication, specify the publication subscription name.</p> <p>for <i>table_rep_def</i> Specifies the name of the table replication definition the subscription is for.</p> <p>for <i>function_rep_def</i> Specifies the name of the function replication definition the subscription is for.</p> <p>for article <i>article_name</i> in <i>pub_name</i> Specifies the name of the article the subscription is for and the name of the publication that contains the article.</p> <p>for publication <i>pub_name</i> Specifies the name of the publication the subscription is for.</p> <p>for database replication definition <i>db_repdef</i> Specifies the name of the database replication definition the subscription is for.</p> <p>with primary at <i>data_server.database</i> Specifies the location of the primary data. If the primary database is part of a warm standby application, <i>data_server.database</i> is the name of the logical data server and database. Include this clause only for a subscription to a publication or a subscription to an article.</p> <p>with replicate at <i>data_server.database</i> Specifies the location of the replicate data. If the replicate database is part of a warm standby application, <i>data_server.database</i> is the name of the logical data server and database.</p>

without purge

Instructs Replication Server to leave rows replicated by a subscription in the replicated copy.

A subscription to a function replication definition is always dropped without purging replicate data. For a subscription to a table replication definition or a publication, you must choose either *without purge* or *with purge*. For a subscription to a database replication definition, you must include *without purge*.

with suspension

Used with the *without purge* clause, suspends the DSI after the subscription is dropped so that you can manually delete subscription rows. If the database is part of a warm standby application, *with suspension* suspends the DSI threads for the active and the standby databases. Delete subscription rows from both databases.

with suspension at active replicate only

Used with the *without purge* clause, suspends the DSI after the subscription is dropped so that you can manually delete subscription rows. In a warm standby application, the standby DSI is not suspended. This allows Replication Server to replicate delete transactions from the active database to the standby database.

incrementally

Used with the *with purge* clause, specifies that deletes are made 1000 rows at a time.

with purge

Used with a table replication definition, article, or publication, instructs Replication Server to remove rows (in the replicate table) that were replicated by a subscription.

A subscription to a function replication definition is always dropped without purging replicate data. For a subscription to a table replication definition or a publication, you must choose either *without purge* or *with purge*.

Examples

Example 1 Drops the `authors_sub` subscription for the `authors_rep` table replication definition. The replicate data is in the `pubs2` database of the `SYDNEY_DS` data server. The rows replicated via the subscription are purged from the replicate table, where they are not part of another subscription:

```
drop subscription authors_sub
for authors_rep
with replicate at SYDNEY_DS.pubs2
with purge
```

Example 2 Drops the titles_sub subscription for the titles_rep table replication definition. The replicate data is in the pubs2 database of the SYDNEY_DS data server. The rows replicated via the subscription remain in the replicate table:

```
drop subscription titles_sub
for titles_rep
with replicate at SYDNEY_DS.pubs2
without purge
```

Example 3 Drops the myproc_sub subscription for the myproc_rep function replication definition. The replicate data is in the pubs2 database of the SYDNEY_DS data server. No subscription data is purged:

```
drop subscription myproc_sub
for myproc_rep
with replicate at SYDNEY_DS.pubs2
```

Example 4 Drops the subscription for the article titles_art that is part of the subscription pubs2_sub for the publication pubs2_pub. The primary data is in the pubs2 database of the TOKYO_DS data server and the replicate data is in the pubs2 database of the SYDNEY_DS data server. The rows that were replicated via the subscription remain in the affected replicate tables. After dropping the article subscription you can drop the article:

```
drop subscription pubs2_sub
for article titles_art in pubs2_pub
with primary at TOKYO_DS.pubs2
with replicate at SYDNEY_DS.pubs2
without purge
```

Example 5 Drops the subscription named pubs2_sub for the pubs2_pub publication, where the primary data is in the pubs2 database of the TOKYO_DS data server and the replicate data is in the pubs2 database of the SYDNEY_DS data server. The rows that were replicated via the subscription are purged from the affected replicate tables, where they are not part of another subscription:

```
drop subscription pubs2_sub
for publication pubs2_pub
with primary at TOKYO_DS.pubs2
with replicate at SYDNEY_DS.pubs2
with purge
```

Example 6 Deletes a database subscription named `pubs2_sub`. The `without purge` option ensures that Replication Server does not remove rows added by the subscription to the replicate:

```
drop subscription pubs2_sub
  for database replication definition pubs2_rep
    with primary at NEWYORK_DS.pubs2
    with replicate at TOKYO_DS.pubs2
    without purge
```

Usage

- When you drop a subscription, Replication Server stops replicating the data specified by the subscription.
- Execute `drop subscription` at the Replication Server where you created the subscription.
- You cannot drop a table replication definition, function replication definition, article, or publication until you have dropped all subscriptions for the object.

The *without purge* clause

- Use `without purge` to drop a subscription to a table or database replication definition or to a publication. Replicated rows remain in the replicate tables.
- When you drop a subscription to a table replication definition or publication, you *must* specify either `without purge` or `with purge`.
- When you drop a subscription to a function replication definition, it is *always* dropped “without purge”—you do not need to specify `without purge`.
- When you drop a publication subscription “without purge,” all of its article subscriptions are dropped together.

The *with purge* clause

- Use the `with purge` clause to delete the rows (in the replicate table) that were replicated by the subscription. All subscription rows are purged unless they belong to another subscription at the replicate site.
- When you use `with purge`, Replication Server selects, from the replicate database, the set of rows that could be deleted. It then evaluates the selected rows against other subscriptions and determines whether to delete the row. The maintenance user for the replicate database must have `select` permission on the table.
- Deletes using `with purge` occur in a single transaction performed by an `rs_select_with_lock` function string in the replicate database.

- Deletes using with purge and incrementally occur 1000 rows at a time. This operation is performed by an `rs_select` function string in the replicate database.
- When you drop a publication subscription “with purge,” its article subscriptions are dropped one at a time in the reverse order that the articles were added to the publication.

Permissions

drop subscription requires “create object” permission at the replicate site and “primary subscribe” permission at the primary Replication Server.

drop subscription ... with purge also requires that the maintenance user have select permission for the replicate table.

See also

check subscription, create subscription, define subscription, drop article, drop function replication definition, drop publication, drop replication definition, resume connection, `rs_select`, `rs_select_with_lock`

drop user

Description	Drops a Replication Server user login name.
Syntax	<code>drop user <i>user</i></code>
Parameters	<i>user</i> The user login name to be dropped.
Examples	Removes the login name “louise” from the Replication Server: <code>drop user louise</code>
Usage	<ul style="list-style-type: none">• Use <code>drop user</code> to remove a Replication Server login name.• Execute this command on the Replication Server where the login name was created.
Permissions	<code>drop user</code> requires “sa” permission.
See also	<code>alter user</code> , <code>create user</code>

grant

Description	Assigns permissions to users.
Syntax	<pre>grant {sa create object primary subscribe connect source} to user</pre>
Parameters	<p>sa Users with “sa” permission can execute any RCL command.</p> <p>create object Allows the recipient to create, alter, and drop Replication Server objects, such as replication definitions, subscriptions, and function strings.</p> <p>primary subscribe Allows recipient to create subscriptions for a replicated table whose primary data is managed by the current Replication Server.</p> <p>connect source This permission is granted to RepAgents and other Replication Servers to log in to the Replication Server.</p> <p>user The login name of a user who is to receive the permission.</p>
Examples	<p>Example 1 Allows the user “thom” to execute any Replication Server command:</p> <pre>grant sa to thom</pre> <p>Example 2 Allows the user “louise” to create subscriptions:</p> <pre>grant primary subscribe to louise</pre>
Usage	<ul style="list-style-type: none">• The “sa” permission cannot be revoked from the “sa” user.• The “connect source” permission is needed by the RSI or RepAgent. Refer to the Replication Server installation and configuration guides for your platform for more information.• For each RCL command described in this manual, the minimum permission required to execute the command is shown. For a list of minimum permissions for all commands, see the <i>Replication Server Administration Guide Volume 1</i>.
Permissions	grant requires “sa” permission.
See also	revoke

ignore loss

Description	Allows Replication Server to accept messages after it detects a loss.
Syntax	<pre>ignore loss from <i>data_server.database</i> [to {<i>data_server.database</i> <i>replication_server</i>}]</pre>
Parameters	<p>from <i>data_server.database</i> Specifies the primary data server and database whose message loss is to be ignored.</p> <p>to <i>data_server.database</i> Specifies the destination data server and database for the lost messages.</p> <p>to <i>replication_server</i> Specifies the destination Replication Server for the lost messages.</p>
Usage	<ul style="list-style-type: none">• Replication Server detects loss when it rebuilds queues or replays transaction logs in recovery mode.• A Replication Server detects message losses on connections to the replicate databases it manages.• For warm standby databases, use the logical connection name for <i>data_server.database</i>, except for losses that Replication Server detects between the active database and the standby database. To ignore these losses, use the physical <i>data_server.database</i> name.• If direct routes exist, the destination Replication Server detects message losses from the source Replication Server. Look in both Replication Server log files to determine whether losses were detected.• When a Replication Server detects losses, it accepts no messages on the connection until ignore loss is executed.• After ignore loss is executed, a few updates may be necessary before messages begin to flow again.• After ignore loss is executed, procedures are required to bring replicated data up to date. <p>See the <i>Replication Server Administration Guide Volume 2</i> for detailed recovery instructions.</p>
Permissions	ignore loss requires “sa” permission.
See also	allow connections, configure route, rebuild queues, set log recovery

move primary

Description	Changes the primary Replication Server for an error class or a function-string class.
Syntax	<pre>move primary of {[replication server] error class function string class} <i>class_name</i> to <i>replication_server</i></pre>
Parameters	<p>replication_server Specify to modify a Replication Server error class. Leave out to modify a data server error class.</p> <p>error class Specifies that the primary Replication Server for an error class is to be changed.</p> <p>function string class Specifies that the primary Replication Server for a function-string class is to be changed.</p> <p><i>class_name</i> The name of the error class or function-string class whose primary Replication Server is to be changed.</p> <p><i>replication_server</i> Specifies the new primary Replication Server for the error class or function-string class. It is the name of the Replication Server where the command is executed, since move primary must be executed at the new primary Replication Server.</p>
Examples	<p>Example 1 Changes the primary Replication Server for the pubs2_db_err_class error class to the SYDNEY_RS Replication Server. The command is entered at SYDNEY_RS:</p> <pre>move primary of error class pubs2_db_err_class to SYDNEY_RS</pre> <p>Example 2 Changes the primary Replication Server for the Replication Server error class my_rs_error_class to the SYDNEY_RS Replication Server. The command is entered at SYDNEY_RS:</p> <pre>move primary of replication server error class my_rs_error_class to SYDNEY_RS</pre>

Example 3 Changes the primary Replication Server for the `sqlserver2_function_class` function-string class to the `SYDNEY_RS` Replication Server. The command is entered at `SYDNEY_RS`:

```
move primary
  of function string class sqlserver2_function_class
  to SYDNEY_RS
```

Usage

- If you have changed the routing configuration, use `move primary` to ensure that error responses and function strings are distributed, via the new routes, to the Replication Servers where they are needed.
- `move primary` must be executed at the new primary Replication Server.
- `move primary` can be used to change the primary Replication Server from A to B only if routes exist from A to B and from B to A.
- There is no primary site for the system-provided `rs_sqlserver_function_class` until you assign one. `rs_default_function_class` and `rs_db2_function_class` are system-provided, cannot be modified, and have no primary site.
- The primary site for a derived function-string class is the site of its parent class, unless the parent class is `rs_default_function_class` or `rs_db2_function_class`. In that case, the primary site of the derived class is the site where it was created.
- If you use `rs_sqlserver_function_class`, you must specify a primary site before you can modify a default function-string. To specify a primary site for the function-string class, execute `create function string class rs_sqlserver_function_class` at the primary site. Then use the `move primary` command to change the primary site for the class.
- There is no primary site for the default error class, `rs_sqlserver_error_class` and `rs_repserver_error_class`, until you assign one. You must specify a primary site before you use `assign action` to change default error actions. To specify a primary site, execute `create error class rs_sqlserver_error_class` or `create replication server error class rs_repserver_error_class` at the primary site. Then you can use `move primary` to change the primary site.

Permissions

`move primary` requires “sa” permission.

See also

`alter error class`, `alter route`, `assign action`, `create error class`, `create function string class`

rebuild queues

Description Rebuilds Replication Server stable queues.

Syntax rebuild queues

Usage

- Rebuild stable queues to recover from a failed or missing partition.

Warning! Use this command only as described in the *Replication Server Administration Guide Volume 2*. `rebuild queues` *deletes* messages from the replication system and may make it more difficult to correct other problems.

- Drop damaged partitions and replace them, if necessary, before you rebuild queues. A dropped partition may not actually be removed from the system until `rebuild queues` is executed.
- `rebuild queues` disconnects all other Replication Servers from the Replication Server where it is executed. Connection attempts are refused until the queues are rebuilt.
- `rebuild queues` clears all of the Replication Server's stable queues, and "gives up" any damaged partitions in use.
- If you start Replication Server in stand-alone mode (using the `-M` command line flag) and then execute `rebuild queues`, Replication Server goes into recovery mode.
- While restoring messages to the rebuilt stable queues, Replication Server determines whether the data cleared from the queues was recovered or lost. Look for error messages in the log file of the Replication Server with the rebuilt queues and in the log files of Replication Servers that have direct routes from it. Loss detection may not complete immediately; it is necessary for new data to flow from each primary database or upstream site.
- If loss is detected, you may need to re-create subscriptions or recover data from offline dumps.
- If a subscription is materializing when you use `rebuild queues`, drop and re-create it. Even if the materialization appears to have completed successfully, some data may have been lost.
- After queues are rebuilt, the Replication Server attempts to restore lost messages by requesting backlogged messages from Replication Servers that have routes to the current Replication Server.

- You cannot rebuild queues for specific database connections or routes.

For help with recovery procedures, see the *Replication Server Administration Guide Volume 2*.

Permissions

rebuild queues requires “sa” permission.

See also

add partition, alter partition, configure connection, create partition, drop partition, ignore loss, resume log transfer, set log recovery

resume connection

Description	Resumes a suspended connection.
Syntax	<code>resume connection to <i>data_server.database</i></code> <code>[skip [<i>n</i>] transaction execute transaction skip to resync marker]</code>
Parameters	<p><i>data_server</i></p> <p>The name of the data server that holds the database whose connection is to be resumed.</p> <p><i>database</i></p> <p>The name of the database whose connection is to be resumed.</p> <p><code>skip [<i>n</i>] transaction</code></p> <p>Instructs Replication Server to skip a specified number of transactions in the connection queue before resuming the connection. The skipped transactions are written to the database exceptions log, and to either the Replication Server log or the alternative log file specified by the <code>sysadmin dump_file</code> command. The maximum number of transactions that resume connection can skip is the number of transactions in the DSI outbound queue.</p> <p>If <i>n</i> is not specified, Replication Server resumes execution with the second transaction in the connection's queue.</p> <p><code>execute transaction</code></p> <p>overrides the Replication Server restriction against the application of system transactions after a DSI startup if the system transaction is the first transaction in the DSI queue.</p> <p><code>skip to resync marker</code></p> <p>instructs Replication Server to skip transactions in the DSI outbound queue for the specified replicate database until Replication Server receives and acknowledges a dump database marker sent by Replication Agent. Replication Server skips processing of records in the outbound queue since the data in the replicate database is expected to be replaced with the dump contents.</p>
Examples	<p>Example 1 Resumes the connection to the pubs2 database in the SYDNEY_DS data server:</p> <pre>resume connection to SYDNEY_DS.pubs2</pre> <p>Example 2 Resumes the connection to the pubs2 database in the SYDNEY_DS dataserver after skipping two transactions. The transactions are logged in the database exceptions log and the Replication Server log:</p> <pre>resume connection to SYDNEY_DS.pubs2 skip 2 transaction</pre>

Example 3 Resumes the connection to the pubs2 database in the SYDNEY_DS dataserver after skipping two transactions. The transactions are logged in the database exceptions log and in the *SYDNEY_RS.log* file. The last sysadmin dump_file command closes the *SYDNEY_RS.log* file:

```
sysadmin dump_file SYDNEY_RS.log
resume connection to SYDNEY_DS.pubs2 skip 2 transaction
sysadmin dump_file
```

Example 4 Instruct Replication Server to remove data from the replicate database outbound queue and wait for a resync marker from the primary database Replication Agent:

```
resume connection to SYDNEY_DS.pubs2 skip to
resync marker
```

Usage

- Resuming a connection allows replication activities for the suspended database to begin again.
- Suspend connections so you can alter them with alter connection or perform maintenance on the suspended database. Connections are also suspended during subscription materialization or dematerialization.
- Replication Server can suspend a database connection because of an error.
- resume connection is also used to resume a connection suspended because of an error.
- If you determine that the system transaction was executed, use the skip transaction clause.
- Use the execute transaction clause *only* if a system transaction has failed to execute and you have corrected the problem that prevented its execution. A system transaction has no enclosing begin tran/commit tran pair. If Replication Server is restarted with a system transaction as the first transaction, you see this message:

```
E. 1998/02/16 14:43:49. ERROR #5152 DSI (206 hookip01.rdb1) - dsisched.c
(2196)
```

```
There is a system transaction whose state is not known. DSI will be shut
down.
```

Determine whether the database has executed this transaction and use skip transaction or execute transaction as appropriate.

- When you set skip to resync, Replication Server does not log the transactions that are skipped in the Replication Server log or in the database exceptions log. Replication Server logs transactions that are skipped when you set skip [n] transaction.

If after executing resume connection with skip to resync marker, Replication Agent does not issue the correct marker or you issue the marker against the wrong connection, or for other reasons the DSI connection is not expected to process the resync database marker, you can resume normal replication processing without waiting for the resync database marker by executing suspend connection, and then resume connection without the skip to resync option.

Note If you execute resume connection with the skip to resync marker option on the wrong connection, data on the replicate database becomes unsynchronized.

Permissions

resume connection requires “sa” permission.

See also

activate subscription, alter connection, assign action, create connection, drop connection, drop subscription, suspend connection

resume distributor

Description	Resumes a suspended Distributor thread for a connection to a database.s
Syntax	<code>resume distributor <i>data_server.database</i> [skip transaction]</code>
Parameters	<p><i>data_server</i></p> <p>The data server name. If the database is part of a warm standby application, <i>data_server</i> is the logical data server name.</p> <p><i>database</i></p> <p>The database name. If the database is part of a warm standby application, <i>database</i> is the logical database name.</p> <p>skip transaction</p> <p>Instructs Replication Server to resume execution with the second transaction in the connection's queue. The first transaction is written to the database exceptions log.</p>
Examples	<p>Resumes the Distributor thread for the logical data server LDS and the pubs2 database:</p> <pre>resume distributor LDS.pubs2</pre>
Usage	<ul style="list-style-type: none">• Use <code>resume distributor</code> to resume a Distributor thread suspended using <code>suspend distributor</code> or suspended by Replication Server.• Use <code>skip transaction</code> to resume connection when distributor is down due to:<ul style="list-style-type: none">• message in inbound queue is longer than 16,000 bytes and site version has not been upgraded to RS 12.5 and up, or• downstream Replication Server cannot accept new feature commands, for example, <code>bigint</code>.
Permissions	<code>resume distributor</code> requires "sa" permission.
See also	<code>suspend distributor</code>

resume log transfer

Description	Allows the RepAgent to connect to the Replication Server.
Syntax	<code>resume log transfer</code> <code>from {<i>data_server.database</i> all}</code>
Parameters	<i>data_server</i> the name of the data server with the database whose RepAgent is to be connected to the Replication Server. <i>database</i> the database whose RepAgent is to connect to the Replication Server. all permits RepAgents for all databases managed by the Replication Server to connect.
Examples	Example 1 The Replication Server will accept connections from any RepAgent: <code>resume log transfer from all</code> Example 2 The Replication Server will accept a connection from a RepAgent for the pubs2 database in the SYDNEY_DS data server: <code>resume log transfer from SYDNEY_DS.pubs2</code>
Usage	<ul style="list-style-type: none">• When you quiesce a Replication Server or the replication system, use <code>suspend log transfer</code> to cause Replication Server to refuse RepAgent connections.• <code>resume log transfer</code> allows the RepAgent threads to connect to a Replication Server upon which <code>suspend log transfer</code> has been executed.• Normally, the RepAgent retries its connection to Replication Server following a <code>suspend log transfer</code> until <code>resume log transfer</code> allows it to reconnect. However, if the RepAgent is down for any reason, <code>resume log transfer</code> does not restart it.• After resuming log transfer from ERSSD, the recovery daemon will automatically restart the ERSSD RepAgent when it wakes up.
Permissions	<code>resume log transfer</code> requires “sa” permission.
See also	<code>admin quiesce_check</code> , <code>admin quiesce_force_rsi</code> , <code>resume connection</code> ,

resume queue

Description	Restarts a stable queue stopped after being passed a message larger than 16K bytes. Applicable only when the Replication Server version is 12.5 or later and the site version has not been similarly upgraded.
Syntax	<code>resume queue, <i>q_number</i>, <i>q_type</i> [, skip transaction with large message]</code>
Parameters	<p><i>q_number</i> The queue number of the stable queue.</p> <p><i>q_type</i> The queue type of the stable queue. Values are “0” for outbound queues, “1” for inbound queues.</p> <p>skip transaction with large message Specifies that the SQM should skip the first large message encountered after restarting.</p>
Examples	<p>Specifies that outbound queue #2 skips the first large message it is passed by the RepAgent:</p> <pre>resume queue, 2, 0, skip transaction with large message</pre>
Usage	<ul style="list-style-type: none">• This command is applicable only when the Replication Server is version 12.5 or later and the site version is not upgraded.• <code>resume queue</code> does not skip any messages if the site version is 12.5 or later.
Permissions	<code>alter queue</code> requires “sa” permission.
See also	<code>alter queue</code>

resume route

Description	Resumes a suspended route.
Syntax	<code>resume route</code> to <i>dest_replication_server</i> [skip transaction with large message]
Parameters	<i>dest_replication_server</i> The name of the destination Replication Server; that is, the suspended route you want to resume. skip transaction with large message Ignore first transaction encountered with a message greater than 16,000 bytes.
Examples	Resumes the route to the SYDNEY_RS Replication Server: <pre>resume route to SYDNEY_RS</pre>
Usage	<ul style="list-style-type: none"> Resuming a route allows Replication Server to begin sending queued messages to the remote Replication Server again. <code>resume route</code> can also be used to resume a route suspended because of an error. <code>skip transaction with large message</code> is applicable only to direct routes where the site version at the replicate site is 12.1 or earlier.
Permissions	<code>resume route</code> requires “sa” permission.
See also	<code>alter route</code> , <code>create route</code> , <code>drop route</code> , <code>suspend route</code>

revoke

Description	Revokes permissions from users.
Syntax	<code>revoke {sa connect source create object primary subscribe} from <i>user</i></code>
Parameters	<p><code>sa</code> Denies permission to execute commands that require “sa” permission.</p> <p><code>connect source</code> Denies permission to execute RCL commands used by RepAgents or other Replication Servers.</p> <p><code>create object</code> Denies permission to create, alter, and drop Replication Server objects such as replication definitions, subscriptions, and function strings.</p> <p><code>primary subscribe</code> Denies permission to create subscriptions for a replicated table if the primary data is managed by the current Replication Server.</p> <p><i>user</i> The login name of the user whose permission is to be revoked.</p>
Examples	<p>Example 1 Prevents user “thom” from executing commands that create or modify Replication Server objects:</p> <pre>revoke create object from thom</pre> <p>Example 2 Prevents user “louise” from creating subscriptions for primary data managed by this Replication Server, unless she has “create object” or “sa” permission at the primary Replication Server:</p> <pre>revoke primary subscribe from louise</pre>
Usage	<ul style="list-style-type: none">• revoke requires “sa” permission.• The “sa” permission cannot be revoked from the “sa” user login name.
Permissions	revoke requires “administrator” permission.
See also	create replication definition, check subscription, create user, grant

set

Description	Controls replication definition properties for a replicate connection.
Syntax	<pre>set {autocorrection dynamic_sql} {on off} for <i>replication_definition</i> with replicate at <i>data_server.database</i></pre>
Parameters	<p>autocorrection Prevents failures that might occur because of missing or duplicate rows in a replicated table. Default is off.</p> <p>dynamic_sql Controls whether the table will be considered for dynamic SQL application. Default is on.</p> <p>on Enables autocorrection or dynamic SQL for the specified replication definition.</p> <p>off Disables autocorrection or dynamic SQL for the specified replication definition.</p> <p><i>replication_definition</i> The name of the replication definition whose autocorrection or dynamic SQL status you are changing.</p> <p><i>data_server</i> The name of the data server with the replicate database for which you are changing the autocorrection or dynamic SQL status. If the replicate database is part of a warm standby application, <i>data_server</i> is the logical data server name.</p> <p><i>database</i> The name of the replicate database where you are changing the autocorrection or dynamic SQL status. If the replicate database is part of a warm standby application, <i>database</i> is the logical database name.</p>
Examples	<p>Example 1 Enables autocorrection for the publishers_rep replication definition in the pubs2 database at the SYDNEY_DS data server:</p> <pre>set autocorrection on for publishers_rep with replicate at SYDNEY_DS.pubs2</pre> <p>Example 2 Disables dynamic SQL for the publishers_rep replication definition in the pubs2 database at the SYDNEY_DS data server:</p> <pre>set dynamic_sql off</pre>

- for publishers_rep
with replicate at SYDNEY_DS.pubs2
- Usage
- Use set dynamic_sql off to disable dynamic SQL commands for the specified replication definition and replicate connection.
 - Use set autocorrection to prevent duplicate key errors that might occur during non-atomic materialization.
 - Autocorrection should be enabled *only* for replication definitions whose subscriptions use non-atomic materialization (create subscription specified without holdlock). After materialization is complete and the subscription is VALID, disable autocorrection to improve performance.
 - Autocorrection is off, by default, for a replication definition.
- How autocorrection works
- set autocorrection determines how Replication Server processes inserts and updates to replicated tables. When autocorrection is on, Replication Server converts each update or insert operation into a delete followed by an insert.
- For example, if a row inserted into the primary version of a table already exists in a replicated copy and autocorrection is off, the operation results in an error. When autocorrection is on, Replication Server converts the insert to a delete followed by an insert so that the insert cannot fail because of an existing row.
- If the primary key has changed in a row that is to be replicated, Replication Server deletes two rows in the replicated table before it inserts the row. It deletes the row in which the primary key matches the before image and the row in which the primary key matches the after image.
- When autocorrection is on, an insert or update at a primary database may cause delete and insert triggers to fire at the replicate database. The delete trigger fires only if the row inserted or updated at the primary database was already present at the replicate database.
 - Replication Server creates entries for replication definitions with autocorrection enabled in the rs_repbjs system table.

Autocorrection and replicated stored procedures

- Replication Server does not perform autocorrection for rows updated at replicate databases as the result of using replicated stored procedures that modify primary data. See the *Replication Server Administration Guide Volume 1* for more information about replicating stored procedures.

Note If you use replicated stored procedures to modify primary data, be sure to write stored procedures at the replicate Replication Server to correct for the failed updates and inserts that can occur during non-atomic materialization. Stored procedures at the replicate Replication Server should simulate autocorrection, treating update and insert operations as combined delete-insert operations. Alternatively, stored procedures can correct failed updates and inserts after they are detected.

Autocorrection and *replicate minimal columns*

- If a replication definition uses replicate minimal columns, you cannot set autocorrection on. If you set autocorrection on before specifying minimal columns (for example, using alter replication definition), autocorrection is not performed. Replication Server logs informational messages for any update operations.

Autocorrection and *text*, *unitext*, or *image* datatypes

- If a replication definition has a text, unitext, or image column in the replicate_if_changed column list, an attempt to enable autocorrection for the replication definition causes an error. Autocorrection requires that all text, unitext, and image columns appear in the always_replicate list for the replication definition.

Autocorrection and bulk copy-in

In normal replication, bulk operation is disabled if autocorrection is on. However, in subscription materialization, bulk copy-in is applied even when autocorrection is enabled except for nonatomic subscriptions recovering from failure.

Permissions

set requires “create object” permission.

See also

alter replication definition, create replication definition, create subscription

set log recovery

Description	Specifies databases whose logs are to be recovered from offline dumps.
Syntax	<code>set log recovery</code> for <i>data_server.database</i>
Parameters	<i>data_server</i> The data server with the database to be recovered. <i>database</i> The database to be recovered.
Usage	<ul style="list-style-type: none">• Execute set log recovery after restarting Replication Server in stand-alone mode.• Execute allow connections after set log recovery to enter recovery mode. Replication Server accepts connections only from RepAgents started in recovery mode for databases named in set log recovery. This ensures that old log records are replayed before new log records are accepted. <p>See the <i>Replication Server Administration Guide Volume 2</i> for detailed recovery procedures.</p>
Permissions	set log recovery requires “sa” permission.
See also	allow connections, ignore loss, rebuild queues

set proxy

Description	Switches to another user.
Syntax	set proxy [to] [<i>user_name</i> [verify password <i>passwd</i>]]
Parameters	<p><i>user_name</i></p> <p>A valid Replication Server login name.</p> <p>verify password</p> <p>Verifies the password of a Replication Server user.</p> <p><i>passwd</i></p> <p>the password of a valid Replication Server user.</p>
Usage	<ul style="list-style-type: none">• set proxy <i>user_name</i> switches to a new user with all the permissions of the new user and none of the permissions of the original user.• The new user can always switch back to the original user, whether or not the new user has “sa” permission, by entering set proxy without a user name.• set proxy <i>user_name</i> verify password <i>passwd</i> allows a user without sa permission to switch to another user—if the correct password for <i>user_name</i> is entered.
Permissions	set proxy <i>user_name</i> requires “sa” permission. Any user can execute set proxy and set proxy <i>user_name</i> verify password <i>passwd</i> .
See also	alter connection, alter route, configure replication server, create connection, create route

show connection

Description	Lists the contents of the connection stack.
Syntax	show connection
Examples	<p>Shows the connection stack after ost_replinuxvm_02, an ID server, creates a gateway to ost_replinuxvm_03:</p> <pre>isql -Usa -P -S ost_replinuxvm_02 1> connect to ost_replinuxvm_03 2> go Gateway connection to 'ost_replinuxvm_03' is created.</pre>

```
1> show connection
2> go
ost_replinuxvm_03
ost_replinuxvm_02 (IDServer)
```

Usage

- Cascaded connections created in the gateway are kept in a connection stack, with the Replication Server that issued the first connect command placed at the bottom of the stack.
- The disconnect command behaves differently in Replication Server 15.1 or earlier. In these versions, a disconnect command terminates the gateway mode, and returns the working server status to the Replication Server that issued the first connect command. When your connection stack includes Replication Server versions 15.2, and 15.1 or earlier, and you issued a disconnect command, the show connection and show server commands may not display the expected output.

Permissions

Any user may execute this command.

See also

connect, disconnect, show server

show server

Description

Displays the current working server given a stack of connections.

Syntax

show server

Examples

Displays the current working server, after a connection from ost_replinuxvm_02 to ost_replinuxvm_03 is created:

```
isql -Usa -P -S ost_replinuxvm_02
1> connect to ost_replinuxvm_03
2> go
Gateway connection to 'ost_replinuxvm_03' is created.

1> show server
2> go
ost_replinuxvm_03
```

Usage

For usage information, see show connection.

Permissions

Any user may execute this command.

See also

connect, disconnect, show connection

shutdown

Description	Shuts down a Replication Server.
Syntax	<code>shutdown</code>
Examples	Instructs the Replication Server to shut down: <code>shutdown</code>
Usage	Use the <code>shutdown</code> command to shut down a Replication Server. This command instructs Replication Server to refuse additional connections, terminate processes, and exit.
Permissions	<code>shutdown</code> requires “sa” permission.

suspend connection

Description	Suspends a connection to a database.
Syntax	<code>suspend connection</code> to <i>data_server.database</i> [with nowait]
Parameters	<i>data_server</i> The name of the data server with the database whose connection is to be suspended. <i>database</i> The name of the database whose connection is to be suspended. with nowait Suspends the connection immediately.
Examples	Suspends the connection to the pubs2 database in the SYDNEY_DS data server: <code>suspend connection to SYDNEY_DS.pubs2</code>
Usage	<ul style="list-style-type: none">• Suspending a connection temporarily halts replication activities for the database.• Connections are suspended so they can be altered with <code>alter connection</code> or so that maintenance can be performed. You can also use <code>suspend connection</code> to control when replicate databases are updated.• While a connection is suspended, Replication Server holds transactions for the database in stable queues.• If <code>suspend connection</code> is executed without the <code>with nowait</code> clause, Replication Server attempts to complete any transaction that is in progress. However, the connection to the data server may be suspended before the transaction is completed.• To reactivate the connection, use <code>resume connection</code>.
Permissions	<code>suspend connection</code> requires “sa” permission.
See also	<code>alter connection</code> , <code>create connection</code> , <code>drop connection</code> , <code>resume connection</code>

suspend distributor

Description	Suspends the Distributor thread for a connection to a primary database.
Syntax	<code>suspend distributor <i>data_server.database</i></code>
Parameters	<p><i>data_server</i></p> <p>The data server name. If the database is part of a warm standby application, <i>data_server</i> is the logical data server name.</p> <p><i>database</i></p> <p>The database name. If the database is part of a warm standby application, <i>database</i> is the logical database name.</p>
Examples	<p>Suspends the Distributor thread for the pubs2 database in the LDS data server:</p> <pre>suspend distributor LDS.pubs2</pre>
Usage	<ul style="list-style-type: none">• Use suspend distributor to suspend a Distributor thread for a logical or physical connection to a primary database.• To resume the Distributor thread, use resume distributor.• The distributor thread reads incoming primary database transactions and forwards them to subscribers. Turn off the distributor to enhance performance in a warm-standby-only environment that has only a standby database and no subscribers.
Permissions	suspend distributor requires “sa” permission.
See also	resume distributor

suspend log transfer

Description	Disconnects a RepAgent from a Replication Server and prevents a RepAgent from connecting.
Syntax	<code>suspend log transfer</code> <code>from {<i>data_server.database</i> all}</code>
Parameters	<i>data_server</i> The data server with the database whose RepAgent is to be suspended. <i>database</i> The database whose RepAgent is to be suspended or whose connections are to be disallowed. all Instructs Replication Server to suspend all RepAgents and to disallow future connections for all RepAgents.
Examples	Example 1 Disconnects the RepAgent for the pubs2 database and does not permit it to reconnect: <pre>suspend log transfer from TOKYO_DS.pubs2</pre> Example 2 Disconnects all connected RepAgents and does not permit any RepAgent to reconnect to the Replication Server: <pre>suspend log transfer from all</pre>
Usage	<ul style="list-style-type: none">• Use suspend log transfer to disconnect a RepAgent. This is the first step in quiescing the replication system. suspend log transfer does not shut down the RepAgent.• To test whether the system is quiesced after suspending a RepAgent, use admin quiesce_check.• To allow RepAgents to connect to the Replication Server, execute resume log transfer.
Permissions	suspend log transfer requires “sa” permission.
See also	admin quiesce_check, admin quiesce_force_rsi, resume log transfer

suspend route

Description	Suspends a route to another Replication Server.
Syntax	<code>suspend route</code> to <i>dest_replication_server</i>
Parameters	<i>dest_replication_server</i> The name of the destination Replication Server, the route to which is to be suspended.
Examples	Suspends the route to the SYDNEY_RS Replication Server: <code>suspend route to SYDNEY_RS</code>
Usage	<ul style="list-style-type: none">• Use <code>suspend route</code> to suspend a route to another Replication Server. This command lets you manage network use by controlling when messages are sent from one Replication Server to another.• While a route is suspended, Replication Server holds messages for the destination Replication Server in a stable queue.• You can suspend only direct routes.• To reactivate a suspended route, use <code>resume route</code>.
Permissions	<code>suspend route</code> requires “sa” permission.
See also	<code>alter route</code> , <code>resume connection</code> , <code>resume route</code> , <code>suspend connection</code>

switch active

Description	Changes the active database in a warm standby application.
Syntax	<pre>switch active for <i>logical_ds.logical_db</i> to <i>data_server.database</i> [with suspension]</pre>
Parameters	<p><i>logical_ds</i> The logical data server name for the logical connection.</p> <p><i>logical_db</i> The logical database name for the logical connection.</p> <p><i>data_server</i> The data server name of the new active database for the logical connection.</p> <p><i>database</i> The database name of the new active database for the logical connection.</p> <p>with suspension Suspends the DSI connection to the new active database after the switch is complete.</p>
Examples	<p>This command starts the switch active process:</p> <pre>switch active for LDS.pubs2 to OSAKA.pubs2</pre> <p>Switch of the active for this logical database is in progress.</p>
Usage	<ul style="list-style-type: none">• switch active is a part of the procedure for switching to the standby database in a warm standby application. See the <i>Replication Server Administration Guide Volume 2</i> for the complete procedure.• switch active returns immediately, but the switch is not complete until admin logical_status displays “None” in the State of Operation in Progress.• Use admin logical_status to monitor the status of the switch active process.• If you use the with suspension option, you must manually resume the DSI connection to the new active database after the switch is complete.• After entering switch active, you can attempt to cancel it using abort switch.
Permissions	switch active requires “sa” permission.
See also	abort switch, admin logical_status, create logical connection, wait for switch

sysadmin apply_truncate_table

Description	Turns on or off the “subscribe to truncate table” option for all existing subscriptions to a particular table, enabling or disabling replication of truncate table.
Syntax	<code>sysadmin apply_truncate_table, data_server, database, {table_owner " ""}, table_name {on 'off'}</code>
Parameters	<p><i>data_server</i> The name of the replicate data server.</p> <p><i>database</i> The name of the replicate database managed by the data server.</p> <p><i>table_owner</i> Identifies the owner of the replicate table. If owner is not specified, Replication Server sets owner to “dbo.”</p> <p><i>table_name</i> Identifies the replicate table for which you want to turn on or off the “subscribe to truncate table” option for existing subscriptions.</p> <p>on Turns on the “subscribe to truncate table” option for existing subscriptions.</p> <p>off Turns off the “subscribe to truncate table” option for existing subscriptions.</p>
Examples	<p>Turns on “subscribe to truncate table” for all subscriptions to the publishers table owned by emily in the pubs2 database:</p> <pre>sysadmin apply_truncate_table, SYDNEY_DS, pubs2, emily, publishers, 'on'</pre>
Usage	<ul style="list-style-type: none">• Use <code>sysadmin apply_truncate_table</code> with Adaptive Server version 11.5 or later databases.• If you did not specify a replicate table owner in the replication definition, enter " (two single-quote characters) or "" (two double-quote characters) for the table owner name.• Subscriptions for a particular table for a particular database must all support or not support replication of truncate table. If, for example, <code>sysadmin apply_truncate_table</code> is off, you cannot create new subscriptions that include the “subscribe to truncate table” option unless you turn <code>sysadmin apply_truncate_table</code> on for all subscriptions for that table.

See `create subscription` or `define subscription` for more information about setting the “subscribe to truncate table” option for new subscriptions.

- Replication Server executes `truncate table` at the replicate database as the maintenance user. Among the permissions granted to maintenance user is “`replication_role`.” If you revoke maintenance user’s “`replication_role`,” you will be unable to replicate `truncate table` unless
 - The maintenance user has been granted “`sa_role`,”
 - The maintenance user owns the table, or
 - The maintenance user is aliased as the Database Owner.
- It is not necessary for warm standby databases to subscribe to `truncate table`; execution of the `truncate table` command is automatically replicated to standby databases. Turn on replication of `truncate table` for standby databases with the `alter logical connection` command.

Permissions

`sysadmin apply_truncate_table` requires “`sa`” permission.

See also

`create subscription`, `define subscription`

sysadmin cdb

Description	Administer the net-change database in real-time loading (RTL) replication to Sybase IQ and high volume adaptive replication (HVAR) into Adaptive Server.
Syntax	To hold, inspect, and release a net-change database, use: To inspect and release net-change database instances, use: sysadmin cdb , <i>q_number</i> , <i>q_type</i> , {hold hold_next unhold}

Note You must execute sysadmin cdb with hold or hold next before you can use sysadmin cdb to display net-change database information if the Data Server Interface Executor (DSI/E) thread is actively processing transactions.

To display information on the net-change database, use:

To display all information on the net-change database, or only information on specific tracking tables, use:

sysadmin cdb,
[*q_number* [, *q_type*] [list [, ["*table_owner*." *table_name*"] |
[[dump_i | dump_d | dump_u | dump_nc], *table_name*] | dump_nc]]

Parameters	<p>hold</p> <p>instructs DSI/E to suspend the current net-change database instances so that you can inspect them.</p> <p>hold_next</p> <p>instructs DSI/E to commit the first transaction that is ready to be committed, release the database instance, then retain the next transaction.</p> <p>unhold</p> <p>instructs DSI/E to release all net-change database instances that DSI is currently retaining and resume normal DSI/E activity.</p> <p><i>q_number</i></p> <p>identifies the outbound DSI stable queue for the replicate database. Examine the output of admin who, sqm command to identify the queue number.</p> <p><i>q_type</i></p> <p>identifies the stable queue type, where 0 is for an outbound queue and 1 is for an inbound queue. Default is 0. If you do not specify <i>q_type</i>, the default value is used.</p>
------------	--

table_name

specifies the replicate table name.

list

displays information about the net-change database. If you do not specify the table name, *list* displays all instances of the outbound DSI stable queue you specify with *q_number*. Specify the table to show only the contents of that table

dump_i

returns a result containing all the columns and rows in the in-memory *Insert_Table* table.

dump_u

returns a result containing all the columns and rows in the in-memory *Update_Table* table.

dump_d

returns a result containing all the columns and rows in the in-memory *Delete_Table* table.

dump_nc

returns a result containing the noncompilable commands that will be applied to the replicate table. For inserts, all the columns are returned. For deletes, only the primary keys are returned. For updates, only the primary key and updated columns are returned.

Examples

Example 1 Instruct DSI/E to suspend a net-change database for inspection, after the database is populated fully. If DSI/E is not actively processing transactions, the command to hold takes effect the next time the net-change database is created and populated. Replication Server suspends the DSI/E after the net-change database is created and populated, and before the net-change database content can be applied to the replicate database. For example to suspend the current net-change database:

```
sysadmin cdb,101,hold
```

Example 2 List active DSI Executor threads and the corresponding status including information on any net-change database Replication Server is processing currently:

```
sysadmin cdb
```

Output shows the RTL status for the two data servers and the respective database, queue number, and queue type for active DSI Executor (DSI/E) threads:

DSName	DBName	Queue	QType	Compile	Hold	CdbName	Commands_in_Group
--------	--------	-------	-------	---------	------	---------	-------------------

-----	-----	-----	-----	-----	-----	-----
IQSRVR2	asiqdemo	105	0	On	No	0
IQSRVR	iqdemo	104	0	On	No	0

The status columns are:

- Compile – status is “On” if RTL is active
- Hold – status is “Yes” if you executed sysadmin cdb with a hold for the same *q_number* and *q_type* to hold a specific DSI/E
- CdbName – the internal name of the net-change database the Replication Server is currently processing or that is in the “hold” state on that DSI/E thread. In this example, Replication Server is not processing any net-change database currently.
- Commands_in_Group – the number of commands that Replication Server is compiling as a group. In this example, no command are being processed.

Example 3 You do not need to suspend the DSI/E by setting it to the hold or hold_next state before you list the information on a specific DSI/E thread. Since the DSI/E is not in a hold or hold_next state, any value may change for successive executions of the command, except for the values under the Queue and QType columns:

```
sysadmin cdb,107,1
```

Output:

Queue	QType	CdbName	TargetDB	Compilable_Tables
-----	-----	-----	-----	-----
107	1	asiqdemo_ws_46_3	asiqdemo_ws	1
Non_Compilable_Tables		Commands_in_Group	Compiled_Rows	Non_Compilable Commands
-----		-----	-----	-----
0		3	2	0

Example 4 Display information on a net-change database that DSI/E is running currently:

Note Before you list information on a net-change database that DSI/E is running currently, you must suspend the database with the “hold” state.

```
sysadmin cdb,107,1,hold
```

```
go
sysadmin cdb,107,1,list
go
```

Output is:

CdbName	Replicate_Table	Status	Cmd_Convert
-----	-----	-----	
asiqdemo_ws_46_3	dbo.test_alltypes_ws_1	compilable	i2di

AutoCorrection	Nb_Columns	PK_Cols	CdbTable
-----	-----	-----	-----
No	25	22	test_allpes_ws_1_46_1

Insert_Table	Inserts	Update_Table	Updates
-----	-----	-----	-----
rs_itest_allpes_ws_1_46_1	1	rs_utest_allpes_ws_1_46_1	0

Delete_Table	Deletes	Non_Compilable_Cmds
-----	-----	-----
rs_dtest_allpes_ws_1_46_1	1	0

Update_Worktable	Delete_Worktable
-----	-----
#rs_dtest_allpes_ws_1_46_1	

Reduced_Inserts	Reduced_Updates	Reduced_Deletes
-----	-----	-----
0	0	0

(1 rows affected)

The columns are:

- CdbName – the internal name of the net-change database the Replication Server is currently processing or that is in the “hold” state on that DSI/E thread.
- Replicate_Table – replicate table name
- Status – “compilable” or “noncompilable” table
- Cmd_Convert – command conversions applied, such none, ud2i, i2di, or i2none
- AutoCorrection – whether autocorrection is applied
- Nb_Columns – number of columns in the net-change database table
- PK_Cols – number of primary key columns in the net-change database table
- CdbTable – unique name of net-change database table

- `Insert_Table` – name of in-memory table for insert operations in the net-change database
- `Inserts` – number of inserts
- `Update_Table` – name of in-memory table for update operations in the net-change database
- `Updates` – number of updates
- `Delete_Table` – name of in-memory table for delete operations in the net-change database
- `Deletes` – number of deletes
- `Non_Compilable_Cmds` – number of noncompilable commands.
- `Update_Worktable` – name of the worktable created on the replicate data server when applying updates. This worktable is populated and joined with the replicate table
- `Delete_Worktable` – name of the worktable created on replicate data server when applying deletes. This worktable is populated and joined with the replicate table
- `Reduced_Inserts` – number of inserts reduced due to compilation
- `Reduced_Updates` – number of updates reduced due to compilation
- `Reduced_Deletes` – number of deletes reduced due to compilation

Example 5 You can list detailed information on a specific table in the net-change database by including the `dump_i`, `dump_u`, `dump_d`, or `dump_nc` options in your query to return information in the table. The options are SQL select statements executed on the net-change table.

For example to display the content of `dbo.test_alltypes_msa_1` and the `Insert_Table` in-memory table:

```
sysadmin cdb,106,0,dump_i,dbo.test_alltypes_msa_1
```

If replication is successful, this is the output:

c1	c2	c3
4	v	ddd
3	upd	qqq

(2 rows affected)

Example 6 To display all noncompilable commands:

```
sysadmin cdb,105,1,dump nc
```

The output is:

[illegible]

Example 7 To display detailed information on a specific table within the net-change database:

```
sysadmin cdb,107,1,hold
go
sysadmin cdb,107,1,list,test_alltypes_ws_1
go
```

The output displays information that includes:

- 1 The status of operations and names of in-memory tables:

```

CdbName              Replicate_Table      Status      Cmd_Convert
-----
asiqdemo_ws_46_3    dbo.test_alltypes_ws_1  compilable  i2di

AutoCorrection      Nb_Columns      PK_Cols      CdbTable
-----
No                  25              22           test_allpes_ws_1_46_1

Insert_Table              Inserts      Update_Table              Updates
-----
rs_itest_allpes_ws_1_46_1  1           rs_utest_allpes_ws_1_46_1  0

Delete_Table              Deletes      Non_Compilable_Cmds
-----
rs_dtest_allpes_ws_1_46_1  1           0

Update_Worktable              Delete_Worktable
-----
#rs_dtest_allpes_ws_1_46_1

Reduced_Inserts      Reduced_Updates      Reduced_Deletes
-----
0                    0                    0
(1 row affected)

```

2 Information on all columns in the table:

```

Colname  Coltype  Maxlength  Cdbtype  Cdbvtype  Primary_key  Changed  HasNull
-----
c1        int       4           8         8          1           1        0
...
c8        money     10          1         0          1           1        0
...
c25       image     5           5         19         0           1        1
(25 rows affected)

```

Usage You can list detailed information on a specific in-memory table in the net-change database by including one of these SQL commands in your query. The in-memory tables are for internal processing and the contents are not disk-resident.

You must execute `sysadmin net_change_db hold` or `sysadmin net_change_db hold next` before you can use `sysadmin net_change_db list` to display net-change database information.

Permissions `sysadmin net_change_db` requires “sa” permission.

See also `admin who`, `admin config`

sysadmin dropdb

Description	Drops a database from the ID Server.
Syntax	<code>sysadmin dropdb, <i>data_server</i>, <i>database</i></code>
Parameters	<i>data_server</i> The name of the data server. <i>database</i> The name of the database you want to drop.
Examples	Drops the pubs2 database in the SYDNEY_DS data server from the ID Server: <code>sysadmin dropdb, SYDNEY_DS, pubs2</code>
Usage	<ul style="list-style-type: none">• Use sysadmin dropdb to drop a database from the ID Server. This command must be executed at an ID Server.• Use sysadmin dropdb only when the ID Server system tables contain information about a database that does not exist in the system. This should happen only after a system failure. For example, if a database is dropped with drop connection, a network failure might prevent the ID Server from being notified so that it can remove the database from its tables. If you attempt to add the same data server and database to the system later, the request will fail because the database and its data server are already registered in the ID Server system tables.• If you reinstall a Replication Server, use sysadmin dropdb to remove the ID Server information for each database the Replication Server managed, including its RSSD. Otherwise, errors occur when you reinstall Replication Server.• If you enter invalid arguments with this command, you are not notified. <hr/> <p>Warning! Never use sysadmin dropdb on any databases that have active connections.</p> <hr/>
Permissions	sysadmin dropdb requires “sa” permission.
See also	sysadmin dropldb

sysadmin dropldb

Description	Drops a logical database from the ID Server.
Syntax	<code>sysadmin dropldb, <i>data_server</i>, <i>database</i></code>
Parameters	<i>data_server</i> The name of the logical data server. <i>database</i> The name of the logical database you want to drop.
Examples	Drops the pubs2 logical database in the LDS logical data server from the ID Server: <code>sysadmin dropldb, LDS, pubs2</code>
Usage	<ul style="list-style-type: none">• Use <code>sysadmin dropldb</code> to drop a logical database from the ID Server. This command must be executed at an ID Server.• Use <code>sysadmin dropldb</code> only when the ID Server system tables contain information about a logical database that does not exist in the system. This should happen only after a system failure. For example, if a logical database is dropped with <code>drop logical connection</code>, a network failure might prevent the ID Server from being notified so that it can remove the logical database from its tables. If you attempt to add the same logical data server and logical database to the system later, the request fails because the logical database and its logical data server are already registered in the ID Server system tables.• If you reinstall a Replication Server, first use <code>sysadmin dropldb</code> to remove the ID Server information for each logical database the Replication Server managed. Otherwise errors occur when you reinstall Replication Server.• If you enter invalid arguments with this command, you are not notified. <hr/> <p>Warning! Never use <code>sysadmin dropldb</code> on any logical databases that have active connections.</p> <hr/>
Permissions	<code>sysadmin dropldb</code> requires “sa” permission.
See also	<code>sysadmin dropdb</code>

sysadmin drop_queue

Description	Deletes a stable queue. Use this command to drop a failed materialization queue.
Syntax	sysadmin drop_queue, <i>q_number</i> , <i>q_type</i>
Parameters	<i>q_number</i> The site ID for the Replication Server or database that is the source or destination for the queue. <i>q_type</i> The queue type.
Usage	<ul style="list-style-type: none">Use sysadmin drop_queue to stop and delete a materialization queue that remains after a subscription experiences an unrecoverable error and must be manually cleaned up. <hr/> <p>Warning! Use sysadmin drop_queue only to drop a failed materialization queue.</p> <hr/> <ul style="list-style-type: none">Use admin who to find the <i>q_number</i> and <i>q_type</i> for a queue. The values appear in the command’s SQM thread output.
Permissions	sysadmin drop_queue requires “sa” permission.
See also	rebuild queues, sysadmin purge_route_at_replicate

sysadmin droprs

Description	Drops a Replication Server from the ID Server.
Syntax	<code>sysadmin droprs, <i>replication_server</i></code>
Parameters	<i>replication_server</i> The name of the Replication Server you want to drop.
Examples	Drops the SYDNEY_RS Replication Server from the ID Server: <code>sysadmin droprs, SYDNEY_RS</code>
Usage	<ul style="list-style-type: none">• Use <code>sysadmin droprs</code> to drop a Replication Server from the ID Server. This command can be executed only at an ID Server.• You can use <code>sysadmin droprs</code> when the ID Server contains information about a Replication Server that does not exist in the replication system. Such a scenario is usually a result of a system failure. For example, if a Replication Server installation fails, the ID Server system tables may contain entries for the Replication Server, preventing subsequent attempts to install the Replication Server.• You are not notified when you enter an invalid argument. <hr/> <p>Warning! Use <code>sysadmin droprs</code> with caution when removing an active Replication Server. For the correct procedure on removing an active Replication Server, see the <i>Replication Server Administration Guide Volume 1</i>.</p> <hr/>
Permissions	<code>sysadmin droprs</code> requires “sa” permission.

sysadmin dump_file

Description	Specifies an alternative log file name for use when dumping a Replication Server stable queue.
Syntax	sysadmin dump_file [, <i>file_name</i>]
Parameters	<i>file_name</i> The name of the new log file that stable queue dumps are to be written to.
Examples	Specifies pubs2.log as the file for logging stable queue output: <pre>sysadmin dump_file, 'pubs2.log'</pre>
Usage	<ul style="list-style-type: none">• Use sysadmin dump_file to specify a log file name before you use sysadmin dump_queue to dump the log to a file.• To reset the current dump file to the default, execute sysadmin dump_file without specifying a file name.• If a file name is specified, the current dump file is closed and a new file is opened. The new file uses the specified file name.• The default dump file is the Replication Server log. Use admin log_name to display the path to this file.• If you enter a log file name containing characters other than letters and numerals, enclose it in quotes.
Permissions	sysadmin dump_file requires “sa” permission.
See also	admin log_name, sysadmin dump_queue, sysadmin sqt_dump_queue

sysadmin dump_queue

Description	Dumps the contents of a Replication Server stable queue.
Syntax	<pre>sysadmin dump_queue {, <i>q_number</i> <i>server</i>[,<i>database</i>]}, <i>qtype</i> { , <i>seg</i>, <i>blk</i>, <i>cnt</i> [, <i>num_cmds</i>] [, {L0 L1 L2 L3}] [, {RSSD client "log" <i>file_name</i>}] "next" [, <i>num_cmds</i>] }</pre>
Parameters	<p><i>q_number</i> <i>server</i>[, <i>database</i>]</p> <p>Identifies the stable queue to dump. Use either <i>q_number</i> or <i>server</i>[, <i>database</i>] to specify the queue number. You can use <code>admin who</code>, <code>admin who, sqm</code>, and <code>admin who, sqt</code> to identify the queue number.</p> <p><i>q_type</i></p> <p>The queue type of the stable queue. Values are 0 for outbound queues and 1 for inbound queues. Use <code>admin who</code>, <code>admin who, sqm</code>, and <code>admin who, sqt</code> to identify the queue type.</p> <p><i>seg</i></p> <p>Identifies the starting segment.</p> <p><i>blk</i></p> <p>Identifies the 16K block in the segment where the dump is to begin. Block numbering starts at 1 and ends at 64.</p> <p><code>sysadmin dump_queue</code> recognizes four special settings for <i>seg</i> and <i>blk</i>:</p> <ul style="list-style-type: none"> • Setting <i>seg</i> to -1 starts with the first active segment in the queue. • Setting <i>seg</i> to -2 starts with the first segment in the queue, including any inactive segments retained by setting a save interval. • Setting <i>seg</i> to -1 and <i>blk</i> to -1 starts with the first undeleted block in the queue. • Setting <i>seg</i> to -1 and <i>blk</i> to -2 starts with the first unread block in the queue. <p><i>cnt</i></p> <p>Specifies the number of blocks to dump. This number can span multiple segments. If <i>cnt</i> is set to -1, the end of the current segment is the last block dumped. If it is set to -2, the end of the queue is the last block dumped.</p>

num_cmds

Specifies the number of commands to dump. This number overrides *cnt*. If *num_cmds* is set to -1, the end of the current segment is the last command dumped. If *num_cmds* is set to -2, the end of the queue is the last command dumped.

L0

Dumps all of the stable queue's content. This is the default behavior if L0, L1, L2, or L3 is not specified.

L1

Dumps only the begin and end commands of transactions found in the stable queue.

L2

Dumps the begin and end commands of the stable queue transactions together with the first 100 characters of all the other commands in the transactions.

L3

Dumps all of the stable queue's content. Except for SQL statements, all other commands are printed as comments. You can use L3 only when you use the *file_name* option or the `sysadmin dump_file` command to specify an alternate log file. You cannot use L3 with RSSD or client options.

RSSD

Forces output to system tables in the RSSD.

client

Forces output to the client that is issuing this command.

"log"

Forces output to the Replication Server log file.

file_name

Forces the output into the *file_name* log file. You can also set an alternate log file using the `sysadmin dump_file` command. The location of this file is recorded in the Replication Server log.

"next"[, *num_cmds*]

Starts from where the last run of `sysadmin dump_queue` for a particular queue and session left off, and dumps the same number of commands or blocks that the last run did. You can use *num_cmds* to override the value of previous *cnt* or *num_cmds*.

If you use "next"[, *num_cmds*] without a prior invocation of `sysadmin dump_queue`, the dump starts from the beginning of the queue with the default values of *seg* -1, *blk* -1, and *cnt* -2, and *num_cmds* is treated as the number of commands.

Examples

Example 1 Acting on queue 103:1, dumps blocks 15–64 of segment 0 and blocks 1–15 of segment 1 into the Replication Server log:

```
sysadmin dump_queue, 103, 1, 0, 15, 65
```

Example 2 Dumps all of queue 103:1 into the RSSD:

```
sysadmin dump_queue, 103, 1, -1, 1, -2, RSSD
```

Example 3 Dumps the contents of queue 103:1 into *SYDNEY_RS.log* log file. The last `sysadmin dump_file` command closes *SYDNEY_RS.log* and any subsequent dumps are directed to the Replication Server log:

```
sysadmin dump_file, SYDNEY_RS.log
sysadmin dump_queue, 103, 1, -1, 1, -2
sysadmin dump_file
```

Example 4 Dumps the contents of the inbound queue for SYDNEY_DS.pubs2 into the Replication Server log:

```
sysadmin dump_queue, SYDNEY_DS, pubs2, 1, -1, 1,
-2, 10, "log"
```

Example 5 Dumps 10 commands of queue 103:1 into the Replication Server log:

```
sysadmin dump_queue, 103, 1, -1, 1, -2, 10, "log"
```

Example 6 Dumps only the begin and end commands of queue 103:1 into the Replication Server log:

```
sysadmin dump_queue, 103, 1, -1, 1, -2, L1
```

Example 7 Dumps the contents of queue 103:1 into the Replication Server log:

```
sysadmin dump_queue, 103, 1, -1, 1, -2, "next"
```

Example 8 Dumps, in chunks, the contents of queue 103:1 into the Replication Server log. "next" dumps the queue from where the last run of sysadmin dump_queue left off. In this example, the first call to sysadmin dump_queue dumps the first ten commands, the second call dumps the next ten commands, and the last call dumps the next 20 commands:

```
sysadmin dump_queue, 103, 1, -1, 1, -2, 10
sysadmin dump_queue, 103, 1, "next"
sysadmin dump_queue, 103, 1, "next", 20
```

Usage

- Use sysadmin dump_queue to dump the contents of a Replication Server stable queue.
- sysadmin dump_queue dumps stable queues into one of the following:
 - Replication Server log
 - Alternate log file
 - RSSD
 - Client issuing the command

To dump queues into the RSSD or client, the last argument of sysadmin dump_queue must be RSSD or client.

If the RSSD or client option is not specified, or if the "log" option is specified, output goes into the Replication Server log.

If an alternative log file for dumping queues is specified through the sysadmin dump_file command or through the *file_name* option, the output goes into the alternative dump file.

- Specify the maximum command length used by this command by setting the queue_dump_buffer_size configuration parameter.

Dumping to the RSSD

If the RSSD option is used, the dump is written into two system tables in the RSSD, rs_queuemsg and rs_queuemsgtxt.

If the queue is dumped into the RSSD, the system tables are first cleared of the segments with the same *q_number*, *q_type*, *seg*, and *blk* as the blocks being dumped.

For information about the contents of the rs_queuemsg system table, see Chapter 8, "Replication Server System Tables."

The rs_queuemsgtxt system table holds the text of commands dumped from the stable queue. If the text of a command exceeds 255 characters, it is stored in multiple rows numbered with the q_seq column.

Dumping to the client

If the client option is used, the dump is written to the client issuing the command, such as isql or Replication Server Manager.

Permissions

sysadmin dump_queue requires “sa” permission.

See also

admin who, rs_queuemsg, rs_queuemsgtxt, sysadmin dump_file

sysadmin dump_thread_stacks

Description	Dumps Replication Server stacks.
Syntax	sysadmin dump_thread_stacks [, <i>module_name</i>]
Parameters	<i>module_name</i> The type of Replication Server thread. The valid module names are the same as the values under the name column displayed by the admin who command.
Examples	<p>Dumps the RSI queue stack:</p> <pre>sysadmin dump_thread_stacks, RSI T. 2006/10/23 15:37:39. (259): RS Thread Type = 'RSI' T. 2006/10/23 15:37:39. (259): RS Thread State = 'Awaiting Wakeup' T. 2006/10/23 15:37:39. (259): RS Thread Info = 'ost_columbia_02' T. 2006/10/23 15:37:39. (259): Open Server Process ID: 50, SRV_PROC address 0xed79c8 T. 2006/10/23 15:37:39. (259): Start of stack trace for spid 50. T. 2006/10/23 15:37:39. (259): Native thread #70, FramePointer: 0xfe34f050 T. 2006/10/23 15:37:39. (259): 0x00362fc8 sqm_read_message (0x3345ed0, 0xfe34fdf4, 0xea60, 0x0, 0xfe34fdf0, 0x47105f0) +0x48 T. 2006/10/23 15:37:39. (259): 0x00300908 _rsi_sender_wrapper (0x30c390, 0x30c230, 0x476f1f0, 0x47105f0, 0x1f2, 0x47105f0) +0x2f28 T. 2006/10/23 15:37:39. (259): 0x002fe960 _rsi_sender_wrapper (0x1d794f0, 0xffffd8f1, 0x268d14, 0xffffd800, 0x800, 0x0) +0xf80 T. 2006/10/23 15:37:39. (259): 0x0054dabc srv__start_function (0xed79c8, 0x0, 0x800, 0x862a04, 0x0, 0x0) +0x1c0 T. 2006/10/23 15:37:39. (259): 0xff265d48 _resume_ret (0x0, 0x0, 0x0, 0x0, 0x0, 0x0) +0x2d0 T. 2006/10/23 15:37:39. (259): End of stack trace for spid 50. T. 2006/10/23 15:37:39. (259):</pre>
Usage	<ul style="list-style-type: none">• Use sysadmin dump_thread_stacks to check the internal processes of Replication Server when Replication Server is unusually slow.• sysadmin dump_thread_stacks is available for these platforms:<ul style="list-style-type: none">• Sun Solaris

- HP/UX
- Linux
- IBM

Permissions `sysadmin dump_thread_stacks` requires “sa” permission.

See also `srv_dbg_stack()` in *Open Server Server-Library/C Reference Manual*

sysadmin dump_tran

Description	Dumps the statements of a specific stable queue transaction into a log file.
Syntax	<pre>sysadmin dump_tran {{, <i>q_number</i>, <i>server</i> [, <i>database</i>]}, <i>q_type</i>, <i>lqid</i> [, <i>num_cmds</i>] [, {L0 L1 L2 L3}] [, {RSSD client "log" <i>file_name</i>}] "next" [, <i>num_cmds</i>}}</pre>
Parameters	<p><i>q_number</i> <i>server</i> [, <i>database</i>]</p> <p>Identifies the stable queue. Use either <i>q_number</i> or <i>server</i> [, <i>database</i>] to specify the queue number. You can use admin who, admin who, sqm, and admin who, sqt to identify the queue number.</p> <p><i>q_type</i></p> <p>The queue type of the stable queue. Values are 0 for outbound queues and 1 for inbound queues. Use admin who, admin who, sqm, and admin who, sqt to identify the queue type.</p> <p><i>lqid</i></p> <p>The local queue ID of any command of a stable queue transaction. <i>lqid</i> identifies the transaction to dump. Format: <i>seg,blk,row</i>.</p> <p><i>num_cmds</i></p> <p>Specifies the number of commands to dump.</p> <p>L0</p> <p>Dumps the contents of the specified transaction. This is the default behavior if L0, L1, L2, or L3 is not specified.</p> <p>L1</p> <p>Dumps only the begin and end commands of the specified transaction.</p> <p>L2</p> <p>Dumps the begin and end commands of the specified transaction, together with the first 100 characters of the other commands in the transaction.</p> <p>L3</p> <p>Dumps all the commands of the specified transaction. All other commands are printed as comments except for SQL statements. You can use L3 only when you use the <i>file_name</i> option or the sysadmin dump_file command to specify an alternate log file. You cannot use L3 with the RSSD or client options.</p> <p>RSSD</p> <p>Forces output to system tables in the RSSD.</p>

client

Forces output to the client that issued the command.

"log"

Forces output to the Replication Server log file.

file_name

Forces the output into the *file_name* log file. You can set an alternate log file using the `sysadmin dump_file` command.

"next"[, num_cmds]

This option continues the last run of `sysadmin dump_tran`.

"next"[, num_cmds] starts from where the last run of `sysadmin dump_tran` for a particular transaction left off, and dumps the same number of commands that the last run did. You can use *num_cmds* to override the value of previous *cnt* or *num_cmds*.

You cannot use *"next"[, num_cmds]* without a prior invocation of `sysadmin dump_tran`.

Examples

Example 1 Dumps the transaction of queue 103:1 with LQID 0:15:2 into the Replication Server log:

```
sysadmin dump_tran, 103, 1, 0, 15, 2
```

Example 2 Dumps 10 commands of the transaction of the inbound queue for SYDNEY_DS.pubs2 with LQID 0:15:2 into the Replication Server log:

```
sysadmin dump_tran, SYDNEY_DS, pubs2, 1, 0, 15, 2,  
10, "log"
```

Example 3 Dumps only the begin and end commands of the transaction of queue 103:1 with LQID 0:15:2 into the Replication Server log:

```
sysadmin dump_tran, 103,1, 0, 15, 2, L1
```

Example 4 Dumps all of the commands of the transaction of queue 103:1 with LQID 0:15:2 into the Replication Server log. All the commands are truncated at 100 characters:

```
sysadmin dump_tran, 103,1, 0, 15, 2, L2
```

Example 5 Dumps the transaction of queue 103:1 with LQID 0:15:2 into the *SYDNEY_RS.log* file:

```
sysadmin dump_tran, 103,1, 0, 15, 2, L3, SYDNEY_RS.log
```

Example 6 Dumps the transaction of queue 103:1 with LQID 0:15:2 into the RSSD:

```
sysadmin dump_tran, 103, 1, 0, 15, 2, RSSD
```

Example 7 Dumps transaction of queue 103:1 with LQID 0:15:2 to the client:

```
sysadmin dump_tran, 103, 1, 0, 15, 2, client
```

Example 8 Dumps, in chunks, the transaction of queue 103:1 with LQID 0:15:2 into the Replication Server log. "next" dumps the transaction from where the last run of sysadmin dump_tran left off. In this example, the first call to sysadmin dump_tran dumps the first 10 commands of the transaction, the second call dumps the next 10 command of the transaction, and the last call dumps the next 20 commands of the transaction:

```
sysadmin dump_tran, 103,1, 0, 15, 2, 10
sysadmin dump_tran, "next"
sysadmin dump_tran, "next", 20
```

Usage

- Use sysadmin dump_tran to dump the contents of a stable queue transaction identified by the LQID.
- Output from sysadmin dump_tran goes to one of the following:
 - Replication Server log
 - Alternate log file
 - RSSD
 - Client issuing the command

To dump a stable queue transaction into the RSSD or a client, the last argument of sysadmin dump_tran must be RSSD or client.

If the RSSD or client option is not specified, or if the log option is specified, output goes to the Replication Server log.

If an alternative log file for dumping the stable queue transaction is specified through the sysadmin dump_file command or through the *file_name* option, the output goes to the alternative dump file.

- Specify the maximum sysadmin dump_tran command length by setting the queue_dump_buffer_size configuration parameter.

Dumping to the RSSD

If the RSSD option is used, the dump is written in two system tables in the RSSD, rs_queuemsg and rs_queuemsgtxt.

If the transaction is dumped to the RSSD, the system tables are first cleared of the segments with the same *q_number*, *q_type*, *seg*, and *blk* as the transaction being dumped.

For information about the contents of the rs_queuemsg system table, see Chapter 8, "Replication Server System Tables."

The `rs_queuemsgtxt` system table holds the text of commands dumped from the stable queue. If the text of a command exceeds 255 characters, it is stored in multiple rows numbered with the `q_seq` column.

Dumping to the client

If the client option is used, the dump is written to the client issuing the command, such as `isql` or Replication Server Manager.

Permissions

`sysadmin dump_tran` requires “sa” permission.

See also

`admin who`, `rs_queuemsg`, `rs_queuemsgtxt`, `sysadmin dump_file`

sysadmin erssd

Description

Allows you to check ERSSD file locations and backup configurations, or perform an unscheduled backup of the ERSSD.

The command returns the status of ERSSD, including:

- ERSSD name
- Database file location
- Transaction log file location
- Transaction mirror location
- Backup start time, start date, and intervals
- Backup directory location

Syntax

```
sysadmin erssd [, backup | dbfile_dir, 'path' | translog_dir, 'path' |
logmirror_dir, 'path' | defrag]
```

Parameters

`backup`

Performs a single unscheduled backup of the ERSSD.

`dbfile_dir, 'path'`

Specifies a new directory for the ERSSD database file.

`translog_dir, 'path'`

Specifies a new directory for the transaction log file.

`logmirror_dir, 'path'`

Specifies a new directory for the transaction log mirror file.

`defrag`

Rebuilds the ERSSD database without empty fragments.

path
The pathname of the new directory.

Note Use these directory path alteration options with caution. Executing sysadmin erssd with these options automatically reboots ERSSD, and may cause system disruption.

Examples This example shows the output of sysadmin erssd:

```
sysadmin erssd
-----

ERSSD Name      ERSSD Database File      ERSSD Transaction Log
-----
erssd.db        /dbfile/erssd.db         /log/erssd.log

ERSSD Transaction Log Mirror ERSSD Backup Start Time
-----
/backup/erssd.mlg          2am

ERSSD Backup Start Date      ERSSD Backup Interval
-----
March 20, 2003              12 hours

ERSSD Backup Location
-----
/backup
```

- Usage
- Using this command with no options displays the database file path, the transaction log path, the transaction log mirror path, and the start-time, start-date, and location of scheduled transactions.
 - Using this command with the backup option performs one unscheduled backup.
 - Using this command with the option dbfile_dir shuts down ERSSD, moves the database to the new directory, updates the Replication Server configuration file, and restarts ERSSD, using the database from the new location.

- Using this command with the option `translog_dir` shuts down ERSSD, moves the transaction log file to the new directory, updates the ERSSD to use the transaction log mirror in the new directory, updates the Replication Server configuration file, and restarts ERSSD.
- Using this command with the option `logmirror_dir` shuts down ERSSD, moves the transaction log mirror file to the new directory, updates the ERSSD to use the transaction log mirror in the new directory, updates the Replication Server configuration file, and restarts ERSSD.
- Use this command with the option `defrag` shuts down ERSSD, rebuilds the database file, and restarts ERSSD.
- Using this command with the options `defrag`, `dbfile_dir`, `translog_dir`, and `logmirror_dir` is expensive. During this operation ERSSD is unavailable and all threads that attempt to access it fail. These threads remain blocked until ERSSD is restarted.
- Your site version must be 15.0 or above to use `defrag`. The defragmented file is automatically upgraded to SQL Anywhere 11.0 by this option, and cannot be downgraded after the command is executed.
- Use this command when you need to move files to larger, faster disks.
- Use single, not double, quotation marks in *path*.

Permissions

You must have "sa" privileges to execute this command.

sysadmin fast_route_upgrade

Description	<p>Updates the route version to the site version of the lower of the primary or replicate Replication Server.</p> <p>Upgrading a route rematerializes the data in system tables and makes information associated with new features available to a newly upgraded Replication Server.</p> <hr/> <p>Note Use <code>sysadmin fast-route-upgrade</code> <i>only</i> if the primary Replication Server has not used new features that require materialization.</p> <hr/>
Syntax	<code>sysadmin fast_route_upgrade, dest_replication_server</code>
Parameters	<p><i>dest_replication_server</i></p> <p>The destination Replication Server for the route.</p>
Examples	<p>Example 1 In these examples, the site version of TOKYO_RS is 1200. SYDNEY_RS has just been upgraded from 11.5 to 12.0; its site version is 1200. Issued at the source Replication Server (SYDNEY_RS) for the route terminating at the Tokyo Replication Server (TOKYO_RS), this command sets the version of the route to 12.0. New features have not yet been used at SYDNEY_RS:</p> <pre>sysadmin fast_route_upgrade, TOKYO_RS</pre> <p>Example 2 Issued at the source Replication Server (TOKYO_RS) for the route terminating at the Sydney Replication Server (SYDNEY_RS), this command is rejected since new features have been used at TOKYO_RS, and you must upgrade the route using Sybase Central's Replication Manager plug-in:</p> <pre>sysadmin fast_route_upgrade, SYDNEY_RS</pre>
Usage	<ul style="list-style-type: none">• Whenever Replication Servers at both ends of a route have been upgraded and site versions set to 11.5 or later, you <i>must</i> upgrade each route that connects the two servers to enable new features to flow through it. Issue this command at the source Replication Server to update the route version.• Use <code>sysadmin fast_route_upgrade</code> to upgrade the route if new features have not been used at the source Replication Server.• If you have used new features at the source Replication Server, the command is rejected and you must upgrade the route using Replication Manager (RM).
Permissions	<code>sysadmin fast_route_upgrade</code> requires "sa" permission.
See also	<code>admin show_route_versions</code> , <code>admin show_site_version</code> , <code>sysadmin site_version</code>

sysadmin hibernate_off

Description	Turns off hibernation mode for the Replication Server and returns it to an active state.
Syntax	<code>sysadmin hibernate_off [, <i>string_ID</i>]</code>
Parameters	<p><i>string_ID</i></p> <p>A valid identifier. If <i>string_ID</i> was specified with <code>sysadmin hibernate_on</code>, you must specify the same one that was used for <code>sysadmin hibernate_on</code>.</p> <p>If you forget the <i>string_ID</i>, you can find it in the text column of the <code>rs_recovery</code> system table.</p> <p>If you need to turn off hibernation mode for a replicate Replication Server after a successful route upgrade or route upgrade recovery, use the Replication Server name for the <i>string_ID</i>.</p>
Examples	<p>This command turns off the hibernation mode of the Replication Server (TOKYO_RS):</p> <pre>sysadmin hibernate_off, TOKYO_RS</pre>
Usage	<ul style="list-style-type: none"> • Hibernation mode is a Replication Server state in which: <ul style="list-style-type: none"> • all Data Definition Language (DDL) commands are rejected, • most service threads, such as Data Server Interface (DSI), distributor, and Replication Server Interface (RSI) sender threads, are suspended, • all routes and connections are suspended, and • RSI users are logged off and not allowed to log back into the Replication Server. • You can execute system information (admin) and system administration (sysadmin) type commands while in hibernation mode. • Execute this command at the Replication Server for which you want turn off hibernation mode. • A destination Replication Server might be in hibernation mode when route upgrade fails. Do not use <code>sysadmin hibernate_off</code> to reactivate the Replication Server. Use Replication Manger to recover the route upgrade. For more information, please see the Replication Manger online help. • Occasionally, a destination Replication Server is placed into hibernation mode after a successful route upgrade. Use <code>sysadmin hibernate_off</code> to reactivate the destination Replication Server.
Permissions	<code>sysadmin hibernate_off</code> requires “sa” permission.

See also

`sysadmin hibernate_on`

sysadmin hibernate_on

Description	<p>Turns on hibernation mode for (or suspends) the Replication Server.</p> <hr/> <p>Warning! The <code>sysadmin hibernate_on</code> command may result in loss detection when Replication Server has routes.</p> <hr/>
Syntax	<code>sysadmin hibernate_on [, <i>string_ID</i>]</code>
Parameters	<p><i>string_ID</i></p> <p>A valid identifier. You must use the same <i>string_ID</i> when you execute <code>sysadmin hibernate_off</code>. You can use <i>string_ID</i> to ensure that no-one else accidentally turns off hibernation mode for the Replication Server while you are working on it.</p> <p>If you forget the <i>string_ID</i>, you can find it in the text column of the <code>rs_recovery</code> system table.</p>
Examples	<p>This command turns on the hibernation mode of the Replication Server (TOKYO_RS):</p> <pre>sysadmin hibernate_on, TOKYO_RS</pre>
Usage	<ul style="list-style-type: none">• Hibernation mode is a Replication Server state in which:<ul style="list-style-type: none">• all Data Definition Language (DDL) commands are rejected,• most service threads, such as Data Server Interface (DSI), distributor, and Replication Server Interface (RSI) sender threads, are suspended,• all routes and connections are suspended, and• RSI users are logged off and not allowed to log back into the Replication Server.• You can execute system information (admin) and system administration (sysadmin) type commands while in hibernation mode.• Execute this command at the Replication Server for which you want turn on hibernation mode.• You can turn hibernation mode on for a Replication Server to help you debug problems.
Permissions	<code>sysadmin hibernate_on</code> requires “sa” permission.
See also	<code>sysadmin hibernate_off</code>

sysadmin issue_ticket

Description	Injects an rs_ticket marker into the inbound queue.
Syntax	<code>sysadmin issue_ticket {,q_number} [{,ds_name, db_name}, h1 [, h2 [, h3 [, h4]]] [,v]</code>
Parameters	<p><i>ds_name</i> Name of the data server on which the database resides.</p> <p><i>db_name</i> Name of the database.</p> <p><i>q_number</i> Identifies the stable queue.</p> <p><i>h1,h2,h3</i> Each each parameter contains from 1– 10 characters; these parameters must follow the database identifier naming convention since the parameters are used as identifiers, in any way you see fit. The header parameter must not start with a number and must not be a reserved word. If number is chosen to be a header parameter, it must be within quotes. For example, '1'.</p> <p><i>h4</i> Contains from 1– 50 characters. Like h1, h2, and h3, you can use this parameter as an identifier in any way you see fit.</p> <p><i>v</i> Identifies the version number of the rs_ticket. It should be either 1 or 2. The default value is 2, if not specified.</p>
Examples	<p>This command injects one transaction in to the Replication Server inbound queue.</p> <pre>sysadmin issue_ticket, 103, 'start' go</pre> <p>where:</p> <p><i>103</i> is the <i>q_number</i> of a logical connection to Replication Server.</p>
Usage	<p>When using the sysadmin issue_ticket command:</p> <ul style="list-style-type: none">• You must have at least one subscription from the replicated database in Replication Server. If there are no subscriptions, the Distributor (DIST) module will not send the rs_ticket marker to the corresponding Data Server Interface (DSI).• The timestamp for the primary database (PDB) and EXEC module is an arbitrary value in the injected rs_ticket marker

- You can specify a stable queue only by using *q_number* or *ds_name*, *db_name*. In a warm standby environment, an inbound queue is related to the logical connection, and Replication Server does not have inbound queue for the standby database. When using `sysadmin issue_ticket` command for warm standby:
 - If the user specifies the stable queue by an existing logical connection or the physical connection for the active database, the specific `rs_ticket` marker is written into Replication Server inbound queue. The corresponding `rs_ticket` record can be found in both the replicate database and the standby database at the primary site.
 - If the user specifies the stable queue by an existing physical connection for the standby database, an error message appears indicating that no such inbound queue exists.

sysadmin log_first_tran

Description	Writes the first transaction in a DSI queue into the exceptions log.
Syntax	sysadmin log_first_tran, [n], <i>data_server</i> , <i>database</i>
Parameters	<p><i>n</i></p> <p>Specifies the number of transactions to write to the database exceptions log, and to either the Replication Server log or the alternative log file specified by the sysadmin dump_file command.</p> <p><i>data_server</i></p> <p>The name of the data server with the database.</p> <p><i>database</i></p> <p>The name of the database from whose DSI queue the first transaction is to be written.</p>
Examples	<p>Example 1 Writes the first transaction in this DSI queue to the exceptions log:</p> <pre>sysadmin log_first_tran, SYDNEY_DS, pubs2</pre> <p>Example 2 Writes the first five transactions in the DSI queue to the database exceptions log, and to either the Replication Server log or the location specified by the sysadmin dump_file command:</p> <pre>sysadmin log_first_tran, 5, SYDNEY_DS, pubs2</pre> <p>Example 3 Writes the first two transactions in the DSI queue to the database exceptions log and to the <i>SYDNEY_RS.log</i> file. The last sysadmin dump_file command closes the <i>SYDNEY_RS.log</i> file:</p> <pre>sysadmin dump_file SYDNEY_RS.log sysadmin log_first_tran, 2, SYDNEY_DS, pubs2 sysadmin dump_file</pre>
Usage	<ul style="list-style-type: none">• Use sysadmin log_first_tran to write the first <i>n</i> transactions in the DSI queue into the exceptions log, and to either the Replication Server log or the alternative log file specified by the sysadmin dump_file command.• This command does not delete the first <i>n</i> transactions from the queue.• The exceptions log consists of three tables, rs_exceptshdr, rs_exceptscmd, and rs_systext. See Chapter 8, “Replication Server System Tables,” for detailed descriptions of these tables.
Permissions	sysadmin log_first_tran requires “sa” permission.
See also	admin who

sysadmin purge_all_open

Description Purges all open transactions from an inbound queue of a Replication Server.

Syntax sysadmin purge_all_open, *q_number*, *q_type*

Parameters *q_number*, *q_type*

Identifies the stable queue to purge. Find these values using admin who, admin who, sqm, and admin who, sqt.

Examples Purges all open transactions from queue 103:1:

```
sysadmin purge_all_open, 103, 1
```

Usage

- Use sysadmin purge_all_open to purge all open transactions from an inbound queue of a Replication Server. Open transactions can only be purged from inbound queues.

Note A transaction is open when the RepAgent has forwarded the transaction begin record, and possibly some commands within the transaction, but has not yet forwarded the transaction commit or abort record.

- sysadmin purge_all_open is useful if you have to truncate a data server log before it has been completely forwarded to the Replication Server, leaving open transactions in the Replication Server inbound queues. These must be removed explicitly using sysadmin purge_all_open.

Warning! Use sysadmin purge_all_open *only* when there are open transactions in the inbound queue and you are certain that the RepAgent will not forward the commit or abort record from the log.

- Replication Server needs enough storage to purge a stable queue. If you do not have enough storage, this error message appears:

```
This RS is out of Disk Space. Use another session to
add disk space for this command to proceed.
```

If this occurs, start another isql session and add stable storage to the Replication Server. sysadmin purge_all_open cannot proceed until sufficient storage is available.

- To review the contents of the transactions being dropped, execute sysadmin sqt_dump_queue before you use this command.

- If the queue has no open transactions, this command leaves the queue unchanged. If the Replication Server is restarted after transactions are purged, they may reappear as a result of recovery operations.

Permissions

sysadmin purge_all_open requires “sa” permission.

See also

admin who, alter partition, create partition, sysadmin purge_first_open, sysadmin sqt_dump_queue

sysadmin purge_first_open

Description	Purges the first open transaction from the inbound queue of a Replication Server.
Syntax	<code>sysadmin purge_first_open, q_number, q_type</code>
Parameters	<i>q_number, q_type</i> Identifies the stable queue to be purged. Find these values using <code>admin who</code> , <code>admin who, sqm</code> , and <code>admin who, sqt</code> .
Examples	Purges the first open transaction from queue 103:1: <pre>sysadmin purge_first_open, 103, 1</pre>
Usage	<ul style="list-style-type: none">• <code>sysadmin purge_first_open</code> removes the first open transaction from a Replication Server's inbound queue. RepAgent threads transfer transactions from the database log one record at a time. A transaction is open when the RepAgent has forwarded the transaction begin record, and possibly some commands within the transaction, but has not yet forwarded the transaction commit or abort record.• <code>sysadmin purge_first_open</code> can be only used with inbound queues.• Replication Server needs enough space to purge the first open transaction from a stable queue. If there is not enough disk space, this error message appears: <pre>This RS is out of Disk Space. Use another session to add disk space for this command to proceed.</pre> If this occurs, start another <code>isql</code> session and add stable storage (disk space) to the Replication Server. <code>sysadmin purge_first_open</code> cannot proceed until sufficient storage is available.• To review the contents of the transaction being dropped, execute <code>sysadmin sqt_dump_queue</code> before you use this command.• To display information about the first transaction in the inbound queue, use <code>admin who, sqt</code>. If the state of the first transaction is "open" (ST:O), it can be dropped from the queue.• The <code>sysadmin purge_first_open</code> command is useful when there is an uncommitted transaction in the Adaptive Server log. The open transaction is delivered by the RepAgent to Replication Server. Because there is an open transaction, Replication Server cannot truncate the inbound queue. If the transaction remains open for a long time, the inbound queue fills and Replication Server may run out of queue space.

- If the first transaction of the queue is not open, this command leaves the queue unchanged. If the Replication Server is restarted after a transaction is dropped, the transaction may reappear as a result of recovery operations.

Warning! Use `sysadmin purge_first_open` only when you have determined (by using `admin who, sqt` and `admin who, sqm`) that the inbound queue is stuck on an uncommitted transaction.

Permissions

`sysadmin purge_first_open` requires “sa” permission.

See also

`admin who`, `alter partition`, `create partition`, `sysadmin dump_queue`, `sysadmin purge_all_open`

sysadmin purge_route_at_replicate

Description	Removes all references to a primary Replication Server from a replicate Replication Server.
Syntax	<code>sysadmin purge_route_at_replicate, replication_server</code>
Parameters	<i>replication_server</i> The name of the primary Replication Server to be purged from the replicate's RSSD.
Examples	Purges the primary Replication Server, TOKYO_RS, from the replicate's RSSD: <code>sysadmin purge_route_at_replicate, TOKYO_RS</code>
Usage	<ul style="list-style-type: none">• Use <code>sysadmin purge_route_at_replicate</code> to remove all subscriptions and route information originating from a specified primary Replication Server after the route is dropped from it. This is useful after <code>drop route with nowait</code> is executed at the primary Replication Server.• If there is a route from the current Replication Server to the specified primary Replication Server, you must drop the route before executing this command.• If a subscription was materializing when <code>drop route with nowait</code> was executed at the primary Replication Server, a materialization queue may be left at the replicate Replication Server. Use <code>sysadmin drop_queue</code> to remove this queue. <hr/> <p>Warning! Use <code>sysadmin purge_route_at_replicate</code> only if the <code>drop route with nowait</code> command was executed at the primary Replication Server or if the primary Replication Server is lost and will not be recovered.</p> <hr/>
Permissions	<code>sysadmin purge_route_at_replicate</code> requires “sa” permission.
See also	<code>drop route</code> , <code>rs_helproute</code>

sysadmin restore_dsi_saved_segments

Description	Restores backlogged transactions.
Syntax	<code>sysadmin restore_dsi_saved_segments, <i>data_server</i>, <i>database</i></code>
Parameters	<i>data_server</i> The name of the data server. <i>database</i> The name of the database.
Examples	Restores backlogged transactions for the pubs2 database in the TOKYO_DS data server: <pre>sysadmin restore_dsi_saved_segments, TOKYO_DS, pubs2</pre>
Usage	<ul style="list-style-type: none">• The DSI must be explicitly suspended before you can use this command to restore saved segments.• Any backlogged transactions saved because a save interval was specified for the connection (using <code>alter connection</code>) are candidates for restoring into the database. The Replication Server uses <code>rs_get_lastcommit</code> to decide which transactions to filter.
Permissions	<code>sysadmin restore_dsi_saved_segments</code> requires “sa” permission.
See also	<code>configure connection</code>

sysadmin set_dsi_generation

Description	Changes a database generation number in the Replication Server to prevent the application of transactions in the DSI stable queue after a replicate database is restored.
Syntax	<code>sysadmin set_dsi_generation, <i>gen_number</i>, <i>primary_data_server</i>, <i>primary_database</i>, <i>replicate_data_server</i>, <i>replicate_database</i></code>
Parameters	<p><i>gen_number</i> The new generation number of the database. The number is an integer between 0 and 65,535.</p> <p><i>primary_data_server</i> The name of the data server at the primary site.</p> <p><i>primary_database</i> The name of the primary database.</p> <p><i>replicate_data_server</i> The name of the replicate data server.</p> <p><i>replicate_database</i> The name of the replicate database.</p>
Examples	<p>Sets new DSI generation number to 105. The previous number was 104 or less:</p> <pre>sysadmin set_dsi_generation 105 NY_DS, ny_db, SF_DS, sf_db</pre>
Usage	<p>Use <code>sysadmin set_dsi_generation</code> during the recovery of a database dump. Changing the generation number except during recovery may cause incorrect data at replicate databases.</p> <p>See the <i>Replication Server Administration Guide Volume 2</i> for a complete description of the recovery procedure.</p>
Permissions	<code>sysadmin set_dsi_generation</code> requires “sa” permission.
See also	<code>admin get_generation</code> , <code>configure connection</code> , <code>dbcc dbrepair</code> , <code>dbcc settrunc</code> , <code>rebuild queues</code>

sysadmin site_version

Description Sets the site version number for the Replication Server. This lets you use the software features in the corresponding release, and prevents you from downgrading to an earlier release. If the Replication Server uses ERSSD, this command also shuts down the ERSSD, upgrades its database file and restarts ERSSD.

Note If your Replication Server uses ERSSD, this command may cause some threads to shutdown since ERSSD is being restarted. Replication should continue after you restart all threads that are shutdown.

Syntax sysadmin site_version [, version]

Parameters *version*
The site version number for Replication Server.

Version number	Site version
Pre-11.5	N/A
11.5	1150
12.0	1200
12.5	1250
12.6	1260
15.0, 15.0.1	1500
15.1	1510
15.2	1520
15.5	1550

No site version numbers exist for releases earlier than 11.5. Maintenance releases may support higher site version numbers.

Examples **Example 1** Displays the current site version number for the Replication Server:

```
sysadmin site_version
```

Example 2 Changes the site version number to correspond to release 15.5:

```
sysadmin site_version, 1550
```

Usage

- To set the site version number for the current Replication Server, execute sysadmin site_version with a *version* parameter.

The site version number you enter must be no higher than the software version number or the release level of Replication Server.

- To display the site version number for Replication Server, execute `sysadmin site_version` without a *version* parameter.
- You can use new software features up to the version set in Replication Server's site version.
- For a newly installed Replication Server of release 15.5, the site version number is 1550.
- For more information about features that were introduced in a particular Replication Server software release, see *Replication Server New Features Guide* for that release.

Warning! When you set the site version number, you cannot downgrade to an earlier release.

- For more information about installing or upgrading Replication Servers, refer to the Replication Server installation and configuration guides for your platform.

Mixed-version replication systems

In a mixed-version replication system, different Replication Servers have different site versions. In such a system, some features are only available to Replication Servers with higher site versions. For example, the site version of a primary Replication Server and one of its replicate Replication Server is 1550, while the site version its other replicate Replication Server is 1260. When a table replication definition has a `timestamp` column, the replicate Replication Server with the lower site version can only subscribe to the `timestamp` as `varbinary (8)`, while the replicate Replication Server with 1550 site version can subscribe to the `timestamp` column directly.

Upgrading routes

- After you have upgraded one or both Replication Servers on either end of a route to a higher release level, and you have set the site versions to a higher level, you need to upgrade the route. Upgrading a route rematerializes the data in system tables and makes information associated with new features available to a newly upgraded Replication Server.

There are two possible scenarios for route upgrade:

- If you have Replication Manger, use the Replication Manger to upgrade routes. For instruction on upgrading routes, please see the Replication Manger online help
- If new features have not been used at the source Replication Server, use `sysadmin fast_route_upgrade` to upgrade routes.

For example, if you upgrade a Replication Server of release 12.6 to release 15.0 and set its site version accordingly, you will need to upgrade a route from another Replication Server of release 15.0. When you upgrade the route, the newly upgraded Replication Server receives information from the 15.0 Replication Server such as additional replication definitions for the table.

See the *Replication Server Configuration Guide* for more information about upgrading routes.

System tables for version information

Version information is stored in the `rs_version` system table. The `rs_routes` system table also contains version information. Route version information is stored in the `rs_routeversions` system table.

Permissions

`sysadmin site_version` requires “sa” permission.

See also

`admin version`, `sysadmin fast_route_upgrade`, `sysadmin system_version`

sysadmin skip_bad_repserver_cmd

Description	<p>Instructs Replication Server to skip a failed replication definition request the next time Replication Agent starts. Use <code>sysadmin skip_bad_repserver_cmd</code> with the replication definition change request change process. See “Replication definition change request process,” in Chapter 9, “Managing Replicated Tables” in the <i>Replication Server Administration Guide Volume 1</i> before you use <code>sysadmin skip_bad_repserver_cmd</code>.</p> <hr/> <p>Warning! Use <code>sysadmin skip_bad_repserver_cmd</code> carefully. If you execute the command, and then restart the Replication Agent without executing the corrected replication definition command in the primary Replication Server, primary data may replicate using the wrong replication definition version.</p> <hr/>
Syntax	<code>sysadmin skip_bad_repserver_cmd, pds_name, pdb_name</code>
Parameters	<p><i>pds_name</i> The primary data server name.</p> <p><i>pdb_name</i> The primary database name.</p>
Examples	<p>In this example, <code>sysadmin skip_bad_repserver_cmd</code> instructs Replication Server and Replication Agent to skip the last failed replication definition command in the <code>pubs2</code> database of the <code>SYDNEY_DS</code> data server:</p> <pre>sysadmin skip_bad_repserver_cmd, SYDNEY_DS, pubs2</pre>
Usage	<ul style="list-style-type: none">• <i>pds_name</i> and <i>pdb_name</i> identify the specific Replication Agent that is affected by the failed replication definition request.• Use <code>sysadmin skip_bad_repserver_cmd</code> to instruct Replication Server to skip a failed replication definition request sent by a Replication Agent. When a replication definition command fails at the primary Replication Server, Replication Agent shuts down. If you restart Replication Agent, the failed command executes again unless Replication Server skips the command. After you execute <code>sysadmin skip_bad_repserver_cmd</code>, execute the corrected replication definition request at the Replication Server before starting the Replication Agent.
Permissions	<code>sysadmin skip_bad_repserver_cmd</code> requires “sa” permission.

See also

admin verify_repserver_cmd, rs_send_repserver_cmd, alter replication definition, alter applied function replication definition, alter request function replication definition, create replication definition, create applied function replication definition, create request function replication definition, drop replication definition

sysadmin sqm_purge_queue

Description	<p>Purges all messages from a stable queue.</p> <hr/> <p>Warning! Purging messages from a stable queue can result in data loss and should be used only with the advice of Sybase Technical Support. Replication Server cannot send purged messages to the destination database or Replication Server, and this causes inconsistencies in the replication system. If a queue contains subscription marker messages or route messages, using this command can have severe consequences.</p> <hr/>
Syntax	<code>sysadmin sqm_purge_queue, <i>q_number</i>, <i>q_type</i></code>
Parameters	<p><i>q_number</i>, <i>q_type</i></p> <p>Identifies the stable queue to be purged. Find these using <code>admin who</code>, <code>admin who, sqm</code>, or <code>admin who, sqt</code>.</p>
Examples	<p>Purges all messages from inbound queue number 103:</p> <pre>sysadmin sqm_purge_queue, 103, 1</pre>
Usage	<ul style="list-style-type: none">• <code>sysadmin sqm_purge_queue</code> removes messages destined to another Replication Server from a stable queue. Use this command when your queues are filled with messages.• <code>sysadmin sqm_purge_queue</code> can only be executed when the Replication Server has been started in standalone mode.
Permissions	Requires “sa” permission.
See also	<code>admin who</code> , <code>repserver</code>

sysadmin sqm_unzap_command

Description	Undeletes a message in a stable queue.
Syntax	sysadmin sqm_unzap_command, <i>q_number</i> , <i>q_type</i> , <i>seg</i> , <i>blk</i> , <i>row</i>
Parameters	<p><i>q_number</i>, <i>q_type</i></p> <p>Identifies the stable queue with the message to be restored. Find these values using <code>admin who</code>, <code>admin who, sqm</code>, and <code>admin who, sqt</code>.</p> <p><i>seg</i></p> <p>Identifies the segment in the stable queue that contains the message to be undeleted.</p> <p><i>blk</i></p> <p>Identifies the 16K block in the segment. Block numbering starts at 1 and ends at 64.</p> <p><i>row</i></p> <p>The row number in the block of the command to be undeleted.</p>
Usage	<ul style="list-style-type: none">• The Replication Server must be in standalone mode to use <code>sysadmin sqm_unzap_command</code>.• <code>sysadmin sqm_unzap_command</code> removes the delete mark from a message in a stable queue. Use this command to restore a message that you marked deleted using <code>sysadmin sqm_zap_command</code>.• Use <code>sysadmin dump_queue</code> to locate the message you want to restore.
Permissions	<code>sysadmin sqm_unzap_command</code> requires “sa” permission.
See also	<code>admin who</code> , <code>sysadmin drop_queue</code> , <code>sysadmin sqm_zap_command</code>

sysadmin sqm_unzap_tran

Description	Restores a specific transaction into the stable queue and returns a message stating the number of restored commands.
Syntax	<pre>sysadmin sqm_unzap_tran {, <i>q_number</i>, <i>server</i> [, <i>database</i>]}, <i>q_type</i>, <i>lqid</i> [, {L0 L1 L2 L3}] [, {RSSD client "log" <i>file_name</i>}]</pre>
Parameters	<p><i>q_number</i> <i>server</i> [, <i>database</i>] Identifies the stable queue. Use either <i>q_number</i> or <i>server</i> [, <i>database</i>] to specify the queue number. You can use admin who, admin who, sqm, and admin who, sqt to identify the queue number.</p> <p><i>q_type</i> The queue type of the stable queue. Values are 0 for outbound queues and 1 for inbound queues. Use admin who, admin who, sqm, and admin who, sqt to identify the queue type.</p> <p><i>lqid</i> The local queue ID of any command of a stable queue transaction. <i>lqid</i> identifies the transaction to restore into the stable queue. Format: <i>seg,blk,row</i>.</p> <p>L0 Dumps the contents of the restored transaction. This is the default behavior if L0, L1, L2, or L3 is not specified.</p> <p>L1 Dumps only the begin and end commands of the restored transaction.</p> <p>L2 Dumps the begin and end commands of the restored transaction together with the first 100 characters of the other commands in the restored transaction.</p> <p>L3 Dumps all the commands of the restored transaction. All other commands are printed as comments, except for SQL statements. You can use L3 only when you use the <i>file_name</i> option or the sysadmin dump_file command to specify an alternate log file. You cannot use L3 with RSSD or client options.</p> <p>RSSD Forces output to system tables in the RSSD.</p> <p>client Forces output to the client that issued the command.</p>

"log"

Forces output to the Replication Server log file.

file_name

Forces output to the *file_name* log file. You can also set an alternate log file using the sysadmin dump_file command.

Examples

Example 1 Restores the transaction of queue 103:1 with LQID 0:15:2 and dumps the transaction into the Replication Server log:

```
sysadmin sqm_unzap_tran, 103, 1, 0, 15, 2
```

Example 2 Restores the transaction of the inbound queue for SYDNEY_DS.pubs2 with LQID 0:15:2 and dumps the transaction into the Replication Server log:

```
sysadmin sqm_unzap_tran, SYDNEY_DS, pubs2, 1, 0, 15,  
2, "log"
```

Example 3 Restores the transaction of queue 103:1 with LQID 0:15:2 and dumps the begin and end commands of the transaction into the Replication Server log:

```
sysadmin sqm_unzap_tran, 103,1, 0, 15, 2, L1
```

Example 4 Restores the transaction of queue 103:1 with LQID 0:15:2 and dumps the transaction into the Replication Server log. All the commands are truncated at 100 characters:

```
sysadmin sqm_unzap_tran, 103,1, 0, 15, 2, L2
```

Example 5 Restores the transaction of queue 103:1 with LQID 0:15:2 and dumps the transaction into the *SYDNEY_RS.log* file:

```
sysadmin sqm_unzap_tran, 103,1, 0, 15, 2, L3,  
SYDNEY_RS.log
```

Example 6 Restores the transaction of queue 103:1 with LQID 0:15:2 and dumps the transaction into the RSSD:

```
sysadmin sqm_unzap_tran, 103, 1, 0, 15, 2, RSSD
```

Example 7 Restores the transaction of queue 103:1 with LQID 0:15:2 and dumps the transaction to the client:

```
sysadmin sqm_unzap_tran, 103, 1, 0, 15, 2, client
```

Usage

- The Replication Server must be in standalone mode to use sysadmin sqm_unzap_tran.

- `sysadmin sqm_unzap_tran` removes the delete mark from a transaction in a stable queue. Use this command to restore a transaction that you marked deleted using `sysadmin sqm_zap_tran`.
- Use `sysadmin dump_queue` to locate the transaction you want to restore.
- `sysadmin sqm_unzap_tran` dumps the restored transaction contents into one of the following:
 - Replication Server log
 - Alternate log file
 - RSSD
 - Client issuing the command

To dump queues into the RSSD or client, the last argument of `sysadmin dump_queue` must be `RSSD` or `client`.

If the `RSSD` or `client` option is not specified, or if the `"log"` option is specified, output goes into the Replication Server log.

If an alternative log file for dumping queues is specified through the `sysadmin dump_file` command or through the `file_name` option, the output goes into the alternative dump file.

Permissions

`sysadmin sqm_unzap_tran` requires “sa” permission.

See also

`admin who`, `sysadmin drop_queue`, `sysadmin sqm_unzap_command`, `sysadmin sqm_zap_command`, `sysadmin sqm_zap_tran`

sysadmin sqm_zap_command

Description	Deletes a single message in a stable queue.
Syntax	sysadmin sqm_zap_command, <i>q_number</i> , <i>q_type</i> , <i>seg</i> , <i>blk</i> , <i>row</i>
Parameters	<p><i>q_number</i>, <i>q_type</i></p> <p>Identifies the stable queue with the message to be deleted. Find these values using admin who, admin who, sqm, and admin who, sqt.</p> <p><i>seg</i></p> <p>Identifies the segment in the stable queue.</p> <p><i>blk</i></p> <p>Identifies the 16K block in the segment. Block numbering starts at 1 and ends at 64.</p> <p><i>row</i></p> <p>The row number in the block of the command to be deleted.</p>
Examples	<pre>sysadmin sqm_zap_command sysadmin sqm_zap_command, 103, 1, 15, 65, 2</pre>
Usage	<ul style="list-style-type: none">• The Replication Server must be in standalone mode to use sysadmin sqm_zap_command.• Use sysadmin dump_queue to locate the message you want to delete.• sysadmin sqm_zap_command marks a message in a stable queue as deleted. When Replication Server processes the queue, it ignores the marked message.• You can restore a message using sysadmin sqm_unzap_command. This command removes the delete mark from the message.• If you delete a message and then restart Replication Server in normal mode, the part of the queue holding the message may have been processed. If it was, you cannot restore the message with sysadmin sqm_unzap_command.
Permissions	sysadmin sqm_zap_command requires “sa” permission.
See also	admin who, sysadmin dump_queue, sysadmin sqm_unzap_command

sysadmin sqm_zap_tran

Description	Deletes a specific transaction from the stable queue and returns a message stating the number of deleted commands.
Syntax	<pre>sysadmin sqm_zap_tran {, <i>q_number</i>, <i>server</i> [, <i>database</i>]}, <i>q_type</i>, <i>lqid</i> [, {L0 L1 L2 L3}] [, {RSSD client "log" <i>file_name</i>}]</pre>
Parameters	<p><i>q_number</i> <i>server</i> [, <i>database</i>] Identifies the stable queue. Use either <i>q_number</i> or <i>server</i> [, <i>database</i>] to specify the queue number. You can use admin who, admin who, sqm, and admin who, sqt to identify the queue number.</p> <p><i>q_type</i> The queue type of the stable queue. Values are "0" for outbound queues and "1" for inbound queues. Use admin who, admin who, sqm, and admin who, sqt to identify the queue type.</p> <p><i>lqid</i> The local queue ID of any command of a stable queue transaction. <i>lqid</i> identifies the transaction to delete from the stable queue. Format: <i>seg,blk,row</i>.</p> <p>L0 Dumps the contents of the deleted transaction. This is the default behavior if L0, L1, L2, or L3 is not specified.</p> <p>L1 Dumps only the begin and end commands of the deleted transaction.</p> <p>L2 Dumps the begin and end commands of the deleted transaction together with the first 100 characters of the other commands in the deleted transaction.</p> <p>L3 Dumps all the commands of the deleted transaction. All other commands are printed as comments, except for SQL statements. You can use L3 only when you use the <i>file_name</i> option or the sysadmin dump_file command to specify an alternate log file. You cannot use L3 with RSSD or client options.</p> <p>RSSD Forces output to system tables in the RSSD.</p> <p>client Forces output to the client that issued the command.</p>

"log"

Forces output to the Replication Server log file.

file_name

Forces output into the *file_name* log file. You can also set an alternate log file using the sysadmin dump_file command.

Examples

Example 1 Deletes the transaction of queue 103:1 with LQID 0:15:2 and dumps the transaction to the Replication Server log:

```
sysadmin sqm_zap_tran, 103, 1, 0, 15, 2
```

Example 2 Deletes the transaction of the inbound queue for SYDNEY_DS.pubs2 with LQID 0:15:2 and dumps the transaction to the Replication Server log:

```
sysadmin sqm_zap_tran, SYDNEY_DS, pubs2, 1, 0, 15,  
2, "log"
```

Example 3 Deletes the transaction of queue 103:1 with LQID 0:15:2 and dumps the begin and end commands of the transaction to the Replication Server log:

```
sysadmin sqm_zap_tran, 103,1, 0, 15, 2, L
```

Example 4 Deletes the transaction of queue 103:1 with lqid 0:15:2 and dumps the transaction to the Replication Server log. All the commands are truncated at 100 characters:

```
sysadmin sqm_zap_tran, 103,1, 0, 15, 2, L2
```

Example 5 Deletes the transaction of queue 103:1 with LQID 0:15:2 and dumps the transaction to the *SYDNEY_RS.log* file:

```
sysadmin sqm_zap_tran, 103,1, 0, 15, 2, L3,  
SYDNEY_RS.log
```

Example 6 Deletes the transaction of queue 103:1 with LQID 0:15:2 and dumps the transaction to the RSSD:

```
sysadmin sqm_zap_tran, 103, 1, 0, 15, 2, RSSD
```

Example 7 Deletes the transaction of queue 103:1 with LQID 0:15:2 and dumps the transaction to the client:

```
sysadmin sqm_zap_tran, 103, 1, 0, 15, 2, client
```

Usage

- The Replication Server must be in standalone mode to use sysadmin sqm_zap_tran.
- Use sysadmin dump_queue to locate the transaction you want to delete.

- `sysadmin sqm_zap_tran` marks a transaction in a stable queue as deleted. When Replication Server processes the queue, it ignores the marked transaction.
- You can restore a transaction using `sysadmin sqm_unzap_tran`. The `sysadmin sqm_unzap_tran` command removes the delete mark from the transaction.
- If you delete a transaction and then restart Replication Server in normal mode, the part of the queue holding the transaction may have been processed. If it was, you cannot restore the transaction with `sysadmin sqm_unzap_tran`.
- `sysadmin sqm_zap_tran` dumps the transaction marked for deletion into one of the following:
 - Replication Server log
 - Alternate log file
 - RSSD
 - Client issuing the command

To dump queues into the RSSD or client, the last argument of `sysadmin dump_queue` must be `RSSD` or `client`.

If the `RSSD` or `client` option is not specified, or if the `"log"` option is specified, output goes into the Replication Server log.

If an alternative log file for dumping queues is specified through the `sysadmin dump_file` command or through the *file_name* option, the output goes into the alternative dump file.

Permissions

`sysadmin sqm_zap_command` requires “sa” permission.

See also

`admin who`, `sysadmin dump_queue`, `sysadmin sqm_unzap_command`, `sysadmin sqm_unzap_tran`, `sysadmin sqm_zap_command`

sysadmin sqt_dump_queue

Description	Dumps the transaction cache for an inbound queue or a DSI queue.
Syntax	<pre>sysadmin sqt_dump_queue {, <i>q_number</i> <i>server</i>[, <i>database</i>]}, <i>q_type</i>, <i>reader</i> [, {open closed read}] [, <i>num_cmds</i>] [, {L0 L1 L2 L3}] [, {RSSD client "log" <i>file_name</i>}]</pre>
Parameters	<p><i>q_number</i> <i>server</i>[, <i>database</i>]</p> <p>Identifies the inbound queue or the DSI queue. Use either <i>q_number</i> or <i>server</i>[, <i>database</i>] to specify the queue number. You can use admin who, admin who, sqm, and admin who, sqt to identify the queue number.</p> <p><i>q_type</i></p> <p>The queue type of the stable queue. Values are 0 for outbound queues and 1 for inbound queues. Use admin who, admin who, sqm, and admin who, sqt to identify the queue type.</p> <p><i>reader</i></p> <p>Identifies the reader you want to dump the stable queue for. This parameter applies to features that require multiple readers, such as warm standby applications. You can get the reader number from admin sqm_readers or from admin who, sqt. If you are not using multiple readers, enter "0" for the reader.</p> <p>open</p> <p>Dumps only open transactions. If you use this option, insert a comma between <i>q_type</i> and the open flag.</p> <p>closed</p> <p>Dumps all the committed transactions found in the SQT cache.</p> <p>read</p> <p>Dumps all restored read transactions found in the SQT cache.</p> <p><i>num_cmds</i></p> <p>Specifies the number of commands to dump. Setting <i>num_cmds</i> to -1 dumps all of the commands in the SQT cache.</p> <p>L0</p> <p>Dumps the all of the SQT cache's content. This is the default behavior if L0, L1, L2, or L3 is not specified.</p> <p>L1</p> <p>Dumps only the begin and end commands of the transactions found in the SQT cache.</p>

L2

Dumps the begin and end commands of the SQT cache transactions together with a shortened version of all other commands in the transactions.

L3

Dumps everything in the cache. Except for SQL statements, all other commands are printed as comments. You can only use L3 when you use the *file_name* option or the sysadmin dump_file command to specify an alternate log file. You cannot use L3 with RSSD or client option.

RSSD

Forces the output to system tables in the RSSD.

client

Forces the output to the client issuing the command.

"log"

Forces the output to the Replication Server log file.

file_name

Forces the output into the alternate log file specified by *file_name*. The alternate log file can also be set using the sysadmin dump_file command. The location of this file is recorded in the Replication Server log.

Examples

Example 1 Dumps all restored transactions in queue 103:1 from the transaction cache:

```
sysadmin sqt_dump_queue, 103, 1, 0
```

Example 2 Dumps all restored transactions in the inbound queue for SYDNEY_DS.pubs2 from the transaction cache into the Replication Server log:

```
sysadmin sqt_dump_queue, SYDNEY_DS, pubs2, 1, 0
```

Example 3 Dumps all restored open transactions in queue 103:1 from the transaction cache into the Replication Server log:

```
sysadmin sqt_dump_queue, 103,1, 0, open
```

Example 4 Dumps all restored closed transactions in queue 103:1 from the transaction cache into the Replication Server log:

```
sysadmin sqt_dump_queue, 103,1, 0, closed
```

Example 5 Dumps all restored read transactions in queue 103:1 from the transaction cache into the Replication Server log:

```
sysadmin sqt_dump_queue, 103,1, 0, read
```

Example 6 Dumps the first 10 commands of restored transactions in queue 103:1 from the transaction cache into the Replication Server log:

```
sysadmin sqt_dump_queue, 103,1, 0, 10
```

Example 7 Dumps the begin and end commands of all restored transactions in queue 103:1 from the transaction cache into the Replication Server log:

```
sysadmin sqt_dump_queue, 103,1, 0, L1
```

Example 8 Dumps all restored transactions in queue 103:1 from the transaction cache into the Replication Server log. All the commands are truncated at 100 characters:

```
sysadmin sqt_dump_queue, 103,1, 0, L2
```

Example 9 Dumps all restored transactions in queue 103:1 from the transaction cache into the *SYDNEY_RS.log* file:

```
sysadmin sqt_dump_queue, 103,1, 0, L3, SYDNEY_RS.log
```

Example 10 Dumps all restored transactions in queue 103:1 from the transaction log into the RSSD:

```
sysadmin sqt_dump_queue, 103,1, 0, RSSD
```

Example 11 Dumps all restored transactions in queue 103:1 from the transaction log to the client:

```
sysadmin sqt_dump_queue, 103,1, 0, client
```

Usage

- Before using `sysadmin sqt_dump_queue`, execute `admin who, sqt` to make sure the transaction cache for the database exists.
- This command dumps all the statements of transactions in the transaction cache.
- `sysadmin sqt_dump_queue` dumps transaction statements into one of the following:
 - Replication Server log
 - Alternate log file
 - RSSD
 - Client issuing the command

To dump transactions into the RSSD or client, the last argument of `sysadmin sqt_dump_queue` must be RSSD or client.

If an alternative log file for dumping transactions is specified through the `sysadmin dump_file` command or through the *file_name* option, the output goes into the alternative dump file.

If the `RSSD` or `client` option is not specified, or the `log` option is specified, output goes into the Replication Server log.

- The output from the `sysadmin sqt_dump_queue` indicates the state of transactions in the transaction cache as open, closed, or read. Open transactions are transactions that do not have a commit yet. Closed transactions have a commit but have not been completely read out yet. Read transactions have been completely read out but have not been deleted yet.
- You can modify the cache size by setting the configuration parameter `sqt_max_cache_size`.

Permissions

`sysadmin sqt_dump_queue` requires “sa” permission.

See also

`admin who`, `sysadmin dump_file`

sysadmin system_version

Description	<p>Displays or sets the system-wide version number for the replication system, allowing you to use the software features in the corresponding release level.</p> <p>Starting with release 11.5, the site version for individual Replication Servers also enables new features. The system version number need not correspond to the current software version.</p>
Syntax	<code>sysadmin system_version [, <i>version</i>]</code>
Parameters	<p><i>version</i></p> <p>The system version number to use for the replication system.</p>
Examples	<p>Example 1 Executed at the ID Server, displays the current system version number:</p> <pre>sysadmin system_version</pre> <p>Example 2 Executed at the ID Server, changes the system version number to correspond to release 15.1. You can use this number if:</p> <ul style="list-style-type: none">• All Replication Servers are at release 15.1• You will not need to downgrade any Replication Server to an earlier release• You will not need to install any Replication Servers of an earlier release <pre>sysadmin system_version, 1510</pre>
Usage	<ul style="list-style-type: none">• To set the system version number, execute <code>sysadmin system_version</code> at the ID Server, and include a <i>version</i> parameter.<ul style="list-style-type: none">• The system version number you enter must be no higher than the lowest software version number—the release level of a Replication Server—of any Replication Server in the replication system.• You cannot set the system version number at any other Replication Server than the ID Server.• To display the current system version number, execute <code>sysadmin system_version</code> at the ID Server, without a <i>version</i> parameter. <p>If you execute this command at another Replication Server, the Replication Server tries to contact the ID Server to determine the current system version number. In rare cases, a Replication Server may be unable to contact the ID Server. For this reason, only the value at the ID Server is guaranteed to be correct.</p>

System version and site version

- Starting with Replication Server release 11.5, you can use certain new software features when the Replication Server's site version number has been set to the current software release—for example, 1510 for release 15.1. See `sysadmin site_version` for more information.

A minimum system version number of 1102 is also required.

- When you install a Replication Server of release 11.5 or higher as the ID Server for a new replication system, the system version number is set to 1102. This number allows you to install additional Replication Servers of release 11.0.2 or later into the system.
- For more information about installing or upgrading Replication Servers, refer to the Replication Server installation and configuration guides for your platform.

Mixed-version replication systems

If all of your Replication Servers are at release 11.0.2 or later, the highest required setting for the system version number is 1102. After setting the system version number to 1102, you may never need to set it again.

A 1102 system version number and site version number for individual Replication Servers allows a mixed-version replication system, in which Replication Servers of different site versions can work together. Each Replication Server can use its full set of available features.

In a mixed-version replication system, some features are only available to Replication Servers with higher site versions. For example, the site version of a primary Replication Server and one of its replicate Replication Server is 1510, while the site version of its other replicate Replication Server is 1260. When a table replication definition has a timestamp column, the replicate Replication Server with the lower site version can only subscribe to the timestamp as varbinary (8), while the replicate Replication Server with 1510 site version can subscribe to the timestamp column directly. See `sysadmin site_version` for more information.

For more information about features that were introduced in a particular Replication Server software release, see the *Replication Server New Features Guide* for that release.

System version and the ID Server

In Replication Servers other than the ID Server, when a command is executed that requires a certain minimum system version, the Replication Server contacts the ID Server to determine the current system version number before allowing use of the command.

System tables for version information

Version information is stored in the `rs_version` system table. The `rs_routes` system table also contains version information.

Permissions `sysadmin system_version` requires “sa” permission.

See also `admin version`, `sysadmin site_version`

validate publication

Description	Sets the status of a publication to VALID, allowing new subscriptions to be created for the publication.
Syntax	<code>validate publication <i>pub_name</i></code> with primary at <i>data_server.database</i>
Parameters	<i>pub_name</i> The name of the publication to be validated. with primary at <i>data_server.database</i> Specifies the location of the primary data. If the primary database is part of a warm standby application, <i>data_server.database</i> is the name of the logical data server and database.
Examples	Validates the publication pubs2_pub: <pre>validate publication pubs2_pub with primary at TOKYO_DS.pubs2</pre>
Usage	<ul style="list-style-type: none">• When all of the articles have been created for a publication, you must validate the publication using <code>validate publication</code> before a replicate site can subscribe to it. Validating a publication verifies that the publication contains at least one article and marks the publication ready for subscription.• Execute <code>validate publication</code> at the Replication Server where you created the publication using <code>create publication</code>.• To check the status of a publication, use <code>check publication</code>. This command displays the number of articles the publication contains and indicates if the publication is valid. <p>See the <i>Replication Server Administration Guide Volume 1 and Volume 2</i> for more information about subscription materialization.</p>
Permissions	<code>validate publication</code> requires “create object” permission.
See also	<code>check publication</code> , <code>check subscription</code> , <code>create publication</code> , <code>create subscription</code> , <code>define subscription</code> , <code>drop publication</code>

validate subscription

Description	For a subscription to a replication definition or a publication, sets the subscription status to VALID. This command is part of the bulk materialization process, or part of the process of refreshing a publication subscription.
Syntax	<pre>validate subscription <i>sub_name</i> for {<i>table_rep_def</i> <i>function_rep_def</i> publication <i>pub_name</i> with primary at <i>data_server.database</i>} with replicate at <i>data_server.database</i></pre>
Parameters	<p><i>sub_name</i> The name of the subscription to be validated.</p> <p>for <i>table_rep_def</i> Specifies the name of the table replication definition the subscription is for.</p> <p>for <i>function_rep_def</i> Specifies the name of the function replication definition the subscription is for.</p> <p>for publication <i>pub_name</i> Specifies the name of the publication the subscription is for.</p> <p>with primary at <i>data_server.database</i> Specifies the location of the primary data. If the primary database is part of a warm standby application, <i>data_server.database</i> is the name of the logical data server and database. Include this clause only for a subscription for a publication.</p> <p>with replicate at <i>data_server.database</i> Specifies the location of the replicate data. If the replicate database is part of a warm standby application, <i>data_server.database</i> is the name of the logical data server and database.</p>
Examples	<p>Example 1 Validates the subscription <code>titles_sub</code> for the table replication definition <code>titles_rep</code>, where the replicate database is <code>SYDNEY_DS.pubs2</code>:</p> <pre>validate subscription titles_sub for titles_rep with replicate at SYDNEY_DS.pubs2</pre> <p>Example 2 Validates the subscription <code>myproc_sub</code> for the function replication definition <code>myproc_rep</code>, where the replicate database is <code>SYDNEY_DS.pubs2</code>:</p> <pre>validate subscription myproc_sub for myproc_rep with replicate at SYDNEY_DS.pubs2</pre>

Example 3 Validates the subscription pubs2_sub for the publication pubs2_pub, where the primary database is TOKYO_DS.pubs2 and the replicate database is SYDNEY_DS.pubs2:

```
validate subscription pubs2_sub
  for publication pubs2_pub
  with primary at TOKYO_DS.pubs2
  with replicate at SYDNEY_DS.pubs2
```

Usage

- Use `validate subscription` to validate a subscription at the primary and replicate Replication Servers. The subscription can be to a table replication definition, function definition replication, or publication.
- This command completes the bulk materialization process. The first step is creating the subscription using `define subscription`. The second step is activating the subscription using `activate subscription`.
- If you have added any new articles to a publication with an existing subscription, you must refresh the publication subscription in order to create new subscriptions for these articles.

Use `define subscription` and `activate subscription` to create and activate the new article subscriptions in the publication subscription. Then manually load the subscription data for the new article subscriptions, and use `validate subscription` to validate the publication subscription.

- Execute `validate subscription` at the Replication Server where you created the subscription using `define subscription`.
- When you validate a publication subscription, all of its article subscriptions are validated at the same time.
- `validate subscription` changes the status of a subscription from `ACTIVE` to `VALID`. Subsequent updates at the primary data server are distributed through the primary Replication Server and applied at the replicate Replication Server.
- This command modifies RSSD tables at multiple sites. Use `check subscription` at both the primary and replicate Replication Servers to see the effects on each.

See the *Replication Server Administration Guide Volume 1 and Volume 2* for more information about subscription materialization.

Permissions

`validate subscription` requires “create object” permission at the site where the data is replicated and “primary subscribe” or “create object” permission at the site where the primary data is stored.

See also

activate subscription, check subscription, create article, create publication, create subscription, define subscription, drop subscription

wait for create standby

Description	A blocking command that allows a client session in the Replication Server to wait for the standby database creation process to complete.
Syntax	<code>wait for create standby for <i>logical_ds.logical_db</i></code>
Parameters	<i>logical_ds</i> The data server name for the logical connection. <i>logical_db</i> The database name for the logical connection.
Usage	<ul style="list-style-type: none">• After the standby database has been created, <code>wait for create standby</code> displays status information.• <code>wait for create standby</code> may be most helpful when used in scripts.
Permissions	<code>wait for create standby</code> requires “sa” permission.
See also	<code>abort switch</code> , <code>switch active</code> , <code>wait for switch</code>

wait for delay

Description	Specifies a time interval at which this command is blocked.
Syntax	wait for delay 'time_string'
Parameters	<p><i>time_string</i></p> <p>The period of time passed before executing. Uses the format hh:mm[:ss[.xxx]] [am pm].</p>
Examples	<p>This command instructs Replication Server to block a command for 1 hour and 30 minutes:</p> <pre>wait for delay '01:30'</pre>
Usage	<ul style="list-style-type: none">• Use wait for delay to instruct Replication Server to wait until the specified period of time has passed. A typical usage is in implementing subscriptions. Usually, wait for delay is issued in between two subscriptions.• The time specified can include hours, minutes, and seconds, up to a maximum of 24 hours.
Permissions	Any user can execute this command.
See also	wait for time

wait for switch

Description	A blocking command that allows a client session in the Replication Server to wait for the switch to the new active database to complete.
Syntax	<code>wait for switch</code> <code>for <i>logical_ds.logical_db</i></code>
Parameters	<i>logical_ds</i> The data server name for the logical connection. <i>logical_db</i> The database name for the logical connection.
Usage	<ul style="list-style-type: none">• After the switch active operation is complete, wait for switch displays status information.• wait for switch may be most helpful when used in scripts.
Permissions	wait for switch requires “sa” permission.
See also	abort switch, switch active, wait for create standby

wait for time

Description	Specifies a time of day at which to unblock this command.
Syntax	<code>wait for time 'time_string'</code>
Parameters	<i>time_string</i> The specific time to execute. Uses the format hh:mm[:ss[.xxx]] [am pm].
Examples	This command instructs Replication Server to wait until 5:30 p.m.: <pre>wait for time '05:30 pm'</pre>
Usage	<ul style="list-style-type: none">• Use <code>wait for time</code> to instruct Replication Server to wait until the specified time.• The time specified can include hours, minutes, and seconds, up to a maximum of 24 hours. If the current time is 6:00 pm, <code>wait for time '5:00 pm'</code> indicates 5:00 p.m. tomorrow.
Permissions	Any user can execute this command.
See also	<code>wait for delay</code>

Replication Server System Functions

This chapter contains reference pages for the Replication Server system functions.

See the *Replication Server Administration Guide Volume 2*, for information about customizing function strings for system functions.

The system functions described in this chapter may have **function-string-class scope** or **replication-definition scope**.

A function that has function-string class scope is defined once, for its class. It is then applied the same way in every database to which the class is assigned.

A function that has replication definition scope is defined once for each replication definition. It is then applied the same way for every operation (update, insert, and so on) that is replicated using the replication definition.

rs_batch_end

Description

rs_batch_end allows users to batch commands into non-Adaptive Server database servers. This function string stores the SQL statements needed to mark the end of a batch of commands.

Examples

Alters rs_batch_end function string so that the SQL output of the function-string class sqlserver_derived_class is END.

```
alter function string publishers.rs_batch_end
for sqlserver_derived_class
output language
'END'
```

Usage

- The rs_batch_end function has function-string class scope.
- This function string is used with rs_batch_start.

- `rs_batch_end` is sent to the replicate data server as the last command in the batch of commands. It is sent only if `use_batch_markers` is set to on.
- `rs_batch_end` precedes `rs_commit` in the order of data server processing.
- `rs_batch_start`, a batch of commands, and `rs_batch_end` may be repeated for a given transaction if more than one batch is required due to commands being flushed by limits such as `dsi_cmd_batch_size`.

See also

`rs_batch_start`

rs_batch_start

Description	<p>rs_batch_start allows users to batch commands into non-Adaptive Server database servers. This function string stores the SQL statements needed to mark the beginning of a batch of commands.</p>
Examples	<p>Alters rs_batch_start function string so that the SQL output of the function-string class sqlserver_derived_class is BEGIN.</p> <pre>alter function string publishers.rs_batch_start for sqlserver_derived_class output language 'BEGIN'</pre>
Usage	<ul style="list-style-type: none">• The rs_batch_start function has function-string-class scope.• Use of rs_batch_start is not necessary for Adaptive Server or any other data server that supports command batching by the function strings rs_begin and rs_commit.• rs_batch_start and the batch of commands following it is sent to the replicate data server only if use_batch_markers is set to on. rs_batch_start is sent after rs_begin.• Replication Server does not use the command separator following rs_batch_start. If the replicate database server requires a command separator following the marker for the beginning of a batch it is included as part of the string for rs_batch_start. This separator must be included as part of the function string whether it is the same or different from the dsi_cmd_separator parameter.• The rs_batch_start, a batch of commands, and rs_batch_end may be repeated if more than one batch is required due to commands being flushed by limits such as dsi_cmd_batch_size.
See also	<p>rs_batch_end</p>

rs_begin

Description

Begins a transaction in a data server.

Examples

Example 1 Creates an `rs_begin` function string for the `oth_sql_class` function-string class. The `rs_origin_xact_name` system variable has a null value if the transaction has no name. Placing “t_” in front of the system variable prevents data server syntax errors and allows the function string to support named and unnamed transactions.

```
alter function string rs_begin
  for oth_sql_class
  output language
  'begin transaction
    t_?rs_origin_xact_name!sys_raw?'
```

Example 2 Creates an `rs_begin` function string for a function-string class for a data server that does not support the begin transaction operation.

```
create function string rs_begin
  for oth_sql_class
  output language ''
```

Usage

- The `rs_begin` function has function-string-class scope.
- Replication Server creates an initial `rs_begin` function string for the system-provided function-string classes during installation.
- If you use a user-created base function-string class, you must create an `rs_begin` function string.
- Create or customize an `rs_begin` function string at the Replication Server that is the primary site for the class.
- Some data servers do not support an explicit begin transaction operation. Instead, they begin transactions implicitly whenever the previous transaction is committed or rolled back. For these data servers, the `rs_begin` function string can be an empty string (“”).
- The function string for this function usually uses the `rs_origin_xact_name` system variable. Its value is received from the RepAgent. The transaction name is assigned in Transact-SQL with `begin transaction`.

See also

`alter function string`, `create function string`, `rs_commit`, `rs_rollback`

rs_check_repl

Description	Checks to see if a table is marked for replication.
Examples	<p>Creates an <code>rs_check_repl</code> function string that executes the <code>rs_check_repl_stat</code> stored procedure.</p> <pre>create function string rs_check_repl for sqlserver_derived_class output language 'execute rs_check_repl_stat @rs_repl_name = ?rs_repl_name!param?'</pre>
Usage	<ul style="list-style-type: none"> • The <code>rs_check_repl</code> function has function-string-class scope. • Replication Server creates an initial <code>rs_check_repl</code> function string for the system-provided function-string classes during installation. • If you use a user-created base function-string class, you must create an <code>rs_check_repl</code> function string. • Create or customize an <code>rs_check_repl</code> function string at the Replication Server that is the primary site for the class.
See also	create function string, create replication definition

rs_commit

Description	Commits a transaction in a data server.
Examples	<p>This example illustrates the default <code>rs_commit</code> function string for the <code>rs_sqlserver_function_class</code> and <code>rs_default_function_class</code> classes. The function string executes a stored procedure named <code>rs_update_lastcommit</code> and then executes the Transact-SQL commit transaction command.</p> <pre>create function string rs_commit for sqlserver_derived_class output language 'execute rs_update_lastcommit @origin = ?rs_origin!sys?, @origin_qid = ?rs_origin_qid!sys?, @secondary_qid = ?rs_secondary_qid!sys?, @origin_time = ?rs_origin_commit_time!sys?; commit transaction'</pre> <p>Here is the text of the <code>rs_update_lastcommit</code> procedure for <code>rs_sqlserver_function_class</code>:</p>

```
/* Create a procedure to update the
** rs_lastcommit table. */
create procedure rs_update_lastcommit
    @origin int,
    @origin_qid binary(36),
    @secondary_qid binary(36),
    @origin_time datetime
as
begin
    update rs_lastcommit
        set origin_qid = @origin_qid,
            secondary_qid = @secondary_qid,
            origin_time = @origin_time,
            commit_time = getdate()
    where origin = @origin
    if (@@rowcount = 0)
    begin
        insert rs_lastcommit (origin,
            origin_qid, secondary_qid,
            origin_time, commit_time,
            pad1, pad2, pad3, pad4,
            pad5, pad6, pad7, pad8)
        values (@origin, @origin_qid,
            @secondary_qid,@origin_time,
            getdate(), 0x00, 0x00, 0x00,
            0x00, 0x00, 0x00, 0x00, 0x00)
    end
end
```

Usage

- The rs_commit function has function-string-class scope.
- Replication Server creates an initial rs_commit function string for the system-provided function-string classes during installation.
- If you use a user-created base function-string class, you must create an rs_commit function string.
- Create or customize an rs_commit function string at the Replication Server that is the primary site for the class.
- Update the rs_lastcommit system table in the rs_commit function string. Updating this table within the transaction maintains data integrity.

Warning! If the rs_lastcommit system table is not updated properly for each transaction committed, after a restart Replication Server may apply transactions more than once or skip transactions.

See also

alter function string, create function string, rs_begin, rs_get_lastcommit,
rs_rollback

rs_datarow_for_writetext

Description Provides an image of the data row associated with a text, unitext, or image column updated with the Transact-SQL writetext command, with the Client-Library function ct_send_data, or with the DB-Library™ functions dbwritetext and dbmoretext.

Examples Executes a stored procedure named capture_datarow, setting the value of @au_id to the value of the au_id column and the value of @copy to the status value for the copy column.

```
create function string
  blurbs_rep.rs_datarow_for_writetext
  for sqlserver_derived_class
  output rpc
  'execute capture_datarow
    @au_id = ?au_id!new?,
    @copy = ?copy!text_status?'
```

Usage

- Replication Server executes rs_datarow_for_writetext before updated text, unitext, or image data is sent to the replicate data server. rs_datarow_for_writetext provides the values of primary key columns and searchable columns from the row so that subscriptions can be processed and data can be transferred to the replicate database.
- rs_datarow_for_writetext accesses the values of all columns in the row except for text, unitext, and image columns. To retrieve information about text, unitext, or image columns, include the *text_status* modifier in the function string. The values returned by *text_status* are described in Table 4-1.
- The rs_datarow_for_writetext function has replication definition scope.
- Replication Server generates an rs_datarow_for_writetext function string for rs_sqlserver_function_class and rs_default_function_class when you create a replication definition.
- If you use a user-created base function-string class, you must create a rs_datarow_for_writetext function string for each replication definition that includes text, unitext, and image columns.
- Create or customize a rs_datarow_for_writetext function string at the Replication Server where you created the replication definition.
- The default generated function string for rs_sqlserver_function_class and rs_default_function_class does not execute commands in the replicate database, since the row image contains no modified data.

- You can create a new `rs_datarow_for_writetext` function string to collect the values of the primary key to pass to a gateway. The *old* and *new* modifiers both provide access to a column's value.
- The *text_status* modifier retrieves the status of the text, unitext, or image column. Table 4-1 lists the possible values for the *text_status* modifier.

Table 4-1: *text_status* values for text, unitext, and image data

Value	Description
0x0001	The column has a null text pointer. There are no modifications to text, unitext, or image columns.
0x0002	Modifications were made at the primary database, which caused a text pointer allocation. Replication Server executes the <code>rs_textptr_init</code> function to allocate a text pointer.
0x0004	The current data value follows. Replication Server executes the <code>rs_writetext</code> function to modify the text, unitext, or image data at the replicate database.
0x0008	The text, unitext, or image column is not replicated. No commands are required in the replicate database because the data did not change value and the text, unitext, or image column has a <code>replicate_if_changed</code> status.
0x0010	The text, unitext, or image column contains a null value after an operation at the primary database. For example, after a text pointer has been allocated, there may be data values in a text or image column and an application at the primary database sets them to null. Replication Server truncates the text, unitext, or image column in the replicate database by setting the values to null if the <i>text_status</i> is not 0x0008.

See also

`rs_get_textptr`, `rs_textptr_init`, `rs_writetext`

rs_delete

Description	Deletes a row in a replicated table.
Examples	<p>Changes the <code>rs_delete</code> function string for the <code>titles_rep</code> replication definition so that it executes a stored procedure named <code>del_title</code>.</p> <pre>alter function string titles_rep.rs_delete for sqlserver_derived_class output rpc 'execute del_title @title=?title!old?'</pre>
Usage	<ul style="list-style-type: none">• Replication Server executes <code>rs_delete</code> to delete a single row in a table. The row is identified by the primary key columns defined in a replication definition for the table.• <code>rs_delete</code> has replication definition scope.• Replication Server generates an <code>rs_delete</code> function string for the system-provided function-string classes when you create a replication definition.• If you use a user-created base function-string class, you must create an <code>rs_delete</code> function string for each replication definition.• Create or customize an <code>rs_delete</code> function string where you created the replication definition.• For the system-provided classes <code>rs_sqlserver_function_class</code> and <code>rs_default_function_class</code>, the <code>rs_delete</code> generated function string uses the Transact-SQL delete command syntax. The row to be deleted is identified with a where clause that specifies the pre-delete values, or before image, of the primary key columns.
See also	<code>create function string</code> , <code>create replication definition</code> , <code>rs_insert</code> , <code>rs_update</code>

rs_dsi_check_thread_lock

Description	Determines whether or not the DSI executor thread is holding a lock that blocks a replicate database process. A return value greater than 0 indicates that the thread is holding resources required by another database process, and that the thread should roll back and retry the transaction.
Examples	Creates the <code>rs_dsi_check_thread_lock</code> function string that checks whether or not the current DSI executor thread is blocking another replicate database process.

```
create function string rs_dsi_check_thread_lock
for sqlserver_derived_class
output language
'select count(*) as seq from master..sysprocesses
where blocked = @@spid and suid = suser_id()'
```

Usage

- Replication Server uses the `rs_dsi_check_thread_lock` function to check whether or not the current DSI executor thread is blocking another replicate database process. It is executed only when more than one DSI thread is defined for a connection with `dsi_commit_control` set on, and a DSI executor thread is ready to commit, but cannot because it is not “next” to commit, and the amount of time specified for `dsi_commit_check_locks_intrvl` has elapsed.
- The function string `rs_dsi_check_thread_lock` query is expected to return a single integer value, column name of `seq`. A return value greater than 0 indicates that the thread is holding resources required by another database process, and that the thread should roll back and retry the transaction.
- `rs_dsi_check_thread_lock` has function string class scope.
- Replication Server creates an initial `rs_dsi_check_thread_lock` function string for the system-provided function string classes during installation.
- You must create a function string for the `rs_dsi_check_thread_lock` function string, if you are using a custom base function string and you want to use the parallel DSI with `dsi_commit_control` set to on. Otherwise, you do not need to create a function string for this function.
- Create or customize an `rs_dsi_check_thread_lock` function string at the Replication Server deployed at the primary site for the class.

See also

`create function string`, `alter function string`

rs_dumpdb

Description

Initiates a coordinated database dump.

Examples

Example 1 Creates an rs_dumpdb function string that dumps the database to a specified dump device and executes a procedure to update the rs_lastcommit system table. This function string works best when there is only one replicate database or when all databases using the function-string class have the same dump device names.

```
create function string rs_dumpdb
for sqlserver_derived_class
output language
'dump database ?rs_destination_db!sys_raw?
to pubs2_dmpdb;
execute rs_update_lastcommit
?rs_origin!sys?,
?rs_origin_qid!sys?,
?rs_secondary_qid!sys?,
?rs_origin_commit_time!sys?'
```

Example 2 This example is better suited to multiple sites and production environments than is the first example. dumpdb_proc manages the backup devices at the replicate sites. The procedure should select a backup device to use, then mark it “used” so that a subsequent dump does not overwrite the previous backup.

```
alter function string rs_dumpdb
for sqlserver_derived_class
output rpc
'execute dumpdb_proc
?rs_dump_dbname!sys?,
?rs_dump_label!sys?,
?rs_dump_timestamp!sys?,
?rs_destination_db!sys?,
?rs_origin!sys?,
?rs_origin_qid!sys?,
?rs_secondary_qid!sys?,
?rs_origin_commit_time!sys?'
```

The procedure uses *rs_origin*, *rs_origin_qid*, and *rs_secondary_qid* to execute *rs_update_lastcommit*. If the server fails after the dump is complete but before the *rs_lastcommit* system table is updated, the backup is restarted when Replication Server resumes.

Note There is no guarantee that the dump and the *rs_update_lastcommit* procedure will execute atomically, because Adaptive Server does not allow the dump command to be included in a transaction with other commands. If the *rs_lastcommit* system table is not updated successfully, an additional dump may be performed.

In the following sample text of the *dumpdb_proc* stored procedure, the dump devices are hard-coded. In a production environment, it is better to manage them in a table.

```
create proc dumpdb_proc
    @dump_dbname varchar(30),
    @dump_label varchar(30),
    @dump_timestamp varbinary(16),
    @destination_dbname varchar(30),
    @origin int,
    @origin_qid binary(36),
    @secondary_qid binary(36),
    @origin_time datetime
as
    print 'Received a dump database command from
Replication Server:'
    declare @message varchar(255)
    select @message = 'dump database ' + @dump_dbname
        + '. Label= ' + @dump_label
        + '. Dest.db = ' + @destination_dbname
        + ' '
    print @message
    if @destination_dbname = 'pubs2'
    begin
        print 'issuing ''dump database pubs2.''
        dump database pubs2 to pubs2_dmplog
        update dmp_count set d_count = d_count + 1
        exec pubs2.dbo.rs_update_lastcommit
            @origin, @origin_qid, @secondary_qid,
            @origin_time
    end
    else if @destination_dbname = 'pubs3'
    begin
        print 'issuing ''dump database pubs3.''
```

```
dump database pubs3 to pubs3_dmplog
update dmp_count set d_count = d_count + 1
exec pubs3.dbo.rs_update_lastcommit
    @origin, @origin_qid, @secondary_qid,
    @origin_time
end
```

Usage

- Replication Server coordinates database dumps by placing rs_dumpdb function calls in the same place in the stream of transactions distributed to each replicate Replication Server.
- rs_dumpdb has function-string class scope.

Note Replication Server does not initialize or generate rs_dumpdb function strings for the system-provided function-string classes. You must create a function string before using a coordinated dump with Adaptive Server.

- Create an rs_dumpdb function string at the Replication Server that is the primary site for the class.
- To account for different dump devices at multiple replicate sites, create a stored procedure in each replicate database that performs a database dump. Then write the rs_dumpdb function string to execute the stored procedure.
- The rs_lastcommit system table should be updated when the rs_dumpdb function string executes so that a restarted Replication Server does not perform duplicate dumps. See rs_commit for information about rs_lastcommit.
- Table 4-2 lists the system variables that can be used in rs_dumpdb function strings.

Table 4-2: System variables for rs_dumpdb function strings

Variable name	Datatype	Description
rs_dump_dbname	varchar(30)	The name of the database where the dump originated.
rs_dump_label	varchar(30)	Label information for the dump. For Adaptive Server, this variable holds a datetime value that is the time the dump originated.
rs_dump_timestamp	varbinary(16)	A timestamp taken when the dump started.

See also

create function string class, rs_commit, rs_dumptran, rs_get_lastcommit

rs_dumptran

Description

Initiates a coordinated transaction dump.

Examples

Example 1 Creates an `rs_dumptran` function string to execute a stored procedure named `dumptran_proc`. The stored procedure manages the dump devices and then executes the `rs_update_lastcommit` stored procedure, passing it the `rs_origin`, `rs_origin_qid`, `-rs_secondary_qid`, and `rs_origin_commit_time` parameters.

```
create function string rs_dumptran
for sqlserver_derived_class
output rpc
'execute dumptran_proc
    ?rs_dump_dbname!sys?,
    ?rs_dump_label!sys?,
    ?rs_dump_timestamp!sys?,
    ?rs_dump_status!sys?,
    ?rs_destination_db!sys?,
    ?rs_origin!sys?,
    ?rs_origin_qid!sys?,
    ?rs_secondary_qid!sys?
    ?rs_origin_commit_time!sys?'
```

If the server crashes after the dump is complete but before the `rs_lastcommit` system table is updated, Replication Server restarts the backup.

Note There is no guarantee that the dump and the `rs_update_lastcommit` procedure will be executed atomically, because Adaptive Server does not allow the dump command to be included in a transaction with other commands. If the `rs_lastcommit` system table is not updated successfully, an additional dump may be performed.

In the following sample text of the `dumptran_proc` stored procedure, the dump devices are hard-coded. In a production environment, it is better to manage them in a table:

```
create proc dumptran_proc
    @dump_dbname varchar(30),
    @dump_label varchar(30),
    @dump_timestamp varbinary(16),
    @dump_status int,
    @destination_dbname varchar(30),
    @origin int,
    @origin_qid binary(36),
    @secondary_qid binary(36),
```

```
@origin_time datetime
as
print 'Received a dump transaction command from Replication Server:'
declare @message varchar(255)
if @dump_status = 0
begin
    select @message = 'dump transaction ' + @dump_dbname + '. Label= '''
        + @dump_label + ''' + '. Dest.db = ''' + @destination_dbname + '''
end
else if @dump_status = 1
begin
    select @message = 'dump transaction standby '
        + @dump_dbname + '. Label= ''' +
        @dump_label + ''' + '. Dest.db = ''' + @destination_dbname + '''
end
print @message
if @destination_dbname = 'pubs2'
begin
    print 'issuing ' 'dump transaction pubs2.' '
    if @dump_status = 0
    begin
        dump transaction pubs2 to pubs2_dmplog
    end
    else if @dump_status = 1
    begin
        dump transaction pubs2 to pubs2_dmplog with standby_access
    end
    update dmp_count set d_count = d_count + 1
    exec pubs2.dbo.rs_update_lastcommit
        @origin, @origin_qid, @secondary_qid,
        @origin_time
end
else if @destination_dbname = 'pubs3'
begin
    print 'issuing ' 'dump transaction pubs3.' '
    if @dump_status = 0
    begin
        dump transaction pubs3 to pubs3_dmplog
    end
    else if @dump_status = 1
    begin
        dump transaction pubs3 to pubs3_dmplog with standby_access
    end
    update dmp_count set d_count = d_count + 1
    exec pubs3.dbo.rs_update_lastcommit
        @origin, @origin_qid, @secondary_qid,
```

```
@origin_time  
end
```

Example 2 Alters the `rs_dumptran` function string that you created in the first example to execute as a remote procedure call.

```
alter function string rs_dumptran  
for sqlserver_derived_class  
output rpc  
'execute dumptran_proc  
  ?rs_dump_dbname!sys?,  
  ?rs_dump_label!sys?,  
  ?rs_dump_timestamp!sys?,  
  ?rs_dump_status!sys?,  
  ?rs_destination_db!sys?,  
  ?rs_origin!sys?,  
  ?rs_origin_qid!sys?,  
  ?rs_secondary_qid!sys?,  
  ?rs_origin_commit_time!sys?!'
```

Usage

- Replication Server coordinates transaction dumps by inserting an `rs_dumptran` function call at the same place in the stream of transactions it distributes to all replicate Replication Servers.
- `rs_dumptran` has function-string-class scope.

Note Replication Server does not initialize or generate `rs_dumptran` function strings for the system-provided function-string classes. You must create a function string before using a coordinated dump with Adaptive Server.

- Create an `rs_dumptran` function string at the Replication Server that is the primary site for the class.
- The `rs_lastcommit` system table should be updated when the `rs_dumptran` function string executes so that a restarted Replication Server does not perform duplicate dumps. See `rs_commit` for information about `rs_lastcommit`.
- To account for different dump devices at multiple replicate sites, create a stored procedure in each replicate database that performs a transaction dump, then write the `rs_dumptran` function string to execute the stored procedure.
- Table 4-3 lists the system variables used in `rs_dumptran` function strings.

Table 4-3: System variables for rs_dumptran function strings

Variable name	Datatype	Description
<i>rs_destination_db</i>	varchar(30)	Name of the database where a transaction was sent.
<i>rs_dump_dbname</i>	varchar(30)	The name of the database where the dump originated.
<i>rs_dump_label</i>	varchar(30)	Label information for the dump. For Adaptive Server, this variable contains a datetime value for the time the dump began.
<i>rs_dump_status</i>	int(4)	Dump status indicator: <ul style="list-style-type: none">• 0 – denotes that the dump transaction command does not contain the parameter with standby_access• 1 – denotes that the dump transaction command contains the parameter with standby_access
<i>rs_dump_timestamp</i>	varbinary(16)	An Adaptive Server database timestamp taken when the dump was started at the origin. The variable is used for informational purposes only.
<i>rs_origin</i>	int(4)	ID of the originating database for a transaction.
<i>rs_origin_commit_time</i>	datetime	The time that a transaction was committed at the origin. Note If you execute <code>select getdate()</code> while ASE is still processing user database recovery, the returned value of <code>select getdate()</code> may be different from the value of <code>rs_origin_begin_time</code> .
<i>rs_origin_qid</i>	varbinary(36)	Origin queue ID of the first command in a transaction.
<i>rs_secondary_qid</i>	varbinary(36)	Queue ID of a transaction in a subscription materialization or dematerialization queue.

See also `create function string`, `rs_commit`, `rs_dumpdb`, `rs_get_lastcommit`

rs_get_charset

Description	Returns the character set used by a data server. This function allows Replication Server to print a warning message if the character set is not what is expected.
Examples	<p>Creates an <code>rs_get_charset</code> function string with output language that calls the <code>sp_serverinfo</code> system procedure and returns the data server's character set.</p> <pre>create function string rs_get_charset for rs_sqlserver2_function_class output language 'sp_serverinfo server_csname'</pre>
Usage	<ul style="list-style-type: none">• <code>rs_get_charset</code> obtains the name of the character set used by a data server. The Replication Server executes this function each time it connects to the data server.• <code>rs_get_charset</code> has function-string class scope.• Replication Server creates an initial <code>rs_get_charset</code> function string for the system-provided function-string classes during installation.• If you use a user-created base function-string class, you must create an <code>rs_get_charset</code> function string.• Create or customize an <code>rs_get_charset</code> function string at the Replication Server that is the primary site for the class.• The default <code>rs_get_charset</code> function string for the <code>rs_sqlserver_function_class</code> and <code>rs_default_function_class</code> classes calls the Adaptive Server stored procedure <code>sp_serverinfo</code> with the argument <code>server_csname</code>.• The data server should return a string with the name of a valid Sybase-supported character set. Valid Sybase character sets are defined in the Sybase release directory in <code>charsets/charset_name/charset.loc</code>, where each <code>charset_name</code> represents the name of a supported character set. For example, the file <code>charsets/iso_1/charset.loc</code> defines the <code>iso_1</code> character set.
See also	<code>create function string</code> , <code>rs_get_sortorder</code>

rs_get_errormode

Description	Returns native error configuration, which determines whether or not the native error is returned directly from the replicate server.
Examples	<p>Example 1 Creates an <code>rs_get_errormode</code> function string for the <code>oth_sql_class</code> function-string class that returns a native error.</p> <pre>create function string rs_get_errormode for oth_sql_class output language 'select yes'</pre> <p>Example 2 Creates an <code>rs_get_errormode</code> function string for the <code>oth_sql_class</code> function-string class that does not returns a native error.</p> <pre>create function string rs_get_errormode for oth_sql_class output language 'select no'</pre>
Usage	<ul style="list-style-type: none">• The <code>rs_get_errormode</code> function has function-string-class scope.• Replication Server creates an initial <code>rs_get_errormode</code> function string for the system-provided function-string classes during installation.• If you use a user-created base function-string class, you must create an <code>rs_get_errormode</code> function string.• Create or customize an <code>rs_get_errormode</code> function string at the Replication Server that is the primary site for the class.• Expected result for the function <code>rs_get_errormode</code> is either <code>yes</code> or <code>no</code>.

rs_get_lastcommit

- Description** Returns rows from the `rs_lastcommit` system table.
- Examples** Creates an `rs_get_lastcommit` function string that executes a stored procedure named `rs_get_lastcommit`. The text of the stored procedure is:
- ```
create procedure rs_get_lastcommit
as
select origin, origin_qid, secondary_qid
from rs_lastcommit

create function string rs_get_lastcommit
for sqlserver_derived_class
output language
'execute rs_get_lastcommit'
```
- Usage**
- Replication Server executes `rs_get_lastcommit` when it starts up a DSI process for a database. The function returns all of the rows in the `rs_lastcommit` system table. Replication Server uses this information to find the last transaction committed from each primary data source.
  - The `rs_lastcommit` system table is updated each time Replication Server commits a transaction in the database.
  - `rs_get_lastcommit` has function-string-class scope.
  - Replication Server creates an initial `rs_get_lastcommit` function string for the system-provided function-string classes during installation.
  - If you use a user-created base function-string class, you must create an `rs_get_lastcommit` function string.
  - Create or customize an `rs_get_lastcommit` function string at the Replication Server that is the primary site for the class.
  - The default `rs_get_lastcommit` function string for the `rs_sqlserver_function_class` and `rs_default_function_class` classes updates the `rs_lastcommit` table by executing a stored procedure named `rs_update_lastcommit` in the `rs_commit` function string.
  - `rs_get_lastcommit` must return columns in the correct order for each primary database whose data is replicated in the database. See Table 4-4.

**Table 4-4: Columns returned by `rs_get_lastcommit`**

| Column name | Datatype | Description                                               |
|-------------|----------|-----------------------------------------------------------|
| origin      | int      | The ID number for the primary database the row represents |

| Column name   | Datatype   | Description                                                                                                                                                                       |
|---------------|------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| origin_qid    | binary(36) | Identifies the last committed transaction in the stable queue for the origin database                                                                                             |
| secondary_qid | binary(36) | If a subscription materialization queue exists for the origin database, this column contains the last transaction in that queue that has been committed in the replicate database |

See also                      create function string, rs\_commit

## rs\_get\_sortorder

|             |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
|-------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Description | Obtains the sort order used by a data server. This function returns a warning message if the sort order does not match that of the Replication Server, and if the sort order is not what is expected.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| Examples    | <p>Creates an <code>rs_get_sortorder</code> function string with output language that calls the <code>sp_serverinfo</code> system procedure and returns the data server's sort order.</p> <pre>create function string rs_get_sortorder   for rs_sqlserver2_function_class   output language   'sp_serverinfo server_soname'</pre>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| Usage       | <ul style="list-style-type: none"> <li>• The <code>rs_get_sortorder</code> function obtains the name of the sort order used by a data server. Replication Server executes this function each time it connects to the data server. If the sort order does not match that of the Replication Server, a warning message is written into the Replication Server error log. If the sort orders match, no warning message is written.</li> <li>• The <code>rs_get_sortorder</code> function has function-string-class scope.</li> <li>• Replication Server creates an initial <code>rs_get_sortorder</code> function string for the system-provided function-string classes during installation.</li> <li>• If you use a user-created base function-string class, you must create an <code>rs_get_sortorder</code> function string.</li> <li>• If you need to create or customize an <code>rs_get_sortorder</code> function string, do so at the Replication Server that is the primary site for the class.</li> <li>• The default <code>rs_get_sortorder</code> function string for the <code>rs_sqlserver_function_class</code> and <code>rs_default_function_class</code> classes calls the Adaptive Server stored procedure <code>sp_serverinfo</code> with the argument <code>server_soname</code>.</li> <li>• An <code>rs_get_sortorder</code> function string should return a string with the name of a valid Sybase-supported sort order. Valid Sybase sort orders for a given character set are defined in the Sybase release directory in <i>charsets/charset_name/sortorder.srt</i>, where <i>charset_name</i> represents the name of a supported character set and <i>sortorder</i> represents the name of a supported sort order for the character set. For example, the file <i>charsets/iso_1/nocase.srt</i> defines the “nocase” sort order for the <code>iso_1</code> character set.</li> </ul> |
| See also    | <code>create function string</code> , <code>rs_get_charset</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |

## rs\_get\_textptr

### Description

Retrieves the description for a text, unitext, or image column.

### Examples

Creates an `rs_get_textptr` function string for the `repcopy` column in the `blurbs` table. The function string name, `copy`, is the name of the text, unitext, or image column in the replication definition.

```
create function string
 blurbs_rep.rs_get_textptr;copy
for sqlserver2_function_class
output language
'select repcopy from blurbs
where au_id = ?au_id!new?'
```

### Usage

- Replication Server calls `rs_get_textptr` to retrieve a text, unitext, or image column description before it sends data with the Client-Library function `ct_send_data`.
- `rs_get_textptr` has replication definition scope.
- When you create a replication definition, Replication Server generates an `rs_get_textptr` function string for the `rs_sqlserver_function_class` and `rs_default_function_class` classes for each replicated text, unitext, or image column in the replication definition.
- If you use a user-created base function-string class, you must create an `rs_get_textptr` function string for each replicated text, unitext, or image column included in the replication definition.
- Create or customize an `rs_get_textptr` function string at the Replication Server where you created the replication definition.
- `rs_get_textptr` must return a text or unitext column description for a text, unitext, or image column in a specified row. The text or unitext column description must conform to Open Server requirements for returning an “I/O descriptor structure.” For information about this structure, refer to the *Open Server Server-Library/C Reference Manual*.

### See also

`rs_datarow_for_writetext`, `rs_textptr_init`, `rs_writetext`

## rs\_get\_thread\_seq

|             |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
|-------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Description | Returns the sequence number for the specified entry in the rs_threads system table.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| Syntax      | <code>rs_get_thread_seq @rs_id</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
| Parameters  | <i>rs_id</i><br>a number of int datatype. It represents the ID of the entry to be checked and matches the value of the id column in the rs_threads system table.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
| Examples    | Creates an rs_get_thread_seq function string that executes a select statement in the rs_threads table.<br><pre>create function string rs_get_thread_seq for sqlserver_derived_class output language 'select seq from rs_threads where id = ?rs_id!param?'</pre>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| Usage       | <ul style="list-style-type: none"><li>• Replication Server executes rs_get_thread_seq to check the completion status of preceding transactions. It is executed only when more than one DSI thread is defined for a connection. The function returns a single row with a single column, seq, which contains the sequence number for the specified ID.</li><li>• The thread invoking this function is blocked until the transaction that last modified the specified entry completes its transaction.</li><li>• rs_get_thread_seq has function-string-class scope.</li><li>• Replication Server creates an initial rs_get_thread_seq function string for the system-provided function-string classes during installation.</li><li>• If you use a user-created base function-string class and you use the parallel DSI feature, you must create a function string for the rs_get_thread_seq function. If you do not use parallel DSI, you do not need to create a function string for this function.</li><li>• Create or customize an rs_get_thread_seq function string at the Replication Server that is the primary site for the class.</li></ul> |
| See also    | configure connection, rs_initialize_threads, rs_set_isolation_level, rs_update_threads                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |

## rs\_get\_thread\_seq\_noholdlock

|             |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
|-------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Description | Returns the sequence number for the specified entry in the rs_threads system table, using the noholdlock option.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
| Syntax      | <code>rs_get_thread_seq_noholdlock @rs_id</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
| Parameters  | <p><i>rs_id</i></p> <p>a number of int datatype. It represents the ID of the entry to be checked and matches the value of the id column in the rs_threads system table.</p>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| Examples    | <p>Creates an rs_get_thread_seq_noholdlock function string that executes a select statement on the rs_threads table.</p> <pre>create function string   rs_get_thread_seq_noholdlock for sqlserver_derived_class output language 'select seq from rs_threads noholdlock   where id = ?rs_id!param?'</pre>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| Usage       | <ul style="list-style-type: none"><li>• rs_get_thread_seq_noholdlock is equivalent to rs_get_thread_seq, except that it is used when dsi_isolation_level is 3. It is executed only when more than one DSI thread is defined for a connection. The row select is done with the noholdlock option. The function returns a single row with a single column, seq, which contains the current sequence number for the specified ID.</li><li>• The rs_get_thread_seq_noholdlock function has function-string class scope.</li><li>• Replication Server creates an initial rs_get_thread_seq_noholdlock function string for the system-provided function-string classes during installation.</li><li>• If you use a user-created base function-string class and you use the parallel DSI feature with transaction isolation level 3, create a function string for rs_get_thread_seq_noholdlock.</li><li>• Create or customize an rs_get_thread_seq_noholdlock function string at the Replication Server that is the primary site for the class.</li></ul> |
| See also    | <code>alter connection, rs_get_thread_seq, rs_initialize_threads, rs_set_isolation_level, rs_update_threads</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |

## rs\_initialize\_threads

|             |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
|-------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Description | Sets the sequence of each entry in the <code>rs_threads</code> system table to 0.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| Syntax      | <code>rs_initialize_threads @rs_id</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| Parameters  | <p><i>@rs_id</i></p> <p>a number from 1 through <i>dsi_num_threads</i>, representing the ID of the entry Replication Server will set to 0.</p>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
| Examples    | <p>Creates an <code>rs_initialize_threads</code> function string that executes a stored procedure named <code>rs_initialize_threads</code>. The text of the stored procedure is:</p> <pre> create procedure rs_initialize_threads     @rs_id int as     delete from rs_threads where id = @rs_id     insert into rs_threads values         (@rs_id, 0, "", "", "", "")  create function string rs_initialize_threads     for sqlserver_derived_class output language 'execute rs_initialize_threads     @rs_id = ?rs_id!param?'</pre>                                                                                                                                                                                                                                                                                                                                                                                            |
| Usage       | <ul style="list-style-type: none"> <li>• <code>rs_initialize_threads</code> Replication Server executes function when a connection is initialized. It is executed only when more than one DSI thread is defined for the connection. It sets the sequence number of each entry in the <code>rs_threads</code> system table to 0.</li> <li>• <code>rs_initialize_threads</code> has function-string-class scope.</li> <li>• Replication Server creates an initial <code>rs_initialize_threads</code> function string for the system-provided function-string classes during installation.</li> <li>• If you use a user-created base function-string class and you use the parallel DSI feature, create a function string for <code>rs_initialize_threads</code>.</li> <li>• Create or customize an <code>rs_initialize_threads</code> function string at the Replication Server that is the primary site for the class.</li> </ul> |
| See also    | <code>create connection</code> , <code>rs_get_thread_seq</code> , <code>rs_get_thread_seq_noholdlock</code> , <code>rs_set_isolation_level</code> , <code>rs_update_threads</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |

## rs\_insert

### Description

Inserts a single row into a table in a replicate database.

### Examples

Replaces the `rs_insert` function string for the publishers table.

```
alter function string publishers.rs_insert
for sqlserver_derived_class
output language
'insert into publishers (pub_id, pub_name, city,
state)
values (?pub_id!new?, ?pub_name!new?,
?city!new?, ?state!new?)'
```

### Usage

- `rs_insert` has replication definition scope.
- Replication Server generates an `rs_insert` function string for the system-provided function-string classes when you create a replication definition.
- If you use a user-created base function-string class, create an `rs_insert` function string for each replication definition.
- Create or customize an `rs_insert` function string at the Replication Server where you created the replication definition.
- The default generated function string for `rs_insert`, for the `rs_sqlserver_function_class` and `rs_default_function_class` classes for each replication definition, uses the Transact-SQL insert command syntax.
- Replication Server cannot send text, untext, or image data to a replicate database in `rs_insert`, but it can report the status of text, untext, or image data with the `text_status` modifier. For a description of the `text_status` modifier, see `rs_datarow_for_writetext`. text, untext, or image data is sent to the replicate database with `rs_get_textptr`, `rs_textptr_init`, and `rs_writetext`.

### See also

`create function string`, `create replication definition`, `rs_datarow_for_writetext`, `rs_delete`, `rs_get_textptr`, `rs_select`, `rs_select_with_lock`, `rs_textptr_init`, `rs_update`,

## rs\_marker

|             |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
|-------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Description | Passes its parameter to Replication Server as an independent command.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| Syntax      | <code>rs_marker @rs_api</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| Parameters  | <i>rs_api</i><br>a varchar(255) character string that contains data used for subscription materialization.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| Examples    | <pre>create function string rs_marker for sqlserver_derived_class output language 'execute rs_marker @rs_api = ?rs_api!param?'</pre>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
| Usage       | <ul style="list-style-type: none"><li>• <code>rs_marker</code> allows Replication Server to insert data into the transaction log so that it can be retrieved by the RepAgent thread.</li><li>• The <code>rs_marker</code> function has function-string-class scope.</li><li>• Replication Server creates an initial <code>rs_marker</code> function string for the system-provided function-string classes during installation.</li><li>• If you use a user-created base function-string class, create a function string for the <code>rs_marker</code> function.</li><li>• Create or customize an <code>rs_marker</code> function string at the Replication Server that is the primary site for the class.</li><li>• Replication Server uses <code>rs_marker</code> during subscription materialization to pass the activate subscription and validate subscription commands to the primary Replication Server via the primary database log.</li><li>• The RepAgent for the primary database must recognize an <code>rs_marker</code> function execution and pass the <code>@rs_api</code> parameter to the primary Replication Server as a command.</li><li>• For Adaptive Server databases, an Adaptive Server replicated stored procedure named <code>rs_marker</code> is created when the database is set up for Replication Server. This stored procedure is marked “replicated” using the <code>sp_setreproc</code> system procedure.</li></ul> |

- When the Adaptive Server RepAgent encounters an rs\_marker execution in the transaction log, it sends the @rs\_api parameter to the primary Replication Server as a command.

---

**Note** Do not change the rs\_marker function string or invoke the rs\_marker stored procedure except when you create bulk subscriptions as described in the *Replication Server Administration Guide Volume 1*.

---

See also

activate subscription, create subscription, sp\_setrepproc, validate subscription

## rs\_non\_blocking\_commit

Description

Requests data servers to immediately send a positive response to a COMMIT statement without waiting for transactions to be written to disk.

Usage

- rs\_non\_blocking\_commit has function-string-class scope.
- rs\_non\_blocking\_commit executes every time DSI connects to the replicate data server when the dsi\_non\_blocking\_commit value is from 1 to 60. If the value of dsi\_non\_blocking\_commit is zero, rs\_non\_blocking\_commit does not execute.
- rs\_non\_blocking\_commit function maps to the “set delayed\_commit on” function string in Adaptive Server 15.0 and later, and to the corresponding “alter session set commit\_write = nowait;” function string in Oracle 10g v2 and later. For all other non-Sybase databases, rs\_non\_blocking\_commit maps to null.
- Replication Server with non-blocking commit enabled, supports replication into Oracle 10gV2 or later because Oracle 10g v2 supports functionality similar to delayed commit.

Replication Server 15.2 heterogeneous datatype support (HDS) scripts have new function strings that support the non-blocking commit feature. Sybase Enterprise Connect Data Access for Oracle supports these function strings. See the *Replication Server Options 15.1 Overview Guide*.

See also

rs\_non\_blocking\_commit\_flush

## rs\_non\_blocking\_commit\_flush

**Description** Sends an insert, delete, or update command to the data servers so that transactions that were sent through a connection configured with `rs_non_blocking_commit` are saved to disk.

**Examples** **Example 1** Creates an instance of an `rs_non_blocking_commit_flush` function string for Adaptive Server:

```
create function string rs_non_blocking_commit_flush
 for sqlserver_derived_class
output language
'set delayed_commit off; begin tran; update rs_lastcommit set
origin_time = getdate() where origin = 0; commit tran;
set delayed_commit on'
```

**Example 2** Creates an instance of an `rs_non_blocking_commit_flush` function string for Oracle:

```
create function string rs_non_blocking_commit_flush
 for oracle_derived_class
output language
'alter session set commit_write = immediate; begin tran;
update rs_lastcommit set origin_time = getdate() where
origin = 0; commit tran; alter session set commit_write = nowait'
```

**Usage**

- `rs_non_blocking_commit_flush` has function-string-class scope.
- `rs_non_blocking_commit_flush` executes at intervals equal to any number of minutes from 1 to 60 that you specify with `dsi_non_blocking_commit`. `rs_non_blocking_commit_flush` does not execute if the value of `dsi_non_blocking_commit` is zero.
- `rs_non_blocking_commit_flush` maps to the corresponding function string in Adaptive Server 15.0 and later, and Oracle 10g v2 and later. For all other non-Sybase databases, `rs_non_blocking_commit_flush` maps to null.
- Replication Server with non-blocking commit enabled, supports replication into Oracle 10gV2 or later because Oracle 10g v2 supports functionality similar to delayed commit.

Replication Server 15.2 heterogeneous datatype support (HDS) scripts have new function strings that support the non-blocking commit feature. Sybase Enterprise Connect Data Access for Oracle supports these function strings. See the *Replication Server Options 15.1 Overview Guide*.

**See also** `rs_non_blocking_commit`

## **rs\_raw\_object\_serialization**

|             |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
|-------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Description | Enables Replication Server to process Java columns in serialized format.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| Usage       | <ul style="list-style-type: none"><li>• <code>rs_raw_object_serialization</code> allows Replication Server to insert serialized data directly into the replicate database.</li><li>• <code>rs_raw_object_serialization</code> has function-string class scope.</li><li>• Replication Server creates an initial <code>rs_raw_object_serialization</code> function string for the system-provided function-string classes <code>rs_sql-server_function_class</code> and <code>rs_default_function_class</code> during installation.</li><li>• Replication Server uses <code>rs_raw_object_serialization</code> when the first Java column is materialized or replicated for a connection, passing the default command set <code>rs_raw_object_serialization</code> on to the Adaptive Server.</li></ul> |

## rs\_repl\_off

|             |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
|-------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Description | Specifies whether transactions executed by the maintenance user in the Adaptive Server database are replicated.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| Examples    | <p>Creates an instance of an <code>rs_repl_off</code> function string.</p> <pre>create function string rs_repl_off for sqlserver_derived_class output language 'set replication off'</pre>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
| Usage       | <ul style="list-style-type: none"><li>• <code>rs_repl_off</code> is executed for the DSI connection to a standby database.</li><li>• <code>rs_repl_off</code> has function-string-class scope.</li><li>• Replication Server creates an initial <code>rs_repl_off</code> function string for the system-provided function-string classes during installation.</li><li>• If you use a user-created base function-string class, create a function string for <code>rs_repl_off</code> if you plan to use it in any way other than the default.</li><li>• Create or customize an <code>rs_repl_off</code> function string at the Replication Server that is the primary site for the class.</li><li>• Standby database connections always use the system-provided class <code>rs_default_function_class</code>, which cannot be modified. Therefore, if you are not using warm standby, you do not need to create a function string for <code>rs_repl_off</code>.</li><li>• You can use <code>alter connection</code> or <code>configure connection</code> to set the <code>dsi_replication</code> configuration parameter and to specify whether or not to execute the <code>rs_repl_off</code> function when connecting to the standby database. Set <code>dsi_replication</code> to “off” to execute <code>rs_repl_off</code>.</li><li>• In a warm standby application, Replication Server sets <code>dsi_replication</code> to “on” for the active database and to “off” for the standby database.</li></ul> |
| See also    | <code>create connection</code> , <code>create function string</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |

## rs\_repl\_on

|             |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
|-------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Description | Sets replication on in Adaptive Server for either a database connection or database connections.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| Examples    | <p>Creates an instance of an <code>rs_repl_on</code> function string:</p> <pre>create function string rs_repl_on for sqlserver_derived_class output language 'set replication on'</pre>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| Usage       | <ul style="list-style-type: none"><li>• <code>rs_repl_on</code> is executed for the DSI connection to a database.</li><li>• <code>rs_repl_on</code> has function-string class scope.</li><li>• Replication Server creates an initial <code>rs_repl_on</code> function string for the system-provided function-string classes during installation.</li><li>• If you use a user-created base function-string class, create a function string for <code>rs_repl_on</code> if you plan to use it in any way other than the default.</li><li>• Create or customize an <code>rs_repl_on</code> function string at the Replication Server that is the primary site for the class.</li></ul> |
| See also    | <code>alter connection</code> , <code>rs_repl_off</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |

## rs\_rollback

|             |                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
|-------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Description | Rolls back a transaction. This function is reserved for future use.                                                                                                                                                                                                                                                                                                                                                                                                          |
| Examples    | <p>This example illustrates the default <code>rs_rollback</code> function string for the <code>rs_sqlserver_function_class</code> and <code>rs_default_function_class</code> classes.</p> <pre>create function string rs_rollback   for sqlserver_derived_class   output language   'rollback transaction'</pre>                                                                                                                                                             |
| Usage       | <ul style="list-style-type: none"><li>• Rolled back transactions retrieved from a primary database transaction log are not distributed to replicate Replication Servers, so this function should never be executed.</li><li>• The <code>rs_rollback</code> function has function-string-class scope.</li><li>• Replication Server creates an initial <code>rs_rollback</code> function string for the system-provided function-string classes during installation.</li></ul> |
| See also    | <code>alter function string</code> , <code>create function string</code> , <code>rs_begin</code> , <code>rs_commit</code>                                                                                                                                                                                                                                                                                                                                                    |

## rs\_select

**Description** Selects rows for subscription materialization from the primary copy of a replicated table and, for subscription dematerialization, from the replicate copy of the table.

**Examples** Creates an instance of an `rs_select` function string. Replication Server uses this function string when a subscription where clause specifies a specific value for the `au_lname` column.

```
create function string
 authors.rs_select;name_select
for flat_file_class
scan 'select * from authors
 where au_lname = ?l_name!user?'
output rpc
'execute name_sel ?l_name!user?, "authors"'
```

**Usage**

- Replication Server executes `rs_select` to retrieve subscription materialization rows from the primary Replication Server when `without holdlock` is included in the `create subscription` command. `without holdlock` is used in non-atomic materialization. The function string used for this operation is in the class assigned to the primary database.
- To retrieve data during atomic materialization, use the function-string class and error class associated with the primary database connection, not the classes associated with the replicate database connection.
- Replication Server also executes `rs_select` to identify rows for subscription dematerialization, if you drop a subscription for a table replication definition using `incrementally with purge`. The function string used for this operation is in the class assigned to the replicate database.
- If `create subscription` does not include `without holdlock`, Replication Server executes the `rs_select_with_lock` function instead of `rs_select`.
- `rs_select` has replication definition scope.
- Replication Server generates `rs_select` function strings for the system-provided function-string classes when you create a replication definition.
- If you use a user-created base function-string class, create `rs_select` function strings for each replication definition to match each possible subscription where clause.
- Create or customize an `rs_select` function string at the Replication Server where you created the replication definition.

- The default generated function strings for `rs_select`, for the `rs_sqlserver_function_class` and `rs_default_function_class` classes for each replication definition, use the Transact-SQL `select` command syntax.
- Function strings for `rs_select` have input and output templates. The input template is a SQL `select` command with a `where` clause that Replication Server matches with the `where` clause in the `create subscription` command.
- If Replication Server cannot match the `where` clause in a `select` operation to a function string input template, it uses a function string with no input template, if one exists.
- An `rs_select` function call fails if Replication Server cannot locate a function string with a matching input template or a function string with no input template.

See also

`alter function string`, `create function string`, `create subscription`, `rs_delete`,  
`rs_insert`, `rs_select_with_lock`, `rs_update`

## rs\_select\_with\_lock

|             |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
|-------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Description | Selects rows for subscription materialization from the primary copy of a replicated table, using a holdlock to maintain serial consistency.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| Examples    | <p>Creates an instance of an rs_select_with_lock function string. Replication Server uses this function string when a subscription where clause specifies a value for the au_lname column.</p> <pre>create function string   authors.rs_select_with_lock;name_select for flat_file_class scan 'select * from authors   where au_lname = ?l_name!user?' output rpc 'execute name_sel_lock ?l_name!user?, "authors"'</pre>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| Usage       | <ul style="list-style-type: none"><li>• Replication Server executes the rs_select_with_lock function to retrieve initial subscription rows from the primary Replication Server when the without holdlock clause is used with create subscription. The without holdlock clause is not used in atomic materialization. The function string used for this operation is in the class assigned to the primary database.</li><li>• Replication Server also executes rs_select_with_lock to identify rows for subscription dematerialization if you drop a subscription for a table replication definition using with purge. The function string used for this operation is in the class assigned to the replicate database.</li><li>• If the without holdlock clause is included in create subscription, Replication Server executes the rs_select function instead of rs_select_with_lock.</li><li>• rs_select_with_lock has replication definition scope.</li><li>• Replication Server generates rs_select_with_lock function strings for the system-provided function-string classes when you create a replication definition.</li><li>• If you use a user-created base function-string class, create an rs_select_with_lock function string for each replication definition to match each possible subscription where clause.</li><li>• Create or customize an rs_select_with_lock function string at the Replication Server where you created the replication definition.</li><li>• The default generated function strings for rs_select_with_lock, for the rs_sqlserver_function_class and rs_default_function_class classes for each replication definition, use the Transact-SQL select...holdlock command syntax.</li></ul> |

- Function strings for `rs_select_with_lock` have input and output templates. The input template is a SQL `select` command with a `where` clause that Replication Server matches with the `where` clause in the `create subscription` command.
- If Replication Server cannot match the `where` clause in a `select` operation to a function-string input template, it uses a function string with no input template, if one exists.
- An `rs_select_with_lock` function call fails if Replication Server cannot locate a function string with a matching input template or a function string with no input template.

See also

`alter function string`, `create function string`, `create subscription`, `rs_delete`, `rs_insert`, `rs_select`, `rs_update`

## rs\_set\_ciphertext

Description

Enables replication of encrypted columns to an Adaptive Server table.

Examples

Alters `rs_set_ciphertext` for non-Adaptive Server databases that do not support “set ciphertext on”:

```
alter function string rs_set_ciphertext
 for some_function_string_class
 output language
 , ,
```

Usage

- `rs_set_ciphertext` is called after `rs_usedb` for any user database connection. Replication Server does not call this function string for Replication Server connections and RSSD connections.
- `rs_set_ciphertext` issues “set ciphertext on” for the `rs_default_function_class` and the `rs_sqlserver_function_class`. For all other function classes, `rs_set_ciphertext` is set to null (an empty string).
- In case of failure, Replication Server continues running and does not report back to the user. This is for backward compatibility with older versions of Adaptive Server that do not support “set ciphertext on”.
- Encrypted columns come to Replication Server in varbinary, encrypted form. For materialization and dematerialization, Replication Server must either “set ciphertext on” for the database connection, or call the Adaptive Server `ciphertext()` function.

- Replication Server always sets the ciphertext property on, whether there is an encrypted column to be replicated, or whether the target database accepts ciphertext property.
- Do not specify encrypted columns as searchable. Replication Server does not know if a varbinary column is ciphertext or plain binary and cannot prevent an encrypted column being a search column.
- Do not map encrypted columns to other than varbinary datatypes. Replication Server does not know if a column is encrypted or not and cannot prevent ciphertext being converted to other datatypes.
- Replication Server cannot encrypt text, unitext, and image columns.

See also                      alter connection, alter function string, create database replication definition, create replication definition

## **rs\_set\_dml\_on\_computed**

Description                      Enables the replication of materialized computed columns to the replicate Adaptive Server database as regular columns.

Usage                              

- rs\_set\_dml\_on\_computed maps to the command set dml\_on\_computed “on” for Adaptive Server replicate databases. For all non-Sybase databases, this function maps to null.
- rs\_set\_dml\_on\_computed has function-string class scope.
- rs\_set\_dml\_on\_computed is always applied at DSI after the use database command when connection is established.
- set dml\_on\_computed “on” is not supported by Adaptive Server version 12.5.x and earlier databases. In case of failure, Replication Server will continue running and will not report back to user.

See also                      create replication definition

## rs\_set\_isolation\_level

**Description** Passes the isolation level for transactions to the replicate data server.

**Examples** Creates an instance of an `rs_set_isolation_level` function string.

```
create function string rs_set_isolation_level
for sqlserver_derived_class
output language
'set transaction isolation level?rs_isolation_level!sys_raw?'
```

- Usage**
- The `rs_set_isolation_level` function passes the transaction isolation level to the replicate data server, and executes every time the DSI connects to the replicate data server if a value has been set for `dsi_isolation_level`. If the `dsi_isolation_level` is the default value, `rs_set_isolation_level` is not executed.
  - Use the `alter connection` or `create connection` with the `set_isolation_level` option to the value for the variable `rs_isolation_level`. The supported Adaptive Server values are 0, 1, 2, and 3. Replication Server supports all other isolation level values supported by other data servers. If no value is supplied for `rs_isolation_level`, Replication Server uses the isolation value of the target data server.
  - Replication Server executes `rs_set_isolation_level` immediately after executing the `rs_usedb` function-string command.
  - The `rs_set_isolation_level` function has function-string class scope.
  - Replication Server creates an initial `rs_set_isolation_level` function string for the Adaptive Server and default function-string classes during installation.
  - If you use a nondefault function-string class and you use the parallel DSI feature, create a function string for the `rs_set_isolation_level` function. The modified function string must contain the variable `rs_isolation_level`.
  - Create or customize an `rs_set_isolation_level` function string at the Replication Server that is the primary site for the class.

**See also** `create connection`, `rs_get_thread_seq`, `rs_initialize_threads`, `rs_update_threads`

## rs\_set\_quoted\_identifier

### Description

Configures a data server connection to accept quoted identifiers.

---

**Note** Data servers such as Adaptive Server, SQL Anywhere, Microsoft SQL Server, Universal Database (UDB), and Oracle handle quoted identifiers differently in terms of length, special characters, and reserved words supported. In a heterogeneous environment, you must ensure that the quoted identifiers being replicated are valid on both the primary and replicate data servers.

---

### Usage

- `rs_set_quoted_identifier` is added to the default function string classes and has function-string-class scope.
- When `dsi_quoted_identifier` is on, Replication Server sends `rs_set_quoted_identifier` to the replicate data server to signal the data server to expect quoted identifiers. If the replicate data server is Adaptive Server, SQL Anywhere, or Microsoft SQL Server, `rs_set_quoted_identifier` is set to `set quoted_identifiers on` command. Otherwise, `rs_set_quoted_identifier` is set to `""`.

### See also

`create connection`, `create replication definition`, `alter connection`, `alter replication definition`

## rs\_setproxy

### Description

Changes the login name in a data server.

### Usage

- `rs_setproxy` has function-string-class scope.
- Replication Server creates an `rs_setproxy` function string for the `rs_sqlserver_function_class` function-string class during installation. The default value is:  
  
`set session authorization "?rs_destination_user!sys"`  
  
The generated string has the syntax of the Adaptive Server `set proxy` command. Use `alter function string` to replace the default function string.
- If a data server does not support network security services or does not have a corresponding `set proxy` command, you can either turn `unified_login` to "not required" or create an empty `rs_setproxy` function string.

- Function-string variable modifiers *sys* contains the login name of a data server. This login name is usually that of the maintenance user or the subscription user.

See also

alter function string, create function string

## rs\_sqldml

Description

A replicated function that carries SQLDML to Replication Server.

Examples

Sends SQLDML to Replication Server as a stored procedure named *rs\_sqldml*:

```
create proc rs_sqldml
 @rs_operator char(1),
 @rs_status int,
 @rs_insert_column varchar(16384),
 @rs_from varchar(16384),
 @rs_where varchar(16384),
 @rs_set varchar(16384),
 @rs_select varchar(16384),
 @rs_owner varchar(255),
 @rs_object varchar(255),
 @rs_rowcount int
```

where:

- *rs\_operator* – any of:
  - U – update
  - D – delete
  - I – insert select
  - S – select into
- *rs\_object* – the operated table name
- *rs\_owner* – the operated table owner. If the owner status of the table is off, owner name will be null.
- *rs\_category* – the SQLDML category:
  - C1 – statements that can be applied at any replicated database and will generate identical result set.
  - C2 – statements that can be applied only at warm standby or MSA database to generate identical result set.

- *rs\_status* – the SQLDML status.
- *rs\_set* – the set clause in an UPDATE statement
- *rs\_where* – the where clause
- *rs\_select* – the select clause in an INSERT SELECT or SELECT INTO statement
- *rs\_from* – the from clause in an INSERT SELECT or SELECT INTO statement
- *rs\_insert\_column* – the column list of an INSERT SELECT statement
- *rs\_rowcount* – the number of impacted rows, which is available only at the end of rs\_sqldml.

#### Usage

- rs\_sqldml is sent to Replication Server as a replicated function. If a SQLDML does not have a responding clause, the parameter will be set to null.
- SELECT INTO cannot be executed inside a user-defined transaction and is replicated as a system transaction.
- RepAgent sends both rs\_sqldml and its affected row log records to Replication Server, and Replication Server decides whether to apply SQLDML or the affected rows to a target.
- Adaptive Server logs execbegin rs\_sqldml to indicate the beginning of a SQLDML, an execend rs\_sqldml to indicate the ending of a SQLDML. SQLDML is packed inside the execbegin command. @rs\_rowcount is packed inside execend command.
- To prevent log SQLDML that changes less than SQLDML replication threshold rows, Adaptive Server performs deferred logging for execbegin. It does not log execbegin when a SQLDML until it changes more than the threshold rows. RepAgent flags the first log record of a SQLDML.
- SQLDML deferred logging is not required. A non-Adaptive Server replication agent, for example, may not perform deferred logging.

## rs\_textptr\_init

|             |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
|-------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Description | Allocates a text pointer for a text, unitext, or image column.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| Examples    | <p>Creates an <code>rs_textptr_init</code> function string for the copy column in the blurbs table.</p> <pre>create function string blurbs_rep.rs_textptr_init;copy for sqlserver2_function_class output language 'update blurbs set copy = NULL   where au_id = ?au_id!new?'</pre>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
| Usage       | <ul style="list-style-type: none"><li>• Replication Server executes <code>rs_textptr_init</code> when a row arrives, indicating that modifications were made at the primary database, which caused a text pointer allocation for the text, unitext, or image column. It is also executed when Replication Server needs to do a <code>writetext</code> operation at the replicate database and the text pointer has not been allocated.</li><li>• The <code>rs_textptr_init</code> function has replication definition scope.</li><li>• For each replicated text, unitext, or image column in a replication definition, Replication Server generates an <code>rs_textptr_init</code> function string for the <code>rs_sqlserver_function_class</code> and <code>rs_default_function_class</code> classes when you create the replication definition.</li><li>• If you use a user-created base function-string class, create an <code>rs_textptr_init</code> function string for each replicated text, unitext, or image column included in the replication definition.</li><li>• Create or customize an <code>rs_textptr_init</code> function string at the Replication Server where you created the replication definition.</li></ul> |
| See also    | <code>rs_get_textptr</code> , <code>rs_datarow_for_writetext</code> , <code>rs_writetext</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |

## rs\_ticket\_report

Description                      Insert ticket to the rs\_ticket\_history table.

Examples                         A sample of the customized rs\_ticket\_report:

```
alter function string rs_ticket_report
for rs_sqlserver_function_class
output language
 'insert rs_ticket_history(h1,h2,h3,h4,
 pdb,prs,rrs,rdb,pdb_t,exec_t, dist_t,rsi_t,
 dsi_t,exec_b,rsi_b,dsi_tnx,dsi_cmd,ticket)
values(?h1!param?, ?h2!param?, ?h3!param?,
 ?h4!param?, ?rs_origin_db!sys?, ?prs!param?,
 ?rrs!param?, ?rs_destination_db!sys?,
 ?pdb!param?, ?exec!param?, ?dist!param?,
 ?rsi!param?, ?dsi!param?, ?b!param?,
 ?rsi_b!param?, ?dsi_t!param?, ?dsi_c!param?,
 ?rs_ticket_param!param?) '
```

Usage

- rs\_ticket\_report has function-string class scope.
- rs\_ticket\_report writes rs\_ticket information to the rs\_ticket\_history table. However, you can customize the rs\_ticket\_report to use the rs\_ticket information as you require.

For information about the rs\_ticket\_history parameters, see rs\_ticket\_history on page 722.

- To disable rs\_ticket\_report, set the connection configuration parameter dsi\_rs\_ticket\_report to off.

See also                         rs\_ticket, rs\_ticket\_history

## rs\_triggers\_reset

**Description** Turns off triggers in Adaptive Server and Oracle.

**Examples** **Example 1** Creates an instance of an `rs_triggers_reset` function string for a user-created base function-string class for Adaptive Server.

```
create function string rs_triggers_reset
for sqlserver2_function_class
output language
'set triggers off'
```

**Example 2** Creates an instance of an `rs_triggers_reset` function string for a user-created base function-string class for Oracle.

```
create function string rs_triggers_reset
for oracle_function_class
output language
'BEGIN rs_trigger_control.enable();; END;;'
```

---

**Note** Unlike Adaptive Server, which has a `set triggers off` command, Oracle does not publish a session-level trigger control. Hence, you need to install the `RS_TRIGGER_CONTROL` package in the replicate Oracle database using create connection using profile to be able to work with `rs_triggers_reset` function string. See Chapter 10, “Oracle Replicate Data Server Issues” in the *Replication Server Heterogeneous Guide*.

---

**Usage**

- By default, the `rs_triggers_reset` function is executed for the DSI connection to a standby database, and is not executed for any other DSI connection.
- `rs_triggers_reset` has function-string-class scope.
- During installation, Replication Server creates an initial `rs_triggers_reset` function string for the system-provided function-string classes.
- Standby database connections always use the system-provided class `rs_default_function_class`, which cannot be modified. For any other database connection, you do not need to create a function string for the `rs_triggers_reset` function, unless:
  - The database connection uses a user-created base function-string class, and
  - You want to set the `dsi_keep_triggers` configuration parameter to “off” for the connection.

- Create an `rs_triggers_reset` function string at the Replication Server that is the primary site for the class.
- Setting `dsi_keep_triggers` to “off” for a database connection to execute `rs_triggers_reset` when the connection is established. The `dsi_keep_triggers` default is “off” for standby databases, and “on” for replicate databases. Use the `alter connection` or `configure connection` command to change this setting.

See also

`create connection`, `create function string`

## rs\_truncate

### Description

Truncates a table or a table partition in a replicate database.

### Examples

**Example 1** Replaces the existing `rs_truncate` function string for the `authors` table with one that executes a Transact-SQL `delete` command, which logs all deletions, instead of the `truncate table` command, which does not log deletions.

```
alter function string authors.rs_truncate
for sqlserver_derived_class
output language
'delete authors'
```

You would want to customize the `rs_truncate` function string for the `authors` table, if:

- The replicate database doesn't support table the Transact-SQL `truncate table` command, or
- You want to have deletions logged at the replicate database.

**Example 2** Replaces the existing `rs_truncate` function string for the `publisher` table to replicate `truncate table partition` as a `delete` command:

```
alter function string publisher.rs_truncate
for rs_sqlserver_function_class
output language
'begin transaction
 if (?1!param? = '') /* No parameter */
 delete publisher
 if (?1!param? = 'A')
 delete publisher where c1 < 1000
 if (?1!param? = 'B')
 delete publisher where c1 >= 1000
commit transaction'
```

**Example 3** Alters the function string to do nothing if there is a parameter so that table partitions are not truncated at replicate:

```
alter function string publisher.rs_truncate
for rs_sqlserver_function_class
output language
'if(?1!param? = '') delete publisher'
```

### Usage

- `rs_truncate` has a replication definition scope. Replication Server executes it to truncate a table or one or more table partitions.

- Replication Server generates an `rs_truncate` function string for the system-provided function-string classes when you create the replication definition.
- If you use a user-created base function-string class, create an `rs_truncate` function string for each replication definition.
- Create or customize an `rs_truncate` function string at the Replication Server where you created the replication definition.
- The default-generated function string for `rs_truncate`, for the `rs_sqlserver_function_class` and `rs_default_function_class` classes for each replication definition, uses the Transact-SQL truncate table command syntax. It deletes all rows in a table without logging the deletion of each individual row.
- Replication Server will reconstruct the same command executed at the primary. This command requires that the replicate site to have the same partition names. If not, DSI will shut down.
- The partition names are passed as parameters to the `rs_truncate` function. `rs_truncate` function string accepts position-based function-string parameters. The following is a position-based variable:

`?n!param?`

The function-string variable `?1!param?` corresponds to the first parameter in the `rs_truncate` function.

- Table 4-5 lists the function string variable modifiers.

**Table 4-5: Function string variable modifiers**

| Modifier                      | Description                                                                                                                                                                                                                                                                                                                               |
|-------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>new, new_raw</code>     | A reference to the new value of a column in a row you are inserting or updating                                                                                                                                                                                                                                                           |
| <code>old, old_raw</code>     | A reference to the existing value of a column in a row you are updating or deleting                                                                                                                                                                                                                                                       |
| <code>user, user_raw</code>   | A reference to a variable that is defined in the input template of an <code>rs_select</code> or <code>rs_select_with_lock</code> function string                                                                                                                                                                                          |
| <code>sys, sys_raw</code>     | A reference to a system-defined variable                                                                                                                                                                                                                                                                                                  |
| <code>param, param_raw</code> | A reference to a function parameter                                                                                                                                                                                                                                                                                                       |
| <code>text_status</code>      | A reference to or a function parameter. If the parameter is not defined through function replication definition or user defined function ( <code>create function</code> ), there must be a number between 1 and 99 (with no leading 0) in place of parameter name which states the parameter position in the function in the LTL command. |

- A function string has a minimum version of 1500 if it contains position-based function-string variables. A replication definition has a minimum version of at least 1500 if it contains a 1500 function string.

See also

alter function string, rs\_datarow\_for\_writetext, rs\_get\_textptr, rs\_insert,  
rs\_delete, rs\_textptr\_init, rs\_writetext, set

## rs\_update

### Description

Updates a single row in a table in a replicate database.

### Examples

Replaces the existing `rs_update` function string for the authors table with one that is similar to the default function string generated by Replication Server for the system-provided function-string classes.

```
alter function string authors.rs_update
for sqlserver_derived_class
output language
'update authors set au_id = ?au_id!new?,
 au_lname = ?au_lname!new?,
 au_fname = ?au_fname!new?,
 phone = ?phone!new?,
 address = ?address!new?,
 city = ?city!new?,
 state = ?state!new?,
 country = ?country!new?,
 postalcode = ?postalcode!new?
where au_id = ?au_id!old?'
```

### Usage

- Replication Server executes `rs_update` to update a single row in a table. The row is identified by the primary key columns defined in a replication definition for the table.
- The `rs_update` function has replication definition scope.
- Replication Server generates an `rs_update` function string for the system-provided function-string classes when you create the replication definition.
- If you use a user-created base function-string class, create an `rs_update` function string for each replication definition.
- Create or customize an `rs_update` function string at the Replication Server where you created the replication definition.
- The default generated function string for `rs_update`, for the `rs_sqlserver_function_class` and `rs_default_function_class` classes for each replication definition, uses the Transact-SQL update command syntax. It replaces all columns in the row, and identifies the row with a `where` clause that specifies the pre-update values, or before image, of the primary key columns.
- When set autocorrection is on, Replication Server does not use `rs_update`. Instead, it calls `rs_delete` to remove the existing row and `rs_insert` to insert the row.

- Replication Server cannot send text, unitext, or image data with `rs_update`, but it can report the status of text, unitext, or image data with the *text\_status* modifier. For a description of the *text\_status* modifier, see `rs_datarow_for_writetext`. Data of type text, unitext, or image is sent to the replicate database with the `rs_get_textptr`, `rs_textptr_init`, `rs_datarow_for_writetext`, and `rs_writetext` functions.

See also

`alter function string`, `rs_datarow_for_writetext`, `rs_get_textptr`, `rs_insert`, `rs_delete`, `rs_textptr_init`, `rs_writetext`, `set`

## rs\_update\_threads

|             |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
|-------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Description | Updates the sequence number for the specified entry in the rs_threads system table.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| Syntax      | <code>rs_update_threads @rs_id, @rs_seq</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| Parameters  | <p><i>rs_id</i><br/>a number of int datatype representing the ID of the entry to be updated.</p> <p><i>rs_seq</i><br/>a number of int datatype representing the new sequence number for the entry.</p>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| Examples    | <p>Creates an rs_update_threads function string that executes a stored procedure named rs_update_threads. The text of the stored procedure is:</p> <pre>create function string rs_update_threads for sqlserver_derived_class output language 'execute rs_update_threads @rs_seq = ?rs_seq!param?, @rs_id = ?rs_id!param!'</pre> <pre>create procedure rs_update_threads @rs_id int, @rs_seq int as update rs_threads set seq = @rs_seq where id = @rs_id</pre>                                                                                                                                                                                                                                                                                                                             |
| Usage       | <ul style="list-style-type: none"><li>• The rs_update_threads function is executed at the start of each transaction when more than one DSI thread is defined for a connection. It is executed only when more than one DSI thread is defined for a connection.</li><li>• The rs_update_threads function has function-string-class scope.</li><li>• Replication Server creates an initial rs_update_threads function string for the system-provided function-string classes during installation.</li><li>• If you use a user-created base function-string class and the parallel DSI feature, create a function string for rs_update_threads.</li><li>• Create or customize an rs_update_threads function string at the Replication Server that is the primary site for the class.</li></ul> |
| See also    | <code>create connection</code> , <code>rs_get_thread_seq</code> , <code>rs_initialize_threads</code> , <code>rs_set_isolation_level</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |

## rs\_usedb

Description

Changes the database context in a data server.

Examples

**Example 1** Changes an existing `rs_usedb` function string to one that is similar to the default function string generated by Replication Server for the system-provided function-string classes.

```
alter function string rs_usedb
for sqlserver_derived_class
output language
'use ?rs_destination_db!sys_raw?'
```

**Example 2** Creates an `rs_usedb` function string with an empty string for an output template for a data server that does not support multiple databases.

```
create function string rs_usedb
for TOKYO_DS
output language ''
```

Usage

- The Replication Server DSI executes the function when it first connects to the data server.
- `rs_usedb` has function-string class scope.
- Replication Server creates an initial `rs_usedb` function string for the system-provided function-string classes during installation.
- If you use a user-created base function-string class, create a function string for the `rs_usedb` function.
- Create or customize an `rs_usedb` function string at the Replication Server that is the primary site for the class.
- The default generated function string for the `rs_usedb` function, for the `rs_sqlserver_function_class` and `rs_default_function_class` classes, has the syntax of the Transact-SQL `use` command.
- If a data server does not support multiple databases or a database context, the output template can be an empty string ('').

See also

`alter function string`, `create function string`

## rs\_writetext

### Description

Modifies text, unitext, or image data in a replicate database.

### Examples

**Example 1** Creates an `rs_writetext` function string that uses the `RPC` method to update the `copy` column in the `blurbs` table.

```
create function string
 blurbs_rep.rs_writetext;copy
for gw_function_class
output rpc
'execute update_blurbs_copy
 @copy_chunk = ?copy!new?,
 @au_id = ?au_id!new?,
 @last_chunk = ?rs_last_text_chunk!sys?,
 @writetext_log = ?rs_writetext_log!sys?'
```

**Example 2** Creates an `rs_writetext` function string that uses the `writetext` method to update the `copy` column. Replication Server modifies the `copy` column by using the I/O descriptor returned by the execution of the `rs_get_textptr` function for the `copy` column.

```
create function string
 blurbs_rep.rs_writetext;copy
for rs_sqlserver2_function_class
output writetext
use primary log
```

For example, if you have a function string for `rs_get_textptr`, then the `rs_writetext` function modifies the `repcopy` column in the `blurbs` table, as follows:

```
create function string
 blurbs_rep.rs_get_textptr;copy
for sqlserver2_function_class
output language
'select repcopy from blurbs
 where au_id = ?au_id!new?'
```

**Example 3** Creates an `rs_writetext` function string that uses the `none` method to specify that the `copy` column should not be updated.

```
create function string
 blurbs_rep.rs_writetext;copy
for rs_sqlserver2_function_class
output none
```

### Usage

- `rs_writetext` has replication definition scope.

- For each replicated text, untext, or image column in a replication definition, Replication Server generates an `rs_writetext` function string for the `rs_sqlserver_function_class` and `rs_default_function_class` classes when you create the replication definition.
- If you use a user-created function-string class, create an `rs_writetext` function string for each replicated text, untext, or image column included in the replication definition.
- Create or customize an `rs_writetext` function string at the Replication Server where you created the replication definition.
- Replication Server supports three output formats for creating an `rs_writetext` function string: RPC, writetext, and none.

#### Using the RPC Method

With the RPC method for creating an `rs_writetext` function string, Replication Server executes a remote procedure call repeatedly, providing up to 255 bytes of the text, untext, or image value on each procedure execution.

The data is passed in the RPC in a varchar parameter for text or untext data or in a varbinary parameter for image data. Replication Server ensures that the data chunks are partitioned on character boundaries for text or untext columns. If a 1-byte character set is in use, the data is sent in 255-byte chunks.

Each time Replication Server executes the RPC, it sets the `rs_last_text_chunk` system variable, an int, to 0 if there is more data to follow or to 1 if this is the last RPC execution for this text column.

- Another int system variable, `rs_writetext_log`, is set to 1 if the writetext logging option was used in the primary database or 0 if the logging option was not used in the primary database.
- The values of other columns in the data row can be accessed by using the *new* or *old* modifier. If you used the Transact-SQL insert command at the primary database, you must use the *new* modifier.
- Use the `text_status` modifier to retrieve the status of a text, untext, or image column. For a description of the `text_status` modifier, see `rs_datarow_for_writetext`.

#### Using the writetext method

The writetext method for creating an `rs_writetext` function string provides the options shown in Table 4-6 to specify the logging behavior in the replicate database.

**Table 4-6: writetext logging options**

| Logging option  | Description                                                                                                                                                                                                         |
|-----------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| use primary log | Log the data in the replicate database transaction log if the logging option was specified in the primary database transaction log. Do not log if logging is not specified in the primary database transaction log. |
| with log        | Log the data in the replicate database transaction log.                                                                                                                                                             |
| no log          | Do not log the data in the replicate database transaction log.                                                                                                                                                      |

The default function string for rs\_sqlserver\_function\_class uses the use primary log option.

Using the none method

The none output template option for rs\_writetext function strings instructs Replication Server not to use the Client-Library function ct\_send\_data to update a text, unitext, or image column value. This option provides necessary flexibility for using text, unitext, or image columns in a heterogeneous environment.

See the *Replication Server Administration Guide Volume 2* for more information.

See also

rs\_get\_textptr, rs\_textptr\_init, rs\_datarow\_for\_writetext

## **Adaptive Server Commands and System Procedures**

This chapter contains reference pages for the Adaptive Server commands and system procedures used with Replication Server.

## dbcc dbrepair

|             |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
|-------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Description | A Transact-SQL command that clears the secondary truncation point for an offline replicated database.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| Syntax      | dbcc dbrepair( <i>database_name</i> , ltmignore)                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
| Parameters  | <p><i>database_name</i></p> <p>The name of the database for which you want to clear the secondary truncation point.</p> <p>ltmignore</p> <p>Deactivates the secondary truncation point in the named database.</p>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| Usage       | <ul style="list-style-type: none"><li>• dbcc dbrepair clears the secondary truncation point for offline databases; dbcc settrunc with the ignore option clears the secondary truncation point for online databases.</li><li>• Sybase recommends that you drain the transaction log and clear the secondary truncation point for a replicated database before starting an upgrade. If you have not performed these two tasks, Adaptive Server does not allow you to bring the database online after upgrade.</li><li>• If you do not drain the transaction log and clear the secondary truncation point before upgrade, use dbcc dbrepair so that Adaptive Server can bring the database online.</li></ul> <p>Before running dbcc dbrepair:</p> <ol style="list-style-type: none"><li>a Start the RepAgent thread on the offline database.</li><li>b Drain the transaction log.</li></ol> <p>If you do not drain the transaction log before running dbcc dbrepair, all transactions in the log are lost.</p> |
| See also    | dbcc settrunc                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |

## dbcc gettrunc

|             |                                                                                                                                                                                                        |
|-------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Description | A Transact-SQL command to retrieve current RepAgent information about an Adaptive Server database.                                                                                                     |
| Syntax      | dbcc gettrunc                                                                                                                                                                                          |
| Usage       | <ul style="list-style-type: none"> <li>Use dbcc gettrunc for RepAgent-enabled databases.</li> <li>The dbcc gettrunc command returns a single row containing the columns shown in Table 5-1:</li> </ul> |

**Table 5-1: Columns returned by dbcc gettrunc**

| Column name           | Contents                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
|-----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>RepAgent</b>       |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| secondary trunc page  | The first page that is not truncated in the database log                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| secondary trunc state | One of the following values: <ul style="list-style-type: none"> <li>1 – Adaptive Server does not truncate the log on or after the truncation page</li> <li>0 – Adaptive Server ignores the truncation page</li> </ul>                                                                                                                                                                                                                                                                                                                                                                   |
| db rep stat           | A mask constructed of the following: <ul style="list-style-type: none"> <li>0x01 – Secondary truncation page is valid</li> <li>0x02 – database contains at least one explicitly replicated table</li> <li>0x04 – database contains replicated stored procedures</li> <li>0x08 – replicate all to standby database</li> <li>0x10 – replicate L1 to standby database</li> <li>0x80 – Replication Agent automatically restarts after an HA failover</li> </ul> RepAgent only: <ul style="list-style-type: none"> <li>0x20 – RepAgent enabled</li> <li>0x40 – autostart RepAgent</li> </ul> |
| generation id         | The database generation ID                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| database id           | The Adaptive Server ID number of the database                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| database name         | The name of the database                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| ltl version           | RepAgent: The log transfer language (LTL) version                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |

**Note** There is no difference in replicating L1 and all to standby database because to date, only support level L1 has been implemented in Adaptive Server version 12.0 and later. For more information, see `sp_reptostandby`.

See also                      `admin get_generation`, `dbcc settrunc`

## dbcc settrunc

|             |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
|-------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Description | A Transact-SQL command that modifies the secondary truncation point information for an Adaptive Server database.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| Syntax      | <pre>dbcc settrunc('ltm', {'valid'   'ignore'})<br/>dbcc settrunc('ltm', 'gen_id', db_generation)</pre>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| Parameters  | <p><b>valid</b><br/>Instructs Adaptive Server to respect the secondary truncation point. This option prevents the Adaptive Server from truncating transaction log records that have not been transferred to Replication Server.</p> <p><b>ignore</b><br/>Instructs Adaptive Server to ignore the secondary truncation point. This allows Adaptive Server to truncate log records that the RepAgent has not yet transferred to the Replication Server.</p> <p><b>gen_id</b><br/>Instructs Adaptive Server to reset the database generation number in the log.</p> <p><b>db_generation</b><br/>The new database generation number. Increment the number after restoring dumps to prevent Replication Server from rejecting new transactions as duplicates.</p> |

---

**Warning!** You cannot execute `dbcc settrunc` when RepAgent is running.

---

|       |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
|-------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Usage | <ul style="list-style-type: none"><li>• Use <code>dbcc settrunc</code> for RepAgent-enabled databases.</li><li>• The secondary truncation point must be valid for Adaptive Server databases containing primary data to be replicated or for databases where replicated stored procedures are stored.</li><li>• When the secondary truncation point is valid, Adaptive Server does not truncate log records that the Replication Server has not yet received from the RepAgent.</li></ul> |
|-------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

- If the secondary truncation point is not modified for an extended period of time, the log may fill up and prevent applications from continuing. You can change the secondary truncation point to ignore—after shutting down the Replication Server and the RepAgent—so that the log can be truncated and applications can continue working. Then use the `rs_zeroltm` procedure to reset the locator value to zero (0). However, note this warning:

---

**Warning!** If you set the secondary truncation point to ignore and then truncate the log, replicated data will be incorrect. You must either re-create subscriptions, reconcile subscriptions by executing `rs_subcmp`, or load database and transaction dumps and replay the lost transactions. See the *Replication Server Administration Guide Volume 2* for instructions for replaying lost transactions. You should increment the database generation number after restoring coordinated dumps. Use `admin get_generation` to find the current generation number.

---

See `rs_zeroltm` on page 634 for details about running this stored procedure.

- Increment the database generation number after restoring to prevent Replication Server from rejecting new log records. See the *Replication Server Administration Guide Volume 2* for information about reloading coordinated dumps.
- If the primary Replication Server is unable to accept transactions and the primary database transaction log is full and must be truncated, you may need to turn off the secondary truncation point and truncate the log in order to allow Adaptive Server transactions to continue. In this situation, use `dbcc settrunc('ltm', 'ignore')` to shut down the Replication Agent and turn off the secondary truncation point in the database.

After using `dbcc settrunc`, you *must* use the `rs_zeroltm` stored procedure to reset the locator value for a database to 0. Otherwise, the log page stored in the `rs_locator` system table may become invalid. Starting the RepAgent may then cause Adaptive Server to register data corruption and to produce errors such as 605 and 813.

- Transactions that execute after you have turned off the secondary truncation point are not transferred to the Replication Server. Therefore, primary and replicate databases may not be in synch.

For this reason, after you have truncated the log and after the Replication Server has been brought up successfully, you may have to alter replication definitions, drop and re-create subscriptions, and re-materialize the data in the replicate database. New columns will be null until the data is re-materialized.

If a relatively small number of transactions did not transfer to the Replication Server, you may instead choose to use the `rs_subcmp` program to reconcile the primary and replicate databases.

See also

`admin get_generation`, `dbcc dbrepair`, `rs_subcmp`, `rs_zeroltm`,  
`sp_config_rep_agent`

# set replication

|             |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
|-------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Description | A Transact-SQL command that enables or disables replication of data definition language (DDL) and/or data manipulation language (DML) commands to the standby database for the current isql session.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| Syntax      | set replication [on   force_ddl   default   off]                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| Parameters  | <div>on</div> <div>Enables replication of DML commands for tables marked with sp_setreptable, if sp_reptostandby is set to “none.” If sp_reptostandby is set to “L1” or “all,” enables replication of DML and DDL commands to the standby database. This is the default setting.</div> <div>force_ddl</div> <div>Always enables replication of DDL commands for the current session. If sp_reptostandby is set to “L1” or “all,” DML commands are replicated for all user tables. If sp_reptostandby is set to “none,” DML commands are replicated for tables marked with sp_setreptable.</div> <div><hr/></div> <div><b>Note</b> Beginning with Replication Server version 12.0, force_ddl as used in the command set replication force_ddl is no longer a reserved word. This does not affect set replication force_ddl functionality; you no longer have to use double quotes when using force_ddl in other object names.</div> <div><hr/></div> <div>default</div> <div>Turns off force_ddl and returns set replication status to “on”—the default.</div> <div>off</div> <div>Turns off replication of marked tables and user stored procedures for the current session. No DML commands and no DDL commands are copied to the standby or replicate database.</div> |
| Usage       | <ul style="list-style-type: none"><li>set replication requires Adaptive Server version 11.5 or later databases.</li></ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
| Permissions | set replication requires “sa” or “dbo” permission and replication_role.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
| See also    | sp_reptostandby, sp_setreptable                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |

## set repmode

|             |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
|-------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Description | Enables or disables, at the session level, the replication of update, delete, insert select, or select into as SQL statements.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
| Syntax      | <pre>set repmode {"on" SQLDML_option   "never"   "off"   'threshold', 'value'}</pre> <p><i>SQLDML_option</i> ::= { U   D   I   S }</p>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| Parameters  | <p><i>SQLDML_option</i></p> <p>Any combination of these DML operations:</p> <ul style="list-style-type: none"> <li>• U – update</li> <li>• D – delete</li> <li>• I – insert select</li> <li>• S – select into</li> </ul> <p>SQL replication settings defined using set repmode overrides those defined using sp_setrepdbmode or sp_setrepdefmode.</p> <p><b>on</b></p> <p>Enables SQL replication of DML operation specified.</p> <p><b>off</b></p> <p>Removes the session-level replication settings of SQL statements and returns to the database-level or table-level settings.</p> <p><b>never</b></p> <p>Specifies not to replicate SQL statements.</p> |
| Examples    | <p><b>Example 1</b> To replicate only select into and delete as SQL statements for the duration of the session, use:</p> <pre>set repmode on 'DS'</pre> <p><b>Example 2</b> To disable SQL statement replication for the duration of the session, regardless of the database or table-level settings, use:</p> <pre>set repmode never</pre> <p><b>Example 3</b> This example illustrates how session-level settings override object-level settings. This example replicates only update statements using SQL statement replication:</p> <pre>set repmode on 'U' go sp_setrepdefmode tabname, on, 'UDI' go</pre>                                              |

**Example 3** This example shows how to define the threshold at the session-level as 1000 rows:

```
set repmode 'threshold', '1000'
go
```

Usage

- You can set the session-level options either at login by using a “login trigger”, or at the beginning of a batch. Your session settings overwrites the table or database settings.
- Session-level settings are active only for the duration of the session. When you set the options inside a stored procedure or a trigger, the settings are reverted back to the table-level or database-level settings when the stored procedure or trigger execution terminates.

See also

sp\_setrepdbmode, sp\_setrepdefmode

## set repthreshold

Description

Specifies the minimum number of rows that a replicated SQL statement must impact before SQL statement replication is activated for the session.

Syntax

set repthreshold *value*

Parameters

*value*

Specifies the minimum number of rows that a replicated SQL statement must impact before SQL statement replication is activated for the session.

Examples

**Example 1** This example shows how to define the threshold at the session-level to 23, in the absence of any threshold setting at the database and table-levels or to override the threshold settings at the table and database levels:

```
set repthreshold 23
go
```

**Example 2** This example shows how to reset the threshold to the default of 50, at the session-level:

```
set repthreshold 0
go
```

**Example 3** You can invoke set repthreshold within an Adaptive Server stored procedure. This example shows how to create the set\_rep\_threshold\_23 stored procedure and invoke it within the my\_proc stored procedure:

- 1 Create the `set_rep_threshold_23` stored procedure:

```
create procedure set_rep_threshold_23
as
set repthreshold 23
update my_table set my_col = 2 (statement 2)
go
```

- 2 Create the `my_proc` stored procedure:

```
create procedure my_proc
as
update my_table set my_col = 1 (statement 1)
exec set_rep_threshold_23
update my_table set my_col = 3 (statement 3)
go
```

- 3 Execute `my_proc` to invoke `set_repthreshold_23`:

```
exec my_proc
go
```

Within the `my_proc` stored procedure, statement 1 executes first with a threshold of 50. Statement 2 executes next with a threshold of 23. Statement 3 executes next with a threshold of 50, because the `set repthreshold 23` command is only valid while executing the `set_rep_threshold_23` procedure.

**Example 4** This example shows how to make the session-level threshold exportable. Therefore, you can set the `export_options` setting to 'on' for a procedure, and set the SQL statement replication threshold, so that procedures in the outer scope use the SQL statement replication threshold set by the stored procedure.

- 1 Create the `set_repthreshold_23` stored procedure and set `export_options` on:

```
create procedure set_repthreshold_23
as
set repthreshold 23 (statement 4)
set export_options on
update my_table set my_col = 2 (statement 2)
go
```

- 2 Create the `my_proc` stored procedure:

```
create procedure my_proc
as
update my_table set my_col = 1 (statement 1)
exec set_rep_threshold_23
update my_table set my_col = 3 (statement 3)
go
```

- 3 Execute my\_proc to invoke set\_repthreshold\_23:

```
exec my_proc
go
```

Statement 1 executes first, with a threshold of 50. Statement 2 executes next with a threshold of 23. Statement 3 executes next with a threshold of 50, because the scope of the set repthreshold 23 command is the scope of the session.

**Example 5** You can create login triggers to set the replication threshold automatically for a specific login ID.

- Create the threshold stored procedure with a threshold setting of 23 and enable export:

```
create proc threshold
as
set repthreshold 23
set export_options on
go
```

- Instruct Adaptive Server to automatically run the threshold stored procedure when user “Bob” logs in:

```
sp_modifylogin Bob, 'login script', threshold
go
```

When Bob logs into Adaptive Server, the SQL statement replication threshold for the session is set to 23.

## Usage

- The default threshold is 50 rows, which means that Adaptive Server uses SQL statement replication if the DML statement affects at least 51 rows. To use the default threshold, set the threshold parameter to 0. The threshold parameter range is 0 to 10,000.
- You can invoke set repthreshold within an Adaptive Server stored procedure.
- The session-level threshold is exportable. Therefore, you can set the export\_options setting ‘on’ for a procedure, and set the SQL statement replication threshold, so that procedures in the outer scope use the SQL statement replication threshold set by the stored procedure
- You can set the session-level threshold either at login by using a “login trigger”, or at the beginning of a batch. Your session settings overwrites the table or database settings.

- Session-level thresholds are active only for the duration of the session. When you set the threshold inside a stored procedure or a trigger, the settings are reverted back to the table-level or database-level settings when the stored procedure or trigger execution terminates.
- The threshold set at the session-level overrides the threshold at the table-level and database-level, and the threshold set for any table overrides the threshold set at the database-level.

See also

`sp_setrepcbmode`, `sp_setrepdefmode`, `set repmode`

## **sp\_configure 'enable rep agent threads'**

|             |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
|-------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Description | Enables or disables RepAgent thread integration in the Adaptive Server.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| Syntax      | <code>sp_configure 'enable rep agent threads'[, 1   0]</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
| Parameters  | <p>1<br/>Enables RepAgent integration for the data server.</p> <p>0<br/>Disables RepAgent integration for the data server.</p>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| Usage       | <ul style="list-style-type: none"><li>• Use <code>sp_configure 'enable rep agent threads'</code> to enable RepAgent for Adaptive Server version 12.0 or later databases.</li><li>• Use <code>sp_configure 'enable rep agent threads'</code> without options to display the current value, default value, and most recently changed value.</li><li>• Enable RepAgent in this order:<ul style="list-style-type: none"><li>• <code>sp_addserver</code> – identifies the Adaptive Server for RepAgent. You need to do this only once.</li><li>• <code>sp_configure 'enable rep agent threads'</code> – enables the data server for RepAgent. You need to do this only once.</li><li>• <code>sp_config_rep_agent</code> – enables the database for RepAgent.</li></ul></li></ul> <p>Refer to the <i>Adaptive Server Enterprise Reference Manual</i> for more information about <code>sp_addserver</code>.</p> |
| Permissions | <p><code>sp_configure</code> requires “sa” or “sso” permission to modify configuration parameters.</p> <p>Anyone can execute <code>sp_configure</code> to display information about parameters and their values.</p>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| See also    | <code>sp_config_rep_agent</code> and more information about <code>sp_configure</code> in the <i>Adaptive Server Enterprise Reference Manual</i> .                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |

## sp\_config\_rep\_agent

**Description** Changes or displays the configuration parameters for the RepAgent thread for an Adaptive Server database.

**Syntax** `sp_config_rep_agent [dbname  
[, {'enable', 'repserver_name',  
'repserver_username', 'repserver_password'} |  
'disable', 'preserve secondary truncpt'] |  
'rs servername', 'repserver_name' |  
'rs username', 'repserver_username' |  
'rs password', 'repserver_password' |  
'scan batch size', 'no_of_qualifying_log_records'] |  
'scan timeout', 'scan_timeout_in_seconds'] |  
'retry timeout', 'retry_timeout_in_seconds'] |  
'skip ltl errors', 'true' | 'false' |  
'batch ltl', 'true' | 'false'] |  
'send warm standby xacts', 'true' | 'false' |  
'send buffer size', '2K' | '4K' | '8K' | '16K'] |  
'connect dataserver', 'connect_dataserver_name'] |  
'connect database', 'connect_database_name'] |  
'send maint xacts to replicate', 'true' | 'false'] |  
'send structured oqids', 'true' | 'false' |  
'short ltl keywords', 'true' | 'false'] |  
'security mechanism', 'mechanism_name'] |  
'unified login', 'true' | 'false'] |  
'mutual authentication', 'true' | 'false'] |  
'msg confidentiality', 'true' | 'false'] |  
'msg integrity', 'true' | 'false'] |  
'msg replay detection', 'true' | 'false'] |  
'msg origin check', 'true' | 'false'] |  
'msg out-of-sequence check', 'true' | 'false'] |  
'skip unsupported features', 'true' | 'false'] |  
'schema cache growth factor', 'growth_factor_value'] |  
'ha failover', 'true' | 'false'] |  
'data limits filter mode', 'off' | 'stop' | 'skip' | 'truncate'] |  
'priority', 'priority_value'] |  
'startup delay', 'delay_value'] |  
'net password encryption', 'true' | 'false'] |  
'cluster instance name', 'coordinator' | 'instance_name'] |  
'bind to engine', 'engine_number'] |  
'ltl batch size', 'ltl_batch_size']]`

**Parameters** *dbname*  
The name of the database for which you want to configure RepAgent.

**enable**

Marks the database as using RepAgent and sets the secondary truncation point to valid.

This command encodes the Replication Server password and inserts the Replication Server name, Replication Server user, and encoded password into the sysattributes table of the specified database.

*repserver\_name*

The name of the Replication Server to which RepAgent connects and transfers log transactions.

*repserver\_username*

The user name that RepAgent thread uses to connect to Replication Server.

*repserver\_password*

The password that RepAgent uses to connect to Replication Server.

If network-based security is enabled and you want to establish unified login, you must specify NULL for *repserver\_password* when enabling RepAgent at the database.

**rs servername [, *repserver\_name*]**

The new or existing name of the Replication Server to which RepAgent connects and transfers log transactions.

**rs username[, *repserver\_username*]**

The new or existing user name that RepAgent thread uses to connect to Replication Server.

**rs password[, *repserver\_password*]**

The new or existing password that RepAgent uses to connect to Replication Server.

**disable**

Unmarks the database as using RepAgent. Use `preserve secondary truncpt` to retain the secondary truncation point. The default sets the secondary truncation point to IGNORE; that is, it disables it.

Use `disable` only when downgrading the Replication Server to an earlier release or changing the primary database to another status. This command truncates all RepAgent entries in the sysattributes table.

**scan batch size[, '*no\_of\_qualifying\_records*']**

Specifies the maximum number of log records to send to Replication Server in each batch. When the maximum number of records is met, RepAgent asks Replication Server for a new secondary truncation point. The default is 1000 records.

`scan timeout[, 'scan_timeout_in_seconds']`

Specifies the number of seconds that RepAgent sleeps once it has scanned and processed all records in the transaction log and Replication Server has not yet acknowledged previously sent records by sending a new secondary truncation point. RepAgent again queries Replication Server for a secondary truncation point after scan timeout seconds. The default is 15 seconds.

RepAgent continues to query Replication Server until Replication Server acknowledges previously sent records either by sending a new secondary truncation point or extending the transaction log.

If Replication Server has acknowledged all records and no new transaction records have arrived at the log, RepAgent sleeps until the transaction log is extended.

`retry timeout[, 'retry_timeout_in_seconds']`

Specifies the number of seconds RepAgent sleeps before attempting to reconnect to Replication Server after a retryable error or when Replication Server is down. The default is 60 seconds.

`skip ltl errors`

Specifies whether RepAgent ignores errors in LTL commands. This option is normally used in recovery mode. When set to “true,” RepAgent logs and then skips errors returned by the Replication Server for distribute commands. When set to “false,” RepAgent shuts down when these errors occur. The default is false.

`batch ltl`

Specifies whether RepAgent sends LTL commands to Replication Server in batches or one command at a time. When set to “true,” the commands are sent in batches. The default is false.

`send warm standby xacts`

Specifies whether RepAgent sends maintenance user transactions, schema changes, and system transactions to the warm standby database. This option should be used only with the RepAgent for the current active database in a warm standby configuration. The default is false.

`send buffer size[, '2K', '4K', '8K', '16K']`

Controls the size of the send buffer that RepAgent uses to communicate with Replication Server. Increasing the size of the send buffer reduces the number of times RepAgent communicates with Replication Server, but increases the amount of memory used.

The default value is 2K.

connect dataserver[, '*connect\_dataserver\_name*']

Specifies the name of the data server RepAgent uses when connecting to Replication Server in recovery mode. This is the data server name RepAgent uses for the connect source command; it is normally the data server for the primary database.

connect database[, '*connect\_database\_name*']

Specifies the name of the temporary database RepAgent uses when connecting to Replication Server in recovery mode. This is the database name RepAgent uses for the connect source command; it is normally the primary database.

send maint xacts to replicate

Specifies whether RepAgent should send records from the maintenance user to the Replication Server for distribution to subscribing sites. The default is "false."

send structured oqids

Specifies whether RepAgent sends origin queue IDs (OQIDs) as structured tokens, which saves space in the LTL and thus improves throughput, or as binary strings. The default value is "false."

short ltl keywords

Specifies whether RepAgent sends an abbreviated form of LTL to Replication Server, requiring less space and reducing the amount of data sent. The default value is "false."

security mechanism [, '*mechanism\_name*']

Specifies the network-based security mechanism RepAgent uses to connect to Replication Server.

unified login

When a network-based security system is enabled, specifies whether RepAgent seeks to connect to other servers with a security credential or password. The default is "false."

mutual authentication

Specifies whether RepAgent should require mutual authentication checks when connecting to Replication Server. The default is "false." This option is not implemented.

msg confidentiality

Specifies whether to encrypt all messages sent to Replication Server. The default is "false."

**msg integrity**

Specifies whether all messages exchanged with Replication Server should be checked for tampering. The default is “false.”

**msg replay detection**

Specifies whether messages received from Replication Server should be checked to make sure they have not been intercepted and replayed. The default is “false.”

**msg origin check**

Specifies whether to check the source of each message received from Replication Server. The default is “false.”

**msg out-of-sequence check**

Specifies whether to check the sequence of messages received from Replication Server. The default is “false.”

**skip unsupported features**

Instructs RepAgent to skip log records for Adaptive Server features unsupported by the Replication Server. This option is normally used if Replication Server is a lower version than Adaptive Server. The default is “false.”

**schema cache growth factor[, 'growth\_factor\_value']**

Controls the duration of time table or stored procedure schema can reside in the RepAgent schema cache before expiring. Larger values mean a longer duration and require more memory. Range is 1 to 10. The default is 1.

**ha failover**

Specifies whether, when Sybase Failover has been installed, RepAgent automatically starts after server failover. The default is “true.”

data limits filter mode[, 'off' | 'stop' | 'skip' | 'truncate']

Specifies how RepAgent handles log records containing new, wider columns and parameters, or larger column and parameter counts, before attempting to send them to Replication Server.

- off – RepAgent allows all log records to pass through.
- stop – RepAgent shuts down if it encounters log records containing wide data.
- skip – RepAgent skips log records containing wide data and posts a message to the error log.
- truncate – RepAgent truncates wide data to the maximum the Replication Server can handle.

---

**Warning!** Sybase recommends that you do not use the `data_limits_filter_mode`, `off` setting with Replication Server version 12.1 or earlier as this may cause RepAgent to skip or truncate wide data, or to stop.

---

The default value of `data limits filter mode` depends on the Replication Server version number. For Replication Server versions 12.1 and earlier, the default value is “stop.” For Replication Server versions 12.5 and later, the default value is “off.”

priority[, 'priority\_value']

Sets relative priority values for individual RepAgents. The value of priority ranges from 0 to 7, where a value of 0 indicates highest priority. The default value is 5.

---

**Note** Sybase recommends that you do not set the value of priority to 0, as it may negatively impact performance.

---

startup delay[, 'delay\_value']

This delays the automatic start-up of RepAgent by a specified duration to allow Replication Server to be running before RepAgent attempts to connect to Replication Server. By default, RepAgent starts without any delay during automatic start-up. Setting a value in seconds results in a delay in RepAgent start-up by the specified number of seconds. Default: 0 (zero) seconds.

net password encryption

Specifies whether connections with a remote server are to be initiated with a client-side password encryption handshake or with the usual unencrypted password handshake sequence. Default: 'true'.

cluster instance name[, 'coordinator' | '*instance\_name*']

Controls the instance where RepAgent is started. By default, RepAgent starts at the instance with the coordinator role. However, you can configure RepAgent to start at any declared instance in the cluster.

bind to engine[, *engine\_number*]

Restricts the RepAgent execution to the engine number specified. You can improve the RepAgent performance by running RepAgent on a dedicated or less utilized engine. The value of *engine\_number* ranges from -1 to (max online engines - 1). Its default is -1, which means RepAgent can execute on any engine.

---

**Note** The bind to engine clause does not restrict other user tasks or system tasks from running on the specified engine number.

---

ltl batch size[, *ltl\_batch\_size*]

Sets the maximum size, in bytes, of LTL data that a RepAgent can send to the Replication Server in a batch. The value of *ltl\_batch\_size* ranges from 16,384 to 2,147,483,647 bytes. Its default value is 16,384 bytes.

You can improve RepAgent performance by increasing the LTL batch size to a bigger number. At the end of each LTL batch, RepAgent checks for errors in the previous batch. Increasing the LTL batch size, decreases the number of times RepAgent checks for LTL errors.

## Examples

**Example 1** Enables RepAgent for the pubs2 database. RepAgent connects to repsvr1 with repusr1 and password reppwd1:

```
sp_config_rep_agent pubs2, 'enable', 'repub1',
'repusr1', 'reppwd1'
```

**Example 2** Displays configuration information for the pubs2 database:

```
sp_config_rep_agent pubs2
```

| Parameter Name   | Default | Config Value | Run Value |
|------------------|---------|--------------|-----------|
| -----            | -----   | -----        | -----     |
| priority         | 5       | 5            | 5         |
| trace flags      | 0       | 0            | 0         |
| scan timeout     | 15      | 15           | 15        |
| retry timeout    | 60      | 60           | 60        |
| rs username      | n/a     | rs1_user     | rs1_user  |
| batch ltl        | true    | true         | true      |
| rs servername    | n/a     | rs1          | rs1       |
| send buffer size | 2k      | 4k           | 4k        |
| trace log file   | n/a     | n/a          | n/a       |

|                               |             |             |             |
|-------------------------------|-------------|-------------|-------------|
| connect database              | n/a         | n/a         | pdb1        |
| connect dataserver            | n/a         | n/a         | pds1        |
| scan batch size               | 1000        | 1000        | 1000        |
| security mechanism            | n/a         | n/a         | n/a         |
| msg integrity                 | false       | false       | false       |
| unified login                 | false       | false       | false       |
| schema cache growth factor    | 1           | 1           | 1           |
| skip ltl errors               | false       | false       | false       |
| msg origin check              | false       | false       | false       |
| short ltl keywords            | false       | false       | false       |
| msg confidentiality           | false       | false       | false       |
| data limits filter mode       | stop        | stop        | stop        |
| msg replay detection          | false       | false       | false       |
| mutual authentication         | false       | false       | false       |
| send structured oqids         | false       | false       | false       |
| send warm standby xacts       | false       | false       | false       |
| msg out-of-sequence check     | false       | false       | false       |
| skip unsupported features     | false       | false       | false       |
| send maint xacts to replicate | false       | false       | false       |
| net password encryption       | true        | true        | true        |
| startup delay                 | 0           | 5           | 5           |
| cluster instance name         | coordinator | coordinator | coordinator |
| bind to engine                | -1          | 2           | 2           |
| ltl batch size                | 16384       | 16384       | 16384       |

**Example 3** Displays values for a specific parameter.

```
sp_config_rep_agent pubs2, 'scan batch size'
```

| Parameter Name  | Default | Config Value | Run Value |
|-----------------|---------|--------------|-----------|
| -----           | -----   | -----        | -----     |
| scan batch size | 1000    | 1000         | 1000      |

**Example 4** Sets scan\_timeout to 60 seconds for the pubs2 database:

```
sp_config_rep_agent pubs2, 'scan timeout', '60'
```

**Example 5** Configures RepAgent to wait 50 seconds before starting:

```
sp_config_rep_agent pubs2, 'startup delay', '50'
```

**Example 6** Starts a disabled RepAgent on ASE1:

```
1> sp_config_rep_agent pdb,
 'cluster instance name','ASE1'
2> go
```

| Parameter Name | Default | Config Value | Run Value |
|----------------|---------|--------------|-----------|
|----------------|---------|--------------|-----------|

|                       |             |       |       |
|-----------------------|-------------|-------|-------|
| -----                 | -----       | ----- | ----- |
| cluster instance name | coordinator | ASE1  | ASE1  |

## Usage

- Use `sp_config_rep_agent` to configure RepAgent for Adaptive Server databases.
- Enable RepAgent in this way:
  - `sp_addserver` – identifies the Adaptive Server for RepAgent. You need to do this only once per screen.
  - `sp_configure 'enable rep agent thread'` – configures the data server for RepAgent. You need to do this only once per screen.
  - `sp_config_rep_agent` – configures the database for RepAgent.

Refer to the *Adaptive Server Enterprise Reference Manual* for more information about `sp_addserver`.

- After you configure the parameters using `sp_config_rep_agent`, you must restart RepAgent using `sp_start_rep_agent` for the new parameters to take effect.
- If you execute `sp_config_rep_agent` without parameters, Adaptive Server displays the default, configured, and runtime values for all databases that are enabled for RepAgent.

If you only enter `dbname`, Adaptive Server displays the default, configured, and runtime values for the specified database.

- Properties specified by `sp_config_rep_agent` are stored in the `sysattributes` table of the database and have an attribute class of RA.
- Use `sp_config_rep_agent` to set the RepAgent configuration parameters after you have enabled RepAgent at the data server using `sp_configure`.
- `repserver_user` must have connect source permission.

## Configuring network-based security

**Note** Network-based security for RepAgent is enabled with `sp_configure` at the Adaptive Server. See the *Adaptive Server Enterprise System Administration Guide* for more information.

- A security mechanism may not support all security properties. Verify the properties of a security mechanism by executing `admin security_property` at the Replication Server. For more information, see `admin security_property` on page 70.

- The security mechanism enabled for the RepAgent must be the same as that enabled for the Replication Server. Security settings at the RepAgent and the Replication Server must be compatible.

| If RepAgent setting is | Setting at Replication Server can be                                                      |
|------------------------|-------------------------------------------------------------------------------------------|
| “true”                 | <ul style="list-style-type: none"><li>• “required”, or</li><li>• “not required”</li></ul> |
| “false”                | “not required”                                                                            |

- If unified\_login is “true,” you must specify the rs\_password parameter as NULL when RepAgent is enabled at the database.
- If you specify one or more security settings, but do not specify a security mechanism, Adaptive Server initializes the default mechanism, the first entry in the SECURITY section in \$SYBASE/\$SYBASE\_ASE/config/libtcl.cfg.

Permissions

sp\_configure\_rep\_agent requires “sa” or “dbo” permission or replication\_role.

See also

sp\_configure 'enable rep agent threads', sp\_help\_rep\_agent, sp\_start\_rep\_agent, sp\_stop\_rep\_agent

## sp\_help\_rep\_agent

|             |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
|-------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Description | Displays static and dynamic information about a RepAgent thread.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| Syntax      | <code>sp_help_rep_agent [dbname[, 'recovery'   'config'   'process'   'scan'   'security'   'all']]</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
| Parameters  | <p><i>dbname</i></p> <p>The name of the database with the RepAgent for which you want information.</p> <p><i>recovery</i></p> <p>Displays recovery status information about the RepAgent.</p> <p><i>config</i></p> <p>Displays configuration information about the RepAgent.</p> <p><i>process</i></p> <p>Displays information about the RepAgent process.</p> <p><i>scan</i></p> <p>Displays log-scanning information about the RepAgent.</p> <p><i>security</i></p> <p>Displays current settings of the network-based security mechanism.</p> <p><i>all</i></p> <p>Displays all the preceding information for the RepAgent connected to the specified database.</p> |

**Examples**                      **Example 1** Displays recovery information.

```
sp_help_rep_agent pubs2, 'recovery'
```

| Rep Agent Recovery Status |            |            |          |               |             |
|---------------------------|------------|------------|----------|---------------|-------------|
| dbname                    | connect    | connect    | status   | rs servername | rs username |
|                           |            | dataserver | database |               |             |
| -----                     | -----      | -----      | -----    | -----         | -----       |
| pubs2                     | sqlserver1 | pubs2      | scanning | repsvr1       | repusr1     |

**Example 2** Displays process information.

```
sp_help_rep_agent pubs2, 'process'
```

| Rep Agent Process Status |       |              |             |            |
|--------------------------|-------|--------------|-------------|------------|
| dbname                   | spid  | sleep status | retry count | last error |
| -----                    | ----- | -----        | -----       | -----      |
| pubs2                    | 40    | not sleeping | 0           | 0          |

**Example 3** Displays scanning information.

```
sp_help_rep_agent pubs2, 'scan'

Replication Agent Scan status

dbname start marker end marker current marker log recs scanned oldest trans.

pubs2 (472675,13) (278622,0) (265736,16) 890 (472675,13)
```

**Usage**

- Use sp\_help\_rep\_agent with RepAgent-enabled databases.
- If you execute sp\_help\_rep\_agent without parameters, Adaptive Server displays information about all databases for which RepAgent is enabled.
- Table 5-2 describes the output for the sp\_help\_rep\_agent 'recovery' system procedure.

**Table 5-2: Column descriptions for sp\_help\_rep\_agent 'recovery' output**

| Column             | Description                                                                                                                                                                                                                                                                                                                                                                                                           |
|--------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| dbname             | The name of the database containing archived logs whose data is transferred to the Replication Server during recovery.                                                                                                                                                                                                                                                                                                |
| connect dataserver | The name of the original data server with the database whose transaction logs were transferred to Replication Server in normal mode. This information is included in the LTL connect source command delivered to Replication Server.                                                                                                                                                                                  |
| connect database   | The name of the original database whose transaction logs were transferred to Replication Server in normal mode. This information is included in the LTL connect source command delivered to Replication Server.                                                                                                                                                                                                       |
| status             | Indicates RepAgent activity. Status values are: <ul style="list-style-type: none"><li>• “not running” – RepAgent is not running.</li><li>• “not active” – RepAgent is not in recovery mode.</li><li>• “initial” – RepAgent is initializing in recovery mode.</li><li>• “end of log” – RepAgent is in recovery mode and has reached the end of the transaction log.</li><li>• “unknown” – none of the above.</li></ul> |
| rs servername      | The name of the Replication Server to which the RepAgent is transferring information. Use this option to override the <i>sysattributes</i> setting.                                                                                                                                                                                                                                                                   |
| rs username        | The login name RepAgent uses to log in to the Replication Server. Use this option to override the <i>sysattributes</i> setting.                                                                                                                                                                                                                                                                                       |

- Table 5-3 describes the output for the sp\_help\_rep\_agent 'config' system procedure.

**Table 5-3: Column descriptions for `sp_help_rep_agent 'config'` output**

| Column                  | Description                                                                                                                                                                                                                                                                                                                            |
|-------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| dbname                  | The name of the database with the data RepAgent is transferring to the Replication Server.                                                                                                                                                                                                                                             |
| auto start              | Contains “true” if the RepAgent starts automatically during server start-up. Otherwise contains “false.”                                                                                                                                                                                                                               |
| rs servername           | The name of the Replication Server to which RepAgent is transferring log transactions.                                                                                                                                                                                                                                                 |
| rs username             | The login name the RepAgent thread uses to log in to the Replication Server. The login name must have been granted connect source permission in the Replication Server.                                                                                                                                                                |
| scan batch size         | The maximum number of log records sent to Replication Server in each batch. The default is 1000.                                                                                                                                                                                                                                       |
| scan timeout            | The number of seconds that RepAgent sleeps when it has scanned and processed all records in the transaction log and Replication Server has not yet acknowledged previously sent records by sending a secondary truncation point.<br>The default is 15 seconds.                                                                         |
| retry timeout           | The number of seconds RepAgent sleeps before attempting to reconnect to Replication Server after a retryable error or when Replication Server is down. The default is 60 seconds.                                                                                                                                                      |
| skip ltl errors         | Contains “true” if RepAgent ignores errors in LTL commands. Contains “false” if RepAgent shuts down when these errors occur. skip ltl errors is normally set to “true” in recovery mode.<br>The default is “false.”                                                                                                                    |
| batch ltl               | Contains “true” if RepAgent batches LTL commands and sends them to Replication Server. Contains “false” if LTL commands are sent to Replication Server as soon as they are formatted.<br>The default is “false.”                                                                                                                       |
| send warm standby xacts | Contains “true” if RepAgent submits schema, system xacts, and all updates, including updates made by the maintenance user, to the Replication Server for application to the standby database in a warm standby application. Contains “false” if RepAgent is not submitting updates to the standby database.<br>The default is “false.” |
| connect dataserver      | The name of the data server RepAgent connects to Replication Server as when running in recovery mode. If RepAgent is not running in recovery mode, contains the name of the data server of the <i>dbname</i> database.                                                                                                                 |
| connect database        | The name of the database RepAgent connects to Replication Server as when running in recovery mode. If RepAgent is not running in recovery mode, contains the <i>dbname</i> database name.                                                                                                                                              |

| Column                           | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
|----------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| send maint commands to replicate | Contains “true” if RepAgent sends records from the maintenance user to replicate databases. Contains “false” if RepAgent does not send records from the maintenance user to replicate databases.<br>The default is “false.”                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| ha failover                      | Specifies whether, when Sybase Failover has been installed, RepAgent starts automatically after server failover.<br>The default is “true.”                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| skip unsupported features        | Instructs RepAgent to skip log records for Adaptive Server features unsupported by the Replication Server. This option is normally used if Replication Server is an earlier version than Adaptive Server.<br>The default is “false.”                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| short ltl keywords               | Specifies whether RepAgent sends an abbreviated form of LTL to Replication Server, requiring less space and reducing the amount of data sent.<br>The default value is “false.”                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| send buffer size                 | Controls the size of the send buffer that RepAgent uses to communicate with Replication Server. Increasing the size of the send buffer reduces the number of times RepAgent communicates with Replication Server, but increases the amount of memory used. Values are “2K,” “4K,” “8K,” and “16K.”<br>The default value is “2K.”                                                                                                                                                                                                                                                                                                                                                                                                                        |
| priority                         | Sets relative priority values for individual RepAgents. The value of priority ranges from 0 to 7, where a value of 0 indicates highest priority. The default value is 5.<br><br><b>Note</b> Sybase recommends that you do not set the value of priority to 0.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| send structured oqids            | Specifies whether RepAgent sends origin queue IDs (OQIDs) as structured tokens, which saves space in the LTL and thus improves throughput, or as binary strings.<br>The default value is “false.”                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| data limits filter mode          | Specifies how RepAgent handles log records containing new, wider columns and parameters, or larger column and parameter counts, before attempting to send them to Replication Server. <ul style="list-style-type: none"><li>• off – RepAgent allows all log records to pass through.</li><li>• stop – RepAgent shuts down if it encounters log records containing wide data.</li><li>• skip – RepAgent skips log records containing wide data and posts a message to the error log.</li></ul> The default value of data_limits_filter_mode depends on the Replication Server version number. For Replication Server versions 12.1 and earlier, the default value is “stop.” For Replication Server versions 12.5 and later, the default value is “off.” |

| Column                     | Description                                                                                                                                                                                                                                            |
|----------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| schema cache growth factor | Controls the duration of time table or stored procedure schema can reside in the RepAgent schema cache before expiring. Larger values mean a longer duration and require more memory. Range is 1 to 10.<br>The default is 1.                           |
| startup delay              | The number of seconds that the RepAgent start-up is delayed. The default is 0.                                                                                                                                                                         |
| cluster instance name      | The name of the cluster instance where the RepAgent is started. The default value is 'coordinator'.                                                                                                                                                    |
| bind to engine             | The engine number where RepAgent is specified to execute on. Range is -1 to (max online engines - 1), where max online engines is an Adaptive Server configuration parameter. The default value is -1, which means RepAgent can execute on any engine. |
| ltl batch size             | The maximum size, in bytes, of LTL data that RepAgent can send to the Replication Server for a given batch. The minimum and default value is 16,384 bytes. The maximum value is 2,147,483,647 bytes.                                                   |

- Table 5-4 describes the output for the `sp_help_rep_agent 'process'` system procedure.

**Table 5-4: Column descriptions for `sp_help_rep_agent 'process'` output**

| Column       | Description                                                                                                                                                                                                                                                                                                                                                                                                                                  |
|--------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| dbname       | The name of the Replication Server to which RepAgent is transferring log transactions.                                                                                                                                                                                                                                                                                                                                                       |
| sleep status | Sleep status values are: <ul style="list-style-type: none"> <li>• “waiting for rewrite” – RepAgent is waiting for a two-phase commit transaction to commit.</li> <li>• “end of log” – RepAgent is at the end of the log, waiting for it to be extended.</li> <li>• “connect retry” – RepAgent is waiting before attempting a connection to Replication Server.</li> <li>• “not sleeping” – none of the above. RepAgent is active.</li> </ul> |
| retry count  | The number of times RepAgent has unsuccessfully attempted to connect to Replication Server since the last successful connection.                                                                                                                                                                                                                                                                                                             |
| spid         | The PID in the dataserver.                                                                                                                                                                                                                                                                                                                                                                                                                   |
| last error   | The error number of the last Replication Server or connection error.                                                                                                                                                                                                                                                                                                                                                                         |

- Table 5-5 describes the output for the `sp_help_rep_agent 'scan'` system procedure.

**Table 5-5: Column descriptions for sp\_help\_rep\_agent 'scan' output**

| Column             | Description                                                                            |
|--------------------|----------------------------------------------------------------------------------------|
| dbname             | The name of the Replication Server to which RepAgent is transferring log transactions. |
| start marker       | Identifies the first record scanned in current batch.                                  |
| end marker         | Identifies the last record to be scanned in current batch.                             |
| current marker     | Identifies the record currently being scanned.                                         |
| log recs scanned   | The number of log records RepAgent has scanned in the current batch.                   |
| oldest transaction | Identifies the oldest transaction in the batch currently being scanned.                |

- Table 5-6 describes output for the sp\_help\_rep\_agent 'security' stored procedure.

**Table 5-6: Column descriptions for sp\_help\_rep\_agent 'security' output**

| Column                  | Description                                                                                                                                           |
|-------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------|
| dbname                  | The name of the Replication Server to which RepAgent is transferring log transactions.                                                                |
| security mechanism      | The name of the enabled security mechanism.                                                                                                           |
| unified login           | Specifies whether RepAgent seeks to connect to Replication Server with a credential ("true") or a password ("false"). The default is "false."         |
| mutual authentication   | Specifies whether RepAgent uses mutual authentication checks when connection to Replication Server. The default is "false."                           |
| msg confidentiality     | Specifies whether RepAgent uses message encryption on all data sent to Replication Server. The default is "false."                                    |
| msg integrity           | Specifies whether RepAgent uses message integrity checks on all data exchanged with Replication Server. The default is "false."                       |
| msg replay detection    | Specifies whether RepAgent checks to detect whether data has been captured and replayed by an intruder. The default is "false."                       |
| msg origin check        | Specifies whether RepAgent verifies the source of data sent from Replication Server. The default is "false."                                          |
| msg out-of-sequence     | Specifies whether RepAgent verifies that messages received from Replication Server are received in the order sent. The default is "false."            |
| net password encryption | Indicates whether or not the connection to a Replication Server is initiated with a client-side password encryption handshake. The default is "true". |

Permissions                      sp\_help\_rep\_agent requires "sa" or "dbo" permission or replication\_role.

See also                            sp\_config\_rep\_agent, sp\_start\_rep\_agent, sp\_stop\_rep\_agent

## sp\_reptostandby

|             |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
|-------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Description | Marks or unmarks database for replication to the standby database. Enables replication of supported schema changes and data changes to user tables.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| Syntax      | <code>sp_reptostandby <i>dbname</i> [, 'L1'   'all'   'none'] [, use_index]</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
| Parameters  | <p><i>dbname</i></p> <p>The name of the active database.</p> <p>L1</p> <p>Sets the schema replication feature set support level to the support level first introduced in Adaptive Server version 12.0. If you upgrade the Adaptive Server to a later version that implements a higher support level (that is, L2, L3, and so on) the support level will remain at the Adaptive Server version 12.0 support level. To date, only support level L1 has been implemented in Adaptive Server version 12.0 and later.</p> <p>all</p> <p>Sets the schema replication feature set support level to the highest support level implemented by the current Adaptive Server. If you upgrade the Adaptive Server to a later version, the highest support level implemented by the later version is enabled automatically.</p> <p>none</p> <p>Unmarks all database tables for replication and turns off data and schema replication to the standby database.</p> <hr/> <p><b>Note</b> If you turn replication off using <code>sp_reptostandby</code> with the <code>none</code> keyword, Adaptive Server locks all user tables in exclusive mode and writes log records for all tables that are unmarked for replication. This can be time-consuming if there are many user tables in the database.</p> <hr/> <p><i>use_index</i></p> <p>Marks the database to use an index for replication on text, unitext, image, or rawobjects columns.</p> |
| Examples    | <p><b>Example 1</b> Sets the replication status for pubs2 to all and creates a global index on the text and image pointers:</p> <pre>sp_reptostandby pubs2, 'all', 'use_index'</pre> <p><b>Example 2</b> Displays the SQL statement replication status at the database level:</p> <pre>1&gt; sp_reptostandby pubs2 2&gt; go</pre>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |

The replication status for database 'pubs2' is 'ALL'.  
The replication mode for database 'pubs2' is ' udis'.  
(return status = 0)

#### Usage

- Use `sp_reptostandby` with Adaptive Server version 11.5 or later databases. You also must enable RepAgent at the active and standby databases.
- Copies data manipulation language (DML) commands, supported data definition language (DDL) commands, and supported system procedures to the standby database.

The supported DDL commands are:

- alter encryption key
- alter key
- alter table
- create default
- create encryption key
- create function
- create index
- create key
- create plan
- create procedure
- create rule
- create schema
- create table
- create trigger
- create view
- drop default
- drop encryption key
- drop function
- drop index
- drop procedure
- drop rule

- drop table
- drop trigger
- drop view
- grant
- installjava
- remove java
- revoke

The supported system procedures are:

- sp\_addalias
- sp\_addgroup
- sp\_addmessage
- sp\_add\_qpgroup
- sp\_addtype
- sp\_adduser
- sp\_bindefault
- sp\_bindmsg
- sp\_bindrule
- sp\_cachestrategy
- sp\_changegroup
- sp\_chgattribute
- sp\_commonkey
- sp\_config\_rep\_agent
- sp\_drop\_all\_qplans
- sp\_drop\_qpgroup
- sp\_dropalias
- sp\_dropgroup
- sp\_dropkey
- sp\_dropmessage

- `sp_droptype`
- `sp_dropuser`
- `sp_encryption`
- `sp_foreignkey`
- `sp_import_qpgroup`
- `sp_primarykey`
- `sp_procxmode`
- `sp_recompile`
- `sp_rename`
- `sp_rename_qpgroup`
- `sp_setrepcol`
- `sp_setrepdefmode`
- `sp_setreplicate`
- `sp_setreptable`
- `sp_unbindefault`
- `sp_unbindmsg`
- `sp_unbindrule`
- If the database is the master database, the DDL commands and system procedures that are supported for replication in a user database are not supported for replication in the master database.

If the database is the master database, the supported DDL commands are:

- `alter role`
- `create role`
- `drop role`
- `grant role`
- `revoke role`

If the database is the master database, the supported system procedures are:

- `sp_addlogin`

- `sp_defaultdb`
- `sp_defaultlanguage`
- `sp_displaylevel`
- `sp_droplogin`
- `sp_locklogin`
- `sp_modifylogin`
- `sp_password`
- `sp_passwordpolicy`

---

**Note** `sp_passwordpolicy` is replicated for all options except for `allow password downgrade`.

---

- `sp_role`

If a DDL command or system procedure contains password information, the password information is sent through the replication environment using the ciphertext password value stored in source ASE system tables.

- `sp_reptostandby` marks the database for replication to the warm standby database. It does not enable replication to replicate databases.
- After `sp_reptostandby` has been executed and the warm standby enabled, you can selectively turn off replication for individual database tables by setting their replication status to `never`. You can use the `set replication` command to control replication of DDL and DML commands and procedures for the `isql` session. See `set replication` for more information.
- By default, `sp_reptostandby` marks text, untext, or image data as `replicate_if_changed`. You cannot change the status to `always_replicate` or `do_not_replicate`.
- If the warm standby application includes normal replication, text, untext, or image data columns may be treated as `always_replicate` or `replicate_if_changed`.
  - If text, untext, or image columns marked by `sp_setreptable` are specified `always_replicate` (the default), all text, untext, or image columns are treated as `always_replicate`.
  - If text, untext, or image columns are specified by `sp_setrepcol` as `do_not_replicate` or `replicate_if_changed`, all text, untext, or image columns are treated as `replicate_if_changed`.

- When the database contains one or more large tables holding text, untext, image, or rawobject columns, the internal process performed by sp\_reptostandby may take a long time. To speed up the process, you can use use\_index which creates a global nonclustered index for every text, untext, image, or rawobject column of tables not explicitly marked for replication.
- With use\_index, a shared-table lock is held while the nonclustered index is created.
- When you run sp\_reptostandby with the none option, and the database is initially marked to use indexes for replication, all those indexes created for replication are dropped.

#### Restrictions and requirements

- The standby database must be of the same or later release level than the active database. Both databases must have the same disk allocations, segment names, and roles. Refer to the *Adaptive Server Enterprise System Administration Guide* for details.
- Login information is not replicated to the standby database.
- Replication of commands or procedures containing the name of another database will fail if the named database does not exist in the standby server.
- Supported DDL commands, such as create table, may not contain local variables.
- Some commands that are not copied to the standby database:
  - select into and update statistics
  - Database or configuration options such as sp\_dboption and sp\_configure
- If the database is the master database:
  - User tables and user stored procedures are not replicated.
  - The target database cannot be materialized with dump or load. Use other methodologies, such as bcp, where the data can be manipulated to resolve inconsistencies.
  - Both the source ASE server and target ASE server must support the master database replication feature.

- Both the source ASE server and the target ASE server must have the same hardware architecture type (32-bit versions and 64-bit versions are compatible) and the same operating system (different versions are compatible).
- If the master database is replicated, the following system procedures must be executed in the master database:
  - `sp_addlogin`
  - `sp_defaultdb`
  - `sp_defaultlanguage`
  - `sp_displaylevel`
  - `sp_droplogin`
  - `sp_locklogin`
  - `sp_modifylogin`
- You cannot use `drop index` to manually drop indexes created for text, untext, image, or rawobject replication. You can use only the supported replication stored procedures `sp_reptostandby`, `sp_setreptable`, and `sp_setrepcol` to change the replication index status.

Permissions

`sp_reptostandby` requires “sa” or “dbo” permission or `replication_role`.

See also

`set replication`, `sp_setrepcol`, `sp_setreptable`, `sp_setreplicate`, `sp_setrepproc`

## sp\_setrepcol

|             |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
|-------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Description | Sets or displays the replication status for text, unitext, or image columns.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| Syntax      | <pre>sp_setrepcol <i>table_name</i> [, {<i>column_name</i>   null}<br/>                        [, {do_not_replicate   always_replicate   replicate_if_changed}]]<br/>                        [, use_index]</pre>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| Parameters  | <p><i>table_name</i></p> <p>The name of the replicated table. You must enable replication for the table using <code>sp_setreptable</code> before you execute <code>sp_setrepcol</code>.</p> <p><i>column_name</i></p> <p>The name of a text, unitext, or image column in the table. Specify null for the column name to set the replication status of all text, unitext, or image columns in the table.</p> <p><code>do_not_replicate</code></p> <p>Prevents Adaptive Server from logging replication information for the text, unitext, or image column. If the column has previously been marked to use an index for replication, setting <code>do_not_replicate</code> removes the index.</p> <p><code>always_replicate</code></p> <p>Causes Adaptive Server to log replication information for the text, unitext, or image column when any column in the row changes. This status adds overhead for replicating text, unitext, or image columns that do not change; however, it protects against data inconsistency from row migration or changes during non-atomic materialization.</p> <p><code>replicate_if_changed</code></p> <p>Causes Adaptive Server to log replication information for the text, unitext, or image column only when the text, unitext, or image column data changes. This status reduces overhead, but it may lead to data inconsistency from row migration or changes during non-atomic materialization.</p> <p><code>use_index</code></p> <p>Marks the column to use an index for replication on text, unitext, image, or rawobjects columns.</p> |
| Examples    | <p><b>Example 1</b> Displays the replication status for all text, unitext, or image columns in the <code>au_pix</code> table. <code>au_pix</code> must be marked for replication using <code>sp_setreptable</code>.</p> <pre>sp_setrepcol au_pix</pre> <p><b>Example 2</b> Displays the replication status for the <code>pic</code> column in the <code>au_pix</code> table. <code>pic</code> must be a text, unitext, or image datatype column.</p> <pre>sp_setrepcol au_pix, pic</pre>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |

**Example 3** Specifies that the pic column (image datatype) in the au\_pix table should have the replicate\_if\_changed status. (In this particular table in the pubs2 database, there are no other text, unitext, or image columns.)

```
sp_setrepcol au_pix, pic, replicate_if_changed
```

**Example 4** Specifies that all text, unitext, or image columns in the au\_pix table should have the replicate\_if\_changed status.

```
sp_setrepcol au_pix, null, replicate_if_changed
```

**Example 5** Marks the column t (text datatype) as replicate\_if\_changed and uses an index for replication:

```
sp_setrepcol t1, t, replicate_if_changed, use_index
```

#### Usage

- Use sp\_setrepcol to specify how text, unitext, or image columns are replicated after you have enabled replication for the table with sp\_setreptable.
- You can also execute sp\_setrepcol with a table name to display the replication status of all of the text, unitext, or image columns in the table, or with the table name and a text, unitext, or image column name to display the replication status of the specified column.
- Using the replicate\_if\_changed option reduces the overhead of replicating text, unitext, or image columns. However, the following restrictions and cautions apply:
  - If you specify the replicate\_if\_changed status for a column, any replication definition that includes the column must also have the replicate\_if\_changed status.
  - If you set the replication status of any column to replicate\_if\_changed, you cannot set autocorrection to “on” for any replication definition that includes the column.
  - If you use non-atomic subscription materialization and you have set the replicate\_if\_changed replication status for any text, unitext, or image columns, Replication Server displays a message in the error log file. This message warns you that the data may be inconsistent if an application modified the primary table during subscription materialization.
  - If your application allows rows to migrate into a subscription and you have set the replicate\_if\_changed replication status for any text, unitext, or imagecolumn, Replication Server displays a warning message in the error log when the row migrates into the subscription and the text or image data is missing.

If a text, unitext, or image column with the `replicate_if_changed` status was not changed in an update operation at the primary table and the update causes the row to migrate into a subscription, the inserted row at the replicate table will be missing the text, unitext, or image data. Run the `rs_subcmp` program to reconcile the data in the replicate and primary tables.

Row migration can occur when subscriptions have where clauses. Updating a column specified in the subscription where clause can cause a row to become valid for, or migrate into, the subscription.

When this happens, Replication Server must execute an insert in the replicate database. An insert requires values for all of the columns, including text, unitext, or image columns that did not change in the primary database.

- When tables are marked with `sp_reptostandby`, you cannot change the replication status of text, unitext, or image columns using `sp_setrepcol`; text, unitext, and image columns are always treated as `replicate_if_changed`.
- If the warm standby application includes normal replication and you have marked tables with `sp_reptostandby` and `sp_setreptable`, text, unitext, or image data columns may be treated as `always_replicate` or `replicate_if_changed`.
  - If text, unitext, or image columns marked by `sp_setreptable` are specified `always_replicate` (the default), all text, unitext, and image columns are treated as `always_replicate`.
  - If text, unitext, or image columns are specified by `sp_setrepcol` as `do_not_replicate` or `replicate_if_changed`, all text, unitext, or image columns are treated as `replicate_if_changed`.
- The order of the precedence on the index status is: column, table, database. If the table is marked to use indexes on text, unitext, image or rawobject columns, but you do not want to use indexes in one of the columns, the column status overrides the table status.
- You cannot use `drop index` to manually drop indexes created for text, unitext, image, or rawobject replication. You can use only the supported replication stored procedures `sp_reptostandby`, `sp_setreptable`, and `sp_setrepcol` to change the replication index status.

#### Permissions

`sp_setrepcol` requires “sa” or “dbo” permission or `replication_role`.

#### See also

`sp_reptostandby`, `sp_setreptable`, `sp_setreplicate`

## sp\_setrepldbmode

|             |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
|-------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Description | Enables or disables replication of SQL statements at the database-level and for one or more specific DML operation type.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| Syntax      | <pre>sp_setrepldbmode dbname [, "option [option [...]]" [, "on"   "off"] ['threshold', 'value']</pre> <p><i>option</i> ::= { U   D   I   S }</p>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
| Parameters  | <p><i>dbname</i></p> <p>The name of the database for which you want to enable SQL statement replication.</p> <p><i>option</i></p> <p>Any combination of these DML operations:</p> <ul style="list-style-type: none"> <li>• U – update</li> <li>• D – delete</li> <li>• I – insert select</li> <li>• S – select into</li> </ul> <p>When the database replication mode is set to any combination of UDIS the RepAgent sends both individual log records and the information needed by Replication Server to build the SQL statement.</p> <p><i>on</i></p> <p>Enables SQL replication of the DML operation specified.</p> <p><i>off</i></p> <p>Disables SQL statement replication at the database level for all types of DML operations, regardless of the operation specified in <i>option</i>.</p> <p><i>'threshold', 'value'</i></p> <p>Specifies the minimum number of rows that a replicated SQL statement must impact before SQL statement replication is activated. Reset <i>value</i> to '0' for the default threshold of 50 rows.</p> |

**Examples** **Example 1** Replicates delete and select into statements:

```
sp_setrepldbmode pdb, 'DS', 'on'
```

**Example 2** Displays the current SQL replication settings:

```
1> sp_setrepldbmode pdb1
2> go
The replication mode for database 'pdb1' is 'us'.
(return status = 0)
```

**Example 3** To disable replication of all SQL statements at database level, use:

```
sp_setrepdbmode pdb, 'D', 'off'
```

**Example 4** To set threshold value at 100 rows:

```
sp_setrepdbmode pubs2, 'threshold', '100'
go
```

**Example 5** This example shows how to set a different threshold at the database and table levels for the pubs2 database and table1 table:

- 1 Reset the threshold at the database-level to the default of 50 rows:

```
sp_setrepdbmode pubs2, 'threshold', '0'
go
```

- 2 Enable SQL statement replication of update, delete, insert, and select into operations for pubs2:

```
sp_setrepdbmode pubs2, 'udis', 'on'
go
```

- 3 Trigger SQL statement replication for table1 in pubs2 only when update, delete, insert, and select into operations execute on table1 and affect more than 1,000 rows:

```
sp_setrepdefmode table1, 'threshold', '1000'
go
```

**Example 6** This example shows how to define the threshold at the database-level for pubs2, and at the same time define different operations for tables, such as table1 and table2:

- 1 Set the threshold at the database-level to trigger SQL statement replication when a data manipulation language (DML) statement affects more than 100 rows:

```
sp_setrepdbmode pubs2, 'threshold', '100'
go
```

- 2 Define a different set of operations for two specific tables, where you want the operations replicated using SQL statement replication. Update, delete, and insert operations are for table1 and delete operations are for table2:

```
sp_setrepdefmode table1, 'udi', 'on'
go
sp_setrepdefmode table2, 'd', 'on'
go
```

In this example, when a delete operation executes against table2 or any DML on table1 executes, the threshold of 100 rows that you defined at the database-level triggers SQL statement replication when reached.

**Usage**

- You can set SQL statement replication at the database level only when the database has been marked for replication by setting `sp_reptostandby` to ALL or L1.
- The default threshold is 50 rows, which means that Adaptive Server uses SQL statement replication if the DML statement affects at least 51 rows. To use the default threshold, set the threshold parameter to 0. The threshold parameter range is 0 to 10,000.
- You can configure replication at the database-level and set the threshold for SQL statement replication at the database-level at the same time. For example:

```
sp_reptostandby pubs2, 'none'
go
sp_setrepdbmode pubs2, 'threshold'
go
```

However, you cannot configure replication at the database-level and define operations also at the database-level as SQL statement replication at the database-level requires that the entire database be replicated and you cannot replicate the operations only. For example, you cannot execute:

```
sp_reptostandby pubs2, 'none'
go
sp_setrepdbmode pubs2, 'udis', 'on'
go
```

- The threshold set at the session-level overrides the threshold at the table-level and database-level, and the threshold set for any table overrides the threshold set at the database-level.

**See also**

`set repmode`, `sp_setrepdefmode`, `set repthreshold`

## sp\_setrepdefmode

|             |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
|-------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Description | Changes or displays the owner status of tables marked for replication, and enables or disables table-level SQL statement replication for a specific DML operation.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| Syntax      | <pre>sp_setrepdefmode <i>table_name</i> [, 'owner_on'   'owner_off'  <br/>    'SQLDML_option' [SQLDML_option [ ...]] [, 'on'   'off'   'never' ]  <br/>    'threshold', 'value']<br/><br/>SQLDML_option ::= { U   D   I }</pre>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
| Parameters  | <p><i>table_name</i></p> <p>The name of a table in the current database that has been marked for replication with sp_setreptable.</p> <p><i>owner_on</i></p> <p>Changes the owner status of the table so the table name and owner name are considered when the table is marked for replication. Enables replication of multiple tables of the same name with different owners.</p> <p><i>owner_off</i></p> <p>Changes the owner status of the table so that only the table name is considered when the table is marked for replication.</p> <p><i>SQLDML_option</i></p> <p>Any of these DML operations:</p> <ul style="list-style-type: none"><li>• U – update</li><li>• D – delete</li><li>• I – insert select</li></ul> <p>When the table replication mode is set to any combination of UDI the RepAgent sends additional information to enable SQL statement replication for the specified DML operation.</p> <p><i>on</i></p> <p>Enables SQL replication of the DML operation specified.</p> <p><i>off</i></p> <p>Removes the table-level replication settings of SQL statements, whether or not the statements are specified in <i>option</i>; the database-level replication settings are followed.</p> <p><i>never</i></p> <p>Disables SQL statement replication, regardless of the database setting, and regardless of whether the UDI parameter is specified.</p> |

'threshold', 'value'

Specifies the minimum number of rows that a replicated SQL statement must impact before SQL statement replication is activated.

#### Examples

**Example 1** Enables SQL statement replication for update, delete and insert select operations on table t:

```
1> sp_setrepdefmode t, 'UDI', 'on'
2> go
```

**Example 2** Sets the threshold to 10. Adaptive Server will use SQL replication on table t if the DML statement affects at least 11 rows:

```
sp_setrepdefmode t, 'threshold', '10'
```

**Example 3** Displays the SQL replication settings and the owner status of table rs\_ticket\_history:

```
1> sp_setrepdefmode rs_ticket_history, 'udi'
2> go

The replication status for 'rs_ticket_history' is
currently owner_off, 'udi'.
The replication threshold for table 'rs_ticket_history'
is '0'.
(return status = 0)
```

**Example 4** Sets the threshold to the default value:

```
sp_setrepdbmode t, 'threshold', '0'
```

#### Usage

- Use sp\_setrepdefmode with RepAgent-enabled Adaptive Server databases.
- If sp\_setrepdefmode is executed with the table name only, it displays the SQL replication settings and owner status of the table.
- Use sp\_setrepdefmode to change the mode of the table. You cannot change the owner mode of tables with sp\_setreptable.
- If the owner\_off option is supplied and the current mode of the table is “owner on,” sp\_setrepdefmode checks that the table name is unique among all replicated tables in owner off mode. If the name is unique, sp\_setrepdefmode changes the table mode to owner off. If the name is not unique, the procedure fails.
- The default threshold is 50 rows, which means that Adaptive Server uses SQL statement replication if the DML statement affects at least 51 rows. To use the default threshold, set the threshold parameter to 0. The threshold parameter range is 0 to 10,000.

|             |                                                                         |
|-------------|-------------------------------------------------------------------------|
| Permissions | sp_setrepdefmode requires “sa” or “dbo” permission or replication_role. |
| See also    | set repmode, sp_setreptable, sp_setrepdbmode, set repthreshold          |

## sp\_setreplicate

|             |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
|-------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Description | <p>This system procedure enables or disables replication for an Adaptive Server table or stored procedure. It also displays the current replication status of a table or stored procedure.</p> <hr/> <p><b>Note</b> This system procedure is still supported, but its capabilities have been incorporated into the system procedures <code>sp_setreptable</code> and <code>sp_setrepproc</code>. <code>sp_setreplicate</code> sets the replication status of columns with text, unitext, or image datatype to <code>do_not_replicate</code>. To replicate text, unitext, or image columns, use the <code>sp_setreptable</code> system procedure instead of <code>sp_setreplicate</code>. To specify individual text, unitext, or image columns for replication, use <code>sp_setrepcol</code> after using <code>sp_setreplicate</code> or <code>sp_setreptable</code>.</p> <hr/> |
| Syntax      | <code>sp_setreplicate [object_name [, {'true'   'false'}]]</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| Parameters  | <p><i>object_name</i><br/>is the name of a table or stored procedure in the current database.</p> <p><code>true</code><br/>enables replication for the table or stored procedure.</p> <p><code>false</code><br/>disables replication for the table or stored procedure.</p>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| Examples    | <p><b>Example 1</b> Displays the replication status for all of the tables and stored procedures in the current database.</p> <pre>sp_setreplicate</pre> <p><b>Example 2</b> Displays the replication status for the publishers table.</p> <pre>sp_setreplicate publishers</pre> <p><b>Example 3</b> Enables replication for the publishers table.</p> <pre>sp_setreplicate publishers, 'true'</pre>                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| Usage       | <ul style="list-style-type: none"> <li>• Use <code>sp_setrepproc</code> to enable or disable replication of stored procedures when you are using function replication definitions. Use either <code>sp_setrepproc</code> or <code>sp_setreplicate</code> to enable or disable replication of stored procedures when you are using table replication definitions.</li> <li>• Use <code>sp_setreplicate</code> with no parameters to display a list of replicated tables or stored procedures in the database.</li> <li>• Use <code>sp_setreplicate object_name</code> without <code>true</code> or <code>false</code> to display the current replication status of the table or stored procedure.</li> </ul>                                                                                                                                                                      |

- If you use `sp_reptostandby` to mark a table for implicit replication to the standby database, text, untext, or image columns set by `sp_setreplicate` or `sp_setrepcol` to `do_not_replicate` are treated as `replicate_if_changed`. Columns set as `always_replicate` or `replicate_if_changed` are treated as marked.
- Because Adaptive Server Enterprise starts a transaction to execute replicated stored procedures, it is important to keep these point in mind when you design procedures:
  - If a replicated stored procedure contains DDL commands (for example, `CREATE TABLE`), Adaptive Server Enterprise generates an error unless the database option “DDL-in-Tran” is enabled on the database.
  - If the replicated stored procedure contains transactions and rollback commands that roll back the transaction, the rollback command rolls back the execution of the entire procedure.
  - Because of the outer transaction, Adaptive Server Enterprise holds all the locks until the execution of the procedure is complete.

See also

`sp_setrepcol`, `sp_setrepproc`, `sp_setreptable`

## sp\_setrepproc

|             |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
|-------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Description | Enables or disables replication for a stored procedure or displays the current replication status of a stored procedure.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| Syntax      | <pre>sp_setrepproc [proc_name [, 'false'   'table'  <br/>                    'function' [, 'log_current'   'log_sproc']]]</pre>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| Parameters  | <p><i>proc_name</i></p> <p>The name of a stored procedure in the current database.</p> <p><i>false</i></p> <p>Disables replication for the stored procedure.</p> <p><i>table</i></p> <p>Enables replication for a stored procedure associated with a table replication definition. This option is equivalent to executing sp_setreplproc on the procedure.</p> <p><i>function</i></p> <p>Enables replication for a stored procedure associated with a function replication definition.</p> <p><i>log_current</i></p> <p>Logs the execution of the stored procedure you are replicating in the current database, not the database where the replicated stored procedure resides.</p> <p><i>log_sproc</i></p> <p>Logs the execution of the stored procedure you are replicating in the database where the stored procedure resides, not in the current database. log_sproc is the default.</p> |
| Examples    | <p><b>Example 1</b> Displays the replication status for all of the stored procedures in the current database. For each procedure, indicates whether it is enabled for replication at all, enabled using a function replication definition, or enabled using a table replication definition.</p> <pre>sp_setrepproc</pre> <p><b>Example 2</b> Displays the replication status for the upd_pubs stored procedure. Indicates whether the stored procedure is enabled for replication at all, enabled using a function replication definition, or enabled using a table replication definition.</p> <pre>sp_setrepproc upd_pubs</pre> <p><b>Example 3</b> Enables replication for the upd_pubs stored procedure for use with a function replication definition. The execution of upd_pubs is logged in the database where upd_pubs resides.</p>                                                  |

```
sp_setrepproc upd_pubs, 'function'
```

**Example 4** Enables replication for the upd\_pubs stored procedure for use with a table replication definition. The execution of upd\_pubs is logged in the database where upd\_pubs resides.

```
sp_setrepproc upd_pubs, 'table'
```

**Example 5** Enables replication for the upd\_pubs stored procedure for use with a function replication definition. The execution of upd\_pubs is logged in the current database.

```
sp_setrepproc upd_pubs, 'function', 'log_current'
```

**Example 6** Enables replication for the upd\_pubs stored procedure for use with a function replication definition. The execution of upd\_pubs is logged in the database where upd\_pubs resides.

```
sp_setrepproc upd_pubs, 'function', 'log_sproc'
```

#### Usage

- Use sp\_setrepproc with no parameters to display all replicated stored procedures in the database.
- Use sp\_setrepproc *proc\_name* with no other parameters to display the current replication status of the stored procedure.
- If you are using Adaptive Server version 11.5 or later, supported DDL commands and stored procedures executed inside a user stored procedure are copied to the standby database if the procedure is enabled for replication with sp\_setrepproc.

Supported DDL commands and stored procedures executed inside a user stored procedure are not copied to the standby database if the procedure is not enabled for replication with sp\_setrepproc.

- Because Adaptive Server starts a transaction to execute replicated stored procedures, keep these points in mind when you design procedures:
  - If a replicated stored procedure contains DDL commands (for example, CREATE TABLE), Adaptive Server Enterprise generates an error unless the database option “DDL-in-Tran” is enabled on the database.
  - If the replicated stored procedure contains transactions and rollback commands that roll back the transaction, the rollback command rolls back the execution of the entire procedure.
  - Because of the outer transaction, Adaptive Server holds all the locks until the execution of the procedure is complete.

See also

`sp_reptostandby`, `sp_setreplicate`, `sp_setreptable`

## sp\_setreptable

|             |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
|-------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Description | Enables or disables replication for an Adaptive Server table or displays the current replication status of a table.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
| Syntax      | <code>sp_setreptable [table_name [, {'true'   'false'   'never'}<br/>[, {owner_on   owner_off   null}] [, use_index]]]</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| Parameters  | <p><i>table_name</i><br/>The name of the table marked for replication.</p> <p><i>true</i><br/>Explicitly marks the table for replication, regardless of whether the database is marked for replication or not.</p> <p><i>false</i><br/>Disables the replication status on a table that has previously been enabled for replication.</p> <p><i>never</i><br/>Disables replication on the table, regardless of the database replication setting.</p> <p><i>owner_on</i><br/>Sets the mode of the table so that both the table name and owner name are considered when the table is marked for replication. Enables tables with the same name but different owner be replicated. This option is for Adaptive Server version 11.5 and later databases.</p> <p><i>owner_off</i><br/>Sets the mode of the table so that only the table name is considered when the table is marked for replication. This is the default. It ensures that the name for each table marked for replication is unique. This option is for Adaptive Server version 11.5 and later databases.</p> <p><i>null</i><br/>Sets the default value of <i>owner_off</i> when you pass it to the <i>owner</i> parameter.</p> <p><i>use_index</i><br/>Marks the table to use an index for replication on text, unitext, image, or rawobjects columns.</p> |
| Examples    | <p><b>Example 1</b> Displays the replication status for all of the tables in the current database:</p> <pre>sp_setreptable</pre> <p><b>Example 2</b> Displays the replication status for the publishers table:</p> <pre>sp_setreptable publishers</pre>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |

**Example 3** Enables replication for the publishers table:

```
sp_setreptable publishers, 'true'
```

**Example 4** Allows multiple tables named publishers each owned by different users to be replicated:

```
sp_setreptable publishers, 'true', owner_on
```

**Example 5** Replicates table named publishers belonging to owner dbo and stored in database pubs2:

```
sp_setreptable 'pubs2.dbo.publishers', 'true', owner_on
```

**Example 6** Marks the table for replication to use indexes on the text, unitext, image, and rawobject columns, and sets owner status to “off”:

```
sp_setreptable t1, true, null, use_index
```

**Example 7** Removes the replication status of table t1, and drops the replication indexes if t1 was initially marked for replication to use indexes:

```
sp_setreptable t1, 'false'
```

**Example 8** To disable replication on table tnever in database pdb, use:

```
sp_reptostandby pdb, 'ALL'
go
sp_setreptable tnever, 'never'
go
```

#### Usage

- Use `sp_setreptable` with no parameters to display a list of replicated tables in the database.
- Use `sp_setreptable table_name` without `true` or `false` to display the current replication status of the table.
- When you include the `owner_on` option, multiple tables with the same table name but different owners may be replicated to replicate and warm standby databases. Make sure that the replication definition on the table also includes owner information or replication may fail.
- If a table has been marked for replication with `sp_setreptable`, you can change the owner mode with the `sp_setrepdefmode` system procedure.
- The replication index status order of precedence is: column, table, database. For example, in a database marked for replication using indexes, the table status overrides the index status.

- When a large table containing one or more text, unitext, image, or rawobject columns is marked for replication, the internal process is performed in a single transaction and may take a long time. To speed up the process, use the `use_index` option to create a global nonclustered index for every text, unitext, image, or rawobject column.
- With `use_index`, a shared-table lock is held while the global nonclustered index is created.
- You cannot use `drop index` to manually drop indexes created for text, unitext, image, or rawobject replication. You can use only the supported replication stored procedures `sp_reptostandby`, `sp_setreptable`, and `sp_setrepcol` to change the replication index status.

Permissions

`sp_setreptable` requires “sa” or “dbo” permission or `replication_role`.

See also

`sp_reptostandby`, `sp_setrepcol`, `sp_setrepdefmode`, `sp_setreplcate`, `sp_setrepproc`

## sp\_start\_rep\_agent

|             |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
|-------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Description | Starts a RepAgent thread for the specified database.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
| Syntax      | <pre>sp_start_rep_agent dbname[, {'recovery'   'recovery_foreground'}<br/>[, 'connect_dataserver',<br/>'connect_database'[, 'repserver_name', repserver_username',<br/>'repserver_password']]]]</pre>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| Parameters  | <p><i>dbname</i><br/>The name of the database for which you want to start a RepAgent.</p> <p><i>recovery</i><br/>Starts the RepAgent in recovery mode, which is used to initiate recovery actions. Recovery mode is used to rebuild queues when queues are lost.<br/><br/>You can also specify the Replication Server name, user name, and password in recovery mode. Specify these parameters to override sysattributes settings.</p> <p><i>recovery_foreground</i><br/><i>recovery_foreground</i> has the same function as <i>recovery</i>. However, it displays the recovery progress information on screen instead of in the Adaptive Server error log. The recovery is complete once the recovery progress information display ends and the command prompt displays.</p> <p><i>connect_dataserver</i><br/>The name of the data server used to recover offline logs.</p> <p><i>connect_database</i><br/>The name of the database used to recover offline logs.</p> <p><i>repserver_name</i><br/>The name of the Replication Server to which RepAgent connects.</p> <p><i>repserver_user_name</i><br/>The user name that RepAgent uses to connect to Replication Server.</p> <p><i>repserver_password</i><br/>The password that RepAgent uses to connect to Replication Server.</p> |
| Examples    | <p><b>Example 1</b> Starts an integrated RepAgent for the pubs2 database. RepAgent connects to the Replication Server specified in <code>sp_config_rep_agent</code>. It starts scanning the transaction log and sends formatted LTL commands to Replication Server.</p> <pre>sp_start_rep_agent pubs2</pre> <p><b>Example 2</b> Starts RepAgent in recovery mode for the pdb2 database connected to the svr2 data server.</p>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |

```
sp_start_rep_agent pubs2 for_recovery, svr2, pdb2
```

**Example 3** Configures RepAgent to print the recovery of database db2 to the client:

```
sp_start_rep_agent db2, recovery_foreground, ds, db1
RepAgent(5). Starting recovery, processing log records
 between (1018, 0) and (2355, 2).
RepAgent(5). Processed 1000 log records.
RepAgent(5). Processed 2000 log records.
RepAgent(5). Processed 3000 log records.
RepAgent(5). Processed 4000 log records.
RepAgent(5). Processed 5000 log records.
RepAgent(5). Processed 6000 log records.
RepAgent(5). Processed 7000 log records.
RepAgent(5). Processed 8000 log records.
RepAgent(5). Processed 9000 log records.
RepAgent(5). Processed 10000 log records.
RepAgent(5). Processed 11000 log records.
RepAgent(5). Processed 12000 log records.
RepAgent(5). Processed 13000 log records.
RepAgent(5). Processed 14000 log records.
RepAgent(5). Processed 15000 log records.
RepAgent(5). Processed 16000 log records.
RepAgent(5). Processed 17000 log records.
RepAgent(5). Processed 18000 log records.
RepAgent(5). Processed 19000 log records.
RepAgent(5). Processed 20000 log records.
RepAgent(5). Processed 20084 log records, recovery
 complete.
Replication Agent thread is started for database 'db2'.
(return status = 0)
```

#### Usage

- Use sp\_start\_rep\_agent with RepAgent-enabled databases.
- Use the sp\_start\_rep\_agent command to start up RepAgent after you have enabled it with sp\_config\_rep\_agent. Once you have started RepAgent with sp\_start\_rep\_agent, it will automatically start up after the dataserver is recovered during server startup.
- Autostart is disabled after you have used sp\_stop\_rep\_agent to shut down RepAgent. Reenable it using sp\_start\_rep\_agent.

- For offline recovery, archived transaction logs may be dumped to a temporary recovery database. You can then transfer records in the transaction log of the temporary recovery database to the replicate database. Execute `sp_start_rep_agent` with either `recovery` or `recovery_foreground`, using the temporary data server and database names, to scan the temporary transaction log.

In recovery, when the RepAgent has completed scanning the transaction log, RepAgent shuts down. After the next transaction dump has been loaded, restart the RepAgent by executing `sp_start_rep_agent` with the options specified earlier.

|             |                                                                                                      |
|-------------|------------------------------------------------------------------------------------------------------|
| Permissions | <code>sp_start_rep_agent</code> requires “sa” or “dbo” permission or <code>replication_role</code> . |
| See also    | <code>sp_help_rep_agent</code> , <code>sp_stop_rep_agent</code>                                      |

## **sp\_stop\_rep\_agent**

|             |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
|-------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Description | Shuts down the RepAgent thread for the specified database.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| Syntax      | <code>sp_stop_rep_agent <i>dbname</i> [, 'nowait']</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| Parameters  | <p><i>dbname</i></p> <p>The name of the database for which you want to shut down the RepAgent.</p> <p><i>nowait</i></p> <p>Shuts down the RepAgent immediately, without waiting for executing operations to complete.</p> <p>The default shuts down RepAgent gracefully at the end of the current batch.</p>                                                                                                                                                                                                                                      |
| Examples    | <p>Shuts down an integrated RepAgent for the pubs2 database. The default shutdown option allows RepAgent to finish processing the current batch.</p> <pre>sp_stop_rep_agent pubs2</pre>                                                                                                                                                                                                                                                                                                                                                           |
| Usage       | <ul style="list-style-type: none"><li>• Use <code>sp_stop_rep_agent</code> with RepAgent-enabled databases.</li><li>• Once you have used <code>sp_stop_rep_agent</code> to shut down RepAgent, it does not automatically start up when the database comes online during server startup. To re-enable automatic startup, execute the <code>sp_start_rep_agent</code> procedure.</li><li>• <code>sp_stop_rep_agent</code> is asynchronous and may take some time to execute. Use <code>sp_who</code> to check the status of the RepAgent.</li></ul> |
| Permissions | <code>sp_start_rep_agent</code> requires “sa” or “dbo” permission or <code>replication_role</code> .                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| See also    | <code>sp_config_rep_agent</code> , <code>sp_help_rep_agent</code> , <code>sp_start_rep_agent</code>                                                                                                                                                                                                                                                                                                                                                                                                                                               |

This chapter contains reference pages for the RSSD stored procedures used with Replication Server. Use the RSSD to execute these stored procedures.

## rs\_capacity

|             |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
|-------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Description | Helps you estimate stable queue size requirements. Use with the <code>rs_fillcaptable</code> stored procedure.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
| Syntax      | <code>rs_capacity TranDuration, FailDuration, SaveInterval, MatRows</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
| Parameters  | <p><i>TranDuration</i><br/>The duration, in seconds, of the longest transaction. The default is up to 5 seconds.</p> <p><i>FailDuration</i><br/>The length in time, in minutes, that the queue must retain information during a failure. The default is 60 minutes.</p> <p><i>SaveInterval</i><br/>The length of time, in minutes, that messages should be retained after they have been confirmed as received. The default is 1 minute.</p> <p><i>MatRows</i><br/>The number of rows to be materialized in a subscription. The default is 1000 rows.</p>                                                    |
| Examples    | <p>For the example scenario described for the <code>rs_fillcaptable</code> stored procedure, use <code>rs_capacity</code> with the following parameters.</p> <pre>rs_capacity 60,      /* TranDuration maximum 60 seconds */ 360,     /* FailDuration 6 hours */ 10,      /* SaveInterval 10 minutes */ 3500     /* Materialize 3500 rows */</pre> <p><code>rs_capacity</code> returns an estimate of the queue sizes needed for each queue. It also gives an estimate of the subscription materialization queue size needed, based on the replication definition and the number of rows to materialize.</p> |
| Usage       | <ul style="list-style-type: none"><li><code>rs_capacity</code> uses the data in the <code>rs_captable</code> table (created using the <code>rs_fillcaptable</code> stored procedure) to calculate estimates of stable queue size requirements. Execute <code>rs_capacity</code> after you have described replication definitions using <code>rs_fillcaptable</code>.</li></ul>                                                                                                                                                                                                                               |
| See also    | <code>rs_fillcaptable</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |

## rs\_delexception

|             |                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
|-------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Description | Deletes a transaction in the exceptions log.                                                                                                                                                                                                                                                                                                                                                                                                         |
| Syntax      | <code>rs_delexception [<i>transaction_id</i>]</code>                                                                                                                                                                                                                                                                                                                                                                                                 |
| Parameters  | <i>transaction_id</i><br>The number of the transaction you want to delete.                                                                                                                                                                                                                                                                                                                                                                           |
| Examples    | Deletes transaction number 1234 from the exceptions log.<br><br><code>rs_delexception 1234</code>                                                                                                                                                                                                                                                                                                                                                    |
| Usage       | <ul style="list-style-type: none"><li>• If you do not specify any parameters, <code>rs_delexception</code> displays a summary of transactions in the exceptions log.</li><li>• If you supply a valid <i>transaction_id</i>, <code>rs_delexception</code> deletes a transaction. You can find the <i>transaction_id</i> for a transaction by using either <code>rs_helpexception</code> or <code>rs_delexception</code> with no parameters.</li></ul> |
| See also    | <code>rs_helpexception</code>                                                                                                                                                                                                                                                                                                                                                                                                                        |

## rs\_dump\_stats

|             |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
|-------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Description | Extracts Replication Server statistics collected in the RSSD by admin stats to a comma-delimited format.                                                                                                                                                                                                                                                                                                                                                                                |
| Syntax      | <code>rs_dump_stats [<i>'comment'</i>]</code>                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| Parameters  | <i>comment</i><br>Is an optional description of the statistics being displayed. It appears on the first line of the output file.                                                                                                                                                                                                                                                                                                                                                        |
| Examples    | Extracts Replication Server statistics with comment “Stats from 01/31/2006.”<br><br><code>rs_dump_stats 'Stats from 01/31/2006'</code><br><br>The columns of counter data are, in order: <ul style="list-style-type: none"><li>• The timestamp of the observation period</li><li>• The number of observations made of the counter during the observation period</li><li>• The total of observed values</li><li>• The last observed value</li><li>• The maximum observed value</li></ul> |

Depending on the counter category (see the chapter “Using Counters to Monitor Performance” in the *Replication Server Administration Guide Volume 2* for a description of counter categories), there may be close correlation between the number of observations and total observations, and between the last and maximum observed values. For example, an observer counter simply counts the number of observations of an event—such as the number of times a message is read from a queue. For an observer counter, the number of observations and the total of observed values are the same. Similarly, the last and maximum observed values are both 1 (unless no messages were read in the observation period, in which case both values would be 0).

---

**Note** Comments to the right of the output are included to explain the example. They are not part of the rs\_dump\_stats output.

---

```
Comment: Stats from 01/31/2006== Provided label
Oct 17 2005 3:13:47:716PM == End of the first observation period
Oct 17 2005 3:14:24:730PM == End of the last observation period
2 == Number of observation periods
0 == Number of minutes in each obs period.
0 if less than one. (Calculated as the number of minutes between the first
and last obs period, divided by the number of observations.)
16384 == Number of bytes in an SQM Block to
 aid calculations
64 == Number of blocks in an SQM Segment
 to aid calculations
CM == Module Name. See rs_help_counter
 for a complete list.
13 == Instance ID. See admin stats for an
 explanation.
-1 == Inst Val/Mod Type. Further instance
 qualification when needed.
dCM == Instance description.
CM: Outbound database connection
requests == Counter description.
CMOBDDBReq == Counter display name.
13003 , , 13, -1 == Counter ID and instance qualifying
 information.
Oct 17 2005 3:13:47:716PM, 52,
52, 1, 1 == Counter data. One row output for
 each observation period. See below for
 explanation.
Oct 17 2005 3:14:24:730PM, 42,
42, 1, 1
ENDOFDATA == End of output for the previous
```

```

 counter
CM: Outbound non-database
connection requests == Start of output for the next counter
CMOBNonDBReq
13004 , , 13, -1
Oct 17 2005 3:13:47:716PM, 2, 2, 1, 1
Oct 17 2005 3:14:24:730PM, 2, 2, 1, 1
ENDOFDATA
.
.
.
CM: Time spent closing an ob fadeout conn
CMOBConnFadeOutClose
13019 , , 13, -1
Oct 17 2005 3:13:47:716PM, 0, 0, 0, 0
Oct 17 2005 3:14:24:730PM, 2, 6, 2, 4
ENDOFDATA
DIST == Start of output for the next
 module/instance
102
-1
DIST, 102 pds03.tpc
DIST: Commands read from inbound queue
CmdsRead
30000 , , 102, -1
Oct 17 2005 3:13:47:716PM, 1, 1, 1, 1
Oct 17 2005 3:14:24:730PM, 1, 1, 1, 1
ENDOFDATA
.
.
.
DSIEXEC: Number of 'message' results
DSIEResMsg
57127 , , 103, 7
Oct 17 2005 3:13:47:716PM, 1, 1, 1, 1
Oct 17 2005 3:14:24:730PM, 1, 1, 1, 1
ENDOFDATA
(return status = 0) == End of output

```

**Usage**

- You can capture the output of `rs_dump_stats` in a text file that can then be analyzed in a spread sheet or other analysis tool.
- If the text file containing the output of `rs_dump_stats` is too large to load in to the analysis tool, you can split the file into multiple files.
  - Each new file must contain the first seven rows and the last row of the original file.

- Between the first seven rows and the last row of each new file, insert all rows associated with a given module instance.

Depending on the analysis tool, it is usually unnecessary to include all instances of one module in the same file.

- `rs_dump_stats` does not remove or alter statistics saved in the RSSD.
- `rs_dump_stats` lists counters with no observations, but does not display counter data rows for them. `rs_dump_stats` displays counter data rows for all counters with at least one observation during the sample period.

See also

`rs_helpcounter`, `admin stats`

## rs\_fillcaptable

|             |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
|-------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Description | Records estimated transaction rates in the rs_captable table for an existing replication definition.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
| Syntax      | <code>rs_fillcaptable RepDefName, InChRateI, InChRateD, InChRateU, OutChRateI, OutChRateD, OutChRateU, InTranRate, OutTranRate, DelFlag</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| Parameters  | <p><i>RepDefName</i><br/>The name of the replication definition.</p> <p><i>InChRateI</i><br/>The number of inserts per second, including inserts that are not replicated. The default is 15 inserts per second.</p> <p><i>InChRateD</i><br/>The number of deletes per second, including deletes that are not replicated. The default is 15 deletes per second.</p> <p><i>InChRateU</i><br/>The number of updates per second, including updates that are not replicated. The default is 15 updates per second.</p> <p><i>OutChRateI</i><br/>The number of inserts per second, excluding inserts that are not replicated. The default is 15 inserts per second.</p> <p><i>OutChRateD</i><br/>The number of deletes per second, excluding deletes that are not replicated. The default is 15 deletes per second.</p> <p><i>OutChRateU</i><br/>The number of updates per second, excluding updates that are not replicated. The default is 15 updates per second.</p> <p><i>InTranRate</i><br/>The number of transactions per second for the database. The default is 5 transactions per second.</p> <p><i>OutTranRate</i><br/>The number of replicated transactions per second for the database. The default is 5 transactions per second.</p> <p><i>DelFlag</i><br/>Set to “n” or “N” to update the row for the replication definition. Set to “y” or “Y” to delete the row for the replication definition from rs_captable. Set <i>DelFlag</i> to “Y” and <i>RepDefName</i> to “ALL,” to clear the entire rs_captable table.</p> |

## Examples

**Example 1** In this example scenario, the overall transaction rate in a primary database is 10 transactions per second. Of these 10 transactions, 8 are replicated. The *InTranRate* for the database is 10 and the *OutTranRate* is 8.

There are two replicated transactions, T1 and T2. T1 executes 5 times per second, performs 2 updates to table1, and performs 1 update to table2. T2 executes 3 times per second, performs 2 inserts to table1, and performs 1 insert to table2.

There are two subscriptions in replicate databases, each receiving one half of the replicated data. The transactions are distributed equally across the two subscriptions. Therefore, the outbound estimates are 50 percent of the inbound estimates.

This table summarizes the information from this example scenario:

|                 |                   | <i>table1</i> |     |     | <i>table2</i> |     |     |
|-----------------|-------------------|---------------|-----|-----|---------------|-----|-----|
|                 |                   | ins           | upd | del | ins           | upd | del |
| <i>Inbound</i>  | T1 (5 per second) | 10            |     |     | 5             |     |     |
|                 | T2 (3 per second) | 6             |     |     | 3             |     |     |
|                 | Totals            | 6             | 10  |     | 8             |     |     |
| <i>Outbound</i> | 50% replicated    | 3             | 5   |     | 4             |     |     |

To get an estimate of stable queue size requirements for this example scenario, first clear the *rs\_captable* table. Then use *rs\_fillcaptable* with the parameters described above. When you are done, use the *rs\_capacity* stored procedure with the new contents of the *rs\_captable* table.

**Example 2** This example clears the *rs\_captable* table.

```
rs_fillcaptable @RepDefName = 'ALL', @DelFlag = 'Y'
```

**Example 3** This example fills the *rs\_captable* table with the appropriate values for the first replication definition.

```
rs_fillcaptable
repdef1, /* replication definition for table1 */
6, /* InChRateI */
0, /* InChRateD */
10, /* InChRateU */
3, /* OutChRateI */
0, /* OutChRateD */
5, /* OutChRateU */
10, /* InTranRate */
8, /* OutTranRate */
n /* DelFlag */
```

**Example 4** This example fills the `rs_captable` table with the appropriate values for the second replication definition.

```
rs_fillcaptble
 repdef2, /* replication definition for table2 */
 8, /* InChRateI */
 0, /* InChRateD */
 0, /* InChRateU */
 4, /* OutChRateI */
 0, /* OutChRateD */
 0, /* OutChRateU */
 10, /* InTranRate */
 8, /* OutTranRate */
 n /* DelFlag */
```

See `rs_capacity` for more information on using the output derived from these examples to complete the estimate of stable queue size requirements.

Usage

- Use `rs_fillcaptble` to describe the transactions for each replication definition you want to include in your stable queue estimate.
- `rs_fillcaptble` maintains a work table named `rs_captable` that contains estimates of change rates for each replication definition in a database.
- Use the output of `rs_fillcaptble` as input for the `rs_capacity` stored procedure.

See also

`rs_capacity`

# rs\_helpclass

|             |                                                                                                                                                                                 |
|-------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Description | Displays error classes and function-string classes and their primary Replication Server and, in the case of inherited classes, the parent class.                                |
| Syntax      | rs_helpclass [ <i>class_name</i> ]                                                                                                                                              |
| Parameters  | <i>class_name</i><br>A string of characters that corresponds to an error class or function-string class name. The string must match an entire name or the first part of a name. |
| Examples    | <b>Example 1</b> Displays information about all error classes and function-string classes for the Replication Server.                                                           |

```
rs_helpclass

Function String Class(es) PRS for CLASS Parent Class

rs_default_function_class Not Yet Defined. Base class
rs_sqlserver_function_class Not Yet Defined. Base class
sqlserver2_function_class TOKYO_RS rs_default_function_class

Error Class(es) PRS for CLASS

rs_db2_error_class Not Yet Defined.
rs_msss_error_class Not Yet Defined.
rs_oracle_error_class Not Yet Defined.
rs_sqlserver_error_class Not Yet Defined.
rs_udb_error_class Not Yet Defined

RepServer Error Class(es) PRS for CLASS

rs_repserver_error_class Not Yet Defined.
```

**Example 2** Displays information about the sqlserver2\_function\_class function-string class.

```
rs_helpclass sqlserver2_function_class
```

|       |                                                                                                                                  |
|-------|----------------------------------------------------------------------------------------------------------------------------------|
| Usage | <b>Note</b> Use the command admin show_function_classes to get more information about error classes and function-string classes. |
|-------|----------------------------------------------------------------------------------------------------------------------------------|

---

- If you do not enter any parameters, rs\_helpclass lists all defined error classes and function-string classes.
- If you supply a *class\_name* string, rs\_helpclass lists error classes and function-string classes that match *class\_name*.

- If a class is not defined at a Replication Server, which is true of default classes for Adaptive Server, `rs_helpclass` lists it as undefined and tells you how to define it.

# rs\_helpclassfstring

|             |                                                                                                                                                                                                                                                                   |
|-------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Description | Displays the function-string information for function strings with function-string-class scope.                                                                                                                                                                   |
| Syntax      | rs_helpclassfstring <i>class_name</i><br>[, <i>function_name</i> ]                                                                                                                                                                                                |
| Parameters  | <i>class_name</i><br>The function-string class for which you want to view function strings.<br><br><i>function_name</i><br>A string of characters that corresponds to a function name. The string must match an entire function name or the first part of a name. |

**Examples**                   **Example 1** Displays parameters and function-string text for all functions of the function-string class rs\_sqlserver\_function\_class.

```
rs_helpclassfstring rs_sqlserver_function_class
```

**Example 2** Displays the function-string text for the rs\_usedb function of rs\_sqlserver\_function\_class.

```
rs_helpclassfstring rs_sqlserver_function_class,
rs_usedb
```

| Function Name | FString Name | FSClass Name                |
|---------------|--------------|-----------------------------|
| -----         | -----        | -----                       |
| rs_usedb      | rs_usedb     | rs_sqlserver_function_class |

| FString Text                    |
|---------------------------------|
| -----                           |
| use ?rs_destination_db!sys_raw? |

- Usage**
- If you do not supply a *function\_name* parameter, rs\_helpclassfstring displays all function strings defined for all functions of the function-string class.
  - If you supply a *function\_name* string, rs\_helpclassfstring displays function strings that match *function\_name*, such as rs\_insert, rs\_delete, rs\_update, and rs\_select, or a user-defined function.
  - Non-customized, inherited function strings are not displayed for derived function-string classes.

## rs\_helpcounter

|             |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
|-------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Description | Displays information about counters.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| Syntax      | <pre>rs_helpcounter [{sysmon   duration   observer   monitor                   must_sample   no_reset   keep_old}                   <i>module_name</i> [, {short   long}]   <i>keyword</i> [, {short   long}]]</pre>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| Parameters  | <p><b>sysmon</b><br/>Specifies those counters most useful for assessing performance and for gathering replication system profile information.</p> <p><b>duration</b><br/>Specifies all counters that measure duration with time intervals measured in one-hundredths of a second.</p> <p><b>observer</b><br/>Specifies counters that record the number of times an event occurs. For example, the number of times a message is read from a queue.</p> <p><b>monitor</b><br/>Specifies counters that record a current value. For example, the size in bytes of the message most recently read from the queue.</p> <p><b>must_sample</b><br/>Specifies counters that must keep sampling even if sampling is not turned on.</p> <p><b>no_reset</b><br/>Specifies counters whose values are not reset when admin stats, reset is executed.</p> <p><b>keep_old</b><br/>Specifies counters that keep both current and previous values.</p> <p><b><i>module_name</i></b><br/>The name of a module: dsi, dsiexec, sqt, cm, dist, rsi, sqm, repagent, and so on.</p> <p><b>short</b><br/>Tells Replication Server to print the display names, module names, and counter descriptions of counters specified.</p> <p><b>long</b><br/>Tells Replication Server to print values for every column in the rs_statcounters table.</p> |

*keyword*

Search keyword. Search in the counter long names, the counter display names, and counter descriptions.

Examples

**Example 1** Lists all module names, and syntax for using rs\_helpcounter.

```
1> rs_helpcounter
2> go

ModuleName

CM
DIST
DSI
DSIEXEC
REAGENT
RSH
RSI
RSIUSER
SERV
SQM
SQMR
SQT
STS
SYNC
SYNCELE
(12 rows affected)
```

How to Use rs\_helpcounter

```

rs_helpcounter -> Shows module names and help.
rs_helpcounter [sysmon | duration | observe | monitor
 | must_sample | no_reset | keep_old]
rs_helpcounter ModuleName [, {short | long }]
rs_helpcounter keyword [, { short | long }]
 where "keyword" is part of the counter name, display name or description
 (return status = 0)
```

**Example 2** Lists the display names, module names, and counter descriptions for the SQM Reader.

| rs_helpcounter sqmr, short |             |                                                                        |
|----------------------------|-------------|------------------------------------------------------------------------|
| Display Name               | Module Name | Counter Description                                                    |
| -----                      | -----       | -----                                                                  |
| --                         |             |                                                                        |
| BlocksRead                 | SQMR        | Number of 16K blocks read from a stable queue by an SQM Reader thread. |

|                     |      |                                                                                                                                                                                                       |
|---------------------|------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| ClocksReadCached    | SQMR | Number of 16K blocks from cache read by an SQM Reader thread.                                                                                                                                         |
| CmdsRead            | SQMR | Commands read from a stable queue by an SQM Reader thread.                                                                                                                                            |
| SQMRReadTime        | SQMR | The amount of time taken for SQMR to read a block.                                                                                                                                                    |
| SleepsStartQR       | SQMR | srv_sleep() calls by an SQM Reader client due to waiting for SQM thread to start.                                                                                                                     |
| SleepsWriteQ        | SQMR | srv_sleep() calls by an SQM read client due to waiting for the SQM thread to write.                                                                                                                   |
| XNLInterrupted      | SQMR | Number of interruptions so far when reading large messages with partial read. Such interruptions happen due to time out, unexpected wakeup, or nonblock read request, which is marked as READ_POSTED. |
| XMLPartials         | SQMR | Partial large messages read so far.                                                                                                                                                                   |
| XNLReads            | SQMR | Large messages read successfully so far. This does not count partial messages, or timeout interruptions.                                                                                              |
| (return status = 0) |      |                                                                                                                                                                                                       |

|             |                                                                                                                                                                                                                                                                                                                                                                               |
|-------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Usage       | <ul style="list-style-type: none"> <li>rs_helpcounter lets you search the rs_statcounters system table.</li> <li>When used with no parameters, rs_helpcounter prints out a list of modules and syntax.</li> <li>For information about counter status and other counter information stored in the RSSD, see the rs_statcounters system table described on page 712.</li> </ul> |
| Permissions | Any user may execute this command.                                                                                                                                                                                                                                                                                                                                            |

# rs\_helpdb

Description Provides information about databases that Replication Server knows about.

Syntax rs\_helpdb [*data\_server*, *database*]

Parameters

*data\_server*  
The data server with the database whose information you want to display.

*database*  
The database whose information you want to display.

Examples rs\_helpdb

|                                        |                             |      |
|----------------------------------------|-----------------------------|------|
| dsname                                 | dbname                      | dbid |
| -----                                  | -----                       | ---- |
| TOKYO_DS                               | TOKYO_RSSD                  | 101  |
| SYDNEY_DS                              | SYDNEY_RSSD                 | 102  |
| TOKYO_DS                               | pubs2                       | 105  |
| controlling_prs                        | errorclass                  |      |
| -----                                  | -----                       |      |
| TOKYO_RS                               | rs_sqlserver_error_class    |      |
| SYDNEY_RS                              | rs_sqlserver_error_class    |      |
| TOKYO_RS                               | rs_sqlserver_error_class    |      |
| repserver_errorclass                   | funcclass                   |      |
| -----                                  | -----                       |      |
| rs_repserver_error_class               | rs_sqlserver_function_class |      |
| rs_repserver_error_class               | rs_sqlserver_function_class |      |
| rs_repserver_error_class               | rs_sqlserver_function_class |      |
| status                                 |                             |      |
| -----                                  |                             |      |
| Log Transfer is ON, Distribution is ON |                             |      |
| Log Transfer is ON, Distribution is ON |                             |      |
| Log Transfer is ON, Distribution is ON |                             |      |

- Usage
- If you do not provide the *data\_server* and *database* parameters, rs\_helpdb returns results for all of the databases in the rs\_databases system table.
  - rs\_helpdb is executed in a Replication Server’s RSSD.
  - For each database, rs\_helpdb provides the following information:  
*dsname* – the name of the data server with the database.  
*dbname* – the name of the database.

*dbid* – the ID number assigned to uniquely identify the database throughout the replication system.

*controlling\_prs* – the Replication Server that manages the database.

*errorclass* – the error class Replication Server uses to handle errors returned from the data server for this database.

*repserver\_errorclass* – the error class that handles errors returned from the Replication Server for this database.

*funcclass* – the function-string class used for the database.

*status* – tells whether log transfer and distribution are on or off for the database.

*ltype* – the type of database connection (logical or physical).

*ptype* – the type of database (active database, standby database, or logical connection).

## rs\_helpdbrep

**Description** Displays information about database replication definitions associated with the current Replication Server.

**Syntax** `rs_helpdbrep [db_repdef[, data_server[, database]]]`

**Parameters**

- db\_repdef*  
Specifies the name of the database replication definition.
- data\_server*  
Specifies the name of the data server whose database replication definition you want to display.
- database*  
Specifies the name of the database whose database replication definition you want to display.

**Examples** **Example 1** In this example, Adaptive Server displays the information of all the database replication definitions found in the current Replication Server:

```
rs_helpdbrep
```

```
DB Rep.Def.Name Primary DS.DB Primary RS Rep.DDL Rep.Sys. Rep.Tab Rep.Func.

```

|         |          |     |     |          |     |     |
|---------|----------|-----|-----|----------|-----|-----|
| db_rep1 | PDS.pdb1 | PRS | Yes | Out-List | All | All |
| db_rep2 | PDS.pdb2 | PRS | Yes | Out-List | All | All |

| Rep.Tran. | Rep.Upd. | Rep.Del. | Rep.Ins. | Rep.Sel. | Creation Date      |
|-----------|----------|----------|----------|----------|--------------------|
| All       | All      | All      | All      | All      | Nov 26 2008 6:58AM |
| All       | All      | All      | All      | All      | Dec 2 2008 6:12PM  |

**Example 2** In this example, Adaptive Server displays information about a single database replication definition, db\_rep1:

```
rs_helpdbrep db_rep1
```

| DB Rep.Def.Name | Primary DS.DB | Primary RS | Rep.DDL | Rep.Sys. | Rep.Tab | Rep.Func. |
|-----------------|---------------|------------|---------|----------|---------|-----------|
| db_rep1         | PDS.pdb1      | PRS        | Yes     | Out-List | All     | All       |

| Rep.Tran. | Rep.Upd. | Rep.Del. | Rep.Ins. | Rep.Sel. | Creation Date      |
|-----------|----------|----------|----------|----------|--------------------|
| All       | All      | All      | All      | All      | Nov 26 2008 6:58AM |

| Rep.Type     | Owner | Name          |
|--------------|-------|---------------|
| Not Rep.Sys. | .     | sp_setrepproc |

| DBRep.Def.Name | DBSub.Name | ReplicationDS.DB | ReplicateRS | Creation Date      |
|----------------|------------|------------------|-------------|--------------------|
| db_rep1        | db_sub1    | RDS1.rdb1        | RRS1        | Nov 26 2008 6:58AM |
| db_rep1        | db_sub2    | RDS2.rdb2        | RRS2        | Nov 26 2008 6:59AM |

#### Usage

- Adaptive Server only displays detail information about named database replication definitions.
- The parameters can contain the wild card '%'. This wild card represents any string. For example, if a string 'abc%' is assigned to *db\_repdef*, rs\_helpdbrep will list all database replication definition that have a database replication definition name prefixed with 'abc'.

#### See also

rs\_helpdbsub

## rs\_helpdbsub

**Description** Displays information about database subscriptions associated with the replicate data server.

**Syntax** `rs_helpdbsub [db_sub[, data_server[, database]]]`

**Parameters** *db\_sub*

Specifies the database subscription.

*data\_server*

Specifies the data server name whose database subscription you want to display.

*database*

Specifies the database name whose database subscription you want to display.

**Examples** In this example, Adaptive Server displays information about a single database subscription, `db_sub1`:

```
rs_helpdbsub db_sub1, RDS1, rdb1
```

| DBSub.Name | ReplicateDS.DB | ReplicateRS | Status at RRS | DBRep.Def.Name |
|------------|----------------|-------------|---------------|----------------|
| db_sub1    | RDS1.rdb1      | RRS1        | Validate      | db_rep         |

| PrimaryDS.DB | PrimaryRS | Method      | Trunc.Table | Creation Date     |
|--------------|-----------|-------------|-------------|-------------------|
| PDS.pdb1     | PRS       | Bulk Create | Yes         | May 2 2003 3:38PM |

**Usage**

- If you do not specify any parameters, `rs_helpdbsub` lists database subscriptions defined in the Replication Server.
- If you supply the *db\_sub* parameter only, `rs_helpdbsub` lists all the database subscriptions defined in the Replication Server that have a database subscription name matching *db\_sub*.
- The parameters can contain the wild card '%'. This wild card represents any string. For example, if a string 'abc%' is assigned to *db\_sub*, `rs_helpdbsub` will list all database subscriptions that have a database subscription name prefixed with 'abc'.

**See also** `rs_helpdbrep`

# rs\_helperror

**Description** Displays the Replication Server error actions mapped to a given data server or Replication Server error number.

**Syntax** rs\_helperror server\_error\_number [, v]

**Parameters** server\_error\_number  
A data server error number.

v  
Displays the Adaptive Server error message text, if it is available.

**Examples** rs\_helperror 2601, v

| DS Error Num                                                                        | Error Action     | Error Class                    |
|-------------------------------------------------------------------------------------|------------------|--------------------------------|
| -----                                                                               | -----            | -----                          |
| 2601                                                                                | Stop Replication | rs_sqlserver_error_class       |
| Adaptive Server Error Message                                                       |                  |                                |
| -----                                                                               |                  |                                |
| Attempt to insert duplicate key row in object '%.*s' with unique index '%.*s'%S_EED |                  |                                |
| RS Error Num                                                                        | Error Action     | Replication Server Error Class |
| -----                                                                               | -----            | -----                          |

- Usage**
- Error action mappings are displayed for all error classes.
  - Use the assign action command to map error actions to data server error numbers.

**See also** assign action

## rs\_helpexception

|             |                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
|-------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Description | Displays transactions in the exceptions log.                                                                                                                                                                                                                                                                                                                                                                                                              |
| Syntax      | <code>rs_helpexception [transaction_id, [, v]]</code>                                                                                                                                                                                                                                                                                                                                                                                                     |
| Parameters  | <p><i>transaction_id</i></p> <p>The number of the transaction for which you want help.</p> <p><i>v</i></p> <p>Includes the text of the transaction in a detailed listing.</p>                                                                                                                                                                                                                                                                             |
| Examples    | <p><b>Example 1</b> Displays summary information on all transactions in the exceptions log.</p> <pre>rs_helpexception</pre> <p><b>Example 2</b> Displays detailed information on transaction number 1234, including the text of the transaction.</p> <pre>rs_helpexception 1234, v</pre>                                                                                                                                                                  |
| Usage       | <ul style="list-style-type: none"><li>• If you do not enter any parameters, <code>rs_helpexception</code> displays a summary list of the transactions in the exceptions log, including all transaction numbers.</li><li>• If you supply a valid <i>transaction_id</i>, <code>rs_helpexception</code> displays a detailed description of a transaction.</li><li>• Use <code>rs_delexception</code> to delete transactions in the exceptions log.</li></ul> |
| See also    | <code>rs_delexception</code>                                                                                                                                                                                                                                                                                                                                                                                                                              |

# rs\_helpfstring

**Description** Displays the parameters and function string text for functions associated with a replication definition.

**Syntax** rs\_helpfstring *replication\_definition*  
[, *function\_name*]

**Parameters** *replication\_definition*  
The table or function replication definition for which you want to view functions.

*function\_name*  
A string of characters that corresponds to a function name. The string must match an entire function name or the first part of a name.

**Examples** **Example 1** Displays parameters and function string text for all functions of the replication definition authors\_rep.

```
rs_helpfstring authors_rep
```

**Example 2** Displays parameters and function string text for the rs\_insert function of the replication definition authors\_rep.

```
rs_helpfstring authors_rep, rs_insert
```

Function String information for Replication Definition.  
'authors\_rep'

Valid Parameters are:

| Parameter Name | Datatype |
|----------------|----------|
| -----          | -----    |
| @au_id         | varchar  |
| @au_lname      | varchar  |
| @au_fname      | varchar  |
| @phone         | char     |
| @address       | varchar  |
| @city          | varchar  |
| @state         | char     |
| @country       | varchar  |
| @postalcode    | char     |

| Rep.Def.Name | Function Name | FString Name | FSClass Name                |
|--------------|---------------|--------------|-----------------------------|
| -----        | -----         | -----        | -----                       |
| authors_rep  | rs_insert     | rs_insert    | rs_sqlserver_function_class |

```
--- Begin FString Text ---

*** System-Supplied Transact-SQL Statement ***
--- End FString Text ---
```

**Usage**

- If you do not supply a *function\_name* parameter, `rs_helpfstring` displays all function strings defined for all functions of the replication definition.
- If you supply a *function\_name* string, `rs_helpfstring` displays function strings that match *function\_name*, such as `rs_insert`, `rs_delete`, `rs_update`, and `rs_select`, or a user-defined function.
- System-generated default function strings have no function string text stored in the RSSD. For these functions strings, `rs_helpfstring` displays the message “System-Supplied Transact-SQL Statement.”

# rs\_helpfunc

**Description** Displays information about functions available for a Replication Server or for a particular replication definition.

**Syntax** rs\_helpfunc [*replication\_definition* [, *function\_name*]]

**Parameters** *replication\_definition*  
The replication definition for which you want function information.

*function\_name*  
A string of characters that corresponds to a function name. The string must match an entire function name or the first part of a name.

**Examples** **Example 1** Displays all available functions, replication definitions, and primary Replication Servers. The class scope of each function is also displayed.

```
rs_helpfunc
```

**Example 2** Displays function information, including function names, parameters, and datatypes, for all functions of the replication definition authors\_rep.

```
rs_helpfunc authors_rep
```

```
Functions and Parameters for Replication Definition:
 'authors_rep'
```

```
System Function Names

rs_insert
rs_delete
rs_update
rs_select
rs_select_with_lock
```

| Parameter(s) | Datatype | Length |
|--------------|----------|--------|
| -----        | -----    | -----  |
| @state       | char     | 2      |
| @postalcode  | char     | 10     |
| @au_id       | varchar  | 11     |
| @phone       | char     | 12     |
| @country     | varchar  | 12     |
| @city        | varchar  | 20     |
| @au_fname    | varchar  | 20     |
| @address     | varchar  | 40     |
| @au_lname    | varchar  | 40     |

**Example 3** Displays parameters and datatypes for the `rs_insert` function of the replication definition `authors_rep`.

```
rs_helpfunc authors_rep, rs_insert
```

Usage

- If you do not specify any parameters, `rs_helpfunc` lists all functions defined in the Replication Server.
- If you supply a *replication\_definition* name, only the functions defined for that replication definition are listed. If you also supply a *function\_name* string, `rs_helpfunc` displays functions whose names match *function\_name*.
- `rs_helpfunc` notifies you if it detects duplicate user-defined functions that may interfere with asynchronous transactions.

# rs\_helppartition

- Description

Displays information about Replication Server partitions.
- Syntax

rs\_helppartition [*partition\_name*]
- Parameters

*partition\_name*

A string of characters that corresponds to a partition name. The string must match an entire partition name or the first part of a name.
- Examples

**Example 1** Displays summary information about all available database partitions for the Replication Server.

```
rs_helppartition

Displaying all partitions known to 'TOKYO_RS'.
Logical Name Size (MB) Segments Allocated (MB)

partition_1 20 3
```

**Example 2** Displays detailed information about the partition named partition\_1.

```
rs_helppartition partition_1

Information for stable device: 'partition_1' on 'TOKYO_RS'.
This device is active.
Physical Name Partition ID

/remote/tyrell2/app/dev/tokyo_rs_p1.dat 101
Partition Size (MB) Segments Allocated (MB)

20 5
Inbound Database Queue(s) on this partition:
Connection Name Number of Segments

LDS.pubs2 1
TOKYO_DS.TOKYO_RSSD 1
Outbound Database Queue(s) on this partition:
Connection Name Number of Segments

LDS.pubs2 1
TOKYO_DS.TOKYO_RSSD 1
Outbound Replication Server Queue(s) on this partition:
Connection Name Number of Segments

SYDNEY_RS 1
```

- Usage

- If you do not specify any parameters, rs\_helppartition lists summary information about all of the Replication Server's partitions.

- If you supply a *partition\_name* string, `rs_helppartition` displays information about any partition whose name matches *partition\_name*.
- If the *partition\_name* string exactly matches a partition name, detailed information about the partition displays, including logical and physical name, total size, number of 1MB segments allocated from each partition, and queues on the partition.
- If the *partition\_name* string does not exactly match a partition name, summary information displays for any partitions whose names match *partition\_name* or for all known partitions.

## rs\_helppub

**Description** Displays information about publications.

**Syntax** `rs_helppub [publication_name, primary_dataserver, primary_db, article_name]`

**Examples** **Example 1**

```
rs_helppub
```

| Publication Name | PRS     | Primary DS.DB |
|------------------|---------|---------------|
| -----            | -----   | -----         |
| funcpub          | prim_rs | P_DS.pdb1     |
| pub1             | prim_rs | P_DS.pdb1     |
| pub2             | prim_rs | P_DS.pdb1     |

| Num Articles | Status | Request Date        |
|--------------|--------|---------------------|
| -----        | -----  | -----               |
| 3            | Valid  | Mar 23 1998 11:51AM |
| 7            | Valid  | Mar 24 1998 10:41AM |
| 3            | Valid  | Mar 24 1998 11:50AM |

```
(return status = 0)
```

**Example 2**

```
rs_helppub funcpub:
```

| Publication Name | PRS     | Primary DS.DB |
|------------------|---------|---------------|
| -----            | -----   | -----         |
| funcpub          | prim_rs | P_DS.pdb1     |

| Num Articles | Status | Request Date |
|--------------|--------|--------------|
| -----        | -----  | -----        |

```

3 Valid Mar 23 1998 11:51AM

Article Name Replication Definition Type
----- -
authors authors
authors authors
publishers publishers

Primary Object Name Replicate Object Name Request Date
----- -----
many_rows_data many_rows_data Mar 23 1998 10:01AM
 Mar 23 1998 11:51AM

Sub Name Replicate DS.DB Owner Req. Date
----- -----
funcsub1 R_DS.rdb1 sa Mar 24 1998 11:12AM

(return status = 0)

```

### Example 3

rs\_helppub funcpub, P\_DS, pdb1, publishers:

```

Article Name Publication Name Replication Definition

publishers funcpub publishers

Primary Object Name Replicate Object Name

publishers publishers

Type Request Date Status

Table Mar 23 1998 11:51AM Valid

Where clauses

 where
 pub_id = "0736"

Sub. Name Replicate DS.DB Owner Req Date

funcsub1 R_DS.rdb1 sa Mar 24 1998 11:12AM

(return status = 0)

```

- Usage**
- If `rs_helppub` is executed at the primary site, information displays for all of the publications created at that site.
  - If `rs_helppub` is executed at the replicate site, information is displayed only for publications for which subscriptions have been created at that site.
  - Use `rs_helppubsub` to display information about subscriptions to publications or articles.
  - Use `check_subscription` to get the most accurate report of subscription status.
- See also** `rs_helppubsub`

## rs\_helppubsub

**Description** Displays information about publication subscriptions and article subscriptions.

**Syntax** `rs_helppubsub subscription_name, publication_name, primary_dataserver, primary_db, replicate_dataserver, replicate_db`

**Examples** **Example 1** Lists all publication subscriptions known at this site:

```
rs_helppubsub
```

| Subscription Name | Publication Name |
|-------------------|------------------|
| -----             | -----            |
| funcsub1          | funcpub          |

| Primary DS.DB | Replicate DS.DB | PRS Status | RRS Status |
|---------------|-----------------|------------|------------|
| -----         | -----           | -----      | -----      |
| P_DS.pdb1     | R_DS.rdb1       | Unknown    | Valid      |

| Owner | Request Date        |
|-------|---------------------|
| ----- | -----               |
| sa    | Mar 24 2007 11:12AM |

(1 row affected)

| Subscription Name | Article Name | Replication Definition |
|-------------------|--------------|------------------------|
| -----             | -----        | -----                  |
| funcsub1          | authors      | authors                |

| PRS Status | RRS Status | Request Date        | Autocorrection |
|------------|------------|---------------------|----------------|
| -----      | -----      | -----               | -----          |
| Unknown    | Valid      | Mar 24 2007 11:11AM | off            |

| Subscribe to Truncate Table | Dynamic SQL |
|-----------------------------|-------------|
|-----------------------------|-------------|

```

Unknown On
(1 row affected, return status = 0)
```

**Example 2** Lists all publication subscriptions named *sub*.

```
rs_helppubsub sub
```

**Example 3** Lists all publication subscriptions named *sub* for publications named *pub*.

```
rs_helppubsub sub, pub
```

**Example 4** Lists all subscriptions named *sub* for the specified publication.

```
rs_helppubsub sub, pub, primary_dataserver, primary_db
```

**Example 5** Lists the publication subscription and the article subscriptions in the group.

```
rs_helppubsub sub, pub, primary_dataserver, primary_db,
replicate_dataserver, replicate_db
```

| Subscription Name |  | Publication Name            |  | Primary DS.DB     |  |
|-------------------|--|-----------------------------|--|-------------------|--|
| -----             |  | -----                       |  | -----             |  |
| sub               |  | pub                         |  | ost_cardhu_2.pdb1 |  |
| Replicate DS.DB   |  | PRS Status                  |  | RRS Status        |  |
| -----             |  | -----                       |  | -----             |  |
| ost_cardhu_2.rdb1 |  | Unknown                     |  | Valid             |  |
|                   |  |                             |  | rdb1_owner        |  |
| Request Date      |  | Subscription Name           |  | Article Name      |  |
| -----             |  | -----                       |  | -----             |  |
| February 25 1998  |  | sub                         |  | article1          |  |
|                   |  |                             |  | article2          |  |
|                   |  | sub                         |  | article3          |  |
|                   |  | sub                         |  | article4          |  |
|                   |  | sub                         |  | article5          |  |
| PRS Status        |  | RRS Status                  |  | Request Date      |  |
| -----             |  | -----                       |  | -----             |  |
| Unknown           |  | VALID                       |  | Feb 25, 1998      |  |
|                   |  |                             |  | repdef1           |  |
|                   |  |                             |  | repdef2           |  |
| Unknown           |  | VALID                       |  | Feb 25, 1998      |  |
| Unknown           |  | VALID                       |  | Feb 25, 1998      |  |
| Unknown           |  | VALID                       |  | Feb 25, 1998      |  |
|                   |  |                             |  | repdef3           |  |
|                   |  |                             |  | repdef4           |  |
|                   |  |                             |  | repdef5           |  |
| Autocorrection    |  | Subscribe to Truncate Table |  | Dynamic SQL       |  |
| -----             |  | -----                       |  | -----             |  |
| on                |  | off                         |  | on                |  |

|     |     |    |
|-----|-----|----|
| off | on  | on |
| off | off | on |
| off | off | on |

- Usage
- `rs_helppub` Use to determine all subscriptions for an article or a publication.
  - Use `check_subscription` to get the most accurate report of subscription status.
- See also
- `rs_helppub`

# rs\_helprep

- Description

Displays information about replication definitions.
- Syntax

rs\_helprep [*replication\_definition*]
- Parameters

*replication\_definition*

A string of characters that corresponds to a replication definition name. The string must match an entire replication definition name or the first part of a name.
- Examples

Example 1

rs\_helprep

| Rep def         | PRS       | Primary DS.DB  | Primary table   | Replicate table    | Type |
|-----------------|-----------|----------------|-----------------|--------------------|------|
| authors         | cardhu_11 | cardhu_10.pdb1 | authors         | ling.authors_r1    | Tbl  |
| authors1        | cardhu_11 | cardhu_10.pdb1 | authors         | authors_r2         | Tbl  |
| discounts       | cardhu_11 | cardhu_10.pdb1 | discounts       | discounts          | Tbl  |
| publishers      | cardhu_11 | cardhu_10.pdb1 | publishers      | ling.publishers_r1 | Tbl  |
| publishers1     | cardhu_11 | cardhu_10.pdb1 | publishers      | publishers_r2      | Tbl  |
| roysched        | cardhu_11 | cardhu_10.pdb1 | roysched        | roysched           | Tbl  |
| rs_classes      | cardhu_11 | cardhu_10.emb  | rs_classes      | Tbl                |      |
| rs_columns      | cardhu_11 | cardhu_10.emb  | rs_columns      | Tbl                |      |
| rs_databases    | cardhu_11 | cardhu_10.emb  | rs_databases    | Tbl                |      |
| rs_erroractions | cardhu_11 | cardhu_10.emb  | rs_erroractions | Tbl                |      |
| rs_funcstrings  | cardhu_11 | cardhu_10.emb  | rs_funcstrings  | Tbl                |      |
| rs_functions    | cardhu_11 | cardhu_10.emb  | rs_functions    | Tbl                |      |
| rs_objects      | cardhu_11 | cardhu_10.emb  | rs_objects      | Tbl                |      |
| rs_routes       | cardhu_11 | cardhu_10.emb  | rs_routes       | Tbl                |      |
| rs_systext      | cardhu_11 | cardhu_10.emb  | rs_systext      | Tbl                |      |

**Example 2** Displays information about the authors replication definition which was created using create function replication definition:

rs\_helprep authors

| Replication Definition Name | PRS        | Type | Creation Date       |
|-----------------------------|------------|------|---------------------|
| authors                     | primary_rs | Tbl  | Nov 26, 2008 1:48PM |

| PDS.DB  | Primary Owner | Primary Table |
|---------|---------------|---------------|
| pds.pdb |               | authors       |

| Replicate Owner | Replicate Table |
|-----------------|-----------------|
|                 |                 |

authors

```

Send Min Cols. Used by Standby Min Vers Dynamic SQL SQL Stmt.Rep.

No No 1000 On UD

Col. Name Rep. Col. Name Datatype Len. Pri. Col. Searchable

au_id au_id varchar 11 1 1
au_lname au_lname varchar 40 0 1
au_fname au_fname varchar 20 0 1

```

**Example 3** Displays information about the R1\_app replication definition which was created using create applied function replication definition:

```
rs_helprep R1_app
```

```

Replication Definition Name PRS Type Creation Date

R1_app ost_replnx4_12 Func Feb 22 2008 12:15PM

PDS.DB Primary Function Replicate Function Used by Standby Func_type

PDS.pdb1 R1 R1_rep No Applied

Parameter Datatype Length Searchable

a int 4 0

Function Name FString Class FString Source FString Name

R1 rs_sqlserver_function_class Class Default R1

Subscriptions known at this Site 'ost_replnx4_12'.

Subscription Name Replicate DS.DB Owner Creation Date

(return status = 0)

```

**Example 4** Displays information about the R1\_req replication definition which was created using create request function replication definition:

```
rs_helprep R1_req
```

```

Replication Definition Name PRS Type Creation Date

```

```

R1_req ost_replnx4_12 Func Feb 22 2008 12:15PM

PDS.DB Primary Function Replicate Function Used by Standby Func_type

PDS.pdb1 R2 R2_rep No Request

Parameter Datatype Length Searchable

a int 4 0

Function Name FString Class FString Source FString Name

R2 rs_sqlserver_function_class Class Default R2

```

Subscriptions known at this Site 'ost\_replnx4\_12'.

```

Subscription Name Replicate DS.DB Owner Creation Date

```

(return status = 0)

**Example 5** Given this table and replication definition:

```

create table t1 (c1 int, c2 int)
create replication definition r1
 with primary at ost_wasatch_08.pdb1
 with all tables named t1
 (c1 int, "c2" int quoted)
 primary key (c1)

```

rs\_helprep r1 displays c2 as a quoted identifier:

```

Replication Definition Name PRS Type Creation Date

r1 ost_wasatch_09 Tbl Nov 11, 2008 2:28PM

PDS.DB Primary Owner Primary Table

ost_wasatch_08.pdb1 t1

Replicate Owner Replicate Table

t1

```

Send Min Cols. Used by Standby Min Vers Dynamic SQL SQL Stmt. Rep.

```

No No 1000 On None

Col. Name Rep. Col. Name Datatype Len. Pri. Col. Searchable

c1 c1 int 4 1 0
"c2" "c2" int 4 0 0

Function Name FString Class FString Source FString Name

-
rs_delete rs_sqlserver_function_class Class Default rs_delete
rs_insert rs_sqlserver_function_class Class Default rs_insert
rs_select rs_sqlserver_function_class Class Default rs_select
rs_select_ rs_sqlserver_function_class Class Default rs_select_
with_lock
rs_truncate rs_sqlserver_function_class Class Default rs_truncate
rs_update rs_sqlserver_function_class Class Default rs_update

Subscriptions known at this Site 'ost_wasatch_09'.
Subscription Name Replicate DS.DB Owner Creation Date

(return status = 0)

```

**Example 6** Given the table and replication definition defined in the preceding example, when you define t1 as a quoted identifier:

```

alter replication definition r1
alter replicate table name "t1" quoted

```

rs\_helprep r1 displays c2 and t1 as quoted identifiers:

```

Replication Definition Name PRS Type Creation Date

r1 ost_wasatch_09 Tbl Nov 11, 2008 2:28PM

PDS.DB Primary Owner Primary Table

ost_wasatch_08.pdb1 "t1"

Replicate Owner Replicate Table

 "t1"

Send Min Cols. Used by Standby Min Vers Dynamic SQL SQL Stmt. Rep.

No No 1000 On None

Col. Name Rep. Col. Name Datatype Len. Pri. Col. Searchable

```

```

c1 c1 int 4 1 0
"c2" "c2" int 4 0 0

Function Name FString Class FString Source FString Name

-
rs_delete rs_sqlserver_function_class Class Default rs_delete
rs_insert rs_sqlserver_function_class Class Default rs_insert
rs_select rs_sqlserver_function_class Class Default rs_select
rs_select_ rs_sqlserver_function_class Class Default rs_select_
with_lock
rs_truncate rs_sqlserver_function_class Class Default rs_truncate
rs_update rs_sqlserver_function_class Class Default rs_update

Subscriptions known at this Site 'ost_wasatch_09'.
Subscription Name Replicate DS.DB Owner Creation Date

(return status = 0)
```

**Example 7** Given the replication definition defined in the preceding example, when you define c2 as not quoted:

```
alter replication definition r1
alter columns c2 not quoted
```

rs\_helprep r1 displays t1 as the only quoted identifier:

```
Replication Definition Name PRS Type Creation Date

r1 ost_wasatch_09 Tbl Nov 11, 2008 2:28PM

PDS.DB Primary Owner Primary Table

ost_wasatch_08.pdb1 "t1"

Replicate Owner Replicate Table

 "t1"

Send Min Cols. Used by Standby Min Vers Dynamic SQL SQL Stmt. Rep.

No No 1000 On None

Col. Name Rep. Col. Name Datatype Len. Pri. Col. Searchable

c1 c1 int 4 1 0
c2 c2 int 4 0 0
```

| Function Name           | FString Class               | FString Source | FString Name            |
|-------------------------|-----------------------------|----------------|-------------------------|
| -----                   | -----                       | -----          | -----                   |
| -                       |                             |                |                         |
| rs_delete               | rs_sqlserver_function_class | Class Default  | rs_delete               |
| rs_insert               | rs_sqlserver_function_class | Class Default  | rs_insert               |
| rs_select               | rs_sqlserver_function_class | Class Default  | rs_select               |
| rs_select_<br>with_lock | rs_sqlserver_function_class | Class Default  | rs_select_<br>with_lock |
| rs_truncate             | rs_sqlserver_function_class | Class Default  | rs_truncate             |
| rs_update               | rs_sqlserver_function_class | Class Default  | rs_update               |

Subscriptions known at this Site 'ost\_wasatch\_09'.

| Subscription Name | Replicate DS.DB | Owner | Creation Date |
|-------------------|-----------------|-------|---------------|
| -----             | -----           | ----- | -----         |

(return status = 0)

**Example 8** To display information about the “authors” replication definition created using create function replication definition, enter:

```
rs_helprep authors
```

See the Ref Objowner and Ref Objname columns in the Output:

| Replication Definition Name | PRS        | Type  | Creation Date       |
|-----------------------------|------------|-------|---------------------|
| -----                       | -----      | ----- | -----               |
| authors                     | primary_rs | Tbl   | Nov 26, 2008 1:48PM |

| PDS.DB | Primary Owner | Primary Table |
|--------|---------------|---------------|
| -----  | -----         | -----         |

|         |         |
|---------|---------|
| pds.pdb | authors |
|---------|---------|

| Replicate Owner | Replicate Table |
|-----------------|-----------------|
| -----           | -----           |

authors

Send Min Cols. Used by Standby Min Vers Dynamic SQL SQL Stmt.Rep.

|       |       |       |       |       |
|-------|-------|-------|-------|-------|
| ----- | ----- | ----- | ----- | ----- |
| No    | No    | 1000  | On    | UD    |

| Col. Name | Rep. Col. Name | Datatype | Len. | Pri. Col. | Searchable |
|-----------|----------------|----------|------|-----------|------------|
| -----     | -----          | -----    | ---- | -----     | -----      |
| au_id     | au_id          | varchar  | 11   | 1         | 1          |
| au_lname  | au_lname       | varchar  | 40   | 0         | 1          |
| au_fname  | au_fname       | varchar  | 20   | 0         | 1          |

| Ref. Objowner | Ref. Objname |
|---------------|--------------|
| -----         | -----        |

table2

#### Usage

- Unless you enter parameters, `rs_helprep` lists summary information for all replication definitions in the Replication Server.
- If you supply a *replication\_definition* string, `rs_helprep` displays information about any replication definition whose name matches *replication\_definition*.
- If the *replication\_definition* string matches exactly one replication definition name, detailed information about that replication definition displays. Information includes the primary Replication Server, data server and database, replication definition columns, functions defined for the replication definition, and subscriptions for the replication definition known by the Replication Server.
- The detailed information displayed is slightly different for table replication definitions, function replication definitions, and system table replication definitions.
- If the *replication\_definition* string does not match exactly one replication definition name, summary information is displayed for any replication definitions that match *replication\_definition*.
- Quoted identifiers are displayed enclosed in double quote characters.
- `rs_helprep` displays information about table references for real-time loading (RTL) and high volume adaptive replication (HVAR).
- `rs_helprep` does not display database replication definition. Use `rs_helpdbrep` to display database replication definition.

## rs\_helprepdb

|             |                                                                                                                                                                                       |
|-------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Description | Displays information about databases with subscriptions for replication definitions in the current Replication Server.                                                                |
| Syntax      | <code>rs_helprepdb [data_server, database]</code>                                                                                                                                     |
| Parameters  | <p><i>data_server</i><br/>The data server with the database whose information you want to display.</p> <p><i>database</i><br/>The database whose information you want to display.</p> |

**Examples** **Example 1** Displays information about all databases with subscriptions for replication definitions in the current Replication Server.

```
rs_helprepdb
```

| dsname    | dbname      | dbid  | controlling_prs |
|-----------|-------------|-------|-----------------|
| -----     | -----       | ----- | -----           |
| SYDNEY_DS | SYDNEY_RSSD | 102   | SYNDEY_RS       |

**Example 2** Displays information about the specified data server and database.

```
rs_helprepdb SYDNEY_DS, pubs2
```

| dsname    | dbname | dbid  | controlling_prs |
|-----------|--------|-------|-----------------|
| -----     | -----  | ----- | -----           |
| SYDNEY_DS | pubs2  | 104   | SYDNEY_RS       |

- Usage**
- Execute `rs_helprepdb` in the RSSD for the primary Replication Server.
  - Unless you specify *data\_server* and *database* parameters, `rs_helprepdb` lists all databases with subscriptions for any of the Replication Server's replication definitions. The database ID and managing Replication Server display for each data server and database.
  - If you supply the *data\_server* and *database* parameters, `rs_helprepdb` displays information about the specified database only.

# rs\_helprepversion

Description                Displays information on replication definition versions in the current Replication Server.

Syntax                    rs\_helprepversion {repdef\_name | repdef\_version\_id}

Parameters                repdef\_name  
                            The replication definition name.

                            repdef\_version\_id  
                            The replication definition version ID.

Examples                  **Example 1** Displays information on all versions of the replication definition when you provide the replication definition name—types11\_pdb1.

```
rs_helprepversion types11_pdb1
```

The output is:

| Repdef Version Name            | Repdef Version ID  | Active Inbound | Active Oubound |
|--------------------------------|--------------------|----------------|----------------|
| -----                          | -----              | -----          | -----          |
| types11_pdb1                   | 0x0107006500000067 | Yes            | No             |
| rs_drp01060065000000674a955c45 | 0x0106006500000067 | No             | Yes            |
| rs_drp01050065000000674a955c40 | 0x0105006500000067 | No             | No             |
| rs_drp01040065000000674a955c3f | 0x0104006500000067 | No             | No             |
| rs_drp01030065000000674a955c3d | 0x0103006500000067 | No             | No             |
| rs_drp01020065000000674a955c3c | 0x0102006500000067 | No             | No             |
| rs_drp01010065000000674a955c3b | 0x0101006500000067 | No             | No             |
| rs_drp01000065000000674a955c3a | 0x0100006500000067 | No             | No             |
| (return status = 0)            |                    |                |                |

**Example 2** If you specify a replication definition version by providing the replication definition version ID, such as 0x0106006500000067 in this example, rs\_helprepversion displays both the general and column information of the replication definition version:

```
rs_helprepversion 0x0106006500000067
```

The output is:

| Repdef Version Name            | Repdef Version ID  | Active Inbound | Active Oubound |
|--------------------------------|--------------------|----------------|----------------|
| -----                          | -----              | -----          | -----          |
| rs_drp01060065000000674a955c45 | 0x0106006500000067 | No             | Yes            |

| Column<br>Name | Replicate<br>Col Name | Datatype   | Len | Pri<br>Col | Searchable | Ref<br>Objowner | Ref<br>Objname |
|----------------|-----------------------|------------|-----|------------|------------|-----------------|----------------|
| charcol        | charcol               | varchar    | 255 | 0          | 0          |                 |                |
| floatcol       | floatcol              | float      | 8   | 0          | 0          |                 |                |
| datecol        | datecol               | datetime   | 8   | 0          | 0          |                 |                |
| smdatecol      | smdatecol             | smalldatet | 4   | 0          | 0          |                 |                |
| moneycol       | moneycol              | money      | 8   | 0          | 0          |                 |                |
| smmoneycol     | smmoneycol            | smallmoney | 4   | 0          | 0          |                 |                |
| intcol         | intcol                | int        | 4   | 0          | 0          |                 |                |
| smintcol       | smintcol              | smallint   | 2   | 0          | 0          |                 |                |
| tinyintcol     | tinyintcol            | tinyint    | 1   | 0          | 0          |                 |                |
| row_num        | row_num               | int        | 4   | 1          | 0          |                 |                |

(return status = 0)

Usage `rs_helprepversion` displays information on replication definition versions:

- Active inbound replication definition version – used by the Executor to pack data into the inbound queue.
- Active outbound replication definition version – used by the Distributor to pack data into the out bound queues.

See also `alter replication definition`, `alter applied function replication definition`, `alter request function replication definition`, `rs_helprep`, `rs_send_repserver_cmd`

# rs\_helproute

|             |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
|-------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Description | Provides status information about routes.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| Syntax      | rs_helproute [ <i>replication_server</i> ]                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
| Parameters  | <div>replication_server</div> <div>The name of a Replication Server for which you want route status information.</div>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
| Examples    | <div>The route from TOKYO_RS to SYDNEY_RS is currently active.</div> <div><pre>rs_helproute  route                                route_status ----- TOKYO_RS -----&gt; SYDNEY_RS         Active</pre></div>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| Usage       | <div><ul style="list-style-type: none"><li>Unless you specify the <i>replication_server</i> parameter, rs_helproute displays information for all the routes known to the current Replication Server.</li><li>If you supply a <i>replication_server</i>, information displays only for routes to and from that Replication Server.</li><li>Replication Server uses a defined protocol to create and drop a route between the source and destination Replication Servers. During this protocol, the route goes through various states. rs_helproute, executed on the RSSD at the source or destination Replication Server, shows the current state of the protocol.</li><li>For each route, rs_helproute returns two types of information:<ul style="list-style-type: none"><li>Route status</li></ul></li></ul></div> <div>Status reflects the state of the route protocol. The information for each route depends on where you execute rs_helproute—at the route’s source or destination.</div> <div><ul style="list-style-type: none"><li>List of system table subscriptions</li></ul></div> <div>If you are creating a route, information is displayed about system table subscriptions that are being created. If you are dropping a route, this list tells you which system table subscriptions are being dropped.</div> <div>Routing protocols usually process system table subscriptions. This information helps you determine which subscriptions prevent you from proceeding to the next step in the protocol. If no system table subscriptions are listed, the protocol is currently not having problems with system table subscriptions.</div> |

Incomplete materialization or dematerialization of system table subscriptions is a common problem. If you notice any problems while creating, dropping, or altering routes, examine `rs_helproute` output for information about subscription status.

## rs\_helpsub

|             |                                           |
|-------------|-------------------------------------------|
| Description | Displays information about subscriptions. |
|-------------|-------------------------------------------|

Syntax

```
rs_helpsub
[subscription_name [, replication_definition
[, data_server, database]]]
```

|            |                                                                                                                                                                                       |
|------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Parameters | <p><i>subscription_name</i></p> <p>A string of characters that corresponds to a subscription name. The string must match an entire subscription name or the first part of a name.</p> |
|------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

*replication\_definition*  
The replication definition subscribed to.

*data\_server*  
The data server with the database containing the subscription's data.

*database*  
The database containing the subscription's data.

**Example 1** Displays summary information about all available subscriptions. The “Unknown” status in the RRS column reflects the fact that the current Replication Server has no knowledge of the subscription status at the listed Replication Server (the primary Replication Server):

rs\_helpsub

**\*\* This Site is primary\_rs \*\***

| Subscription Name | Rep. Def. Name | Replicate DS.DB | A/C | Status at |       |
|-------------------|----------------|-----------------|-----|-----------|-------|
|                   |                |                 |     | RRS       | PRS   |
| authors_1         | authors        | RDS.rdb         | 0   | Unknown   | Valid |
| many_rows_1       | many_rows      | RDS.rdb         | 0   | Unknown   | Valid |
| publishers_1      | publishers     | RDS.rdb         | 0   | Unknown   | Valid |
| titleauthor_1     | titleauthor    | RDS.rdb         | 0   | Unknown   | Valid |
| titles_1          | titles         | RDS.rdb         | 0   | Unknown   | Valid |

## Dynamic SQL

\_\_\_\_\_

On

On

On

On

On

```
(return status = 0)
```

**Example 2** Displays detailed information about the authors\_sub subscription:

```

rs_helpsub authors_sub
Subscription Name Rep. Def. Name Replicate DS.DB A/C RRS PRS

authors_sub authors_rep RDS.rdb 0 Defined Unknown

Dynamic SQL Owner Creation Date

On sa Oct 2 2007

Subscription Text

create subscription authors_sub
 for authors_rep
 with replicate at RDS.rdb
 where
 state = "CA"
(return status = 0)

```

**Usage**

- If you do not specify any parameters, `rs_helpsub` lists summary information about all subscriptions defined in the Replication Server. Information include replication definitions, replicate data server and database, autocorrection status, and subscription materialization status at the replicate and primary Replication Server.
- If you supply a *subscription\_name* string, `rs_helpsub` displays information about any subscription whose name matches *subscription\_name*.
- If the *subscription\_name* string matches exactly one subscription name, the owner, creation date, and text of the subscription also display.
- If the *subscription\_name* string does not match exactly one subscription name, summary information displays for any subscriptions whose names match *subscription\_name*.
- If you also supply a *replication\_definition*, `rs_helpsub` displays information only for subscriptions to that replication definition.
- `rs_helpsub` does not display subscription replication definition. Use `rs_helpdbsub` to display subscription replication definition.

# rs\_helpuser

|             |                                                                            |
|-------------|----------------------------------------------------------------------------|
| Description | Displays information about user login names known to a Replication Server. |
| Syntax      | rs_helpuser [user]                                                         |
| Parameters  | <i>user</i><br>The user login name about which you want information.       |
| Examples    | <b>Example 1</b> Displays information about all users.                     |

```
rs_helpuser
Users and Privileges Known at Site repl_rs

Primary Users
User Name Permission(s) Name

TOKYO_RS_id_user no grants
sa sa
TOKYO_RS_ra connect source
TOKYO_RS_rsi connect source
repuser create object
TOKYO_RSSD_prim connect source, primary subscr

Maintenance Users
User name Destination DS.DB

TOKYO_RSSD_maint TOKYO_DS.TOKYO_RSSD
pubs2_maint TOKYO_DS.pubs2
pubs2_maint SYDNEY_DS.pubs2sb
```

**Example 2** Displays information about the pubs2\_maint user.

```
rs_helpuser pubs2_maint
Users and Privileges Known at Site TOKYO_RS

Primary User(s)
User Name Permission Name

pubs2_maint TOKYO_DS.pubs2
pubs2_maint SYDNEY_DS.pubs2sb
```

|       |                                                                                                                                                                                                                                                                                                               |
|-------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Usage | <ul style="list-style-type: none"><li>Unless you enter parameters, rs_helpuser displays information about all user login names known to the current Replication Server.</li><li>If you supply a <i>user</i> login name parameter, rs_helpuser displays information about that user login name only.</li></ul> |
|-------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

## rs\_helpreptable

**Description** Displays information about replication definitions created against a primary table.

**Syntax** `rs_helpreptable database, [owner,] table`

**Parameters** *database*

The database where the table is created.

*owner*

The owner of the table.

*table*

The name of the table.

**Examples** `rs_helpreptable pdb1, authors`

| Replication<br>definition<br>name | Primary<br>owner | Primary<br>table | Primary<br>owner | Replicate<br>table | Used<br>standby | Min vers |
|-----------------------------------|------------------|------------------|------------------|--------------------|-----------------|----------|
| authors                           |                  | authors          | ling             | authors_r1         | Yes             | 1000     |
| authors1                          |                  | authors          |                  | authors_r2         | No              | 1000     |

**Usage**

- Only user-defined table replication definitions are displayed.

## rs\_init\_erroractions

|             |                                                                                                                                                                                                                                                                                                                                                                                         |
|-------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Description | Initializes a new error class.                                                                                                                                                                                                                                                                                                                                                          |
|             | <hr/> <b>Note</b> rs_init_erroractions will be deprecated. To initialize new classes, Sybase suggests that you use create error class with the set template to option. <hr/>                                                                                                                                                                                                            |
| Syntax      | rs_init_erroractions <i>new_error_class</i> , <i>template_class</i>                                                                                                                                                                                                                                                                                                                     |
| Parameters  | <i>new_error_class</i><br>The name of the new error class you have created.<br><br><i>template_class</i><br>The name of the error class that you want to serve as a template for the new error class.                                                                                                                                                                                   |
| Examples    | Creates the error class <i>new_class</i> , based on the template error class, <i>rs_sqlserver_error_class</i> .<br><br><pre>rs_init_erroractions new_class,<br/>rs_sqlserver_error_class</pre>                                                                                                                                                                                          |
| Usage       | <ul style="list-style-type: none"><li>• The template error class may be a user-defined error class or a system-provided error class such as <i>rs_sqlserver_error_class</i>.</li><li>• Use the create error class command to create the new error class in the primary Replication Server for that error class. Then use <i>rs_init_erroractions</i> to initialize the class.</li></ul> |
| See also    | create error class                                                                                                                                                                                                                                                                                                                                                                      |

## rs\_send\_repserver\_cmd

|             |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
|-------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Description | Executes replication definition change requests directly at the primary database.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| Syntax      | <code>rs_send_repserver_cmd 'rs_api'</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
| Parameters  | <p><i>rs_api</i></p> <p>Contains the replication definition Replication Command Language (RCL) command and parameters you specify for <code>rs_send_repserver_cmd</code>. <i>rs_api</i> is a varchar parameter with a maximum length of 16370 bytes for Adaptive Server, 4000 bytes for Oracle, and 8000 bytes for Microsoft SQL Server.</p> <p>Enclose <i>rs_api</i> in single quotes, and replace each single quote inside the string with two single quotes.</p> <p>If the parameter length for <i>rs_api</i> is too short for a create or alter replication definition request, you can split the request into two or more requests.</p>                                                                                                                                                                                                                              |
| Examples    | <p><b>Example 1</b> Execute the “authors” alter replication definition request at the primary database to drop the address, city, state, and zip columns:</p> <pre>exec rs_send_repserver_cmd 'alter replication definition authors drop address, city, state, zip'</pre> <p><b>Example 2</b> If a replication definition RCL is longer than the maximum length allowed for <i>rs_api</i>, you can split the request into two or more requests.</p> <pre>exec rs_send_repserver_cmd 'alter replication definition authors drop address, city' go exec rs_send_repserver_cmd 'alter replication definition authors drop state, zip'</pre> <p><b>Example 3</b> In this example, you need to enclose “authors” in double quotes, and ‘off’ in two single quotes:</p> <pre>exec rs_send_repserver_cmd 'alter replication definition "authors" replicate sqldml ``off``'</pre> |
| Usage       | <ul style="list-style-type: none"> <li>Before you use <code>rs_send_repserver_cmd</code> at the primary database, use <code>admin verify_repserver_cmd</code> to verify that you can execute the replication definition request successfully at the Replication Server. See “admin verify_repserver_cmd” on page 97.</li> <li>Replication Server supports <code>rs_send_repserver_cmd</code> for these replication definition commands: <ul style="list-style-type: none"> <li>alter replication definition</li> <li>create replication definition</li> </ul> </li> </ul>                                                                                                                                                                                                                                                                                                 |

- drop replication definition
- alter applied function replication definition
- create applied function replication definition
- alter request function replication definition
- create request function replication definition

---

**Note** Besides Adaptive Server, Replication Server extends support for `rs_send_repserver_cmd` to supported versions of these non-ASE databases: Microsoft SQL Server and Oracle. See the *Release Bulletin* for Replication Agent for the supported database versions.

---

- When you execute `rs_send_repserver_cmd` at the primary database, the Replication Agent sends the RCL command stored in `rs_api` to the Replication Server, which then executes the RCL command. This ensures that Replication Server replicates the primary data with the proper replication definition version—primary data before the `rs_send_repserver_cmd` is replicated with the old replication definition version, while primary data after the `rs_send_repserver_cmd` is replicated with the new replication definition version.
- You do not always need to issue replication definition change requests directly from a primary data server. For example, you can execute the alter replication definition request directly from the primary Replication Server in these situations:
  - If there is no subscription to the replication definition
  - If there are subscriptions to the replication definition, but there is no data in the primary database log for the table or stored procedure
  - If you are adding or dropping a searchable column to or from a table replication definition
  - If you are adding or dropping a searchable parameter to or from a function replication definition
  - If you are altering a replication definition to turn Dynamic SQL on or off

---

**Warning!** As Replication Server accepts all commands that Replication Agent sends to Replication Server, you must control access to `rs_send_repserver_cmd` at the primary database.

---

**See also**

alter replication definition, create replication definition, drop replication definition, alter applied function replication definition, create applied function replication definition, alter request function replication definition, create request function replication definition, admin verify\_repserver\_cmd, sysadmin  
skip\_bad\_repserver\_cmd

## rs\_ticket

|             |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
|-------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Description | A stored procedure in the primary database that monitors Replication Server performance, module heartbeat, replication health and table-level quiesce.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| Syntax      | <code>rs_ticket h1 [, h2 [, h3 [, h4]]]</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| Parameters  | <code>h1 [, h2 [, h3 [, h4]]]</code><br>Header information in short varchar strings.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| Examples    | <p><b>Example 1</b> Executes <code>rs_ticket</code> at regular intervals:</p> <pre>Exec rs_ticket 'heartbeat', 'beat-sequence-number'</pre> <p><b>Example 2</b> To measure performance, execute the following from the primary database:</p> <pre>Exec rs_ticket 'start'<br/>Execute replication benchmarks<br/>Exec rs_ticket 'stop'</pre>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| Usage       | <ul style="list-style-type: none"><li>The <code>rs_ticket</code> stored procedure has a ticket version number <code>V=2</code> and a ticket size of 1024 bytes.</li><li>If your application understands only version 1 ticket, call <code>rs_ticket_v1</code> to generate ticket in version 1 format. The <code>rs_ticket_v1</code> syntax is:<br/><pre>rs_ticket_v1 h1 [, h2 [, h3 [, h4]]]</pre></li><li><code>rs_ticket</code> executes the following command:<br/><pre>rs_marker 'rs_ticket rs_ticket_param'</pre><p>To avoid issuing wrongly formatted <code>rs_marker</code> and to enforce the <code>rs_ticket_param</code> standard, you should invoke <code>rs_ticket</code> instead of <code>rs_marker</code>. If you call <code>rs_marker</code> directly and form an incorrect <code>rs_marker</code> subcommand, the Replication Server refuses the <code>rs_marker</code> and shuts down the RepAgent connection. In this case, you must skip <code>rs_marker</code> from the transaction log, which may cause data loss.</p></li><li>The Replication Server EXEC, DIST, RSI, and DSI modules parse and process <code>rs_ticket</code> subcommand:<ul style="list-style-type: none"><li>When EXEC processes <code>rs_ticket</code>, it appends a timestamp, and then the total bytes received from RepAgent after <code>rs_ticket_param</code>. An EXEC timestamp takes the form "EXEC(spид)=mm/dd/yy hh:mm:ss.ddd". The byte information is "B(spид)=ddd". EXEC writes <code>rs_ticket</code> back to inbound queue.</li></ul></li></ul> |

- When DIST processes `rs_ticket`, it appends another timestamp to `rs_ticket_param`. A DIST timestamp takes the form "DIST(spид)=mm/dd/yy hh:mm:ss.ddd".
- When RSI processes `rs_ticket`, it appends yet another timestamp to `rs_ticket_param`. An RSI timestamp takes the form "RSI(spид)=mm/dd/yy hh:mm:ss.ddd".
- When DSI processes `rs_ticket`, it appends yet another timestamp to `rs_ticket_param`. A DSI timestamp takes the form "DSI(spид)=mm/dd/yy hh:mm:ss.ddd".
- There are no subscriptions for `rs_ticket`. DIST does not send `rs_ticket` to DSI unless there is at least one subscription from the replicate site.
- `rs_ticket` is lightweight and nonintrusive and can be used in test environments as well as production environments.
- `rs_ticket` lets you know, without quiescing the Replication Server, when the data has been completely flushed out of replication path.
- The movement of `rs_ticket` is tracked by the EXEC, DIST, RSI, and DSI threads through RSTicket counter. Each thread has one RSTicket counter which is increased by one whenever the corresponding thread receives `rs_ticket`. This counter is never reset.

You can monitor the module that `rs_ticket` has reached by sampling the RSTicket counters. RMS or other Replication Server monitoring tool uses these counters to produce EXEC, DIST, RSI, and DSI heartbeat.

You can also monitor the health of the replication path by sending an `rs_ticket` at primary and checking the RSTicket counters. If RSTicket counter of a module is not increasing, it shows that replication path at this stage is broken.

- Use `rs_ticket` only when Replication Server is 15.0 or higher.

See also

`rs_ticket_report`, `rs_ticket_history`

## rs\_zerolrm

|             |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
|-------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Description | Resets the locator value for a database to zero (0). Use this stored procedure after you have used the Adaptive Server command <code>dbcc settrunc</code> to disable the secondary truncation point and truncate the logs, but before you restart Replication Server.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| Syntax      | <code>rs_zerolrm data_server, database</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| Parameters  | <p><i>data_server</i></p> <p>The data server with the database whose locator value you want to reset.</p> <p><i>database</i></p> <p>The database whose locator value you want to reset.</p>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| Examples    | Resets the locator value to 0 for the TOKYO_DS data server and the pubs2 database.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
| Usage       | <pre>rs_zerolrm TOKYO_DS, pubs2</pre> <ul style="list-style-type: none"><li>• Use this command for RepAgent-enabled databases.</li><li>• Use <code>dbcc settrunc</code> to disable the secondary truncation point and truncate the log, before using <code>rs_zerolrm</code>.</li><li>• The locator value for a replicated database is maintained by the Replication Server and stored in the <code>rs_locator</code> table. Its value normally matches that of the secondary truncation point stored in the Adaptive Server.</li></ul> <p>If the transaction log fills up, you may have to use the <code>dbcc settrunc</code> command to disable the secondary truncation point and truncate the log. <code>dbcc settrunc</code> resets the secondary truncation point, and the locator value and the secondary truncation point no longer match. Execute <code>rs_zerolrm</code> to bring the values back in sync: Setting the locator value to zero with <code>rs_zerolrm</code> tells Replication Server to get the new secondary truncation point from Adaptive Server and set the locator to that value.</p> |
| See also    | <code>dbcc settrunc</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |

## Executable Programs

This chapter contains reference pages for the Replication Server executable programs. These include Replication Server and the `rs_subcmp` procedure.

## repserver

|             |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
|-------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Description | The Replication Server executable program.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
| Syntax      | <code>repserver [-C <i>config_file</i>] [-i <i>id_server</i>] [-S <i>rs_name</i>]<br/>[-l <i>interfaces_file</i>] [-E <i>errorlog_file</i>] [-M] [-v] [-K <i>keytab_file</i>]</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| Parameters  | <p><b>-C <i>config_file</i></b><br/>Specifies the name and location of the Replication Server configuration file. The <code>rs_init</code> program creates a configuration file which, by default, is named <i>Rep_Server_name.cfg</i>, where <i>Rep_Server_name</i> is the name of the Replication Server. You can specify this file name by using the <code>-C</code> flag. If you do not use the <code>-C</code> flag, <code>repserver</code> looks for the configuration file named <i>config.rs</i> in the directory where you started the Replication Server.</p> <p><b>-i <i>id_server</i></b><br/>Specifies the name of the ID Server for the replication system. The ID Server must be the first Replication Server started. It must be running and accessible before you can start a new Replication Server. The name of the ID Server is stored in the configuration file. Use the <code>-i</code> option to specify a different ID Server.</p> <p><b>-S <i>rs_name</i></b><br/>The name to use for the current Replication Server. If network-based security and unified login are enabled, specifies the name of the principal user.</p> <p><b>-l <i>interfaces_file</i></b><br/>Specifies the name and location of the interfaces file where the Replication Server is defined. The interfaces file must also have entries for the data servers and other Replication Servers that the current Replication Server communicates with. Interfaces files at replicate sites must have entries for the primary Replication Server and the primary data server. If you do not use the <code>-l</code> flag, Replication Server looks for the default interfaces file in the Sybase release directory.</p> <p>Refer to the Replication Server installation and configuration guides for your platform for more information about the interfaces file, including the default interfaces file name for your platform.</p> <p><b>-E <i>errorlog_file</i></b><br/>Specifies the name and location of the Replication Server error log file, into which <code>repserver</code> writes error messages. If you do not use the <code>-E</code> flag, the default error log file name and location is <i>repserver.log</i> in the directory where you started the Replication Server.</p> |

**-M**

Starts the Replication Server in standalone mode, which is used to initiate recovery actions. See the *Replication Server Administration Guide Volume 2* for more information about running Replication Server in standalone mode.

**-v**

Prints the version number of the Replication Server.

**-K *keytab\_file***

Should be used only with DCE network security. Specifies the name and location of the DCE keytab file that contains the security credential for the user logging into the server. Keytab files can be created with the DCE dcecp utility. See your DCE documentation for more information.

---

**Note** The **-K *keytab\_file*** option is only applicable for Windows platforms.

---

## Examples

**Example 1** Starts the Replication Server named TOKYO\_RS, using the configuration file TOKYO\_RS.cfg.

```
repserver -STOKYO_RS -CTOKYO_RS.cfg
```

**Example 2** Starts the Replication Server named SYDNEY\_RS, using the configuration file SYDNEY\_RS.cfg. TOKYO\_RS is the ID Server for the replication system.

```
repserver -SSYDNEY_RS -CSYDNEY_RS.cfg -iTOKYO_RS
```

**Example 3** Starts Replication Server and specifies an interfaces file, *my\_newinterfaces*, that overrides a default interfaces file or LDAP directory service.

```
repserver -STOKYO_RS _CTOKYO_RS.cfg
-I$SYBASE/SYBASE_RS/my_newinterfaces
```

## Usage

- Use the **repserver** command to start the Replication Server executable program. Normally, you start Replication Server by executing the run file created by **rs\_init**.
- On UNIX systems, this executable program is called **repserver**. On PC systems, the program is called **repsrvr**.
- The **repserver** executable program is located in the *bin* subdirectory of the Sybase release directory. Refer to the Replication Server installation and configuration guides for your platform for more information.
- The **repserver** command should be executed by the “sybase” user so that the Replication Server can access its disk partitions.

- The interfaces file must contain definitions of the other Replication Servers and data servers that the current Replication Server communicates with. Interfaces files at replicate sites must have entries for the primary Replication Server and the primary data server.
- If a password is stored in encrypted form, you cannot edit it directly by editing the Replication Server configuration file. To change an encrypted password in this file, use the rs\_init program. Refer to the Replication Server installation and configuration guides for your platform for more information.
- The RSSD\_primary\_user and the RSSD\_maint\_user are automatically assigned to the rs\_systabgroup group by rs\_init at Replication Server configuration time. This enables these users to modify the system tables. You can add other user login names to this group with the Adaptive Server system procedure sp\_changegroup. See the *Adaptive Server Enterprise System Administration Guide* for more information.
- If any of the network-based security parameters for the RSSD are present, the use\_security\_services parameter is set “on” and network-based security is initiated automatically.

#### Replication Server configuration file

The following table lists the parameters in the Replication Server configuration file.

**Table 7-1: Replication Server configuration file parameters**

| Configuration parameter     | Description                                                                                                                                                                                                                                                               |
|-----------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>CONFIG_charset</i>       | The character set used to write the Replication Server configuration file. Use this parameter only if this character set differs from the Replication Server’s character set. It can be any character set that is compatible with the Replication Server’s character set. |
| <i>erssd_backup_dir</i>     | ERSSD backup directory.                                                                                                                                                                                                                                                   |
| <i>erssd_dbfile</i>         | ERSSD database file.                                                                                                                                                                                                                                                      |
| <i>erssd_errorlog</i>       | ERSSD error log.                                                                                                                                                                                                                                                          |
| <i>erssd_logmirror</i>      | ERSSD transaction log mirror file.                                                                                                                                                                                                                                        |
| <i>erssd_ping_cmd</i>       | Allows user to specify a different command to ping ERSSD. For debug purposes only.                                                                                                                                                                                        |
| <i>erssd_port</i>           | ERSSD port number for network listener. The port number is obtained from the interface file.                                                                                                                                                                              |
| <i>erssd_release_dir</i>    | Allows user to specify a different release directory. For debug purposes only. The default is <code>\$\$SYBASE/\$SYBASE_REP/ASA11</code> .                                                                                                                                |
| <i>erssd_ra_release_dir</i> | Allows a user to specify a different release directory for ERSSD Replication Agent. For debug purposes only.                                                                                                                                                              |

| Configuration parameter      | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
|------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>er SSD_ra_start_cmd</i>   | Allows user to specify a different command to start ERSSD Replication Agent. For debug purposes only.                                                                                                                                                                                                                                                                                                                                                                                  |
| <i>er SSD_start_cmd</i>      | Allows user to specify a different command to start ERSSD. For debug purposes only.                                                                                                                                                                                                                                                                                                                                                                                                    |
| <i>er SSD_translog</i>       | ERSSD transaction log file.                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
| <i>ID_pw</i>                 | The password for the ID Server user ( <i>ID_user</i> ).                                                                                                                                                                                                                                                                                                                                                                                                                                |
| <i>ID_pw_enc</i>             | The encrypted password for the ID Server user ( <i>ID_user</i> ).                                                                                                                                                                                                                                                                                                                                                                                                                      |
| <i>ID_server</i>             | The name of the Replication Server that is the designated ID Server for the replication system.                                                                                                                                                                                                                                                                                                                                                                                        |
| <i>ID_user</i>               | The login name on the ID Server for other Replication Servers to use.                                                                                                                                                                                                                                                                                                                                                                                                                  |
| <i>RS_charset</i>            | <p>The character set for the Replication Server to use. You can specify any Sybase-supported character set.</p> <p>In setting up a replication system, it is highly recommended, though not required, that all servers at a given Replication Server site use the same character set. It is also recommended that all of the Replication Servers in your replication system use compatible character sets.</p> <p>Refer to the <i>Replication Server Design Guide</i> for details.</p> |
| <i>RS_language</i>           | The language used by the Replication Server to print its messages to the error log file and to its clients. You can specify any language to which the Replication Server has been localized that is compatible with the character set chosen.                                                                                                                                                                                                                                          |
| <i>RS_send_enc_pw</i>        | <p>Ensures that all Replication Server client connections are made with encrypted passwords except for the first connection to the RSSD. Values are on and off.</p> <p>Default: off</p>                                                                                                                                                                                                                                                                                                |
| <i>RS_sortorder</i>          | <p>The sort order that Replication Server uses. The sort order controls what rows of a table belong in a subscription that has a <i>where</i> clause involving character data. It also controls how identifiers you enter are recognized.</p> <p>You can specify any Sybase-supported sort order that is compatible with the character set chosen. All sort orders in your replication system should be the same.</p>                                                                  |
| <i>RS_unicode_sort_order</i> | <p>The Unicode sort order Replication uses. You can specify any Sybase-supported Unicode sort order.</p> <p>Default: binary</p>                                                                                                                                                                                                                                                                                                                                                        |
| <i>RSSD_database</i>         | The name of the RSSD.                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| <i>RSSD_embedded</i>         | Indicates whether RSSD is embedded or not.                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| <i>RSSD_ha_failover</i>      | <p>Specifies whether HA failover is allowed or not.</p> <p>Default: No.</p>                                                                                                                                                                                                                                                                                                                                                                                                            |
| <i>RSSD_maint_pw</i>         | The password for the RSSD maintenance user.                                                                                                                                                                                                                                                                                                                                                                                                                                            |
| <i>RSSD_maint_pw_enc</i>     | The encrypted password for the RSSD maintenance user.                                                                                                                                                                                                                                                                                                                                                                                                                                  |

| Configuration parameter          | Description                                                                                                                                                                                                                                                                                                                                                                                      |
|----------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>RSSD_maint_user</i>           | The login name for the RSSD maintenance user. This login name is automatically assigned to the <code>rs_systabgroup</code> group, whose users can modify the system tables. You can add other user login names to this group with the Adaptive Server system procedure <code>sp_changegroup</code> . See the <i>Adaptive Server Enterprise System Administration Guide</i> for more information. |
| <i>RSSD_msg_confidentiality</i>  | Specifies whether Replication Server sends and receives encrypted data. If set to “required”, outgoing and incoming data must be encrypted. If set to “not_required”, outgoing data is not encrypted and incoming data may be encrypted or not encrypted. This option is not implemented.<br>Default: <code>not_required</code>                                                                  |
| <i>RSSD_msg_integrity</i>        | Specifies whether data are checked for tampering. Valid entries are “required” and “not_required”. This option is not implemented.<br>Default: <code>not_required</code>                                                                                                                                                                                                                         |
| <i>RSSD_msg_origin_check</i>     | Specifies whether the origin of data should be checked. Valid entries are “required” and “not_required”. This option is not implemented.<br>Default: <code>not_required</code>                                                                                                                                                                                                                   |
| <i>RSSD_msg_replay_detection</i> | Specifies whether data should be checked to make sure they have not been read or intercepted. Valid entries are “required” and “not_required”. This option is not implemented.<br>Default: <code>not_required</code>                                                                                                                                                                             |
| <i>RSSD_msg_sequence_check</i>   | Specifies whether data should be checked to make sure the sequence hasn’t changed. Valid entries are “required” and “not_required”. This option is not implemented.<br>Default: <code>not_required</code>                                                                                                                                                                                        |
| <i>RSSD_mutual_auth</i>          | Specifies whether the RSSD must provide proof of identity before Replication Server establishes a connection. Valid entries are “required” and “not_required”. This option is not implemented.<br>Default: <code>not_required</code>                                                                                                                                                             |
| <i>RSSD_primary_user</i>         | The login name for the RSSD primary user. <code>rs_init</code> automatically assigns this user to the <code>rs_systabgroup</code> group during installation. You can add other user login names to this group using the Adaptive Server system procedure <code>sp_changegroup</code> . See the <i>Adaptive Server Enterprise System Administration Guide</i> for more information.               |
| <i>RSSD_primary_pw</i>           | The password for the RSSD primary user.                                                                                                                                                                                                                                                                                                                                                          |
| <i>RSSD_primary_pw_enc</i>       | The encrypted password for the RSSD primary user.                                                                                                                                                                                                                                                                                                                                                |
| <i>RSSD_sec_mechanism</i>        | The security mechanism Replication Server uses for initial contact with the RSSD at startup. Thereafter, network security information for contact with the RSSD is read from the <code>rs_config</code> file. This option is not implemented.                                                                                                                                                    |
| <i>RSSD_server</i>               | The name of the Adaptive Server with the RSSD.                                                                                                                                                                                                                                                                                                                                                   |
| <i>RS_ssl_identity</i>           | SSL identity file.                                                                                                                                                                                                                                                                                                                                                                               |

| Configuration parameter   | Description                                                                                                                                                                                                                                                                                                              |
|---------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>RS_ssl_pw</i>          | Password for the SSL private key                                                                                                                                                                                                                                                                                         |
| <i>RS_ssl_pw_enc</i>      | The encrypted password for the SSL private key.                                                                                                                                                                                                                                                                          |
| <i>RSSD_unified_login</i> | Specifies whether Replication Server seeks to connect to the RSSD with a credential at startup. Thereafter, network security information for contact with the RSSD is read from the <i>rs_config</i> file. Valid entries are “required” and “not_required”. This option is not implemented.<br><br>Default: not_required |
| <i>trace</i>              | Turns on a Replication Server trace. You can use multiple instances of this parameter to set the different traces available. Spaces are not allowed. For example:<br><br><i>trace=DSI,DSI_BUF_DUMP</i><br><i>trace=DIST,DIST_TRACE_COMMANDS</i>                                                                          |
| <i>trace_file</i>         | Indicates the name of the Replication Server log file.                                                                                                                                                                                                                                                                   |

## rs\_subcmp

### Description

An executable program that compares the data of a replicated table to the primary version of the table. `rs_subcmp` also performs schema comparison between replicated and primary tables and between replicated and primary databases. These features aid in finding—and optionally reconciling—missing, orphaned, and inconsistent rows and schemas. On UNIX systems, this program is called `rs_subcmp`. On PC systems, the program is called `subcmp`.

The `rs_subcmp` program is located in the *bin* subdirectory of the Sybase release directory. See the Replication Server installation and configuration guides for your platform for more information.

For `rs_subcmp` to work, the `SYBASE` environment variable, and the library path environment variable must be set. If you use `rs_subcmp` for schema comparison, ensure that `rs_subcmp` can locate the *ddlgen* executable file and that the *ddlgen* can successfully run in your Replication Server environment. See the Usage section for instructions.

`rs_subcmp` is intended to reconcile Sybase databases only.

### Syntax

```
rs_subcmp [-R | -r] [-v] [-V] [-z[1 | 2]]
[-f config_file] [-F]
-S primary_ds [-D primary_db]
-s replicate_ds [-d replicate_db]
-t table_name [-T primary_table_name]
-c select_command [-C primary_select_command]
-u user [-U primary_user]
[-p passwd] [-P primary_passwd]
[-B primary_init_batch]
[-b replicate_init_batch]
[-n num_iterations] [-w wait_interval]
[-e float_precision] [-E real_precision]
[-k primary_key_column [-k primary_key_column]...]
[-i identity_column]
[-l text_image_column_name
[-l text_image_column_name]...]
[-L text_image_length_in_kilobytes]
[-N text_image_column_name
[-N text_image_column_name]...]
[-Z language]
[-o sort_order]
[-O sort_order]
[-J rs_subcmp_charset]
[-j rep_charset]
[-a replicate_column_name primary_column_name
[-a replicate_column_name primary_column_name]...]
[-q unicode_sort_order]
[-Q unicode_sort_order]
```

`[-x schema_flag]`  
`[-X filter_flag]`  
`[-I interface_file]`

#### Parameters

- R**  
Reconciles the replicate data with the primary data, making a final verification of data inconsistencies at the primary database. `rs_subcmp` inserts, deletes, and updates rows at the replicate database so that the replicate data matches the primary data.
- r**  
Reconciles the replicate data with the primary data, without making a final verification of data inconsistencies at the primary database, as **-R** does. `rs_subcmp` inserts, deletes, and updates rows at the replicate database so that the replicate data matches the primary data.
- v**  
Prints version information.
- V**  
(Visual) prints the results of the comparison on the display (standard output). If you do not use the **-V** flag, `rs_subcmp` does not report differences between rows. Values of text, unitext, or image data are not printed. Instead, `rs_subcmp` reports whether the inconsistency is in the text, unitext, or image columns or in the columns of other datatypes.
- z**  
Enables trace. **-z1**, the default, provides basic trace information, such as comparisons of column headings. **-z1** also prints information about numeric precision differences. **-z2** provides trace information on comparisons of all rows and commands.
- f *config\_file***  
Specifies the name of the configuration file for `rs_subcmp`.
- F**  
Displays the format (syntax) to use for the *config\_file*. A configuration file must use the syntax displayed with the **-F** option, and must contain all required syntax parameters.
- S *primary\_ds***  
The name of the data server with the primary data for the subscription.
- D *primary\_db***  
The name of the database where the primary data for the subscription is stored.

**-s** *replicate\_ds*

The name of the data server with the replicate copy of the data.

**-d** *replicate\_db*

The name of the database with the replicate copy of the data.

**-t** *table\_name*

The name of the table in the primary and replicate databases with the data to be compared. If the name is different in the databases, use the **-T** option to specify the name of the table in the primary database. You can include table owner name information here.

**-T** *primary\_table\_name*

The name of the table in the primary database. Use this option when the table name is different in the primary and replicate databases. You can include table owner name information here.

**-c** *select\_command*

A select command that retrieves the subscription's data from both the primary and replicate copies of the data. Use **-C** to specify a different command for the primary data. select commands must order rows based on the primary key.

You can include columns with text, unitext, or image datatypes in the select command, with the following requirements:

- Columns with text, unitext, or image datatypes cannot be primary key columns.
- You must place columns with text, unitext, or image datatypes at the end of the select list.
- By default, the replicate table does not allow null values for text or image columns. You must include the **-N** flag in the `rs_subcmp` executable to indicate that a null value is allowed in the text, unitext, or image column of the replicate table.

**-C** *primary\_select\_command*

A select command that retrieves the subscription's data from the primary copy of the data. Use this option and **-c** when you need a different select command for the primary and replicate databases. select commands must order rows based on the primary key.

**-u** *user*

The login name used to log into the primary and replicate data servers. If you need different login names, use the **-U** option to specify a different primary data server login name.

**-U *primary\_user***

The login name used to log into the primary data server. Use this option and the **-u** option when different login names are required for the primary and replicate data servers.

**-p *passwd***

The password to use with the *user* login name and, if supplied, the *primary\_user* login name. If you omit this option, **rs\_subcmp** uses a null password. If you specify a different password for the *primary\_user* login name, specify it with the **-P** option.

**-P *primary\_passwd***

The password to use with the *primary\_user* login name.

**-B *primary\_init\_batch***

A command batch to be executed when initially connecting to the primary database. The batch can be used for any purpose, such as to set the isolation level. The batch is run after **rs\_subcmp** logs into the primary database.

**-b *replicate\_init\_batch***

A command batch to be executed when initially connecting to the replicate database. The batch can be used for any purpose, such as to turn off triggers when running **rs\_subcmp** in a warm standby application, or to set the isolation level. The batch is run after **rs\_subcmp** logs into the replicate database.

**-n *num\_iterations***

The number of times that **rs\_subcmp** examines the inconsistent rows it finds. The default is 10 iterations. The first iteration may find many inconsistencies due to normal time lag in replication. Additional iterations allow **rs\_subcmp** to distinguish true inconsistencies from the inconsistent rows that are corrected through normal replication activity.

**-w *wait\_interval***

The number of seconds **rs\_subcmp** waits before beginning another iteration. The default is 5 seconds.

**-e *float\_precision***

Sets the number of decimal places in exponential notation that floating point values are expected to agree. By default, this is set to the maximum precision supported by the platform.

**-E *real\_precision***

Sets the number of decimal places in exponential notation that real values are expected to agree. By default, this is set to the maximum precision supported by the platform.

**-k *primary\_key\_column***

A column name that is part of the primary key for the table. The primary key must be unique and it cannot be a text, untext, or image column. Use the -k option for each column in the primary key. If the primary and replicate column names are different, the name specified here is the replicate column name.

**-i *identity\_column***

The name of the xidentity column in the replicate table.

**-l *text\_image\_column\_name***

Turns off logging of updates to a replicate text, untext, or image column. By default, text, untext, or image column updates are logged.

**-L *text\_image\_length***

Sets the longest value the data server returns for text, untext, or image columns. The default value is 2048K.

**-N *text\_image\_column\_name***

Indicates that a null value is allowed in the text, untext, or image column of the replicate table. By default, the replicate table does not allow null values for text, untext, or image columns.

**-Z *language***

The name of the language in which rs\_subcmp generates error and informational messages. If not specified, it uses the language specified in the “default” locale entry for your platform.

**-o *sort\_order***

The name of the sort order used in your replication system. rs\_subcmp uses this information to compare primary key columns.

**-O *sort\_order***

The name of the sort order used in your replication system. rs\_subcmp uses this information to compare all columns.

**-J *rs\_subcmp\_charset***

The name of the character set used by rs\_subcmp error and informational messages and in all configuration parameters and command line options. If you do not specify *rs\_subcmp\_charset*, it is set to the character set specified in the “default” locale entry for your platform.

**-j *rep\_charset***

The name of the character set used by the replicate data server. The rs\_subcmp program uses this character set when comparing and reconciling the replicate and primary versions of a table. If you do not specify a *rep\_charset*, it is set to the *rs\_subcmp\_charset* character set.

**-a** *replicate\_column\_name primary\_column\_name*

Specifies the primary column name associated with a replicate column. Use this option if a replicate column name is different from that of the primary column.

---

**Note** When you use the -a option, the replicate column name must come before the associated primary column name.

---

**-q** *unicode\_sort\_order*

Specifies the Unicode sort order *rs\_subcmp* uses to compare Unicode primary key columns.

**-Q** *unicode\_sort\_order*

Specifies the Unicode sort order *rs\_subcmp* uses to compare all Unicode columns.

**-x** *schema\_flag*

Specifies the *rs\_subcmp* comparison type. The possible values of the *schema\_flag* are:

- 0 – data comparison. This is the default value.
- 1 – database schema comparison between two databases.
- 2 – table schema comparison between two tables.

**-X** *filter*

Specifies the schema types and subtypes included or excluded from the comparison. If the value starts with “+”, only the schema types are selected for comparison, and the subschema types are ignored. Otherwise, the schema types and subschema types are both not selected and not used for comparison. For a list of schema types and schema subtypes supported by *rs\_subcmp*, see Table 7-4 and Table 7-5.

**-I** *interface\_file*

Specifies the interface file location. For more information on the interface file, see the Replication Server configuration guides for your platform.

**-g**

Creates reconciliation file for inconsistent data.

**-h**

Performs fast comparison.

**-H** *normalization\_option*

Indicates how to normalize the data when performing fast comparison. For a list of normalization options supported by *rs\_subcmp*, see Table 7-6.

## Examples

**Example 1** Starts rs\_subcmp using a configuration file called *titleauthor.cfg*.

```
rs_subcmp -ftitleauthor.cfg
```

The configuration file consists of the following:

```
titleauthor.cfg - Reconcile
SYDNEY_DS.pubs2.dbo.titleauthor with
TOKYO_DS.pubs2.dbo.titleauthor.
#
PDS = TOKYO_DS
RDS = SYDNEY_DS
PDB = pubs2
RDB = pubs2
PTABLE = titleauthor
RTABLE = titleauthor
PSELECT = select au_id, title_id, au_ord,\ royaltyper
 from titleauthor order by au_id,\ title_id
RSELECT = select au_id, title_id, au_ord,\ royaltyper
 from titleauthor order by au_id,\ title_id
PUSER = repuser
RUSER = repuser
PPWD = piglet
RPWD = piglet
KEY = au_id
KEY = title_id
RECONCILE = Y
VISUAL = Y
NUM_TRIES = 3
WAIT = 10
```

rs\_subcmp compares the primary and replicate tables called titleauthor and generates the following output:

```
$SYBASE/bin/rs_subcmp -f ttl_au.cmp
INCONSISTENT ROWS:
```

| _____Replicate row_____ |          |        |            |
|-------------------------|----------|--------|------------|
| au_id                   | title_id | au_ord | royaltyper |
| -----                   |          |        |            |
| 672-71-3249             | TC7777   | 1      | 40         |

| _____Primary row_____ |          |        |            |
|-----------------------|----------|--------|------------|
| au_id                 | title_id | au_ord | royaltyper |
| -----                 |          |        |            |
| 672-71-3249           | TC7777   | 1      | 50         |

**Example 2** Starts `rs_subcmp` using a configuration file called `subcmp.cfg`. Command line flags override the configuration file settings, to reconcile differences in the primary and replicate versions of the authors table, performing a final verification.

```
rs_subcmp -R -fsubcmp.cfg -STOKYO_DS -Dpubs2 \
-sSYDNEY_DS -dpubs2 -tauthors
```

The primary data server and database are TOKYO\_DS and pubs2. The replicate data server and database are SYDNEY\_DS and pubs2.

**Example 3** Compares all schemas between two databases using the `config.cfg` file:

```
rs_subcmp -f config.cfg
```

The configuration file contains:

```
PDS = PASE
RDS = R2ASE
PDB = pubs2
PTABLE = authors
RTABLE = authors
PUSER = sa
RUSER = sa
PPWD =
RPWD =
SCHEMAFLAG = 1
```

**Example 4** Compares schema between two databases without a configuration file:

```
rs_subcmp -Spds -srds -Dpdb -drdb -Usa -usa -Psa_pwd
-psa_pwd -x1
```

**Example 5** Compares schema of two databases excluding index, trigger, and datatype:

```
rs_subcmp -Spds -srds -Dpdb -drdb -Usa -usa -Psa_pwd
-psa_pwd -x1 -XitD
```

**Example 6** Compares all table schemas and user schemas:

```
rs_subcmp -Spds -srds -Dpdb -drdb -Usa -usa -Psa_pwd
-psa_pwd -x1 -X+TU
```

#### Usage

- Run `rs_subcmp` when primary changes do not occur.
- The `SYBASE` environment variable, and the library path environment variable must be set for `rs_subcmp` to work.

Set the `SYBASE` environment variable to the Sybase release directory.

Set the library path variable to `$SYBASE/$SYBASE_OCS/lib` (UNIX) or `%SYBASE%\%SYBASE_OCS%\lib` (Windows):

- For Solaris and Linux, the library path variable is `LD_LIBRARY_PATH`.
- For HP, the library path variable is `SHLIB_PATH`.
- For RS6000, the library path variable is `LIBPATH`.
- For Windows, the library path variable is `PATH`.
- `rs_subcmp` must be able to locate and successfully run the *ddlgen* executable file for schema comparison to work. You can set the location of *ddlgen* through the `DDLGENLOC` environment variable. If `DDLGENLOC` is not set, `rs_subcmp` looks for *ddlgen* at its default location, which is at `$SYBASE/ASEP/bin/`. To ensure that *ddlgen* runs successfully, the environment variables that *ddlgen* uses must be set correctly.
- The following requirements apply to `rs_subcmp`:
  - If you provide a configuration file and also use command line options, the command line values override the values in the configuration file.
  - The lowercase options `-d`, `-c`, `-u`, `-p`, and `-t` provide values for both primary and replicated data. Use the uppercase options to override the values for primary data.
  - The only required uppercase option is `-S`.
  - The primary key specified with `-k` must be unique. If you do not specify any primary key columns with the `-k` option, all columns are considered to be part of the primary key.
  - Use a positive integer in `-L` to specify a new value, overwriting the default value of 26K, for the byte length of text and image columns:  
  
`-L = <new_value>`  
  
For instance, if you want text and image columns to be 65,536 bytes, enter:  
  
`-L = <64>`
- These options can be used to specify a non-default table owner or a different primary replicate table or column name:
  - For options `-t`, `-T`, `-c`, and `-C`, table owner information can be included (for example, `ling.authors`).

- Owner, table, and column names specified for the `-c` option should be those of the replicate table.
- Owner, table, and column names specified for the `-C` option should be those of the primary table.
- The column name specified for the `-k` option is the column name of the replicate table.
- `rs_subcmp` creates a report file after every schema comparison. The report file details the comparison result between two tables or two databases. The report file is named *reportPROCID.txt*. If inconsistencies exist, `rs_subcmp` creates a reconciliation script named *reconcilePROCID.sql*. The report file and the reconciliation script are saved in the same directory from which `rs_subcmp` executed.
- The reconciliation file's SQL statements cannot contain text, unitext, or image.
- `rs_subcmp` creates a reconciliation file if you specify the `-g` option. The file is named *reconcile\_file\_PROCID.sql* and is located at the current working directory.

#### Return codes

The following return codes can be returned by `rs_subcmp`:

**Table 7-2: *rs\_subcmp* return codes**

| Return code | Meaning                                                    |
|-------------|------------------------------------------------------------|
| 0           | The replicated and primary tables are the same.            |
| 1           | An error occurred while executing <code>rs_subcmp</code> . |
| 2           | The replicated and primary tables are different.           |

#### Configuration file

You can create a file containing `rs_subcmp` parameters and specify it on the command line using the `-f` flag. Each line in the configuration file consists of a parameter name, an equal sign (=), and a value.

The following table lists the parameters that can be used in the `rs_subcmp` configuration file and the corresponding command line option for each parameter.

**Table 7-3: *rs\_subcmp* configuration file parameters**

| Configuration parameter | Command-line option | Value                      |
|-------------------------|---------------------|----------------------------|
| <i>PDS</i>              | <code>-S</code>     | Primary data server name   |
| <i>RDS</i>              | <code>-s</code>     | Replicate data server name |

| Configuration parameter | Command-line option | Value                                                                                                                                                                                                     |
|-------------------------|---------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>PDB</i>              | -D                  | Primary database name                                                                                                                                                                                     |
| <i>RDB</i>              | -d                  | Replicate database name                                                                                                                                                                                   |
| <i>PTABLE</i>           | -T                  | Primary table name                                                                                                                                                                                        |
| <i>RTABLE</i>           | -t                  | Replicate table name                                                                                                                                                                                      |
| <i>PUSER</i>            | -U                  | Primary user name                                                                                                                                                                                         |
| <i>RUSER</i>            | -u                  | Replicate user name                                                                                                                                                                                       |
| <i>PPWD</i>             | -P                  | Primary password                                                                                                                                                                                          |
| <i>RPWD</i>             | -p                  | Replicate password                                                                                                                                                                                        |
| <i>KEY</i>              | -k                  | Primary key element in replicate table                                                                                                                                                                    |
| <i>PINITBATCH</i>       | -B                  | Primary database connection initialization batch. Can span multiple lines if newline characters are preceded by a “\” (backslash). Up to 1024 characters per line and 64K characters total are allowed.   |
| <i>RINITBATCH</i>       | -b                  | Replicate database connection initialization batch. Can span multiple lines if newline characters are preceded by a “\” (backslash). Up to 1024 characters per line and 64K characters total are allowed. |
| <i>PSELECT</i>          | -C                  | Primary select command. Can span multiple lines if newline characters are preceded by a “\” (backslash). Up to 1024 characters per line and 64K characters total are allowed.                             |
| <i>RSELECT</i>          | -c                  | Replicate select command. Can span multiple lines if newline characters are preceded by a “\” (backslash). Up to 1024 characters per line and 64K characters total are allowed.                           |
| <i>RECONCILE</i>        | -r                  | Reconcile differences (Y or N)                                                                                                                                                                            |
| <i>RECONCILE_CHECK</i>  | -R                  | Reconcile differences with primary verification (Y or N)                                                                                                                                                  |
| <i>TRACE</i>            | -z                  | Enable trace with optional level (optional integer)                                                                                                                                                       |
| <i>FPPRECISION</i>      | -e                  | Expected floating point precision (integer—default is platform-dependent)                                                                                                                                 |
| <i>RPPRECISION</i>      | -E                  | Expected real precision (integer—default is platform-dependent)                                                                                                                                           |
| <i>WAIT</i>             | -w                  | Seconds between comparisons (integer—default is 5 seconds)                                                                                                                                                |
| <i>NUM_TRIES</i>        | -n                  | Number of comparisons (integer—default is 10 iterations)                                                                                                                                                  |
| <i>VISUAL</i>           | -V                  | Print results (Y or N)                                                                                                                                                                                    |
| <i>IDENTITY</i>         | -i                  | identity column name in replicate table                                                                                                                                                                   |
| <i>TXT_IMG_LEN</i>      | -L                  | The longest value, in kilobytes, the data server returns for text, unitext, or image columns.                                                                                                             |
| <i>NO_LOG</i>           | -l                  | Do not log updates for this replicate text, unitext, or image column                                                                                                                                      |

| Configuration parameter            | Command-line option | Value                                                                                                                                                                                                                                                                            |
|------------------------------------|---------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>NULLABLE</i>                    | -N                  | The text, unitext, or image column in the replicate table accepts null values.                                                                                                                                                                                                   |
| <i>LANGUAGE</i>                    | -Z                  | Language of rs_subcmp error and informational messages                                                                                                                                                                                                                           |
| <i>SORT_ORDER</i>                  | -o                  | Use the specified sort order to compare primary key columns.                                                                                                                                                                                                                     |
| <i>SORT_ORDER_ALL_COLS</i>         | -O                  | Use the specified sort order to compare all columns.                                                                                                                                                                                                                             |
| <i>SCHARSET</i>                    | -j                  | Character set of rs_subcmp                                                                                                                                                                                                                                                       |
| <i>RCHARSET</i>                    | -J                  | Character set of the replicate data server                                                                                                                                                                                                                                       |
| <i>REP_PRI_COLNAME</i>             | -a                  | Replicate-Primary column name pair                                                                                                                                                                                                                                               |
| <i>UNICODE_SORT_ORDER</i>          | -q                  | The Unicode sort order rs_subcmp uses to compare Unicode primary key columns.                                                                                                                                                                                                    |
| <i>UNICODE_SORT_ORDER_ALL_COLS</i> | -Q                  | The Unicode sort order rs_subcmp uses to compare all Unicode columns.                                                                                                                                                                                                            |
| <i>SCHEMAFLAG</i>                  | -x                  | The rs_subcmp comparison type.                                                                                                                                                                                                                                                   |
| <i>FILTER</i>                      | -X                  | The filter used to indicate the schema and schema subtypes included or excluded in the schema comparison. See Table 7-4 and Table 7-5 for a list of schema types and schema subtypes supported by rs_subcmp.                                                                     |
| <i>IFILE</i>                       | -l                  | The interface file location.                                                                                                                                                                                                                                                     |
| <i>RECONCILE_FILE</i>              | -g                  | Indicates whether to create a reconciliation file or not.<br>Values:<br><ul style="list-style-type: none"> <li>• Y – create reconciliation file.</li> <li>• N – do not create reconciliation file.</li> </ul> Default: N                                                         |
| <i>FASTCMP</i>                     | -h                  | Indicates whether to perform fast comparison or not.<br>Values:<br><ul style="list-style-type: none"> <li>• Y – perform fast comparison using compressed data.</li> <li>• N – perform normal comparison.</li> </ul> Default: N                                                   |
| <i>HASH_OPTION</i>                 | -H                  | Indicates the normalization option used for fast comparison. If this parameter is not included in the configuration file, rs_subcmp normalizes the data using native byte order and character set. See Table 7-6 for a list of the normalization options supported by rs_subcmp. |

#### Requirements for *select* commands

- The select commands specified by -c (RSELECT) and -C (PSELECT) must return columns with the same names and datatypes from both the primary and the replicate databases.

- You must have a clustered index on the primary key or an order by clause in the select command. select commands must order rows based on the primary key. If *rs\_subcmp* does not receive rows in the correct order, it may delete rows in the replicate table.
- Do not select *rs\_address* datatypes with the *-c* or *-C* options. If replicate tables contain columns using the *rs\_address* datatype, the primary and replicate versions of these columns may not be identical. Replication Server filters out updates to these columns so as not to replicate them unnecessarily.

#### How *rs\_subcmp* works

- *rs\_subcmp* logs into the primary and replicate databases and executes the supplied select commands. It verifies that the commands return the same columns, based on the name and datatype of each column. If the returned columns match, *rs\_subcmp* compares the primary and replicate rows and creates these lists:
  - Missing rows – rows at the primary, but not at the replicate
  - Orphaned rows – rows at the replicate, but not at the primary
  - Inconsistent rows – rows at the replicate and the primary with matching primary keys, but differences in other columns
- After the three lists are compiled *rs\_subcmp* iterates for the specified number of times, checking:
  - If missing rows appear at the replicate
  - If orphaned rows disappear from the replicate
  - If inconsistent rows match
  - If the new replicate row value matches the primary row value from the previous iteration
- After the specified number of iterations, the contents of the three lists are printed to the standard output if you specified the *-V* option.

#### Reconciling inconsistencies

- *rs\_subcmp* reconciles missing, orphaned, and inconsistent rows if you specify the *-R* or *-r* option.
- If you specify the *-r* option, *rs\_subcmp* reconciles the primary and replicate copies. It passes the final lists and modifies the replicate table as follows:
  - Inserts rows remaining in the missing rows list

- Deletes rows remaining in the orphaned rows list
- Updates inconsistent rows to match the primary rows
- If you specify the -R option, `rs_subcmp` reconciles the replicate table to the primary version in the same way as with the -r option. However, before it inserts a missing row or deletes an orphaned row, it logs into the primary database and performs a `select` on the row to verify that:
  - The row still exists (in the case of a missing row in the replicate table), or
  - The row does not exist (in the case of an orphaned row in the replicate table).

#### Reconciling IDENTITY columns

- If the values in an identity column for a row are inconsistent, `rs_subcmp` reconciles them by deleting the row in the replicate database before inserting the row from the primary database.

#### Reconciling *text*, *unitext*, or *image* datatypes

- Unlike other datatypes, inconsistencies in *text*, *unitext*, or *image* values are not stored in a list. To reconcile a missing or inconsistent row that contains a *text* or *image* value, `rs_subcmp` logs back into the primary database and re-executes the `select` statement. If the inconsistent or missing row is found, `rs_subcmp` modifies the replicate table by updating or inserting the row. However, if the inconsistent or missing row is not found in the primary table, `rs_subcmp` takes the following actions:
  - For an inconsistent row, `rs_subcmp` deletes the row from the replicate table
  - For a missing row, `rs_subcmp` takes no action
- Using the Adaptive Server option `set textsize` as part of the `select` statement can limit the amount of text compared. For example, the following example shows the effect of setting the `textsize` to 10. The first `select` statement returns 30 characters of text:

```
set textsize 30 select * from zetext
```

| a     | b      | c            |
|-------|--------|--------------|
| ----- | -----  | -----        |
| abba  | apples | odd one here |
| beta  | banana | rotten       |
| caro  | celery | not carrots  |

The next select statement sets the size of the text to 10:

```
1> set textsize 10 select * from zetext
2> go
```

| a    | b      | c           |
|------|--------|-------------|
| abba | apples | odd one     |
| beta | banana | rotten      |
| caro | celery | not carrots |

(3 rows affected)

Using *rs\_subcmp* in international environments

- *rs\_subcmp* provides support for international environments with the *-Zlanguage*, *-o sort\_order*, *-O sort\_order*, *-q unicode\_sort\_order*, *-Q unicode\_sort\_order*, *-J rs\_subcmp\_charset*, and *-j rep\_charset* options.
- *rs\_subcmp* performs character set conversion when comparing and reconciling the replicate and primary versions of a table. The method is similar to how Replication Server converts character sets, so you can expect to see similar results.

For example, if the primary and replicate data server's character sets are incompatible, no conversion takes place. If the character sets are incompatible but a single character from the primary data server's character set has no representation in the replicate server's character set, the character is replaced with a "?" and processing continues.

- `rs_subcmp` uses the character set of the replicate data server in all operations involving user data. To specify the replicate data server's character set, use the `-j` command line option or the *RCHARSET* configuration file parameter.

---

**Note** `rs_subcmp` does not have a parameter for the primary data server's character set because all data operations are done in the replicate data server's character set. The program depends on the primary data server to convert all character data to the replicate data server's character set. This is comparable to how Replication Server works during subscription materialization.

---

- You can also specify a character set for `rs_subcmp` if it is different from the replicate data server's character set. To do this, use the `-J` command line option or the *SCHARSET* configuration file parameter. When you specify a character set, `rs_subcmp` converts its string-type configuration parameters from the `rs_subcmp` character set to the replicate data server's character set.

#### Requirements for character sets and sort orders

- The following requirements apply for specifying character sets and sort orders in `rs_subcmp`:
  - All characters in object names (including servers, databases, tables, and column names) must be compatible with the *rs\_subcmp\_charset* and *rep\_charset* character sets; otherwise `rs_subcmp` will fail to execute.
  - If the character sets of the replicate and primary data servers differ, the replicate data server's character set must be installed at the primary data server. This enables the primary data server to do character set translation.
  - If the replicate and primary data servers use different sort orders and the `where` clause of the `select` statement includes character or text datatypes, results may be confusing. To avoid confusion, run `rs_subcmp` first without the `-r` or `-R` (reconcile) options and with the `-V` (visual) option to see the potential effects on your data.

#### Using sort orders

- You can specify nonUnicode sort order in two ways: using the `-o` option or using the `-O` option.
- If you specify the `-o` option, `rs_subcmp`:

- a Performs a simple binary comparison of the primary key columns.
- b If the primary keys match, rs\_subcmp performs a binary comparison of the remaining columns. If they don't match, an inconsistent row is reported.
- c If the primary key columns do not match, rs\_subcmp compares them using the specified sort order.
  - If the primary key columns don't match, the row is reported missing or orphan.
  - If the primary key columns test equal using the sort order, the row is reported inconsistent.
- If you specify the -O option, rs\_subcmp:
  - Performs a column comparison using the specified sort order for all columns of types char, varchar, and text.
  - Does not perform a binary comparison.
- If no sort order is specified, rs\_subcmp performs a simple binary comparison on each column of the primary and replicate row.

#### Using Unicode sort orders

- You can specify Unicode sort order in two ways: using the -q option or using the -Q option.
- If you specify the -q option, rs\_subcmp:
  - a Performs a simple binary comparison of the Unicode primary key columns.
  - b If the primary keys match, rs\_subcmp performs a binary comparison of the remaining columns. If they don't match, an inconsistent row is reported.
  - c If the primary key columns do not match, rs\_subcmp compares them using the specified sort order.
    - If the Unicode primary key columns don't match, the row is reported missing or orphan.
    - If the primary key columns test equal using the sort order, the row is reported inconsistent.
- If you specify the -Q option, rs\_subcmp:
  - Performs a column comparison using the specified sort order for all Unicode columns.

- Does not perform a binary comparison.
- If no sort order is specified, `rs_subcmp` performs a simple binary comparison on each Unicode column of the primary and replicate row.

#### Schema types and schema subtypes

Table 7-4 and Table 7-5 list the schema and schema subtypes supported by `rs_subcmp`.

**Table 7-4: Schema types supported by `rs_subcmp`**

| Type | Description                                                                                                |
|------|------------------------------------------------------------------------------------------------------------|
| A    | All aliases in the database.                                                                               |
| D    | All defaults in the database.                                                                              |
| E    | All user-defined datatypes in the database.                                                                |
| G    | All groups in the database.                                                                                |
| R    | All rules in the database.                                                                                 |
| T    | All user tables in the database. Includes table elements such as indexes, keys, constraints, and triggers. |
| U    | All users in the database.                                                                                 |
| V    | All views in the database.                                                                                 |
| P    | All procedures in the database.                                                                            |

**Table 7-5: Schema subtypes supported by `rs_subcmp`**

| Type | Description    |
|------|----------------|
| c    | Constraint     |
| d    | Bind default   |
| f    | Foreign key    |
| g    | Grant          |
| i    | Index          |
| m    | Procedure mode |
| p    | Primary key    |
| r    | Bind rule      |
| t    | Trigger        |

#### Normalization options for faster comparison

Table 7-6 lists the normalization options for faster comparison supported by `rs_subcmp`.

**Table 7-6: Normalization options supported by rs\_subcmp**

| Normalization option | Description                                                                       |
|----------------------|-----------------------------------------------------------------------------------|
| lsb                  | Normalizes all byte-order-dependent data to lsb-first (little-endian) byte order. |
| msb                  | Normalizes all byte-order-dependent to msb-first (big-endian) byte order.         |
| unicode              | Normalizes the character data to Unicode (UTF-16).                                |
| unicode_lsb          | Normalizes lsb in conjunction with Unicode for platform independence.             |
| unicode_msb          | Normalizes msb in conjunction with Unicode for platform independence.             |

## Replication Server System Tables

This chapter lists the system tables in the Replication Server System Database (RSSD) or Embedded RSSD (ERSSD). System tables are stored in a dedicated database—Adaptive Server for RSSD or SQL Anywhere ERSSD.

Access to the system tables is restricted to users with sa permission, or members of the rs\_systabgroup group. The system tables are maintained by RCL commands and must not be directly modified. To alter server values found in the rs\_config table, use the configure replication server command.

For more information about the rs\_systabgroup group, see repserver on page 636. For information about configure replication server, see configure replication server on page 206.

The system tables include the user-defined datatype rs\_id that is defined as binary(8). It is used for columns that hold object names. For more information about identifiers, see “Identifiers” on page 35.

The rs\_lastcommit and rs\_threads system tables are documented in this chapter, although these tables are created and stored in each user database, not in the RSSD or ERSSD.

## rs\_articles

Description Stores information about articles known to this Replication Server.

| Column       | Datatype     | Description                                                                         |
|--------------|--------------|-------------------------------------------------------------------------------------|
| articlename  | varchar(255) | Name of the article                                                                 |
| articleid    | rs_id        | Unique article ID                                                                   |
| type         | char(1)      | <ul style="list-style-type: none"><li>• T – table</li><li>• P – procedure</li></ul> |
| primaryname  | varchar(255) | Primary table or procedure name                                                     |
| primaryowner | varchar(30)  | Primary table owner name                                                            |
| objid        | rs_id        | ID of the corresponding replication definition                                      |
| pubid        | rs_id        | ID of the publication to which this article belongs                                 |
| requestdate  | datetime     | Date and time the article was added to the publication                              |
| minvers      | int          | Minimum Replication Server version required to support this article                 |

Indexes

- Unique clustered index on (articlename, pubid)
- Unique index on (articleid)

## rs\_classes

Description Stores the names of function-string classes and error classes.

| Column         | Datatype    | Description                                                                                                                                                                                                                   |
|----------------|-------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| classname      | varchar(30) | Class name                                                                                                                                                                                                                    |
| classid        | rs_id       | ID for this class                                                                                                                                                                                                             |
| classtype      | char(1)     | One of the following values: <ul style="list-style-type: none"> <li>• R - Replication Server error class</li> <li>• F – function-string class</li> <li>• E – data server error class</li> <li>• D – datatype class</li> </ul> |
| prsid          | int         | ID of the site where this class is primary                                                                                                                                                                                    |
| parent_classid | rs_id       | ID for the parent class if this is a derived class<br>0 if this is a base class; default is 0                                                                                                                                 |
| attributes     | int         | 0x01 – Default class<br><br>For rs_default_function_class and rs_db2_function_class, the default is 1.<br>Otherwise, the default is 0.                                                                                        |

Indexes

- Unique clustered index on (classname, classtype)
- Unique index on (classid)

## rs\_columns

Description                      Contains information about the columns of replication definitions.

| Column  | Datatype     | Description                                                                    |
|---------|--------------|--------------------------------------------------------------------------------|
| prsid   | int          | Primary Replication Server for this object                                     |
| objid   | rs_id        | Table/function replication definition ID or function ID this column belongs to |
| colname | varchar(255) | Column or parameter name                                                       |
| colnum  | smallint     | Column number                                                                  |

| Column        | Datatype | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
|---------------|----------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| coltype       | tinyint  | Datatype of the column or parameter: <ul style="list-style-type: none"> <li>• 0 – char</li> <li>• 1 – binary</li> <li>• 4 – text</li> <li>• 5 – image</li> <li>• 6 – tinyint</li> <li>• 7 – smallint</li> <li>• 8 – int</li> <li>• 9 – real</li> <li>• 10 – float</li> <li>• 11 – bit</li> <li>• 12 – datetime</li> <li>• 13 – smalldatetime</li> <li>• 14 – money</li> <li>• 15 – smallmoney</li> <li>• 16 – numeric</li> <li>• 17 – decimal</li> <li>• 18 – varchar</li> <li>• 19 – varbinary</li> <li>• 25 – unichar</li> <li>• 27 – date</li> <li>• 28 – time</li> <li>• 29 – unitext</li> <li>• 30 – bigint</li> <li>• 31 – usmallint</li> <li>• 32 – uint</li> <li>• 33 – ubigint</li> <li>• 35 – bigdatetime</li> <li>• 36 – bigtime</li> <li>• 110 – univarchar</li> </ul> |
| length        | int      | Length of the declared data                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| searchable    | tinyint  | 1 if searchable key, 0 if not                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| primary_col   | tinyint  | 1 if primary key, 0 if not                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| fragmentation | tinyint  | 1 if fragmentation key, 0 if not                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
| rowtype       | tinyint  | 1 if row is to be replicated, 0 if not                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |

| Column            | Datatype     | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
|-------------------|--------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| status            | int          | Mask, can be one or more of the following: <ul style="list-style-type: none"> <li>• 0x01 – column is declared an <b>identity</b> column</li> <li>• 0x02 – column is declared a <b>timestamp</b> column</li> <li>• 0x04 – column is an <b>rs_address</b> datatype</li> <li>• 0x08 – column has a status of <b>replicate_if_changed</b></li> <li>• 0x10 – column allows null values in the replicate table (only for <b>text</b>, <b>unitext</b>, or <b>image</b> columns)</li> <li>• 0x20 – column is sent to standby connection (only in internal replication definitions)</li> <li>• 0x40 – column is marked as dropped from the internal replication definition (only in internal replication definitions)</li> <li>• 0x200 – published as <b>identity</b></li> <li>• 0x400 – published as <b>timestamp</b></li> <li>• 0x1000 – declared as <b>Java</b> column</li> <li>• 0x2000 – published as <b>Java</b> column</li> </ul> |
| basecolnum        | smallint     | Column position in base replication definition. Default is <i>colnum</i> value.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
| repl_colname      | char(255)    | Column name in replicate table. Default is <i>colname</i> value.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| declared_dtid     | rs_id        | Datatype ID. For a user-defined datatype, this is a foreign key to the table.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
| publ_dtid         | rs_id        | Published datatype as specified in the replication definition. If no published datatype is specified, <i>publ_dtid</i> is equal to <i>declared_dtid</i> .                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| publ_base_coltype | tinyint      | The base datatype of the published datatype. If no published datatype is specified, <i>publ_base_coltype</i> is equal to <i>coltype</i> .                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| publ_length       | int          | The maximum length of the published datatype.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
| version           | rs_id        | Identifies a replication definition version.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| ref_objowner      | varchar(30)  | Name of the replicate object owner, as specified in replication definition. Blank if the owner is not specified.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| ref_objname       | varchar(255) | Object name defined in the replicate database.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |

## Indexes

- Unique clustered index on (version, colname)
- Unique index on (objid, basecolnum)
- Unique index on (objid, colname)
- Unique index on (objid, colnum)
- Unique index on (version, colnum)

## rs\_config

### Description

Holds a set of default configuration parameter values that you can modify using the configure replication server command. You also can set certain parameters for specific targets using the alter connection, alter logical connection, or alter route command.

See the *Replication Server Administration Guide Volume 1* for more information about the configuration parameters in the rs\_config table.

| Column     | Datatype     | Description                                                                                                                                                                                 |
|------------|--------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| optionname | varchar(30)  | Name of the parameter, for example: memory_max, cm_max_connections<br>To view a list of these parameters with their descriptions, execute a select * statement against the rs_config table. |
| objid      | rs_id        | ID of the object this option references. If set to 0, this applies to the whole system.                                                                                                     |
| charvalue  | varchar(255) | Character value for parameter.                                                                                                                                                              |
| status     | tinyint      | This column is not used.                                                                                                                                                                    |
| comments   | varchar(255) | Comment about the parameter.                                                                                                                                                                |

### Indexes

Unique clustered index on (optionname, objid)

## rs\_databases

Description Stores database names known at a Replication Server site.

| Column       | Datatype    | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
|--------------|-------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| dsname       | varchar(30) | Data server name                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| dbname       | varchar(30) | Database name                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
| dbid         | int         | Unique identifier for the database                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| dist_status  | cs_int      | Status of the connection. Can be: <ul style="list-style-type: none"><li>• 0x1 – valid</li><li>• 0x2 – suspended</li><li>• 0x4 – suspended by a standby-related action</li><li>• 0x8 – waiting for a marker</li><li>• 0x10 – will issue dbcc ('lrm', 'ignore')</li><li>• 0x20 – waiting for dump marker to initialize a standby database</li><li>• 0x40 – switching related duplicate detection when <i>ltype</i> is equal to 'P'</li><li>• 0x40 – allow switching when <i>ltype</i> is equal to 'L'</li><li>• 0x80 – temporarily not doing any grouping</li><li>• 0x100 – waiting for a resync marker</li></ul> |
| src_status   | cs_int      | Status of the source: <ul style="list-style-type: none"><li>• 0x1 – valid</li><li>• 0x2 – suspended</li><li>• 0x4 – suspended by a standby-related action</li><li>• 0x10 – DIST thread is suspended</li></ul>                                                                                                                                                                                                                                                                                                                                                                                                   |
| attributes   | tinyint     | One of the following values: <ul style="list-style-type: none"><li>• 1 – distribution</li><li>• 2 – source</li></ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
| errorclassid | rs_id       | Error class for this database                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
| funcclassid  | rs_id       | function-string class for this database                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| prsid        | int         | ID of Replication Server managing this database                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
| rowtype      | tinyint     | Indicates the row type: <ul style="list-style-type: none"><li>• 1 – row is replicated</li><li>• 0 – row not replicated</li></ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| sorto_status | tinyint     | Indicates if the sort order check has been completed. One of the following values: <ul style="list-style-type: none"><li>• 0 – not checked</li><li>• 1 – checked</li></ul>                                                                                                                                                                                                                                                                                                                                                                                                                                      |

| Column          | Datatype | Description                                                                                                                                                                                                                                     |
|-----------------|----------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| ltype           | char(1)  | The type of database this row represents. One of the following values: <ul style="list-style-type: none"> <li>• P – physical database</li> <li>• L – logical database connection</li> </ul>                                                     |
| ptype           | char(1)  | The type of database in a warm standby application. One of the following values: <ul style="list-style-type: none"> <li>• A – the active database</li> <li>• S – the standby database</li> <li>• L – the logical database connection</li> </ul> |
| ldbid           | int      | The dbid for the logical connection the database is associated with. If there is no logical connection, ldbid is the same as dbid.                                                                                                              |
| enable_seq      | int      | The sequence number used during an active database switch or the creation of a standby database.                                                                                                                                                |
| rs_errorclassid | rs_id    | Replication Server error class for this database                                                                                                                                                                                                |

- Indexes
- Unique clustered index on (dsname, dbname, ltype)
  - Unique index on (ptype, ldbid)
  - Unique index on (dbid, ltype)
  - Unique index on (dsname, dbname, ptype)

## rs\_datatype

Description Stores attribute information for all user-defined datatypes (UDDs) in a replication definition.

| Column  | Datatype    | Description                                                                                                                         |
|---------|-------------|-------------------------------------------------------------------------------------------------------------------------------------|
| prsid   | int         | Can be: <ul style="list-style-type: none"> <li>• ID of primary Replication Server</li> <li>• 0 for globally defined UDDs</li> </ul> |
| classid | rs_id       | ID of datatype class to which the datatype belongs                                                                                  |
| dname   | varchar(30) | Unique name of datatype                                                                                                             |
| dtid    | rs_id       | Unique ID of datatype                                                                                                               |

| Column       | Datatype | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
|--------------|----------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| base_coltype | tinyint  | <p>ID of base datatype for the datatype. Can be:</p> <ul style="list-style-type: none"><li>• 0 – char</li><li>• 1 – binary</li><li>• 2 – longchar (not used)</li><li>• 3 – longbinary (not used)</li><li>• 4 – text</li><li>• 5 – image</li><li>• 6 – tinyint</li><li>• 7 – smallint</li><li>• 8 – int</li><li>• 9 – real</li><li>• 10 – float</li><li>• 11 – bit</li><li>• 12 – datetime</li><li>• 13 – smalldatetime</li><li>• 14 – money</li><li>• 15 – smallmoney</li><li>• 16 – numeric</li><li>• 17 – decimal</li><li>• 18 – varchar</li><li>• 19 – varbinary</li><li>• 21 – sensitivity</li><li>• 25 – unichar</li><li>• 27 – date</li><li>• 28 – time</li><li>• 29 – unitext</li></ul> |

| Column         | Datatype     | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
|----------------|--------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|                |              | <ul style="list-style-type: none"> <li>• 30 – bigint</li> <li>• 31 – usmallint</li> <li>• 32 – uint</li> <li>• 33 – ubigint</li> <li>• 35 – bigdatetime</li> <li>• 36 – bigtime</li> <li>• 101 – numeric (literal)</li> <li>• 102 – money (literal)</li> <li>• 103 – real (literal)</li> <li>• 104 – float (literal)</li> <li>• 105 – identity (literal)</li> <li>• 106 – timestamp (literal)</li> <li>• 107 – sensitivity (literal)</li> <li>• 110 – univarchar</li> </ul>                                                                                                                                                                                                                |
| length         | int          | Maximum length of a value of the datatype. For UDDs with masks defined as decimal or money, the value is the maximum precision plus four.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| status         | int          | Status. (See the status column in the rs_columns table.)                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
| length_err_act | tinyint      | Action to be taken if value exceeds length identified in length. Can be: <ul style="list-style-type: none"> <li>• 1 – error</li> <li>• 2 – continue</li> <li>• 3 – truncate left</li> <li>• 4 – truncate right</li> <li>• 5 – round up</li> <li>• 6 – round up and continue on error</li> <li>• 7 – round up and use default on error</li> <li>• 8 – round up and use minimum on error</li> <li>• 9 – round up and use maximum on error</li> <li>• 10 – round down</li> <li>• 11 – round down and continue on error</li> <li>• 12 – round down and use default on error</li> <li>• 13 – round down and use minimum on error</li> <li>• 14 – round down and use maximum on error</li> </ul> |
| mask           | varchar(255) | Datatype mask. Datatype must have base datatype of char for non-null mask.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
| scale          | int          | Maximum number of digits to the right of decimal point. Valid only for masks of money or decimal.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |

| Column                   | Datatype    | Description                                                                                                                                                                                                                              |
|--------------------------|-------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| default_len              | tinyint     | Length of value in default_val column.                                                                                                                                                                                                   |
| default_val              | binary(255) | Default value. Supplies missing components for target value during translation to this datatypes.                                                                                                                                        |
| delim_pre_len            | tinyint     | Length of delim_pre value.                                                                                                                                                                                                               |
| delim_pre                | binary(30)  | Postfixing character or character string used when mapping a non-Java value into a function string. An empty string if the delimiter prefix for the base datatype is used.                                                               |
| delim_post_len           | tinyint     | Length of delim_post.                                                                                                                                                                                                                    |
| delim_post               | binary(30)  | Postfixing character or character string used when mapping a non-Java value into a function string. An empty string if the delimiter prefix for the base datatype is used.                                                               |
| min_boundary_len         | tinyint     | Length of value in min_boundary column. <ul style="list-style-type: none"> <li>• 1 – error</li> <li>• 2 – use default</li> <li>• 3 – use minimum</li> <li>• 4 – use maximum</li> </ul>                                                   |
| min_boundary             | binary(255) | Minimum acceptable value for datatype.                                                                                                                                                                                                   |
| min_boundary_err_act     | tinyint     | Action to be taken if the value exceeds the minimum boundary set by min_boundary. Can be: <ul style="list-style-type: none"> <li>• 1 – error</li> <li>• 2 – use default</li> <li>• 3 – use minimum</li> <li>• 4 – use maximum</li> </ul> |
| max_boundary_len         | tinyint     | Length of value in max_boundary.                                                                                                                                                                                                         |
| max_boundary             | binary(255) | Maximum acceptable value for datatype.                                                                                                                                                                                                   |
| maximum_boundary_err_act | tinyint     | Action to be taken if a value exceeds the maximum boundary set by max_boundary. Can be: <ul style="list-style-type: none"> <li>• 1 – error</li> <li>• 2 – use default</li> <li>• 3 – use minimum</li> <li>• 4 – use maximum</li> </ul>   |

| Column       | Datatype | Description                                                                                                                                                                                                 |
|--------------|----------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| rowtype      | tinyint  | Indicates whether a row is local to Replication Server or distributed to all Replication Servers in the domain. Can be: <ul style="list-style-type: none"> <li>• 0 – local</li> <li>• 1 – global</li> </ul> |
| canonic_type | tinyint  | DSI uses the value of canonic_type to convert the UDD to the correct data type when sending dynamic SQL execute commands. A value of 255 indicates that this datatype is incompatible with dynamic SQL.     |

|         |                                                                                                                                                                                              |
|---------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Indexes | <ul style="list-style-type: none"> <li>• Unique index on (dtid)</li> <li>• Unique index on (name)</li> <li>• Non-unique index on (classid)</li> <li>• Non-unique index on (prsid)</li> </ul> |
|---------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

## rs\_dbreps

Description Stores all information about database replication definitions except name sets. It is replicated to all sites with a version number of 12.6 or later.

| Column      | Datatype      | Description                                                             |
|-------------|---------------|-------------------------------------------------------------------------|
| dbrepid     | rs_id         | Database replication definition ID                                      |
| dbrepname   | varchar (255) | Database replication definition name                                    |
| prsid       | int           | Primary Replication Server ID                                           |
| dbid        | int           | Primary database ID                                                     |
| ownerid     | rs_id         | Replication Server user who created the database replication definition |
| requestdata | datetime      | Time the database replication definition was created                    |

| Column  | Datatype | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
|---------|----------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| status  | int      | Bitmap of subset content: <ul style="list-style-type: none"><li>• 0x0001 – table list appears</li><li>• 0x0002 – tables are negated</li><li>• 0x0004 – function list appears</li><li>• 0x0008 – functions are negated</li><li>• 0x0010 – transaction list appears</li><li>• 0x0020 – transactions are negated</li><li>• 0x0040 – system procedure list appears</li><li>• 0x0080 – system procedure is negated</li><li>• 0x0100 – do not replicate DDL</li><li>• 0x0200 – update list appears</li><li>• 0x0400 – update statement replication is negated for the list</li><li>• 0x0800 – delete list appears</li><li>• 0x1000 – delete statement replication is negated for the list</li><li>• 0x2000 – select into list appears</li><li>• 0x4000 – select into statement replication is negated for the list</li><li>• 0x8000 – insert select list appears</li><li>• 0x10000 – insert select statement replication is negated for the list</li></ul> |
| minvers | int      | Earliest version of Replication Server to which this table can be replicated.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |

Indexes                      Unique indexes on (dbrepid, dbid, and dbrepname).

## rs\_dbsubsets

Description                      Stores the name sets for database replication definitions. It is replicated to all sites with a version number of 12.6 or later.

| Column  | Datatype | Description                        |
|---------|----------|------------------------------------|
| dbrepid | rs_id    | Database replication definition ID |
| prsid   | int      | Primary Replication Server ID      |

| Column  | Datatype                                             | Description                                                                                                                                                                                                                                                                                                    |
|---------|------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| type    | char                                                 | Item type: <ul style="list-style-type: none"><li>• T – table name.</li><li>• F – function name.</li><li>• X – transaction name.</li><li>• P – system procedure name.</li><li>• U – update command</li><li>• L – delete command</li><li>• I – insert select command</li><li>• S – select into command</li></ul> |
| owner   | varchar (30)                                         | Owner name of a table or function, or the user name that executed a transaction or system procedure.<br>An * indicates all owners or users.                                                                                                                                                                    |
| name    | varchar (255)                                        | Table, function, transaction, or system procedure name.<br>An * indicates all tables, functions, transactions, and system procedures.                                                                                                                                                                          |
| Indexes | Unique index on (dbrepid, subtype, owner, and name). |                                                                                                                                                                                                                                                                                                                |

# rs\_diskaffinity

Description Stores information about the affinity between disk partition and database connection or route.

| Column         | Datatype | Description                                                                                                                    |
|----------------|----------|--------------------------------------------------------------------------------------------------------------------------------|
| partition_id   | int      | Partition ID assigned by Replication Server                                                                                    |
| dbid_or_siteid | int      | An ID for a Replication Server or database                                                                                     |
| status         | int      | Status of the affinity. Valid values are: <ul style="list-style-type: none"><li>0x01 - valid</li><li>0x02 - obsolete</li></ul> |

Indexes Unique clustered index on (dbid\_or\_siteid)

## rs\_diskpartitions

**Description** Stores information about the disk partitions that Replication Server uses for stable message queues.

| Column       | Datatype     | Description                                                                                                                                              |
|--------------|--------------|----------------------------------------------------------------------------------------------------------------------------------------------------------|
| name         | varchar(255) | Operating system name for the disk device                                                                                                                |
| logical_name | varchar(30)  | User-assigned name for the partition                                                                                                                     |
| id           | int          | Partition ID assigned by Replication Server                                                                                                              |
| num_segs     | int          | Total size of the partition in segments                                                                                                                  |
| status       | int          | Status of the disk partition. Valid values are: <ul style="list-style-type: none"> <li>• 1 – online</li> <li>• 2 – partition is being dropped</li> </ul> |
| vstart       | int          | Offset at which Replication Server starts writing to the partition (in MB)                                                                               |

**Indexes**

- Unique clustered index on (logical\_name)
- Unique index on (name)

## rs\_erroractions

Description                      Maps a data server error number to an action to be taken by a Replication Server.

| Column       | Datatype | Description                                                                                                                                                                                                                                                                                                                                                                                                                    |
|--------------|----------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| ds_errorid   | int      | Data server error number                                                                                                                                                                                                                                                                                                                                                                                                       |
| errorclassid | rs_id    | Error class ID (see rs_classes)                                                                                                                                                                                                                                                                                                                                                                                                |
| action       | tinyint  | Action to take when error occurs: <ul style="list-style-type: none"><li>• 1 – ignore the error</li><li>• 2 – stop replication</li><li>• 3 – output a warning message</li><li>• 4 – write an entry in the exceptions log</li><li>• 5 – retry the transaction and then log the transaction if it still fails</li><li>• 6 – retry the transaction a certain number of times and then stop replication if it still fails</li></ul> |
| prsid        | int      | Site where this row is primary                                                                                                                                                                                                                                                                                                                                                                                                 |

- Indexes
- Unique index on (ds\_errorid, errorclassid)
  - Clustered index on (errorclassid)

## rs\_exceptscmd

- Description** Stores the information used to retrieve the text of transactions from the exceptions log. The text, stored in the `rs_systext` system table, includes:
- Source command – the text of the user transaction received by Replication Server.
  - Output command – the text of the transaction that Replication Server prepared for the database from function strings. The output command can be either a language command or an RPC.

`rs_exceptscmd` has one row for each source command or output command.

| Column                        | Datatype             | Description                                                                                                                                                   |
|-------------------------------|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>sys_trans_id</code>     | <code>rs_id</code>   | System-assigned transaction ID for the transaction                                                                                                            |
| <code>src_cmd_line</code>     | <code>int</code>     | Command-line number of the source within the logged transaction                                                                                               |
| <code>output_cmd_index</code> | <code>int</code>     | Line number of the output command within the logged transaction                                                                                               |
| <code>cmd_type</code>         | <code>char(1)</code> | Command type: <ul style="list-style-type: none"> <li>• S – source command</li> <li>• L – language output command</li> <li>• R – RPC output command</li> </ul> |
| <code>cmd_id</code>           | <code>rs_id</code>   | Index into <code>rs_systext</code>                                                                                                                            |

**Indexes** Unique index on (`cmd_id`)

## rs\_exceptshdr

**Description** Stores information about failed transactions. The source and output commands of the transactions are stored in the system tables `rs_exceptscmd` and `rs_systext`. All rows for a transaction in `rs_exceptscmd` and `rs_exceptshdr` are identified by the column `sys_trans_id`.

| Column                       | Datatype                  | Description                                                                                                                                                                                                                                                                                                                                                                                                                            |
|------------------------------|---------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>sys_trans_id</code>    | <code>rs_id</code>        | System-assigned transaction ID for this transaction                                                                                                                                                                                                                                                                                                                                                                                    |
| <code>rs_trans_id</code>     | <code>binary(120)</code>  | Replication Server-generated unique transaction ID                                                                                                                                                                                                                                                                                                                                                                                     |
| <code>app_trans_name</code>  | <code>varchar(30)</code>  | User-specified transaction name                                                                                                                                                                                                                                                                                                                                                                                                        |
| <code>orig_siteid</code>     | <code>int</code>          | ID of the origin database                                                                                                                                                                                                                                                                                                                                                                                                              |
| <code>orig_site</code>       | <code>varchar(30)</code>  | Data server name for the origin database                                                                                                                                                                                                                                                                                                                                                                                               |
| <code>orig_db</code>         | <code>varchar(30)</code>  | Name of the origin database                                                                                                                                                                                                                                                                                                                                                                                                            |
| <code>orig_time</code>       | <code>datetime</code>     | Time the transaction was initiated                                                                                                                                                                                                                                                                                                                                                                                                     |
| <code>orig_user</code>       | <code>varchar(30)</code>  | User who submitted the transaction at the origin site                                                                                                                                                                                                                                                                                                                                                                                  |
| <code>error_siteid</code>    | <code>int</code>          | ID of the site where the error occurred                                                                                                                                                                                                                                                                                                                                                                                                |
| <code>error_site</code>      | <code>varchar(30)</code>  | Name of the data server where the error occurred                                                                                                                                                                                                                                                                                                                                                                                       |
| <code>error_db</code>        | <code>varchar(30)</code>  | Name of the database where the error occurred                                                                                                                                                                                                                                                                                                                                                                                          |
| <code>log_time</code>        | <code>datetime</code>     | Time the error occurred                                                                                                                                                                                                                                                                                                                                                                                                                |
| <code>ds_error</code>        | <code>int</code>          | Data server error number                                                                                                                                                                                                                                                                                                                                                                                                               |
| <code>ds_errmsg</code>       | <code>varchar(255)</code> | Data server error message                                                                                                                                                                                                                                                                                                                                                                                                              |
| <code>error_src_line</code>  | <code>int</code>          | Line number of the command that caused the error                                                                                                                                                                                                                                                                                                                                                                                       |
| <code>error_proc</code>      | <code>varchar(255)</code> | Procedure during which the error occurred                                                                                                                                                                                                                                                                                                                                                                                              |
| <code>err_output_line</code> | <code>int</code>          | Line number of the output command that caused the error                                                                                                                                                                                                                                                                                                                                                                                |
| <code>log_reason</code>      | <code>char(1)</code>      | Why the transaction was logged: <ul style="list-style-type: none"><li>• O – indicates an orphan transaction in the DSI queue</li><li>• E – a data server error mapped to LOG or RETRY_LOG</li><li>• S – indicates the transaction was skipped because the resume connection command was executed with the skip transaction option</li><li>• D – the transaction was logged by a sysadmin <code>log_first_tran</code> command</li></ul> |
| <code>trans_status</code>    | <code>smallint</code>     | Transaction status—one or more of the following: <ul style="list-style-type: none"><li>• 0x0001 – orphan transaction</li><li>• 0x0002 – logged transaction was going to primary site</li><li>• 0x0004 – conflicting transaction</li></ul>                                                                                                                                                                                              |
| <code>retry_status</code>    | <code>smallint</code>     | Retry status for the transaction—one of the following: <ul style="list-style-type: none"><li>• 1 – retry succeeded</li><li>• 2 – transaction has not committed</li></ul>                                                                                                                                                                                                                                                               |
| <code>app_usr</code>         | <code>varchar(30)</code>  | Name of the user who applied the transaction at the error site                                                                                                                                                                                                                                                                                                                                                                         |

| Column  | Datatype    | Description                                                        |
|---------|-------------|--------------------------------------------------------------------|
| app_pwd | varchar(30) | Password of the user who applied the transaction at the error site |
| Indexes |             | Unique index on (sys_trans_id)                                     |

## rs\_exceptslast

**Description** Stores the origin ID, secondary queue ID, and associated information about the last logged transaction written into the exceptions log.

| Column        | Datatype   | Description                                                                                                                                                                                                                                                                                                                                    |
|---------------|------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| error_db      | int        | Database where the error occurred                                                                                                                                                                                                                                                                                                              |
| origin        | int        | Origin database of the transactions                                                                                                                                                                                                                                                                                                            |
| origin_qid    | binary(36) | qid of the last transaction from this origin                                                                                                                                                                                                                                                                                                   |
| secondary_qid | binary(36) | Secondary qid of the last logged transaction from this origin                                                                                                                                                                                                                                                                                  |
| status        | tinyint    | Status of the transaction: <ul style="list-style-type: none"><li>• 0 – Valid: no transactions were lost for this origin</li><li>• 1 – Detecting losses: you should determine if any transactions have been lost in this origin</li><li>• 2 – Rejecting messages after loss detected: transactions were probably lost for this origin</li></ul> |
| origin_time   | datetime   | Time at origin for the transaction                                                                                                                                                                                                                                                                                                             |
| log_time      | datetime   | Time the transaction was logged                                                                                                                                                                                                                                                                                                                |
| lorigin       | int        | Logical database where the message originated                                                                                                                                                                                                                                                                                                  |

**Indexes**

- Unique index on (error\_db, origin)
- Unique index on (error\_db, origin, status)

## rs\_funcstrings

Description Stores the function strings associated with each function.

| Column     | Datatype     | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
|------------|--------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| prsid      | int          | Site where this row is primary                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| classid    | rs_id        | Class the function string belongs to                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| funcid     | rs_id        | Function this string is for                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| name       | varchar(255) | Function string name                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| fstringid  | rs_id        | ID for this function string                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| attributes | smallint     | Attributes of the function string: <ul style="list-style-type: none"> <li>• 0x01 – conflicting function</li> <li>• 0x02 – RPC</li> <li>• 0x04 - altered</li> <li>• 0x08 – used for all functions other than rs_writetext and indicates that a function has no output command and that nothing is sent to the replicate data server.</li> <li>• 0x10 – default input</li> <li>• 0x20 – default output</li> <li>• 0x40 – writetext output is used for an rs_writetext function string</li> <li>• 0x80 – writetext output is used with the with log option for an rs_writetext function string</li> <li>• 0x100 – a function string for an rs_writetext, rs_textptr_init, or rs_get_textptr function</li> <li>• 0x200 – writetext output is used with the no log option for an rs_writetext function string</li> <li>• 0x400 – function string includes one or more variables that will access the values of non-key columns</li> <li>• 0x800 – the <i>rs_default_fs</i> system variable was used in output language template</li> <li>• 0x1000 – none output is used for an rs_writetext function string</li> </ul> |
| parameters | smallint     | Number of parameters in this function string                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| param_hash | int          | Hash value of input template                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| expiredate | datetime     | Date the function string should expire. This is used for dynamic function string expiration                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| rowtype    | tinyint      | 1 if this row is replicated, 0 if not                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| minvers    | int          | Minimum version required to support the function string. This means that if a function string has minvers value of 15.0, it will not replicate to sites below 15.0                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |

Indexes

- Unique clustered index on (classid, funcid, name)

- Unique index on (fstringid)

## rs\_functions

Description Stores information about Replication Server functions.

| Column      | Datatype     | Description                                                                                                           |
|-------------|--------------|-----------------------------------------------------------------------------------------------------------------------|
| prsid       | int          | Site where the function is primary                                                                                    |
| funcname    | varchar(255) | Name of the function                                                                                                  |
| funcid      | rs_id        | ID of the function                                                                                                    |
| objid       | rs_id        | Object to which the function applies. NULL_OBJECT_ID (0x00000000) is stored in this column for class-scope functions. |
| conflicting | tinyint      | 1 if the function is conflicting, 0 if not                                                                            |
| userdefined | bit          | 1 if this is a user-defined function, 0 if not                                                                        |
| rowtype     | tinyint      | 1 if this row is replicated, 0 if not                                                                                 |

Indexes

- Clustered index on (objid)
- Unique index on (objid, funcname)
- Unique index on (funcid)

# rs\_idnames

Description Stores the names of Replication Servers and databases known to the ID server. This table is relevant only at the ID Server site.

| Column | Datatype    | Description                                                                                                                      |
|--------|-------------|----------------------------------------------------------------------------------------------------------------------------------|
| name1  | varchar(30) | Replication Server or data server name                                                                                           |
| name2  | varchar(30) | Database name; "" for a Replication Server                                                                                       |
| type   | int         | Replication Server or database: <ul style="list-style-type: none"><li>• 8 – Replication Server</li><li>• 9 – database</li></ul>  |
| id     | int         | Unique ID assigned to the Replication Server or database                                                                         |
| ltype  | char(1)     | The type of the database: <ul style="list-style-type: none"><li>• P – Physical database</li><li>• L – Logical database</li></ul> |

Indexes Unique clustered index on (name1, name2, ltype)

## rs\_ids

Description Stores the last ID used for various types of objects.

| Column   | Datatype    | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
|----------|-------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| typename | varchar(30) | Name of this object type. For example, “subscriptions,” “objects”                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
| objid    | int         | Last ID used for this object type                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
| objtype  | tinyint     | <ul style="list-style-type: none"> <li>Object type: <ul style="list-style-type: none"> <li>1 – Subscriptions</li> <li>2 – Objects</li> <li>3 – Classes</li> <li>4 – Users</li> <li>5 – Functions</li> <li>6 – Function strings</li> <li>7 – Error log</li> </ul> </li> <li>Exception log types: <ul style="list-style-type: none"> <li>12 – Reject transaction</li> </ul> </li> <li>Site ID types: <ul style="list-style-type: none"> <li>8 – Replication Server ID</li> <li>9 – Database ID</li> </ul> </li> <li>Stable queue parameters: <ul style="list-style-type: none"> <li>10 – Disk partition IDs</li> </ul> </li> <li>Counter used by subscriptions module: <ul style="list-style-type: none"> <li>13 – Counter for subscriptions module</li> </ul> </li> <li>Recovery manager IDs: <ul style="list-style-type: none"> <li>14 – Recovery ID type</li> <li>15 – Rematerialization ID</li> <li>16 – Publication ID</li> <li>17 – Article ID</li> <li>18 – where clause ID</li> <li>19 – UDD ID</li> </ul> </li> </ul> |

Indexes Unique clustered index on (objtype)

## rs\_lastcommit

**Description** Replication Server uses the information in this table to find the last transaction committed from each data source.

The rs\_lastcommit table is stored in each user database, not in the RSSD.

| Column           | Datatype    | Description                                                                                                                                                                        |
|------------------|-------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| origin           | int         | ID number for the primary database a row represents.                                                                                                                               |
| origin_qid       | binary      | Identifies the last committed transaction in the stable queue for the origin database.                                                                                             |
| secondary_qid    | binary      | If a subscription materialization queue exists for the origin database, this column contains the last transaction in that queue that has been committed in the replicate database. |
| origin_time      | datetime    | Time at origin for the transaction.                                                                                                                                                |
| dest_commit_time | datetime    | Time the transaction was committed at the destination.                                                                                                                             |
| pad1             | binary(255) | Filler to pad the row so only one row fits on a database page.                                                                                                                     |
| pad2             | binary(255) | Filler to pad the row so only one row fits on a database page.                                                                                                                     |
| pad3             | binary(255) | Filler to pad the row so only one row fits on a database page.                                                                                                                     |
| pad4             | binary(255) | Filler to pad the row so only one row fits on a database page.                                                                                                                     |
| pad5             | binary(255) | Filler to pad the row so only one row fits on a database page.                                                                                                                     |
| pad6             | binary(255) | Filler to pad the row so only one row fits on a database page.                                                                                                                     |
| pad7             | binary(255) | Filler to pad the row so only one row fits on a database page.                                                                                                                     |
| pad8             | binary(83)  | Filler to pad the row so only one row fits on a database page.                                                                                                                     |

**Indexes** Unique clustered index on (origin)

## rs\_locator

**Description** Stores the last locator field received by stable queues from each of their senders.

| Column  | Datatype   | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
|---------|------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| sender  | int        | Sender site ID                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| type    | char(1)    | Who is using this row: <ul style="list-style-type: none"> <li>• R – RSI (route)</li> <li>• D – distributor locator used for subscriptions</li> <li>• E – executor for Replication Agent</li> <li>• U – locator at last system upgrade</li> <li>• W – distributor locator used for a warm standby application</li> <li>• C – locator for the last successfully executed replication definition request sent by Replication Agent.</li> <li>• F – locator for failed command sent by Replication Agent.</li> <li>• S – locator for failed command that Replication Server skipped.</li> </ul> |
| locator | binary(36) | Last queue ID received from this sender                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |

When you execute `rs_send_repserver_cmd` at the primary database, `rs_locator` stores the OQID of the RCL in type according to whether the RCL inside '`rs_apl`' executed:

- Successfully – type C
- Unsuccessfully – type F

When you execute `sysadmin skip_bad_repserver_cmd` for the RCL that failed, Replication Server updates the `rs_locator` entry to type "S." Replication Server skips the failed command the next time Replication Agent sends the command.

**Indexes** Unique clustered index on (sender, type)

## rs\_maintusers

**Description** Stores the user login names and passwords Replication Server uses to access other Replication Servers and data servers.

| Column           | Datatype    | Description                                                                                                      |
|------------------|-------------|------------------------------------------------------------------------------------------------------------------|
| destid           | int         | Site ID for the Replication Server or database to be logged into                                                 |
| username         | varchar(30) | User name for the Replication Server RSI user or for the database maintenance user                               |
| password         | varchar(30) | Password                                                                                                         |
| use_enc_password | int         | <ul style="list-style-type: none"><li>• 0 – use normal passwords</li><li>• 1 – use encrypted passwords</li></ul> |
| enc_password     | varchar(66) | Encrypted user password                                                                                          |

**Indexes** Unique clustered index on (destid)

## rs\_msgs

Description Stores the localized error messages used during installation and by some Replication Server stored procedures.

| Column   | Datatype     | Description                                                                                                                                 |
|----------|--------------|---------------------------------------------------------------------------------------------------------------------------------------------|
| msgnum   | int          | Unique ID number for the message                                                                                                            |
| langname | char(30)     | Local language name of this version of the message text. Corresponds to the @@ <i>language</i> global variable in the RSSD Adaptive Server. |
| msgtxt   | varchar(255) | Text of the message, in the localized language.                                                                                             |

Indexes Unique clustered index on (msgnum, langname)

## rs\_objects

Description Stores replication definitions, one per row.

| Column  | Datatype     | Description                                                                                                                                                         |
|---------|--------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| prsid   | int          | Primary Replication Server where this object was created                                                                                                            |
| objname | varchar(255) | Object name                                                                                                                                                         |
| objid   | rs_id        | Object ID                                                                                                                                                           |
| dbid    | int          | Unique ID for data server and database                                                                                                                              |
| objtype | char(1)      | One of the following object types: <ul style="list-style-type: none"><li>• R – table replication definition</li><li>• F – function replication definition</li></ul> |

| Column     | Datatype | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
|------------|----------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| attributes | int      | <p>Mask, can be one or more of the following:</p> <ul style="list-style-type: none"> <li>• 0x01 – generate dynamic function strings.</li> <li>• 0x02 – replication definition has <b>bigdatetime</b> or <b>bigtime</b> columns and can be propagated only to Replication Server 15.5 or later.</li> <li>• 0x04 – minimum columns enabled for replication definition.</li> <li>• 0x08 – replication definition has <b>identity</b> column.</li> <li>• 0x10 – <b>replicate_if_changed</b> status.</li> <li>• 0x20 – replication definition has a drop pending.</li> <li>• 0x40 – replication definition has <b>text</b>, <b>unitext</b>, or <b>image</b> column.</li> <li>• 0x80 – replication definition is used by a standby.</li> <li>• 0x0100 – replication definition's columns are sent to standby database.</li> <li>• 0x0200 – replication definition is propagated to Replication Servers version 11.0.x or earlier.</li> <li>• 0x0400 – replication definition has been used as a base replication definition for the primary table.</li> <li>• 0x0800 – replication definition is internal only.</li> <li>• 0x1000 – object or column names differ in the primary and replicate tables.</li> <li>• 0x4000 – replication definition has column-level translations.</li> <li>• 0x8000 – replication definition has columns declared with UDDs.</li> <li>• 0x10000 – replication definition has <b>char</b>, <b>varchar</b>, <b>binary</b>, or <b>varbinary</b> columns with more than 255 bytes and can be propagated only to Replication Server 12.5 or later.</li> <li>• 0x20000 – replication definition has <b>unichar</b> or <b>univarchar</b> columns and can be propagated only to Replication Server 12.5 or later.</li> <li>• 0x40000 – replication definition has <b>date</b> or <b>time</b> columns and can be propagated only to Replication Server 12.6 or later.</li> <li>• 0x80000 – replication definition has <b>timestamp</b> columns. Propagated to Replication Server 15.1 as <b>timestamp</b>, and propagated to Replication Server 15.0.1 or earlier as <b>varbinary</b>.</li> <li>• 0x200000 – applied function replication definition and can be propagated only to Replication Server 15.1.</li> <li>• 0x400000 – request function replication definition and can be propagated only to Replication Server 15.1.</li> <li>• 0x800000 – dynamic SQL is not used on the table.</li> <li>• 0x2000000 - update is enabled for SQL replication.</li> <li>• 0x4000000 - delete is enabled for SQL replication.</li> <li>• 0x8000000 - insert select is enabled for SQL replication</li> <li>• 0x10000000 - SQL replication is disabled by repserver internally</li> </ul> |

| Column          | Datatype     | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
|-----------------|--------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| ownertype       | char(1)      | Type of owner of this object: <ul style="list-style-type: none"> <li>• U – user</li> <li>• S – System</li> </ul>                                                                                                                                                                                                                                                                                                                                                |
| crdate          | datetime     | Date and time created                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| parentid        | rs_id        | Reserved for future use.                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| ownerid         | rs_id        | ID of the user who created this object                                                                                                                                                                                                                                                                                                                                                                                                                          |
| rowtype         | tinyint      | 1 if row is replicated, 0 if not                                                                                                                                                                                                                                                                                                                                                                                                                                |
| phys_tablename  | varchar(255) | Primary table name – used when communicating with data server about this object                                                                                                                                                                                                                                                                                                                                                                                 |
| deliver_as_name | varchar(255) | Name of the replicate table or stored procedure                                                                                                                                                                                                                                                                                                                                                                                                                 |
| phys_objowner   | char(30)     | Name of the primary table owner, as specified in replication definition.<br>Blank if the table owner is not specified.                                                                                                                                                                                                                                                                                                                                          |
| repl_objowner   | char(30)     | Name of the replicate table owner, as specified in replication definition.<br>Blank if the table owner is not specified.                                                                                                                                                                                                                                                                                                                                        |
| has_baserepdef  | rs_id        | If this is not a base replication definition, the value of has_baserepdef matches that of objid for the base replication definition. Or, has the following value:<br>0x00 - Base replication definition                                                                                                                                                                                                                                                         |
| minvers         | int          | Specifies the minimum version of a replication definition, and thus the Replication Server to which it can propagate. Can be: <ul style="list-style-type: none"> <li>• 1200 – propagates to Replication Server version 12 or later</li> <li>• 1150 – propagates to Replication Server version 11.5 or later</li> <li>• 1000 or 0 (zero) – propagates to any Replication Server</li> <li>• 0 (zero) – for function and system replication definitions</li> </ul> |
| version         | rs_id        | Uniquely identifies a replication definition version.                                                                                                                                                                                                                                                                                                                                                                                                           |
| active_inbound  | int          | Executor uses this column to decide which replication definition version to use. Executor uses the replication definition version whose value for the active_inbound column is 0.                                                                                                                                                                                                                                                                               |
| attributes2     | int          | <ul style="list-style-type: none"> <li>• 0x01 – Distributor uses this to identify replication definition versions which are not in use by Distributor.</li> <li>• 0x02 – The standby DSI uses this to identify replication definition versions which are not in use by the standby DSI.</li> </ul>                                                                                                                                                              |

Replication Server may create a new replication definition version when you alter a replication definition. When that happens, Replication Server changes the name of the old replication definition version to a unique name with a “rs\_drp” prefix. Replication Server treats replication definitions with names prefixed with “rs\_drp” or “rs\_in” as internal replication definitions.

Replication Server deletes an internal replication definition when the data associated with the replication definition is no longer in the replication system.

To exclude internal replication definitions from a query to the `rs_objects` table, add to the `where` clause of the query:

```
and objname not like 'rs_drp%' and objname not like
'rs_in%'
```

For example, this query returns the name of all replication definitions created against the `ling.authors` primary table, and excludes the internal replication definitions:

```
select objname from rs_objects where prsid = 16777317
and dbid = 104 and phys_tablename = 'authors' and
phys_objowner = 'ling' and objname not like 'rs_drp%'
and objname not like 'rs_in%'
```

#### Indexes

- Unique clustered index on (`objname`)
- Unique index on (`dbid`, `phys_tablename`, `phys_objowner`, `objtype`, `has_baserepdef`, `active_inbound`)
- Unique index on (`objid`)
- Unique index on (`version`)

## rs\_oqid

**Description** Stores the last queue ID received from an origin site, and is also used to coordinate the resetting of truncation points.

| Column          | Datatype   | Description                                                                                                                                                        |
|-----------------|------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| origin_site_id  | int        | Site ID of the origin site                                                                                                                                         |
| q_number        | int        | Queue number                                                                                                                                                       |
| q_type          | int        | Queue type                                                                                                                                                         |
| origin_q_id     | binary(36) | Command ID at the origin database                                                                                                                                  |
| local_q_id      | binary(36) | Local ID for the queue                                                                                                                                             |
| valid           | int        | Validation status: <ul style="list-style-type: none"><li>• 0 – valid</li><li>• 1 – detecting losses</li><li>• 2 – rejecting messages after loss detected</li></ul> |
| origin_lsite_id | int        | Site ID of the logical database of the origin site                                                                                                                 |

**Indexes**

- Unique clustered index on (origin\_site\_id, q\_number, q\_type)

## rs\_profdetail

**Description** Records details associated with a Replication Server profile.

| Column      | Datatype     | Description                                                                                                                                                                                                                                                                                                                                            |
|-------------|--------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| profid      | rs_id        | Profile ID, a foreign key to the rs_profile table                                                                                                                                                                                                                                                                                                      |
| name        | varchar(255) | Profile name. Can be an empty string                                                                                                                                                                                                                                                                                                                   |
| pdetailtype | int          | Specifies the action that must be taken for the profile: <ul style="list-style-type: none"><li>• 1 – Appends the connection configuration to the create connection command</li><li>• 2 – Executes class-level translation definition in the RSSD.</li><li>• 3 – Executes the replicate database object definition in the replicate database.</li></ul> |
| pdetailid   | rs_id        | Profile detail ID, a foreign key to the rs_systext table                                                                                                                                                                                                                                                                                               |
| sequence    | int          | Indicates the profile detail sequence within a profile. Replication Server uses sequence to determine the order in which profile detail actions will be executed.                                                                                                                                                                                      |

**Note** The create connection options for Replication Server are always executed first regardless of how they are identified by sequence in the profile detail.

**Indexes**

- Unique index on (profid, sequence)

- Unique index on (id)
- Non-unique index on (profid)

## rs\_profile

Description Stores currently defined Replication Server profiles.

| Column   | Datatype     | Description                                                                            |
|----------|--------------|----------------------------------------------------------------------------------------|
| name     | varchar(255) | Profile name                                                                           |
| vers     | varchar(255) | Profile version. Can be an empty string                                                |
| id       | rs_id        | Profile ID                                                                             |
| type     | char(1)      | Profile type: <ul style="list-style-type: none"><li>• C – Connection profile</li></ul> |
| comments | varchar(255) | Profile description and information                                                    |

Indexes

- Unique index on (name, vers, type)
- Non-unique index on (type)

## rs\_publications

Description Stores information about publications known to this Replication Server.

| Column      | Datatype     | Description                                                                                                 |
|-------------|--------------|-------------------------------------------------------------------------------------------------------------|
| prsid       | int          | Primary Replication Server where the publication was created                                                |
| pubname     | varchar(255) | Name of the publication                                                                                     |
| pubid       | rs_id        | Unique publication ID                                                                                       |
| pdbid       | int          | Unique ID for the publication's primary data server and database                                            |
| requestdate | datetime     | Date and time the last article was added to the publication                                                 |
| ownerid     | rs_id        | ID of the user who created the publication                                                                  |
| status      | int          | Publication status: <ul style="list-style-type: none"><li>• 0x00 – Invalid</li><li>• 0x01 – Valid</li></ul> |
| minvers     | int          | Minimum Replication Server version required to support this publication                                     |

Indexes

- Unique clustered index on (pubname, pdbid)
- Unique index on (pubid)

## rs\_queuemsg

**Description** When you dump Replication Server queues into the RSSD, the queue entries are stored in rs\_queuemsg. If this table already has rows for a segment, those rows are deleted from the table before the latest rows from that segment are dumped.

| Column          | Datatype    | Description                                                                |
|-----------------|-------------|----------------------------------------------------------------------------|
| q_number        | int         | Queue number                                                               |
| q_type          | int         | Queue type                                                                 |
| q_seg           | int         | Queue segment                                                              |
| q_blk           | int         | Queue block                                                                |
| q_row           | int         | Queue row                                                                  |
| len             | int         | Length of the queue entry                                                  |
| origin_site_id  | int         | Origin site ID                                                             |
| origin_q_id     | binary(36)  | Queue ID assigned by the origin                                            |
| origin_time     | datetime    | Time transaction was initiated                                             |
| origin_user     | varchar(30) | User who submitted transaction at origin site                              |
| tran_name       | varchar(30) | Transaction name                                                           |
| local_q_id      | binary(36)  | Queue ID assigned by the local Replication Server                          |
| status          | int         | Message status                                                             |
| reserved        | int         | Reserved for future use                                                    |
| tran_len        | smallint    | Length of tran_id                                                          |
| txt_len         | smallint    | Length of command                                                          |
| tran_id         | binary(120) | Transaction ID                                                             |
| lorigin_site_id | int         | Site ID of the logical connection that is the source of the queue entries. |
| version         | int         | Release version of the message                                             |

**Indexes** Unique clustered index on (q\_number, q\_type, q\_seg, q\_blk, q\_row)

## rs\_queuemsgtxt

Description

Stores the command or text portion of messages in stable queues. Each stable queue entry is represented by one or more rows in this table. Multiple rows are needed when the length of data in the stable queue entry exceeds the maximum command field length of 255 bytes.

| Column   | Datatype     | Description                                        |
|----------|--------------|----------------------------------------------------|
| q_number | int          | Queue number                                       |
| q_type   | int          | Queue type                                         |
| q_seg    | int          | Segment that contains the message                  |
| q_blk    | int          | Block within the segment that contains the message |
| q_row    | int          | Row within the block that contains the message     |
| q_seq    | int          | Sequence number of the row for this entry          |
| txt      | varchar(255) | Text of the entry                                  |
| txtbin   | binary(255)  | Text in binary                                     |

Indexes

Unique default index on (q\_number, q\_type, q\_seq, q\_seg, q\_blk, q\_row)

## rs\_queues

**Description** Stores information to allow site recovery. Used by the Replication Server stable queue manager and guaranteed delivery system.

| Column  | Datatype | Description                                                                                                                                                                                                                                                                                                                                                                                                                        |
|---------|----------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| number  | int      | Queue ID. This column displays a number representing either: <ul style="list-style-type: none"> <li>• The source database for an inbound queue, or</li> <li>• The destination database or Replication Server for an outbound queue</li> </ul> Values correspond to entries for databases in the dbid column in the rs_databases system table and to entries for Replication Servers in the id column in the rs_sites system table. |
| type    | int      | Queue type: <ul style="list-style-type: none"> <li>• 0 – outbound queue</li> <li>• 1 – inbound queue</li> <li>• large negative number – a subscription materialization queue</li> </ul>                                                                                                                                                                                                                                            |
| state   | int      | Current state of this queue: <ul style="list-style-type: none"> <li>• 0 – failure</li> <li>• 1 – active</li> <li>• 2 – deleting</li> </ul>                                                                                                                                                                                                                                                                                         |
| twosave | int      | Indicates the number of seconds the Replication Server maintains an SQM segment after all messages in the segment have been acknowledged by targets. A setting of -1 indicates a strict setting.                                                                                                                                                                                                                                   |
| truncs  | int      | The number of truncation points                                                                                                                                                                                                                                                                                                                                                                                                    |

**Indexes** Unique clustered index on (number, type)

## rs\_recovery

**Description** Logs actions that must be performed by Replication Server upon recovery, if there is a failure.

| Column  | Datatype    | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
|---------|-------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| action  | int         | Represents the recoverable actions: <ul style="list-style-type: none"><li>• 1 – create_route</li><li>• 2 – drop_route</li><li>• 3 – standalone mode</li><li>• 4 – rebuild queues</li><li>• 5 – log recovery</li><li>• 6 – restart LTM at the top of the log</li><li>• 7 – create standby</li><li>• 8 – switch active</li><li>• 9 – strict save interval for DSI or materialization queue</li><li>• 10 – quit DSI secondary duplicate detection after switch active</li><li>• 11 – drop standby</li><li>• 12 – alter distributor locator</li><li>• 13 – delete segments with replication definitions</li><li>• 14 – drop pending replication definitions</li><li>• 15 – drop pending table or function replication definition with reference counter</li><li>• 16 – create schema replication definition (for auto-generating schema replication definition)</li></ul> |
| id      | rs_id       | Each row is assigned a unique ID.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| seqnum  | int         | For actions with multiple rows, this column stores the sequence number of each row.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
| state   | int         | Contains the current state for recoverable actions that move through a finite number of states.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| text    | binary(255) | Data required to complete the action.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
| textlen | int         | Length of the text data.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |

**Indexes** Unique index on (id)

## rs\_repdb

**Description** Contains information about all of the databases known by a primary Replication Server. This information is stored when a subscription is entered for a database at a replicate site.

| Column       | Datatype    | Description                                   |
|--------------|-------------|-----------------------------------------------|
| dbid         | int         | Unique database ID                            |
| dsname       | varchar(30) | Data server name                              |
| dbname       | varchar(30) | Database name                                 |
| controllerid | int         | Managing Replication Server for this database |

**Indexes**

- Clustered index on (controllerid)
- Unique index on (dbid)
- Unique index on (dsname, dbname)

## rs\_repobjs

**Description** Stores autocorrection flags for replication definitions at replicate Replication Servers. Set the flag to on or off using the set autocorrection command.

| Column     | Datatype | Description                                                                                                                                                              |
|------------|----------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| objid      | rs_id    | Replication definition object ID                                                                                                                                         |
| dbid       | int      | ID of the database where the replicate data is stored                                                                                                                    |
| attributes | int      | Valid value: <ul style="list-style-type: none"><li>• 0x01 – autocorrection flag is on</li><li>• 0x02 – Dynamic SQL is not used for the replication definition.</li></ul> |

**Indexes** Unique clustered index on (objid, dbid)

## rs\_routes

Description Stores routing information about network traffic.

| Column       | Datatype | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
|--------------|----------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| dest_rsid    | int      | ID of a data server or Replication Server                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| through_rsid | int      | Destination is reached through this Replication Server. For a direct route, the value of through_rsid is the same as that of dest_id.                                                                                                                                                                                                                                                                                                                                                                                                                           |
| source_rsid  | int      | Replication Server where this route is defined                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| status       | tinyint  | Status of the route: <ul style="list-style-type: none"> <li>• 1 – being initialized</li> <li>• 2 – route is valid at this site (route is valid when status is 2 at both the source and destination Replication Servers)</li> <li>• 3 – dropping this route gracefully</li> <li>• 4 – dropping this route immediately</li> </ul>                                                                                                                                                                                                                                 |
| suspended    | tinyint  | One of the following values: <ul style="list-style-type: none"> <li>• 0 – route is active</li> <li>• 1 – route is suspended</li> <li>• 2 – route is being rebuilt. In the process of setting the truncation point.</li> <li>• 3 – route is suspended. In the process of setting the truncation point.</li> <li>• 8 (mask) – for an RSI outbound queue, instructs the replicate Replication Server to set the <code>locator</code> field in the <code>rs_locator</code> table to 0, for this sending Replication Server.</li> </ul>                              |
| src_version  | int      | Version of source Replication Server for this route. Note that this version is the RSI version (not what appears in the <code>rs_config</code> stored procedure under <code>current_rssd_version</code> ). <ul style="list-style-type: none"> <li>• 1000 – version assigned to any pre-10.1 Replication Server</li> <li>• 1010 – version 10.1</li> <li>• 1100 – version 11.0</li> <li>• 1150 – version 11.5</li> <li>• 1200 – version 12.0</li> </ul> Refer to the <i>Release Bulletin for Replication Server</i> for any additional supported version numbers. |

Indexes Unique clustered index on (dest\_rsid, source\_rsid)

## rs\_routeversions

**Description** Stores version information about the Replication Servers on each end of a route.

| Column           | Datatype | Description                                                                                                                                                                                                                                                                   |
|------------------|----------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| dest_rsid        | int      | ID of the destination Replication Server                                                                                                                                                                                                                                      |
| source_rsid      | int      | ID of the source Replication Server where this route is defined                                                                                                                                                                                                               |
| dest_rssd_id     | int      | ID of the RSSD of the destination Replication Server                                                                                                                                                                                                                          |
| route_version    | int      | The minimum site version of the destination and source Replication Server                                                                                                                                                                                                     |
| min_path_version | int      | Reserved for future use                                                                                                                                                                                                                                                       |
| marker_serial_no | int      | For internal use                                                                                                                                                                                                                                                              |
| status           | int      | Route status: <ul style="list-style-type: none"><li>• 0x00 – Valid</li><li>• 0x01 – Route upgrade/recovery in progress, or route upgrade/recovery needed.</li><li>• 0x02 – Route upgrade/recovery complete. This is a temporary status used by Replication Manager.</li></ul> |
| proposed_version | int      | New route value in transition                                                                                                                                                                                                                                                 |

**Indexes** Unique clustered index on (dest\_rsid, source\_rsid)

## rs\_rules

**Description** Stores subscription rules. This table has one row for each term in a subscription clause.

| Column        | Datatype    | Description                                                                                                                                                                                                                |
|---------------|-------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| prsid         | int         | Primary Replication Server for this object                                                                                                                                                                                 |
| subid         | rs_id       | ID of the subscription this rule applies to. Or, for a subscription to an article, the ID of the where clause to which this rule applies.                                                                                  |
| objid         | rs_id       | ID for the table or function replication definition for this subscription                                                                                                                                                  |
| dbid          | int         | ID for the database where the subscribed data is stored                                                                                                                                                                    |
| subtype       | int         | Subscription type: <ul style="list-style-type: none"> <li>• 0x01 – Range subscription</li> <li>• 0x02 – Equality subscription</li> <li>• 0x80 – Article subscription</li> </ul>                                            |
| primary_sre   | int         | If set, the subscription should be included in the subscription resolution engine at the primary Replication Server                                                                                                        |
| replicate_sre | int         | If set, the subscription should be included in the subscription resolution engine at the replicate Replication Server                                                                                                      |
| colnum        | smallint    | The value of the base column number                                                                                                                                                                                        |
| valuetype     | tinyint     | Datatype of operand, for example, SYBCHAR                                                                                                                                                                                  |
| low_flag      | tinyint     | Bitmap for the type of the low value: <ul style="list-style-type: none"> <li>• 0x01 – exclusive</li> <li>• 0x02 – inclusive</li> <li>• 0x04 – infinity</li> <li>• 0x08 – equality</li> <li>• 0x20 – rs_address</li> </ul>  |
| high_flag     | tinyint     | Bitmap for the type of the high value: <ul style="list-style-type: none"> <li>• 0x01 – exclusive</li> <li>• 0x02 – inclusive</li> <li>• 0x04 – infinity</li> <li>• 0x08 – equality</li> <li>• 0x20 – rs_address</li> </ul> |
| low_len       | int         | Length of low value                                                                                                                                                                                                        |
| high_len      | int         | Length of high value                                                                                                                                                                                                       |
| low_value     | binary(255) | Binary representation of low value                                                                                                                                                                                         |
| high_value    | binary(255) | Binary representation of high value                                                                                                                                                                                        |
| dtid          | rs_id       | ID of the declared datatype of the columns as defined in the replication definition.                                                                                                                                       |

Indexes

- Unique index on (subid, colnum, primary\_sre, replicate\_sre, subtype)
- Unique index on (subid, colnum)
- Clustered index on (objid, subtype, dbid)

## rs\_schedule

**Description** Stores information about the schedules you create in Replication Server.

| Column     | Datatype     | Description                                                                                                                                         |
|------------|--------------|-----------------------------------------------------------------------------------------------------------------------------------------------------|
| sched_name | varchar(30)  | Name of the schedule.                                                                                                                               |
| sched_time | varchar(255) | The day and time string in the form of restricted UNIX cron style that indicates the time that Replication Server performs the specified operation. |
| status     | int          | Switch on or switch off the schedule. Valid values are: <ul style="list-style-type: none"> <li>• 0 – off</li> <li>• 1 – on</li> </ul>               |
| type       | int          | Type of command to run in the schedule. Values are: <ul style="list-style-type: none"> <li>• 0 – shell command</li> </ul>                           |
| ownerid    | rs_id        | ID of the user who created the schedule.                                                                                                            |

**Indexes** Unique clustered index on (sched\_name)

# rs\_schedule.txt

Description Stores the command portion of the schedules you create in Replication Server. Each schedule entry is represented by one or more rows in this table. Multiple rows are needed when the command exceeds the maximum command field length of 255 bytes.

| Column        | Datatype     | Description                                  |
|---------------|--------------|----------------------------------------------|
| schedule_name | varchar(30)  | Name of the schedule.                        |
| sequence      | int          | Sequence number of the row for the schedule. |
| textval       | varchar(255) | Full path of shell command.                  |

- Indexes
- Unique clustered index on (schedule\_name, sequence)
  - Partial index on (schedule\_name)

## rs\_segments

**Description** Replication Server uses raw disk space to store message data. This table holds information about the allocation of each segment.

| Column           | Datatype | Description                                                                                                                                                                                                                                                                   |
|------------------|----------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| partition_id     | int      | Unique ID for the partition                                                                                                                                                                                                                                                   |
| q_number         | int      | Queue that this partition belongs to                                                                                                                                                                                                                                          |
| q_type           | int      | Type of this queue                                                                                                                                                                                                                                                            |
| partition_offset | int      | Offset of segment within partition                                                                                                                                                                                                                                            |
| logical_seg      | int      | Offset of segment within queue                                                                                                                                                                                                                                                |
| used_flag        | int      | Current status of segment: <ul style="list-style-type: none"> <li>• 0 – inactive</li> <li>• 1 – active</li> <li>• <math>n</math> – save interval: <math>n</math> indicates the actual time (measured in seconds from a base date) when this segment can be deleted</li> </ul> |
| version          | int      | Current version of the segment. The version number increases after each use.                                                                                                                                                                                                  |
| flags            | int      | Set to 1 on the last segment of the DSI queue after switch active                                                                                                                                                                                                             |

**Indexes** Unique clustered index on (partition\_id, partition\_offset)

## rs\_sites

Description Stores the names of Replication Servers known at a site.

| Column | Datatype    | Description                                 |
|--------|-------------|---------------------------------------------|
| name   | varchar(30) | Replication Server name                     |
| id     | int         | Site ID assigned to this Replication Server |
| status | tinyint     | Not used                                    |

Indexes

- Unique index on (name)
- Unique clustered index on (id)

## rs\_statcounters

Description Stores descriptive information about each counter. These values do not change.

| Column         | Datatype     | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
|----------------|--------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| counter_id     | int          | Unique counter identification number                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| counter_name   | varchar(60)  | Descriptive counter name                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| module_name    | varchar(30)  | Name of module to which the counter belongs                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
| display_name   | varchar(30)  | Counter name used for RCL commands                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
| counter_status | int          | Counter status. Bit-mask values are: <ul style="list-style-type: none"><li>• 0x001 – internal use, does not display</li><li>• 0x002 – internal use, does not display</li><li>• 0x004 – sysmon (counter flushed as output of admin statistics, sysmon)</li><li>• 0x008 – must sample (counter sampled at all times)</li><li>• 0x010 – no reset (counter is never reset)</li><li>• 0x020 – duration (counter records amount of time to complete an action, usually in .01 seconds)</li><li>• 0x040 – internal use, does not display</li><li>• 0x080 – keep old (previous value of counter retained, usually to aid calculation during next observation period)</li><li>• 0x100 – internal use, does not display</li><li>• 0x200 – observer</li><li>• 0x400 – monitor</li><li>• 0x800 – internal use, does not display</li></ul> |
| description    | varchar(255) | Description of counter                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |

Indexes Unique, clustered key rs\_key\_statcounters on (counter\_id)

## rs\_statdetail

Description Stores counter metrics that have been flushed to the RSSD.

| Column        | Datatype     | Description                                                                                                                                                                                                                                                                  |
|---------------|--------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| run_id        | rs_id        | Number assigned to the run or observation period                                                                                                                                                                                                                             |
| instance_id   | int          | An ID that identifies a module instance.<br><br>Counters are grouped by modules. A module may have one instance or multiple instances. Defined module IDs are used when available. For example, the instance_id for a DSI module is the database ID associated with the DSI. |
| instance_val  | int          | An ID that identifies a module instance when instance_id can not identify it uniquely.                                                                                                                                                                                       |
| counter_id    | int          | Unique counter identification number                                                                                                                                                                                                                                         |
| counter_obs   | int          | Number of observations                                                                                                                                                                                                                                                       |
| counter_total | int          | Total of observed values for the run or observation period                                                                                                                                                                                                                   |
| counter_last  | int          | Last observed value for the run or observation period                                                                                                                                                                                                                        |
| counter_max   | int          | Maximum observed value for the run or observation period                                                                                                                                                                                                                     |
| label         | varchar(255) | Descriptive information about the module instance associated with the counter, such as the data server and database name.                                                                                                                                                    |

Indexes Unique, nonclustered key rs\_key\_statdetail on (run\_id, instance\_id, instance\_val, counter\_id)

## rs\_statrun

Description Stores descriptive information about each observation period or run.

| Column       | Datatype    | Description                                       |
|--------------|-------------|---------------------------------------------------|
| run_id       | rs_id       | Number assigned to an observation period or run   |
| run_date     | datetime    | Date and time of observation period or run        |
| run_interval | int         | Duration of observation period or run in seconds  |
| run_user     | varchar(30) | Name of user who flushed the counters to the RSSD |
| run_status   | int         | Status of run                                     |

Indexes

Unique, nonclustered key rs\_key\_statdetail on (run\_id)

## rs\_subscriptions

Description Stores information about subscriptions, triggers, and fragments.

| Column      | Datatype    | Description                                                                                                                                                                                                                                                                                                                        |
|-------------|-------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| subname     | varchar(30) | Name of the subscription, trigger, or fragment.                                                                                                                                                                                                                                                                                    |
| subid       | rs_id       | ID for this subscription or fragment.                                                                                                                                                                                                                                                                                              |
| type        | int         | Object type: <ul style="list-style-type: none"> <li>• 0x00 – Subscription</li> <li>• 0x01 – Range subscription</li> <li>• 0x02 – Equality subscription</li> <li>• 0x04 – Entire table</li> <li>• 0x08 – Subscription for publication</li> <li>• 0x40 – Database subscription</li> <li>• 0x80 – Subscription for article</li> </ul> |
| objid       | rs_id       | ID for the table replication definition, function replication definition, article, or publication for this subscription. Or, ID for fragment, or event for this trigger.                                                                                                                                                           |
| dbid        | int         | ID of the database this object belongs to.                                                                                                                                                                                                                                                                                         |
| pdbid       | int         | For system table replication and publication or article subscriptions, the value of pdbid is the ID of the primary database for the replication definition. Otherwise, value is 0.                                                                                                                                                 |
| requestdate | datetime    | Date and time the last DDL request (create, drop, alter) was entered.                                                                                                                                                                                                                                                              |
| pownerid    | rs_id       | User ID at the primary Replication Server.                                                                                                                                                                                                                                                                                         |
| rownerid    | rs_id       | User ID at the replicate Replication Server.                                                                                                                                                                                                                                                                                       |

| Column          | Datatype | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
|-----------------|----------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| status          | int      | <ul style="list-style-type: none"><li>• Byte 1 holds the replicate database materialization status:<ul style="list-style-type: none"><li>• 0x01 – Subscription is new</li><li>• 0x02 – Bulk subscription is activating or atomic/non-atomic subscription has completed building materialization queue</li><li>• 0x04 – Bulk/non-atomic subscription is active</li><li>• 0x08 – Bulk subscription is validating or non-atomic has materialized</li><li>• 0x10 – Subscription is valid</li><li>• 0x40 – Subscription is valid at the standby</li><li>• 0x40 – Subscription removed at standby</li><li>• 0x80000000 – Database subscription is using dump marker to coordinate materialization</li></ul></li><li>• Byte 2 holds the primary database dematerialization status:<ul style="list-style-type: none"><li>• 0x100 – New</li><li>• 0x0200 – Activating</li><li>• 0x0400 – Active</li><li>• 0x0800 – Valid</li></ul></li><li>• Byte 3 holds the replicate database dematerialization status:<ul style="list-style-type: none"><li>• 0x00010000 – Dematerializing at replicate</li><li>• 0x00020000 – Removing at replicate</li><li>• 0x00100000 – Dematerializing at primary</li></ul></li><li>• Byte 4 holds suspect or rematerialization status for a publication subscription:<ul style="list-style-type: none"><li>• 0x02000000 – Suspect because of switch active</li><li>• 0x04000000 – Suspect on drop at standby</li><li>• 0x10000000 – The article subscriptions within this publication subscription are materializing one at a time</li><li>• 0x20000000 – In the process of creating new article subscriptions</li><li>• 0x40000000 – include truncate table</li></ul></li></ul> |
| recovering      | int      | <p>Subscription recovery status:</p> <ul style="list-style-type: none"><li>• 0x0 – Subscription is OK</li><li>• 0x1 – Recovering</li><li>• 0x2 – Pending</li></ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| error_flag      | int      | If set, subscription is unrecoverable                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| materializing   | int      | If set, subscription is materializing                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| dematerializing | int      | If set, subscription is dematerializing                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |

| Column              | Datatype | Description                                                                                                                                                                                                                                                                                                                                                                                                                   |
|---------------------|----------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| primary_sre         | int      | If set, the subscription should be included in the subscription resolution engine at the primary Replication Server                                                                                                                                                                                                                                                                                                           |
| replicate_sre       | int      | If set, the subscription should be included in the subscription resolution engine at the replicate Replication Server                                                                                                                                                                                                                                                                                                         |
| materialization_try | int      | Number of times this atomic materialization has been tried                                                                                                                                                                                                                                                                                                                                                                    |
| method              | int      | Method for materializing the subscription: <ul style="list-style-type: none"> <li>• 0x00 – Default method</li> <li>• 0x01 – Atomic</li> <li>• 0x02 – Bulk</li> <li>• 0x04 – Suspend</li> <li>• 0x08 – Incremental</li> <li>• 0x10 – Non-atomic</li> <li>• 0x80 – Bulk materialization with suspended standby DSI</li> </ul> <p><b>Note</b> For function replication definitions, this column is always set to 0x02 (bulk)</p> |
| generation          | binary   | Generation number for the origin queue ID of the materialization queue                                                                                                                                                                                                                                                                                                                                                        |
| parentid            | rs_id    | ID for the subscription for a publication if the current subscription is for an article.                                                                                                                                                                                                                                                                                                                                      |
| security            | int      | Security settings: <ul style="list-style-type: none"> <li>• 0x001 – unified_login is “required”</li> <li>• 0x002 – mutual_auth is “required”</li> <li>• 0x004 – msg_confidentiality is “required”</li> <li>• 0x08 – msg_integrity is “required”</li> <li>• 0x10 – msg_origin_check is “required”</li> <li>• 0x20 – msg_reply_detection is “required”</li> <li>• 0x40 – msg_sequence_check is “required”</li> </ul> Default: 0 |
| mechanism           | char(30) | Name of security mechanism<br>Default: NULL                                                                                                                                                                                                                                                                                                                                                                                   |

- Indexes
- Unique clustered index on (subid)
  - Unique index on (objid, dbid, subname)
  - Unique index on (subid, recovering, error\_flag, materializing, dematerializing, primary\_sre, replicate\_sre)
  - Unique index on (subid, status)

- Unique index on (objid)
- Unique index on (pdbid)

## rs\_systext

**Description** Stores the text of repeating groups for various other tables such as rs\_funcstrings.

| Column   | Datatype     | Description                                                                                                                                                                                                                                                                               |
|----------|--------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| prsid    | int          | Replication Server where the object is defined                                                                                                                                                                                                                                            |
| parentid | rs_id        | ID of the object this text is for                                                                                                                                                                                                                                                         |
| texttype | char(1)      | Type of object this row is for: <ul style="list-style-type: none"><li>• S – input template for function string</li><li>• O – output template for function string</li><li>• C – command from a logged transaction in the exceptions log</li><li>• P – Replication Server profile</li></ul> |
| sequence | int          | Sequence of the text                                                                                                                                                                                                                                                                      |
| textval  | varchar(255) | The text                                                                                                                                                                                                                                                                                  |

**Indexes** Unique clustered index on (parentid, texttype, sequence)

# rs\_tbconfig

Description

Replication Server uses the information in this table to support referential constraints.

rs\_tbconfig is not a replicated system table.

| Column     | Datatype     | Description                                                                                                                                                                                   |
|------------|--------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| optionname | varchar(30)  | Name of the parameter, for example: memory_max, cm_max_connections<br>To view a list of these parameters with their descriptions, execute a select * statement against the rs_tbconfig table. |
| dbid       | int          | Unique identifier for the database.                                                                                                                                                           |
| objname    | varchar(255) | Object name defined in the replicate database.                                                                                                                                                |
| objowner   | varchar(30)  | Name of the replicate object owner, as specified in replication definition.<br>Blank if the owner is not specified.                                                                           |
| charvalue  | varchar(255) | Character value for parameter.                                                                                                                                                                |
| status     | tinyint      | This column is not used.                                                                                                                                                                      |
| comments   | varchar(255) | Comment about the parameter.                                                                                                                                                                  |

Indexes

Unique clustered index on (optionname, dbid, objname, objowner).

## rs\_threads

**Description** Replication Server uses the information in this table to detect deadlocks and to perform transaction serialization between parallel DSI threads. An entry is updated in this table each time a transaction is started and more than one DSI thread is defined for a connection.

The rs\_threads table is stored in each user database, not in the RSSD.

| Column | Datatype  | Description                                                                                                                       |
|--------|-----------|-----------------------------------------------------------------------------------------------------------------------------------|
| id     | int       | The entry ID number. There are two entries for each parallel DSI thread.                                                          |
| seq    | int       | The sequence number of the last update made to this entry. The sequence number starts at 0 each time the connection is restarted. |
| pad1   | char(255) | Filler to pad the row so that only one row fits on a database page.                                                               |
| pad2   | char(255) | Filler to pad the row so that only one row fits on a database page.                                                               |
| pad3   | char(255) | Filler to pad the row so that only one row fits on a database page.                                                               |
| pad4   | char(255) | Filler to pad the row so that only one row fits on a database page.                                                               |

**Indexes** Unique clustered index on (id)

## rs\_ticket\_history

Description Stores rs\_ticket information.

| Column  | Datatype      | Description                                                                                |
|---------|---------------|--------------------------------------------------------------------------------------------|
| cnt     | int identity  | Ticket unique sequence.                                                                    |
| h1      | varchar(10)   | Ticket header. Set as “-” if a header is not present.                                      |
| h2      | varchar(10)   | Ticket header. Set as “-” if a header is not present.                                      |
| h3      | varchar(10)   | Ticket header. Set as “-” if a header is not present.                                      |
| h4      | varchar(50)   | Ticket header. Set as “-” if a header is not present.                                      |
| pdb     | varchar(30)   | Primary database name.                                                                     |
| prs     | varchar(30)   | Primary Replication Server name. Set to “-” if no primary Replication Server is specified. |
| rrs     | varchar(30)   | Replicate Replication Server name.                                                         |
| rdb     | varchar(30)   | Replicate database name.                                                                   |
| pdb_t   | datetime      | The time the rs_ticket stored procedure was executed at the primary database.              |
| exec_t  | datetime      | The time the ticket passed through the Replication Server executor thread.                 |
| dist_t  | datetime      | The time the ticket passed through the DIST thread.                                        |
| rsi_t   | datetime      | The time the ticket passed through the RSI thread.                                         |
| dsi_t   | datetime      | The time the ticket passed through the DSI thread.                                         |
| rdb_t   | datetime      | The time the ticket arrived at the replicate database.                                     |
| exec_b  | int           | Total bytes received by the EXEC thread.                                                   |
| rsi_b   | int           | Total bytes received by the RSI thread.                                                    |
| dsi_tnx | int           | Total number of transactions observed by DSI.                                              |
| dsi_cmd | int           | Total number of commands observed by DSI.                                                  |
| ticket  | varchar(1024) | Raw ticket.                                                                                |

Indexes Unique clustered index rs\_ticket\_idx on rs\_ticket\_history(cnt)

## rs\_translation

Description Stores information about class-level datatype translations.

| Column        | Datatype | Description                                                                                                                                                                                                     |
|---------------|----------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| prsid         | int      | ID of the primary Replication Server                                                                                                                                                                            |
| classid       | rs_id    | Function-string class ID of connection                                                                                                                                                                          |
| type          | char(1)  | Type of translation. Can be: <ul style="list-style-type: none"> <li>• D – class-level</li> </ul>                                                                                                                |
| source_dtid   | rs_id    | ID of source datatype                                                                                                                                                                                           |
| target_dtid   | rs_id    | ID of target datatype                                                                                                                                                                                           |
| target_length | int      | Maximum length for a value of the target datatype                                                                                                                                                               |
| target_status | int      | See status column in rs_columns table                                                                                                                                                                           |
| rowtype       | tinyint  | Indicates whether a row is local to the Replication Server or distributed to all Replication Servers in the domain. Can be: <ul style="list-style-type: none"> <li>• 0 – local</li> <li>• 1 – global</li> </ul> |

Indexes

- Unique, compound index on (classid, source\_dtid)
- Non-unique index on (classid, prsid)

## rs\_users

Description Stores a row for each user with access to the Replication Server.

| Column           | Datatype    | Description                                                                                                                                                                                                  |
|------------------|-------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| username         | varchar(30) | Name of the user                                                                                                                                                                                             |
| uid              | rs_id       | ID of the user                                                                                                                                                                                               |
| password         | varchar(30) | Password                                                                                                                                                                                                     |
| permissions      | smallint    | Mask indicating roles a user can have: <ul style="list-style-type: none"><li>• 0x0001 – sa</li><li>• 0x0002 – connect source</li><li>• 0x0004 – create object</li><li>• 0x0008 – primary subscribe</li></ul> |
| use_enc_password | int         | <ul style="list-style-type: none"><li>• 0 – use normal passwords</li><li>• 1 – use encrypted passwords</li></ul>                                                                                             |
| enc_password     | varchar(66) | Encrypted password                                                                                                                                                                                           |

Indexes

- Unique index on (username)
- Unique index on (uid)

## rs\_version

**Description** Stores version number information for the replication system. At local Replication Servers, only the local version number and the system-wide version number are stored. At the ID Server, version information is stored for all Replication Servers in the replication system.

| Column  | Datatype | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
|---------|----------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| siteid  | int      | ID number of the Replication Server: <ul style="list-style-type: none"> <li>• 0 – site ID for the system-wide version number</li> <li>• 1 – site ID for the site version number</li> <li>• <math>n</math> – site ID of individual Replication Servers</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
| version | int      | Version number: <ul style="list-style-type: none"> <li>• 1000 – version 10.0 (assigned to any Replication Server whose version is unknown)</li> <li>• 1003 – version 10.0.3</li> <li>• 1011 – version 10.1.1</li> <li>• 1100 – version 11.0</li> <li>• 1101 – version 11.0.1</li> <li>• 1102 – version 11.0.2</li> <li>• 1103 – version 11.0.3</li> <li>• 1150 – version 11.5</li> <li>• 1200 – version 12.0</li> <li>• 1210 – version 12.1</li> <li>• 1250 – version 12.5</li> <li>• 1260 – version 12.6</li> <li>• 1500 – versions 15.0, 15.0.1</li> <li>• 1510 – version 15.1</li> <li>• 1520 – version 15.2</li> <li>• 1550 – version 15.5</li> </ul> See the <i>Release Bulletin for Replication Server</i> for any additional supported version numbers. |

For more information about system-wide version numbers, see `admin security_property` on page 70.

**Indexes** Unique clustered index on (siteid)

## rs\_whereclauses

Description Stores information about where clauses used in articles known to this Replication Server.

| Column    | Datatype | Description                                                                              |
|-----------|----------|------------------------------------------------------------------------------------------|
| articleid | rs_id    | ID of the article included in this where clause                                          |
| wclauseid | rs_id    | ID of this where clause                                                                  |
| type      | int      | <ul style="list-style-type: none"><li>• 0x01 – Range</li><li>• 0x02 – Equality</li></ul> |

Indexes Unique clustered index on (wclauseid)

# Replication Monitoring Services API

This chapter contains the reference pages for the Replication Monitor Service (RMS) API. Table 9-1 provides a brief description of the API commands.

**Table 9-1: RMS API commands**

| Command             | Description                                                                                                                                                                                                                                               |
|---------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| add event trigger   | Sets up a trigger, such as a process or a script, that is executed by the RMS when a specific event occurs.                                                                                                                                               |
| add server          | Adds a server to be monitored by the RMS.                                                                                                                                                                                                                 |
| configure component | Returns configuration parameters for a component; or sets the value of the specified configuration parameter. Components are monitored objects within a server, including Replication Server and Adaptive Server Enterprise.                              |
| configure RMS       | Returns the configuration parameter information for the RMS, or sets the value of a specified RMS configuration parameter.                                                                                                                                |
| configure server    | Returns configuration parameter information for a Replication Server or Replication Agent, or sets the value of a specified configuration parameter. Also retrieves and sets RMS-specific parameters.                                                     |
| connect to server   | Provides a pass-through mode that enables you to send commands to a server that is monitored by the RMS. Result sets generated by commands are passed back to the client.                                                                                 |
| create group        | Enables you to define a set of servers and issue commands to all members of the group.                                                                                                                                                                    |
| delete group        | Deletes a logical group that was added using the create group command.                                                                                                                                                                                    |
| disconnect server   | Disconnects from a server where a pass-through connection was established.                                                                                                                                                                                |
| drop event trigger  | Removes a trigger that the RMS is monitoring, using the add <i>event triggers</i> command.                                                                                                                                                                |
| drop server         | Drops a server that is being monitored by the RMS.                                                                                                                                                                                                        |
| filter connection   | Returns current filter settings or sets the filter setting for a connection. This command can filter either the Replication Agent thread or the DSI thread status.                                                                                        |
| get component       | Returns a list of Replication Server or Adaptive Server Enterprise components that are monitored by the RMS. Components are monitored objects within a server.                                                                                            |
| get group           | Returns a result set that contains either a list of the groups and a roll-up status for each group, or status of each server and a roll-up status for the specified group. Roll-up status shows the lowest status reported for a component in the groups. |

| Command                   | Description                                                                                                                                                                                                                                           |
|---------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| get heartbeat             | Retrieves a list of the heartbeat processes that have been defined in the RMS.                                                                                                                                                                        |
| get heartbeat tickets     | Retrieves a set of tickets from the <code>rms_ticket_history</code> table, for the heartbeat process and date and time range specified.                                                                                                               |
| get network spec          | Retrieves the connection information for all servers known to the RMS. This list is retrieved from the RMS's <i>interfaces</i> file or LDAP server. The list consists of the server name, host computer name, and the port number used by the server. |
| get rmiaddress            | Retrieves the address of the Remote Method Invocation (RMI) service.                                                                                                                                                                                  |
| get servers               | Returns a list of servers that are monitored by the RMS, and the status of the RMS environment. The RMS status is a roll-up of the monitored servers.                                                                                                 |
| get status descriptions   | Retrieves the list of status descriptions for a server or component.                                                                                                                                                                                  |
| get threads               | Displays information about threads running in the Replication Server.                                                                                                                                                                                 |
| get triggers              | Displays information about the triggers that are monitored by the RMS.                                                                                                                                                                                |
| get version               | Retrieves the version number of RMS.                                                                                                                                                                                                                  |
| resume component          | Resumes a component in a specified server. The command resumes a DSI thread, Replication Agent thread, RepAgent thread, queue, or a route in a Replication Server.                                                                                    |
| resume Replication Agent  | Resumes replication in a Replication Agent.                                                                                                                                                                                                           |
| shutdown server           | Issues a shutdown command to a server or to the RMS.                                                                                                                                                                                                  |
| suspend component         | Suspends a component in a specified server. The command suspends a DSI thread, Replication Agent thread, RepAgent thread, or route in a Replication Server.                                                                                           |
| start heartbeat           | Sets up and starts a heartbeat process from a specified primary connection to a specified replicate connection.                                                                                                                                       |
| stop heartbeat            | Stops the heartbeat process between the primary and replicate databases. Optionally, truncates the <code>rms_ticket_history</code> table.                                                                                                             |
| suspend Replication Agent | Suspends replication in a Replication Agent.                                                                                                                                                                                                          |
| trace                     | Displays trace information in the RMS log file.                                                                                                                                                                                                       |

To use the RMS API commands, these permissions must be set for each server that is monitored by RMS:

| Server                   | Permission                                                                                                                                                     |
|--------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Adaptive Server          | The user must have “sa” or “dbo” permissions or Replication role for any primary database. The user must have “sa” or “dbo” permissions for any RSSD database. |
| Replication Server       | The user must have “sa” permissions.                                                                                                                           |
| Replication Agent        | The server does not have different user permissions.                                                                                                           |
| Mirror Replication Agent | The server does not have different user permissions.                                                                                                           |
| DirectConnect™           | The user must have permission to successfully log into the back end server. The RMS does not attempt to read or write to the back end database.                |
| SA                       | The user must have permission to log into the SA database. The RMS does not attempt to read or write to the database.                                          |

| Server      | Permission                                                                                                          |
|-------------|---------------------------------------------------------------------------------------------------------------------|
| IQ          | The user must have permission to log into the IQ server. The RMS does not attempt to read or write to the database. |
| Remote RMS  | The server does not have different user permissions.                                                                |
| Open Server | The user must have permission to establish a connection to the Open Server.                                         |

## add event trigger

**Description** Adds a trigger that is executed by the RMS when a specific event occurs in the replication domain. A trigger identifies a process or script that is executed by the RMS.

**Syntax**

```
add {status | latency | size} trigger
 [{connection | logical connection | route | queue | rep agent |
 partition} [component_name]]
 [with primary primary_connection]
 for server_name
 {status changes to state |
 size {exceeds | falls below} size_threshold |
 latency {exceeds | falls below} latency_threshold}
 [wait wait_interval]
 [continuous continuous_flag]
 execute command
```

**Parameters**

*status*, *latency*, *size*  
Type of trigger.

*connection*, *logical connection*, *route*, *queue*, *rep agent*, *partition*  
Specifies the type of component to be monitored. Components are monitored objects within a server. Replication Server components are connections, logical connections, routes, queues, and partitions; Adaptive Server Enterprise components are RepAgent threads.

*component\_name*  
Specifies the name of the component to be monitored.

with primary *primary\_connection*  
Identifies the primary connection for a connection latency trigger. The trigger executes the script if the latency threshold between the primary connection and the replicate connection is not satisfied.

for *server\_name*  
Specifies the name of the server to be monitored. If the command is to add a trigger for a component, then the server is the owner of the component.

size exceeds, falls below *size\_threshold*

Indicates whether the trigger should execute when the size exceeds the threshold or when it falls below the threshold.

latency exceeds, falls below *latency\_threshold*

Indicates whether the trigger should execute when the latency exceeds the threshold or when it falls below the threshold.

status changes to *state*

Specifies the state of the server or component to monitor. If *state* changes to the specified value, the trigger executes. The state value is dependent on the object type. See Appendix C, “RMS Server and Component States” for information about the state codes.

wait *wait\_interval*

Specifies the number of seconds to wait before triggering the event. This allows the object time to recover. If you do not include the wait option, the event triggers immediately.

continuous *continuous\_flag*

A Boolean flag that, if set to true, causes the RMS to execute the trigger’s script at every subsequent monitoring interval until the state changes. If you do not set this flag, the RMS executes the trigger script only once.

execute *command*

Specifies the command to be executed when the event is triggered. The command is operating-system-specific.

## Examples

**Example 1** Adds a trigger that executes the script *email.sh* when the status of the server named INVENTORY\_RS is changed to “DOWN”:

```
add status trigger for INVENTORY_RS
status changes to DOWN
execute /sybase/RMS/scripts/email.sh
```

**Example 2** Adds a trigger that executes the script *email.sh* after 120 seconds. Since the status of the connection “inventory\_pds.pdb1” of server INVENTORY\_RS is changed to “SUSPENDED”, it will execute script at every subsequent monitoring interval until the state changes:

```
add status trigger connection inventory_pds.pdb1 for
INVENTORY_RS
status changes to Suspended
wait 120
continuous true
execute /sybase/RMS/scripts/email.sh
```

**Example 3** Adds a trigger to the Replication Server INVENTORY\_RS partition “p1” that executes the script *email.sh* when the partition usage exceeds 80 percent. The script is executed at every subsequent monitoring interval as long as the partition usage exceeds 80 percent:

```
add size trigger partition p1 for INVENTORY_RS
size exceeds 80
continuous true
execute /sybase/RMS/scripts/email.sh
```

**Example 4** Adds a trigger to the Replication Server INVENTORY\_RS that executes the script *email.sh* when the sum of all partition usage exceeds 75 percent:

```
add size trigger partition for INVENTORY_RS
size exceeds 75
execute /sybase/RMS/scripts/email.sh
```

**Example 5** Adds a trigger to the queue “inventory\_pds.vendor(Inbound)” of Replication Server INVENTORY\_RS that executes the script *email.sh* when the queue size falls below 100 megabytes. The script is executed at every subsequent monitoring interval as long as the queue size is less than 100 MB:

```
add size trigger queue inventory_pds.vendor(Inbound)
for INVENTORY_RS
size falls below 100
continuous true
execute /sybase/RMS/scripts/email.sh
```

**Example 6** Adds a trigger to the replicate connection “inventory\_rds.vendor” of replicate Replication Server INVENTORY\_RS that will execute the script *email.sh* when the latency from the primary connection “inventory\_pds.vendor” exceeds 5 minutes (300 seconds):

```
add latency trigger connection inventory_rds.vendor
with primary inventory_pds.vendor
for INVENTORY_RS
latency exceeds 300
execute /sybase/RMS/scripts/email.sh
```

#### Usage

- You can add one status trigger for each server or component status. For example, you can add a trigger for a Replication Server when the status changes to “DOWN” or “SUSPECT”, but you cannot add two triggers to the “DOWN” status.
- You must set *server\_name* to the name of the replicate Replication Server when adding a latency connection trigger. In the following example, INVENTORY\_RS is the replicate Replication Server:

```
add latency trigger connection inventory_rds.vendor
with primary inventory_pds.vendor
for INVENTORY_RS
latency exceeds 300
execute /sybase/RMS/scripts/email.sh
```

- You must set the configuration parameter ltl\_origin\_time\_require to “true” when setting up a latency connection trigger where the primary connection is from a Replication Agent or MRA. To set the parameter, connect to the Replication Agent or MRA and execute:

```
ra_config ltl_origin_time_required, true
```

- add event trigger returns the following result set:

**Table 9-2: Column descriptions for add event trigger**

| Column | Description                 |
|--------|-----------------------------|
| Action | The name of the action      |
| Result | The result of the execution |

See also

drop event trigger, get triggers

# add server

|             |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
|-------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Description | Adds a server to be monitored by the RMS.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| Syntax      | <pre>add {ASA   ASE   DirectConnect   IQ   Replication Agent   MRA   Replication Server   RMS   Open Server   dbltm} server_name     set username [to] user     [set password [to] passwd]     [set charset [to] charset]     [set language [to] lang]     [set rssid_username [to] rssid_user]     [set rssid_password [to] rssid_passwd]     [set rssid_charset [to] rssid_charset]     [set rssid_language [to] rssid_lang]     [set monitoring [to] {'true'   'false'}]     [set interval [to] interval]     [set connection_ds [to] ds]     [set connection_db [to] db]</pre> |
| Parameters  | <p>ASA, ASE, DirectConnect, IQ, Replication Agent, MRA, Replication Server, RMS, Open Server, dbltm</p> <p>Specifies the type of server to add to the RMS. You can add a remote RMS to a controlling RMS.</p>                                                                                                                                                                                                                                                                                                                                                                      |

*server\_name*

Specifies the name of the server as listed in the RMS *interfaces* file or LDAP server.

*user*

Specifies the user name that the RMS uses when establishing a connection to the server. The user name must have the required permissions to allow the RMS to monitor the server.

*passwd*

Specifies the corresponding password that the RMS uses when establishing a connection.

---

**Note** Do not include the set password clause if the password is NULL.

---

*charset*

Specifies the character set that the RMS uses when establishing a connection to the server. If you do not specify *charset*, jConnect uses the server's default character set.

*lang*

Specifies the language that the RMS uses when establishing a connection to the server. If you do not specify the language, jConnect uses the server's default language.

*rssd\_user*

Specifies the user name that the RMS uses when establishing a connection to the server that contains the RSSD. The user name must have the required permissions to allow the RMS to monitor the server. This parameter is required for a Replication Server.

*rssd\_passwd*

Specifies the corresponding password that the RMS uses when establishing a connection to the server that contains the RSSD.

*rssd\_charset*

Specifies the character set that the RMS uses when establishing a connection to the server that contains the RSSD. If you do not provide the *charset*, jConnect uses the server's default character set.

*rssd\_lang*

Specifies the language that the RMS uses when establishing a connection to the server that contains the RSSD. If you do not provide the language, jConnect uses the server's default language.

*monitoring*

Specifies whether the RMS is monitoring the state of the server and its components. If this value is false, monitoring for this server is disabled. If this value is true (the default), RMS automatically monitors this server.

*interval*

Specifies the number of seconds between monitoring cycles. If the monitoring property is set to true, then RMS performs periodic monitoring based on the value of *interval*. For example, if the value is set to 120, the RMS checks the health of the server every 120 seconds. The range of values is 30 seconds to 1 hour and the default value for the interval is the value of *ping\_interval* in RMS config.

*ds*

Specifies the name of the primary data server. The dbltm sends *ds.db* to the Replication Server when replicating transactions. The *ds* must match the server name used in the Replication Server connection. This parameter is optional and is valid only for a dbltm server.

*db*

Specifies the name of the primary database. The dbltm server sends *ds.db* to the Replication Server when replicating transactions. The *db* must match the database name used in the Replication Server connection. This parameter is optional and is valid only for a dbltm server.

## Examples

**Example 1** Adds a Replication Server named INVENTORY\_RS to the RMS. Uses the user name “sa” without a password, character set, or language when establishing a connection. Uses the user name “sa” and the password “sa\_pwd” when establishing a connection to the RSSD:

```
add replication server INVENTORY_RS
 set username to sa
 set rssid_username to sa
 set rssid_password to sa_pwd
```

**Example 2** Adds a server named INVENTORY\_PDS to the RMS. Sets the user name, password, language, monitoring and interval:

```
add ASE INVENTORY_PDS
 set username to sa
 set password to sa_ps
 set language to Japanese
 set monitoring to true
 set interval to 120
```

## Usage

- Use the RSSD options when adding a Replication Server to the RMS. You need not add the server that contains the RSSD to the RMS.

- The server name must be in the *interfaces* file or LDAP server that is used by the RMS.
- When you issue *add server*, the RMS attempts to connect to the specified server and automatically determines its type and version. If the type or version is invalid or cannot be determined, or the server is already being monitored, the RMS returns an error message.
- If the new server is a Replication Server, supply the user name for the RSSD.
- The *add server* command returns the following result set:

**Table 9-3: Column descriptions for *add server***

| Column | Description                 |
|--------|-----------------------------|
| Action | The name of the action      |
| Result | The result of the execution |

See also

configure server, connect to server, disconnect server, drop server, get servers, shutdown server

## configure component

|             |                                                                                                                                                                                                                                                                                                                                                                                                                                            |
|-------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Description | Returns configuration parameter information for a component in either a Replication Server or an Adaptive Server; or sets the value of a specified configuration parameter. Components are monitored objects within a server. Replication Server components are connections, logical connections, and routes; Adaptive Server Enterprise components are RepAgent threads.                                                                  |
| Syntax      | <pre>configure {connections   logical connections   routes   repagents} component_name [for] {server_name   group_name} [param[= value]]</pre>                                                                                                                                                                                                                                                                                             |
| Parameters  | <p>connections, logical connections, routes, repagents</p> <p>Specifies the type of component to configure. Replication Server components are connections, logical connections, routes; Adaptive Server Enterprise components are RepAgent threads.</p> <p><i>component_name</i></p> <p>Specifies the name of the component to configure.</p> <p><i>server_name</i></p> <p>Specifies the server that contains the requested component.</p> |

*group\_name*

Specifies the name of a group. You can modify the *group\_name* parameter for each different component in the group.

*param*

Specifies the name of a component's configuration parameter.

*value*

The value to be assigned to the configuration parameter specified in the *param* option.

## Examples

**Example 1** Returns a list of all configuration parameters for the connection “inventory\_pds.vendor” in the server INVENTORY\_RS:

```
configure connection inventory_pds.vendor
for INVENTORY_RS
```

**Example 2** Returns the dsi\_cmd\_batch\_size configuration parameter information for the connection “inventory\_pds.vendor” in the server INVENTORY\_RS:

```
configure connection inventory_pds.vendor
for INVENTORY_RS dsi_cmd_batch_size
```

**Example 3** Sets the dsi\_cmd\_batch\_size configuration parameter to 15000 for the connection “inventory\_pds.vendor” in the server INVENTORY\_RS:

```
configure connection inventory_pds.vendor
for inventory_rs dsi_cmd_batch_size = 15000
```

## Usage

configure *component* returns the following result set if a *value* parameter is not included:

**Table 9-4: Column descriptions for configure component**

| Column           | Description                                                                                           |
|------------------|-------------------------------------------------------------------------------------------------------|
| Server           | The name of the server that contains the parameters.                                                  |
| Component Name   | The name of the component that contains the parameter.                                                |
| Component Type   | The type of the component (connection, route, or RepAgent).                                           |
| Category         | The name of the category for the parameter. Categories are used to group related parameters together. |
| Parameter Name   | The name of the parameter.                                                                            |
| Current Value    | The current value of the parameter.                                                                   |
| Pending Value    | The pending value becomes the value of the parameter after the component is restarted.                |
| Default Value    | The default value of the parameter.                                                                   |
| Legal Values     | A string that defines the legal values for the parameter. This can be a list, or a numeric range.     |
| Restart Required | A flag indicating whether the server must be restarted for the parameter to take effect.              |

See also

get component, resume component, suspend component

## configure *RMS*

**Description** Returns configuration parameter information for the Replication Monitoring Services, or sets the value of a specified RMS configuration parameter.

**Syntax** `configure [param [= value]]`

**Parameters** *param*

Specifies the name of an RMS configuration parameter.

*value*

The value to be assigned to the configuration parameter specified in the *param* option.

Table 9-5 identifies all parameters for the RMS and their associated values:

Table 9-5: RMS parameters

| Parameter     | Value                                                                                                                                   |
|---------------|-----------------------------------------------------------------------------------------------------------------------------------------|
| Logconfig     | The path to the RMS log config file.                                                                                                    |
| Name          | The name of the RMS server. This name must appear in the Sybase interfaces file.                                                        |
| Password      | The password used to connect to the RMS. The value of this parameter is not displayed by the configure command.                         |
| ping_interval | The number of seconds between the end of one monitoring cycle and the beginning of the next. It ranges from 30 seconds to 3600 seconds. |
| Port          | The IP port used by the RMS. It ranges from 1024 to 65,535.                                                                             |
| SybaseHome    | The Sybase home directory. This directory contains the interfaces file.                                                                 |
| Username      | The user name to connect to the RMS.                                                                                                    |
| Version       | The version string of the RMS. This is a read-only parameter.                                                                           |
| includeLDAP   | A flag that turns LDAP support on or off.                                                                                               |
| ldapTimeout   | A user-configurable timeout value.                                                                                                      |

Examples

**Example 1** Returns the list of RMS configuration parameters and their current value in this format:

```
configure

Parameter Name Parameter Type Current Value

includeldap boolean false
ldaptimeout integer 35
logconfig string ../plugins/
 com.sybase.rms/
 log4j.properties
name string RedtailRMS

Pending Value Default Value Legal Values

NULL false List: true,false
NULL 180 N/A
N/A ../log4j.properties N/A
NULL Rms N/A

Category Restart Required

Rms false
```

```
Rms false
Rms N/A
Rms true
```

#### Description

```

A flag that turns LDAP support on or off.
A user configurable timeout value.
The path to the RMS log config file.
The name of the RMS server.
```

```
...
```

**Example 2** Configures a user name of “sa” for the RMS:

```
configure username=sa
```

#### Usage

The `configure RMS` command returns this result set, if you do not include a value parameter:

**Table 9-6: Default RMS result set**

| Column           | Description                                                                                                                     |
|------------------|---------------------------------------------------------------------------------------------------------------------------------|
| Parameter Name   | The name of the parameter, such as <code>logconfig</code> , <code>name</code> , <code>port</code> , and <code>password</code> . |
| Parameter Type   | The type of parameter, such as <code>boolean</code> , <code>integer</code> , <code>string</code> , and <code>password</code> .  |
| Current Value    | The current value of the parameter.                                                                                             |
| Pending Value    | The value the parameter will be after the server is restarted.                                                                  |
| Default Value    | The default value of the parameter.                                                                                             |
| Legal Values     | A string that defines the legal values for the parameter. This can be a list, or a numeric range.                               |
| Category         | The name of the category for the parameter. You can use categories to group related parameters together.                        |
| Restart Required | A flag indicating whether the server must be restarted for the parameter to take effect.                                        |
| Description      | The parameter description.                                                                                                      |

#### See also

`get version`, `resume Replication Agent`, `suspend Replication Agent`, `trace`

# configure server

**Description** Returns configuration parameter information for a Replication Server or Replication Agent and Mirror Replication Agent (MRA), or sets the value of a specified configuration parameter. Also retrieves and sets RMS-specific parameters.

**Syntax** configure server {*server\_name* | *group\_name*} [*RMS*] [*param* [= *value*]]

**Parameters**

*server\_name*  
Specifies the server to be configured.

*group\_name*  
Specifies the name of a group. Modify *group\_name* for each server in the group.

*RMS*  
Specifies RMS parameters.

*param*  
Specifies the name of a server's configuration parameter.

*value*  
The value assigned to the configuration parameter specified in the *param* option.

**Examples** **Example 1** Returns a list of all configuration parameters for the server INVENTORY\_RS:

```
configure server INVENTORY_RS
```

**Example 2** Returns the *memory\_limit* configuration parameter information for the server INVENTORY\_RS:

```
configure server INVENTORY_RS memory_limit
```

| Parameter Name | Parameter Type | Current Value | Pending Value | Default Value |
|----------------|----------------|---------------|---------------|---------------|
| memory_limit   | NULL           | 20            | 55            | NULL          |

| Legal Values | Category | Restart Required | Description |
|--------------|----------|------------------|-------------|
| NULL         | NULL     | NULL             | NULL        |

**Example 3** Sets the *memory\_limit* configuration parameter to 50 for the server INVENTORY\_RS:

```
configure server inventory_rs memory_limit = 50
```

**Example 4** Retrieves all RMS-specific parameters:

```
configure server INVENTORY_RS RMS
```

**Example 5** Changes the user name used by the RMS to connect to the server:

```
configure server INVENTORY_RS RMS username = 'rsa'
```

Usage

- configure server supports Replication Server, Replication Agent, and remotely monitored RMS configurations.
- configure server can retrieve and set RMS-specific parameters for all types of servers. The server and the RMS use these parameters to communicate.
- configure server returns the following result set if you do not include a value parameter:

**Table 9-7: Default configure server result set**

| Column           | Description                                                                                           |
|------------------|-------------------------------------------------------------------------------------------------------|
| Parameter Name   | The name of the parameter.                                                                            |
| Parameter Type   | The type of parameter.                                                                                |
| Current Value    | The current value of the parameter.                                                                   |
| Pending Value    | The pending value becomes the value of the parameter after the server is restarted.                   |
| Default Value    | The default value of the parameter.                                                                   |
| Legal Values     | A string that defines the legal values for the parameter. This can be a list or a numeric range.      |
| Category         | The name of the category for the parameter. Categories are used to group related parameters together. |
| Restart Required | A flag indicating whether the server must be restarted in order for the parameter to take effect.     |
| Description      | The parameter description.                                                                            |

See also

add server, connect to server, disconnect server, drop server, get servers, shutdown server

## connect to server

|             |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
|-------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Description | Provides a pass-through mode that enables you to send commands to a server that is monitored by the RMS. The result sets generated by the commands are passed back to the client. You can connect to one server at a time to send commands.                                                                                                                                                                                                                                                      |
| Syntax      | <code>connect [to] <i>server_name</i> [username=<i>username</i> [,password = <i>pwd</i>]]</code>                                                                                                                                                                                                                                                                                                                                                                                                 |
| Parameters  | <p><i>server_name</i></p> <p>Specifies the name of the server to which to connect.</p> <p><i>username</i></p> <p>An optional parameter that specifies a user name to use when connecting to the server. If you omit this parameter, the RMS uses the name used when the server was added.</p> <p><i>pwd</i></p> <p>The password associated with the user name.</p>                                                                                                                               |
| Examples    | <p>Establishes a connection to the server INVENTORY_RS:</p> <pre>connect to INVENTORY_RS</pre>                                                                                                                                                                                                                                                                                                                                                                                                   |
| Usage       | <ul style="list-style-type: none"><li>• Issuing the connect command establishes a connection to the server. The message Established a connection to the server <b><i>server_name</i></b> indicates the connection is established.</li><li>• Subsequent commands are passed directly to the server until the client issues a disconnect command. Use ISQL commands appropriate for the server; for example, Transact-SQL for Adaptive Server Enterprise, or RCL for Replication Server.</li></ul> |
| See also    | add server, configure server, disconnect server, drop server, get servers, shutdown server                                                                                                                                                                                                                                                                                                                                                                                                       |

## create group

|             |                                                                                                 |
|-------------|-------------------------------------------------------------------------------------------------|
| Description | Defines a logical group of servers, and enables you to issue commands to the group.             |
| Syntax      | <code>create group <i>group_name</i><br/>[add] <i>server_name</i> [, <i>server_name</i>]</code> |
| Parameters  | <p><i>group_name</i></p> <p>Specifies the name of the new group.</p>                            |

*server\_name*

Specifies a server to add to the group.

#### Examples

Adds a group called “inventory\_mra” that contains three Mirror Replication Agent (MRA) servers:

```
create group inventory_mra
add ny_mra, chi_mra, la_mra
```

#### Usage

- A group name must be unique.
- All servers in a group must be the same type (that is, all servers must be MRAs, Replication Servers, and so on).
- A server can belong to more than one group.
- create group returns the following result set:

**Table 9-8: Column descriptions for create group**

| Column | Description                                                                           |
|--------|---------------------------------------------------------------------------------------|
| Action | The name of the action                                                                |
| Result | The result of the execution, such as Successfully created the group <i>group_name</i> |

#### See also

delete group, get group

## delete group

#### Description

Deletes a logical group that was added using the create group command.

#### Syntax

```
delete group group_name
```

#### Parameters

*group\_name*

Specifies the name of the group to delete.

#### Examples

Deletes the group named “inventory\_mra:”

```
delete group inventory_mra
```

#### Usage

- Deleting a group does not drop the servers from the RMS.
- delete group returns the following result set:

**Table 9-9: Column descriptions for delete group**

| Column | Description                                                                           |
|--------|---------------------------------------------------------------------------------------|
| Action | The name of the action                                                                |
| Result | The result of the execution, such as Successfully dropped the group <i>group_name</i> |

See also `create group`, `get group`

## disconnect server

|             |                                                                                                                                                                                                                                                                                            |
|-------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Description | Disconnects from a server where a pass-through connection was established. The client can connect through the RMS to a managed server using the <code>connect</code> command. Subsequent commands are forwarded to the server until the client issues the <code>disconnect</code> command. |
| Syntax      | <code>disconnect</code>                                                                                                                                                                                                                                                                    |
| Examples    | From the client, disconnects from a server:<br><br><code>disconnect</code>                                                                                                                                                                                                                 |
| Usage       | Issuing the <code>disconnect</code> command breaks the connection to the server. The message <code>Disconnected from the server <b>servername</b></code> indicates the connection no longer exists.                                                                                        |
| See also    | <code>add server</code> , <code>configure server</code> , <code>connect to server</code> , <code>drop server</code> , <code>get servers</code> , <code>shutdown server</code>                                                                                                              |

## drop event trigger

|             |                                                                                                                                                                                                                                                                                                                                                                 |
|-------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Description | Removes a trigger that the RMS is monitoring. A trigger identifies a process or script that is executed by the RMS. Set triggers up using the <code>add trigger</code> command.                                                                                                                                                                                 |
| Syntax      | <code>drop {status   latency   size} trigger</code><br><code>    [{connection   logical connection   route   queue   rep agent  </code><br><code>      partition} [<i>component_name</i>]]</code><br><code>    [with primary <i>primary_connection</i>]</code><br><code>    for <i>server_name</i></code><br><code>    {status changes to <i>state</i>  </code> |

|            |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
|------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|            | size {exceeds   falls below} <i>size_threshold</i>  <br>latency {exceeds   falls below} <i>latency_threshold</i>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| Parameters | <p>status, latency, size<br/>Specifies the type of trigger.</p> <p>connection, logical connection, route, queue, rep agent, partition<br/>Specifies the type of component.</p> <p><i>component_name</i><br/>Specifies the name of the component. Components are monitored objects within a server. Replication Server components are connections, logical connections, routes, queues, and partitions; Adaptive Server Enterprise components are RepAgent threads.</p> <p>with primary <i>primary_connection</i><br/>Identifies the primary connection of the latency connection trigger to drop. This parameter is required when dropping a latency connection.</p> <p><i>server_name</i><br/>Specifies the name of the server for which the trigger is defined that is being dropped.</p> <p><i>state</i><br/>Specifies the state of the event trigger that is being dropped. See Appendix C, “RMS Server and Component States” for state information.</p> <p>size exceeds, falls below <i>size_threshold</i><br/>Indicates the size trigger to drop.</p> <p>latency exceeds, falls below <i>latency_threshold</i><br/>Indicates the latency trigger to drop.</p> |
| Examples   | <p><b>Example 1</b> Removes the “DOWN” status trigger for the server INVENTORY_RS:</p> <pre>drop status trigger for INVENTORY_RS status changes to DOWN</pre> <p><b>Example 2</b> Removes the “SUSPENDED” status trigger for the connection “inventory_pds.pdb1” of server INVENTORY_RS:</p> <pre>drop status trigger connection inventory_pds.pdb1 for inventory_rs status changes to SUSPENDED</pre> <p><b>Example 3</b> Drops a partition size trigger:</p> <pre>drop size trigger partition p1 for INVENTORY_RS size exceeds 80</pre>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |

**Example 4** Drops a latency connection trigger:

```
drop latency trigger
 connection inventory_rds.vendor
 with primary inventory_pds.ventory
 for INVENTORY_RS
 latency exceeds 300
```

Usage drop trigger returns the following result set:

**Table 9-10: Column descriptions for drop event trigger**

| Column | Description                 |
|--------|-----------------------------|
| Action | The name of the action      |
| Result | The result of the execution |

See also add event trigger, get triggers

## drop server

Description Drops a server that is being monitored by the RMS.

Syntax drop server *server\_name*

Parameters *server\_name*  
Specifies the name of the server to be removed from the RMS.

Examples Drops the server named INVENTORY\_RS from the RMS. The agent no longer monitors the server:

```
drop server inventory_rs
```

Usage drop server returns the following result set:

**Table 9-11: Column descriptions for drop server**

| Column | Description                 |
|--------|-----------------------------|
| Action | The name of the action      |
| Result | The result of the execution |

See also add server, configure server, connect to server, disconnect server, get servers, shutdown server

## filter connection

|             |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
|-------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Description | Returns current filter settings, or sets the filter setting for a connection. The command can filter either the Replication Agent thread or the DSI thread status.                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| Syntax      | <code>filter connection for replication_server_name [{rep agent   dsi} [= {on   off}]]</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| Parameters  | <p><i>connection</i><br/>Specifies the name of the connection to filter.</p> <p><i>replication_server_name</i><br/>The name of the Replication Server to filter.</p> <p><i>rep agent, dsi</i><br/>Specifies the part of the connection to filter.</p> <p><i>on, off</i><br/>Sets filtering for the connection to either on or off.</p>                                                                                                                                                                                                                                                                            |
| Examples    | <p><b>Example 1</b> Returns the list of filter set for “inventory_pds.vendor” connection in prs1:</p> <pre>filter inventory_pds.vendor for prs1</pre> <p><b>Example 2</b> Hides the status of the DSI thread for the connection “inventory_pds.vendor” in prs1:</p> <pre>filter inventory_pds.vendor dsi for prs1 dsi = on</pre> <p><b>Example 3</b> Turns rep agent filtering off for the connection “inventory_pds.item” in prs1:</p> <pre>filter inventory_pds.item for prs1 rep agent = off</pre>                                                                                                             |
| Usage       | <ul style="list-style-type: none"> <li>When a filter is turned on, the connection status is displayed as “Hidden.” The status of the connection is not rolled up into the status of the Replication Server.</li> <li>If the rep agent filter is turned on, the RMS does not report the status of the Replication Agent thread or RepAgent thread in the Adaptive Server Enterprise, Replication Agent, or the Replication Server.</li> <li>When you invoke the filter command with no options specified, it returns a list of specified connections.</li> <li>filter returns the following result set:</li> </ul> |

Table 9-12: filter connection result set (list of filtered connections)

| Column     | Description                        |
|------------|------------------------------------|
| RepServer  | The name of the Replication Server |
| Connection | The name of the connection         |
| DSI        | The filtering value of DSI         |
| rep agent  | The filtering value of rep agent   |

- The filter command returns the following result set, if you have turned filtering on or off for the connection:

Table 9-13: filter connection result set (filtering turned on/off)

| Column | Description                 |
|--------|-----------------------------|
| Action | The name of the action      |
| Result | The result of the execution |

See also `get network spec`, `get threads`

# get component

|             |                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
|-------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Description | Returns a list of components that are monitored by the RMS. Components are monitored objects within a server. Replication Server components are connections, logical connections, routes, queues, and partitions; Adaptive Server Enterprise components are RepAgent threads.                                                                                                                                                                                          |
| Syntax      | <pre>get {connections   logical connections   routes   queues   partitions       repagents}     for server_name [, component_name]...</pre>                                                                                                                                                                                                                                                                                                                            |
| Parameters  | <p>connections, logical connections, routes, queues, partitions, repagents</p> <p>Returns the specified type of component monitored by the RMS. For example, returns all connections in a specified Replication Server monitored by the RMS.</p> <p><i>server_name</i></p> <p>Specifies the server that contains the requested components. If the server does not contain any of the requested components, <code>get component</code> returns an empty result set.</p> |

*component\_name*

Specifies a specific component or list of components to return. Components are monitored objects within a server. Replication Server components include connections, logical connections, routes, queues, and partitions. Adaptive Server Enterprise components are RepAgent threads.

**Examples**

**Example 1** Returns a list of all connections being monitored by the RMS in the Replication Server INVENTORY\_RS:

```
get connections for INVENTORY_RS
```

**Example 2** Returns a list of all RepAgent threads being monitored by the RMS in the Adaptive Server Enterprise server called INVENTORY\_PDS:

```
get repagents for INVENTORY_PDS
```

**Example 3** Returns the information for the route named “inventory\_rs.euro\_sales” for the Replication Server INVENTORY\_RS:

```
get routes for INVENTORY_RS, inventory_rs.euro_sales
```

**Usage**

- Components monitored by a remote RMS are also returned by this command.
- `get connections` supports retrieving connections that are associated with a data server or a Replication Agent process. It supports servers other than a Replication Server:
  - ASE – `get connections` returns the connection information for each database in the ASE. The RMS searches all of the Replication Servers in the RMS looking for connections named *ASE\_name.database*.
  - Replication Agent/MRA – `get connections` returns the information for the primary connection associated with the Replication Agent. The name of the connection associated with the Replication Agent or MRA is stored in the configuration parameters `rs_source_ds` and `rs_source_db`. `get connections` searches all of the Replication Servers in the RMS to find the connection.
  - dbltm – `get connections` returns the information for the primary connection associated with the dbltm. The connection information for the dbltm is optionally provided when the server is added to the environment. If the information is not available, `get connections` returns an empty result set and writes a warning message to the RMS log indicating the information is missing.
  - DirectConnect – `get connections` returns the information of all of the connections where the data server matches the name of the DirectConnect server.

- SA/IQ – get connections returns the information where the data server matches the name of the SA or IQ server. SA or IQ server does not use database names.
- If the specified server is not monitored by the RMS, the *get component* command returns an error message.
- *get component* returns the following result set (some results vary by component type):

**Table 9-14: Column descriptions for get component result set**

| Column                 | Description                                                                                                                                                                                                                               |
|------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Server                 | The name of the server that contains the components.                                                                                                                                                                                      |
| Name                   | The name of the component.                                                                                                                                                                                                                |
| Type                   | The type of the component (connection, route, queue, RepAgent).                                                                                                                                                                           |
| Last Monitored         | A timestamp indicating that last time the component was monitored by the RMS. The timestamp is in the format MM/DD/YYYY HH:MM:SS.                                                                                                         |
| State                  | The description that defines the state of the component.                                                                                                                                                                                  |
| State Constant         | The integer constant that defines the state of the component. See Appendix C, “RMS Server and Component States” for state information.                                                                                                    |
| Description            | The reason string that describes the state of the component.                                                                                                                                                                              |
| More Descriptions      | Indicates whether additional information is available. If true, then the status of the component contains multiple descriptions. Use the <i>get status descriptions</i> command to retrieve a list of all descriptions for the component. |
| Intermediate RepServer | Identifies the intermediate site for the route. Intermediate RepServer should be blank if the route is a direct route                                                                                                                     |
| Queue Number           | The queue number.                                                                                                                                                                                                                         |
| Queue Type             | The queue type.                                                                                                                                                                                                                           |
| Size column            | The queue size.                                                                                                                                                                                                                           |

See also

configure component, get status descriptions, get servers, resume component, suspend component

## get group

**Description** Returns a result set that contains either a list of the groups and a roll-up status for each group, or status of each server in a group and a roll-up status for the specified group. Roll-up status shows the lowest status reported; for example, if any server in a group is not UP, then the group status is reported as “SUSPECT”.

**Syntax** `get group [group_name]`

**Parameters** *group\_name*  
Specifies the name of the group for which to retrieve the list of servers.

**Examples** **Example 1** Returns a list of the groups names, and a roll-up status for each group:

```
get group
```

| Group Name | State    | State   | Description              | More         |
|------------|----------|---------|--------------------------|--------------|
|            | Constant |         |                          | Descriptions |
| -----      | -----    | -----   | -----                    | -----        |
| group1     | 4        | Suspect | inventory_rsl is Suspect | False        |

**Example 2** Returns the status of each list of server names that the group “inventory\_mra” contains and a roll-up status for the group:

```
get group inventory_mra
```

| Group Name    | Server Name | Server Type       | Last Monitored      |
|---------------|-------------|-------------------|---------------------|
| -----         | -----       | -----             | -----               |
| inventory_mra | RAObeta     | Replication Agent | 12/16/2005 13:38:30 |

Version String

```

Sybase Replication Agent for Unix & Windows/12.6.0.5001/B/generic/
JDK 1.4.2/main/5001/VM: Sun Microsystems Inc. 1.4.2_05/OPT/Wed May 4
02:42:07 MDT 2005
```

| State Constant | State | Description                   | More Descriptions |
|----------------|-------|-------------------------------|-------------------|
| -----          | ----- | -----                         | -----             |
| 6              | Admin | Waiting for operator command. | false             |

**Usage**

- If you do not provide a *group\_name* parameter, get group returns a result set that contains a roll-up status for each group:

**Table 9-15: Column descriptions for get group (group list, and roll-up for each group)**

| Column            | Description                                                                                                                                      |
|-------------------|--------------------------------------------------------------------------------------------------------------------------------------------------|
| Group Name        | The name of the group.                                                                                                                           |
| State Constant    | The integer constant that defines the state of the group.                                                                                        |
| State             | The description that defines the state of the group. This is a string representation of the State Constant column.                               |
| Description       | The reason string that describes the state of the group. If there is more than one description, this field should contain the first description. |
| More Descriptions | A flag that indicates whether there is more than one description string that describes the status of the group.                                  |

- If you provide a *group\_name* parameter, get group returns a result set that contains the status of each server:

**Table 9-16: Column descriptions for get group (individual server, and roll-up for specified group)**

| Column            | Description                                                                                                                                                                                                             |
|-------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Group Name        | The name of the group.                                                                                                                                                                                                  |
| Server Name       | The name of the server.                                                                                                                                                                                                 |
| Server Type       | The type of the server (Replication Server, Adaptive Server Enterprise, Replication Agent, and so on).                                                                                                                  |
| Last Monitored    | A timestamp indicating that last time the server was monitored by the RMS. The timestamp is in the format <i>MM/DD/YYYY HH:MM:SS</i> .                                                                                  |
| Version String    | Returns the server version string.                                                                                                                                                                                      |
| State Constant    | The numeric status of the server.                                                                                                                                                                                       |
| State             | The description that defines the state of the server. This is a string representation of the state constant.                                                                                                            |
| Description       | The reason string that describes the state of the server.                                                                                                                                                               |
| More Descriptions | Indicates whether additional information is available. If true, then the status of the server contains multiple descriptions. Use <i>get status descriptions</i> to retrieve a list of all descriptions for the server. |

See also

create group, delete group, get status descriptions

## get heartbeat

**Description** Retrieves the heartbeats that have been defined in the RMS. A heartbeat is a process that runs the Replication Server `rs_ticket` stored procedure at the primary database at a specified interval. The output, or heartbeat ticket, is stored in a table in the replicate database.

**Syntax** `get heartbeat [for ds.db]`

**Parameters** *ds.db*  
The name of a connection that is participating in a heartbeat process. This name can be either a primary or replicate connection.

**Examples** **Example 1** Retrieves all heartbeats defined in the RMS:

```
get heartbeat
```

**Example 2** Retrieves heartbeats defined for the “inventory\_pds.pdb1” connection:

```
get heartbeat for inventory_pds.pdb1
```

**Usage** `get heartbeat` returns the following result set:

**Table 9-17: Column descriptions for Get Heartbeat**

| Column    | Type    | Description                                                                                                                                                                                                                               |
|-----------|---------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Primary   | varchar | The name of the primary data server and database.                                                                                                                                                                                         |
| Replicate | varchar | The name of the replicate data server and database.                                                                                                                                                                                       |
| Interval  | int     | The interval in seconds that the RMS executes the <code>rs_ticket</code> command.                                                                                                                                                         |
| Max Rows  | int     | The maximum number of rows that the <code>rms_ticket_history</code> table can contain. The RMS tests the size of the table at every heartbeat interval. If the size is greater than <i>max_rows</i> , the RMS removes the oldest entries. |

**See also** `get heartbeat tickets`, `start heartbeat`, `stop heartbeat`

## get heartbeat tickets

**Description** Retrieves a set of tickets from the `rms_ticket_history` table for the heartbeat process and date and time range specified. The ticket output includes a set of date and time fields for each step in the replication process. The date and time are synchronized to the replicate data server system time.

**Syntax** `get heartbeat tickets from pds.pdb to rds.rdb  
[start date time]  
[end date time]  
[last num_tickets]`

**Parameters** *pds.pdb*

The name of the primary data server and database.

*rds.rdb*

The name of the replicate data server and database.

*start date time*

The starting date and time for the range of tickets. The RMS retrieves ticket information starting with this time and ending at either the end time, or the end of the table. If you do not provide this parameter, the RMS starts at the oldest ticket in the table.

*end date time*

The ending date and time for the range of tickets. The RMS retrieves ticket information starting at the specified time until this time. If you do not provide this parameter, the RMS includes all tickets starting with the start time.

*last num\_tickets*

Retrieves the specified number of tickets from the table. You cannot use this parameter with the `start` and `end` parameters

**Examples**

**Example 1** Retrieves all rows from the `rms_ticket_history` table:

```
get heartbeat tickets
from inventory_pds.vendor to inventory_dss.vendor
```

**Example 2** Retrieves all rows between Oct 29th and November 3rd:

```
get heartbeat tickets
from inventory_pds.vendor to inventory_dss.vendor
start Oct 29, 2005 12:00am
end Nov 3, 2005 12:00am
```

**Example 3** Retrieves all rows in the table starting at October 29th at 1:30:

```
get heartbeat tickets
from inventory_pds.vendor to inventory_dss.vendor
```

```
start 10/29 1:30pm
```

**Example 4** Retrieves the 500 latest rows in the table:

```
get heartbeat tickets
 from inventory_pds.vendor to inventory_dss.vendor
last 500
```

#### Usage

- The `start` and `end` parameters support multiple date and time formats; for example, you can enter the date in the format MM/DD/YYYY (such as 10/29/2005), or in the format MMM DD, YYYY (such as Oct 29, 2005). The time fields support an entry without seconds or milliseconds, as well as localized date and time formats.
- All dates in the result set are synchronized to the replicate data server system time. Before the result set is generated, the RMS retrieves the date and time from the data servers and Replication Servers, and adjusts the time by the difference between the server's time and the RMS system's time.
- The `get heartbeat ticket` command returns the following result set:

**Table 9-18: Column descriptions for `get heartbeat tickets`**

| Column    | Type     | Description                                                                                          |
|-----------|----------|------------------------------------------------------------------------------------------------------|
| Primary   | varchar  | The name of the primary data server and database.                                                    |
| Replicate | varchar  | The name of the replicate data server and database.                                                  |
| PDB       | datetime | The time that the <code>rs_ticket</code> stored procedure was executed at the primary database.      |
| EXEC      | datetime | The time the ticket passed through the primary Replication Server executor thread.                   |
| Bytes     | int      | Total bytes the executor thread received from the RepAgent or Replication Agent.                     |
| DIST      | datetime | The time the ticket passed through the primary Replication Server distributor thread.                |
| DSI       | datetime | The time the ticket passed through the replicate Replication Server DSI thread.                      |
| RDB       | datetime | The time the ticket arrived at the replicate data server. The result set is sorted by the RDB field. |

See also

`get heartbeat`, `start heartbeat`, `stop heartbeat`

# get network spec

**Description** Retrieves the connection information for all servers known to the RMS. This list is retrieved from the RMS *interfaces* file or LDAP server. The list consists of the server name, host computer name, and the port number used by the server.

**Syntax** `get network spec [[monitored] | [server_name [,server_name]]]`

**Parameters**

`monitored`  
Returns the list of servers that the RMS is currently monitoring.

`server_name`  
Specifies the name of a server or set of servers for which to retrieve information.

**Examples** **Example 1** Retrieves a list of all servers from the RMS *interfaces* file or LDAP server:

```
get network spec
```

**Example 2** Retrieves the connection information for the set of servers managed by the RMS:

```
get network spec monitored
```

**Example 3** Retrieves the connection information for the servers INVENTORY\_RS and INVENTORY\_ASE:

```
get network spec INVENTORY_RS, INVENTORY_ASE
```

- Usage**
- Returns an empty result set if the requested server does not exist or the *interfaces* file or LDAP server is not available.
  - `get network spec` returns the following result set:

**Table 9-19: Column descriptions for `get network spec`**

| Column | Description                                             |
|--------|---------------------------------------------------------|
| Name   | The name of the server                                  |
| Host   | The name of the computer that hosts the server          |
| Port   | The port number of the host on which the server listens |

**See also** `filter connection`

## get rmiaddress

|             |                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
|-------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Description | <p>Retrieves the address of Remote Method Invocation (RMI) service. RMI enables an object running in one Java virtual machine (VM) to invoke methods on an object running in another Java VM. RMI provides remote communication between programs written in Java.</p> <p>RMS provides client applications the ability to register callback routines that are executed when a specific event occurs. The RMS provides asynchronous callbacks using the remote RMI feature.</p> |
| Syntax      | <code>get rmiaddress</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
| Parameters  | <p><code>rmiaddress</code></p> <p>Returns the server and port used for RMI service.</p>                                                                                                                                                                                                                                                                                                                                                                                       |
| Examples    | <p>Retrieves the address of the RMI service:</p> <pre>get rmiaddress</pre> <pre>Rmi Address ----- rmi://redtail:9999/</pre>                                                                                                                                                                                                                                                                                                                                                   |
| Usage       | <code>get rmiaddress</code> returns the address of the RMI service.                                                                                                                                                                                                                                                                                                                                                                                                           |

## get servers

|             |                                                                                                                                                                                                                                                                                                                                                     |
|-------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Description | <p>Returns the status for each of the servers that are monitored by the RMS, followed by the status of the RMS environment. The RMS status is a roll-up of the monitored servers, and shows the lowest status reported; for example, if the status of any server in the list is not “UP”, then the status for the RMS is reported as “SUSPECT”.</p> |
| Syntax      | <code>get servers [[for group <i>group_name</i>] [{ASA   ASE   DirectConnect   IQ   Replication Agent   MRA   Replication Server   RMS   Open Server   [<i>server_name</i>,...}]]</code>                                                                                                                                                            |
| Parameters  | <p>ASA, ASE, DirectConnect, IQ, Replication Agent, MRA, Replication Server, RMS, Open Server</p> <p>Returns only the specified type of server monitored by the RMS. For example, returns all Replication Servers monitored by the RMS.</p> <p><i>group_name</i></p> <p>Specifies a group for which servers are returned.</p>                        |

Examples

*server\_name*  
Specifies a specific server or list of servers to return. If the server is not monitored by the RMS, an empty result set is returned.

**Example 1** Returns the status for all servers monitored by the RMS, followed by the status for the RMS environment:

```
get servers
```

**Example 2** Returns a list of all Adaptive Server Enterprise servers monitored by the RMS:

```
get servers ASE
```

**Example 3** Returns a list that contains the information for the servers INVENTORY\_RS and INVENTORY\_PDS”

```
get servers INVENTORY_RS, INVENTORY_PDS
```

Usage

Servers monitored by a remote RMS are also returned by this command.

**Table 9-20: Column descriptions for get servers**

| Column            | Description                                                                                                                                                                                                                               |
|-------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Name              | The server name.                                                                                                                                                                                                                          |
| Type              | The server type (Replication Server, Adaptive Server Enterprise, Replication Agent, and so forth).                                                                                                                                        |
| Last Monitored    | A timestamp indicating that last time the server was monitored by the RMS. The timestamp is in the format <i>MM/DD/YYYY HH:MM:SS</i> .                                                                                                    |
| Version String    | The complete version string of the server.                                                                                                                                                                                                |
| State Constant    | The integer constant that defines the state of the server. See Appendix C, “RMS Server and Component States” for server state information.                                                                                                |
| State             | The description that defines the state of the server. This is a string representation of the state constant.                                                                                                                              |
| Description       | A string that describes the state of the server.                                                                                                                                                                                          |
| More Descriptions | Indicates whether additional information is available. If true, then the status of the server contains multiple descriptions. Use the <code>get status descriptions</code> command to retrieve a list of all descriptions for the server. |

See also

`add server`, `configure server`, `connect to server`, `disconnect server`, `drop server`, `get component`, `get status descriptions`, `shutdown server`

## get status descriptions

|             |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
|-------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Description | <p>Retrieves the list of status descriptions for a server or component. Components are monitored objects within a server. The state of a server or component consists of a state integer constant and a list of description strings. The <code>get server</code> and <code>get component</code> commands return the first description in the list and a flag that indicates whether the description list contains more than one string.</p> <p>Client applications can use <code>get server</code> or <code>get component</code> to display the state of all servers monitored by the RMS. If more information is needed, the application can display all descriptions.</p>                                                                                                                                                                                  |
| Syntax      | <pre>get status descriptions {[for {connection   logical connection   route   queue                           rep agent   partition}                         component_name] for server_name   for group_name}</pre>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| Parameters  | <p><code>connection</code>, <code>logical connection</code>, <code>route</code>, <code>queue</code>, <code>rep agent</code>, <code>partition</code><br/>Returns status descriptions for the specified server or component.</p> <p><i>component_name</i><br/>Specifies the name of the component for which to return status descriptions. Components are monitored objects within a server. Replication Server components are connections, logical connections, routes, queues, and partitions. Adaptive Server Enterprise components are RepAgent threads.</p> <p><i>server_name</i><br/>Specifies the name of the server for which to return status descriptions. The server name is also used when returning status descriptions for components.</p> <p><i>group_name</i><br/>Specifies the name of the group for which to return status descriptions.</p> |
| Examples    | <p><b>Example 1</b> Retrieves all description strings for the server name <code>INVENTORY_RS</code>:</p> <pre>get status descriptions for INVENTORY_RS</pre> <p><b>Example 2</b> Retrieves all description strings for the group name “group1”:</p> <pre>get status descriptions for group1</pre> <p><b>Example 3</b> Retrieves all description strings for the connection “inventory_pds.pdb1” in the server <code>INVENTORY_ASE</code>:</p> <pre>get status descriptions for connection inventory_pds.pdb1 for INVENTORY_ASE</pre>                                                                                                                                                                                                                                                                                                                         |
| Usage       | <ul style="list-style-type: none"> <li>• <code>get status descriptions</code> returns all strings in the description list (including the first description).</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |

- You can use get status descriptions to return the status descriptions for the RMS.
- get status descriptions returns a result set that contains a single string column that contains one status description. The result set returns multiple rows, one for each description.

See also

get component, get servers

## get threads

|             |                                                                                                                                                                                                                                                                                                                  |
|-------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Description | Displays information about threads running in the Replication Server.                                                                                                                                                                                                                                            |
| Syntax      | get threads [for] <i>server_name</i> [{dist   dsi   rsi   sqm   sqt}]                                                                                                                                                                                                                                            |
| Parameters  | <p><i>server_name</i><br/>Specifies the Replication Server that contains the threads.</p> <p>dist   dsi   rsi   sqm   sqt<br/>Specifies the thread type. If no type is specified, the summary list of threads is returned.</p>                                                                                   |
| Examples    | <p><b>Example 1</b> Returns the summary list of all threads in the Replication Server INVENTORY_RS:</p> <pre>get threads for INVENTORY_RS</pre> <p><b>Example 2</b> Returns the thread information for all route threads in the Replication Server INVENTORY_RS:</p> <pre>get threads for INVENTORY_RS rsi</pre> |
| Usage       | get threads executes the admin who command for the specified Replication Server. The result set is identical to the admin who result set.                                                                                                                                                                        |
| See also    | filter connection, resume component, suspend component                                                                                                                                                                                                                                                           |

## get triggers

|             |                                                                                      |
|-------------|--------------------------------------------------------------------------------------|
| Description | Displays information about the triggers that are monitored by the RMS.               |
| Syntax      | get status triggers<br>[[connection   logical connection   route   queue   rep agent |

```
partition}
component_name for server_name]
```

**Parameters****status**

Specifies the type of trigger.

**connection, logical connection, route, queue, rep agent, partition**

Specifies the type of component to be monitored. Components are monitored objects within a server. Replication Server components are connections, logical connections, routes, queues, and partitions. Adaptive Server Enterprise components are RepAgent threads.

**component\_name**

Specifies the name of the component to be monitored.

**server\_name**

Specifies the name of the server to be monitored.

**Examples**

**Example 1** Returns the list of all triggers in the RMS:

```
get triggers
```

**Example 2** Returns the list of all triggers defined for the Replication Server INVENTORY\_RS:

```
get triggers for INVENTORY_RS
```

**Example 3** Returns the list of all triggers defined for the connection “inventory\_pds.vendor” in the Replication Server INVENTORY\_RS:

```
get triggers connection inventory_pds.vendor for
INVENTORY_RS
```

**Usage**

get triggers returns the following result set:

**Table 9-21: Column descriptions for get triggers**

| Column             | Description                                                                                                                                                                                                                      |
|--------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Type               | The type of the trigger.                                                                                                                                                                                                         |
| Server Type        | The server type of the trigger.                                                                                                                                                                                                  |
| Server Name        | The server name of the trigger.                                                                                                                                                                                                  |
| Component Type     | The component type of the trigger.                                                                                                                                                                                               |
| Component Name     | The component name of the trigger.                                                                                                                                                                                               |
| Primary Connection | The name of the primary connection.                                                                                                                                                                                              |
| Change Value       | The value of the server or component that will cause the RMS to execute the trigger's script.                                                                                                                                    |
| Change State       | The state string of the server or component that will cause the RMS to execute the trigger's script.                                                                                                                             |
| Wait               | The number of seconds to wait after the initial state change before executing the trigger's script. If <i>waitInterval</i> is set to zero, the script executes immediately.                                                      |
| Continuous         | A Boolean flag that, if set to true, causes the RMS to execute the trigger's script at every subsequent monitoring interval until the state changes. If the flag is not set, then the RMS executes the trigger script only once. |
| Script             | The operating system script that the RMS executes when the event occurs.                                                                                                                                                         |

See also [add event trigger](#), [drop event trigger](#)

## get version

|             |                                      |
|-------------|--------------------------------------|
| Description | Retrieves the version string of RMS. |
|-------------|--------------------------------------|

Syntax `get version`

Parameters version

Returns a string containing several pieces of version information separated by slashes.

|          |                                          |
|----------|------------------------------------------|
| Examples | Retrieves the version string of the RMS: |
|----------|------------------------------------------|

version

Replication Monitoring Services/15.0/P/generic/JDK 1.4.2.03/main/

Build 102/VM:

Sun Microsystems Inc. 1.5.0\_05/Opt/Wed Dec 7 15:26:13 CST 2005

Usage                      get version returns the version string of the RMS.  
See also                    configure RMS, resume Replication Agent, suspend Replication Agent, trace

## log level

Description               Returns the current log level setting. log level also changes log level settings of RMS.

Syntax                    log level [= {debug | info | warn | error | fatal}]

Parameters               debug, info, warn, error, fatal  
                            The log level value.

Examples                  **Example 1** Returns the current log level setting:  
                            log level

**Example 2** Sets the log level to error:  
                            log level = error

Usage                      The log level has the following order: debug, info, warn, error, fatal. You must set the log level to at least info to trace log level messages.

## resume component

Description               Resumes a component in a specified server. The command resumes a DSI thread, Replication Agent thread, queue, or route in a Replication Server, or a RepAgent thread in an Adaptive Server Enterprise.

Syntax                    resume {dsi | queue | rep agent | route} *component\_name*  
                            for {*server\_name* | *group\_name*} [skip transaction | execute transaction]

Parameters               dsi, queue, rep agent, route  
                            Specifies the component type to resume. The component is a database name, if resuming a RepAgent thread in an Adaptive Server Enterprise. Otherwise, the component is a connection, queue, or route name.

*component\_name*  
                            Specifies the name of the component to resume.

*group\_name*

Specifies the name of a group. Each component in the group is resumed.

*server\_name*

Specifies the name of either a Replication Server or an Adaptive Server Enterprise that contains the component.

*skip transaction*

If the option is provided for a DSI connection, instructs the Replication Server to resume execution with the second transaction in the connection's queue. The first transaction is written to the database exceptions log.

If the option is provided for a queue, specifies that the SQM should skip the first large message encountered after restarting.

If this option is provided for a route, ignore the first transaction encountered with a wide message greater than 16K bytes.

*execute transaction*

Overrides the Replication Server restriction against the application of system transactions after a DSI start-up if the system transaction is the first transaction in the DSI queue.

Examples

**Example 1** Resumes the DSI thread for the connection

“inventory\_pds.vendor” in the Replication Server INVENTORY\_RS. Does not wait for the current operation to complete:

```
resume dsi inventory_pds.vendor for INVENTORY_RS with
nowait
```

**Example 2** Resumes the Replication Agent thread for the connection

“inventory\_pds.vendor” in the Replication Server INVENTORY\_RS:

```
resume rep agent inventory_pds.vendor for INVENTORY_RS
```

**Example 3** Starts the RepAgent thread for the database vendor in the Adaptive Server Enterprise INVENTORY\_PDS:

```
resume rep agent vendor for INVENTORY_PDS
```

Usage

- The rep agent component type is used to resume either a Replication Agent thread for a connection in a Replication Server, or a RepAgent thread in an Adaptive Server Enterprise.
- The skip transaction option is valid with a Replication Server DSI connection, queue, or route.
- The execute transaction option is valid only for a Replication Server DSI connection.resume issues the sp\_start\_rep\_agent when resuming a RepAgent thread in an Adaptive Server Enterprise.

- resume returns the following result set.

**Table 9-22: Column descriptions for resume component**

| Column | Description                 |
|--------|-----------------------------|
| Action | The name of the action      |
| Result | The result of the execution |

See also `configure component`, `get component`, `get threads`, `suspend component`

## resume *Replication Agent*

|             |                                                                                                                                                                                                  |
|-------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Description | Resumes replication in a Replication Agent.                                                                                                                                                      |
| Syntax      | <code>resume {<i>server_name</i>   <i>group_name</i>}</code>                                                                                                                                     |
| Parameters  | <p><i>server_name</i><br/>Specifies the name of the Replication Agent to resume.</p> <p><i>group_name</i><br/>Specifies the name of a group. Each Replication Agent in the group is resumed.</p> |
| Examples    | <p>Resumes the Replication Agent “sales_ra:”</p> <pre>resume sales_ra</pre>                                                                                                                      |
| Usage       | None                                                                                                                                                                                             |
| See also    | <code>configure RMS</code> , <code>get version</code> , <code>suspend Replication Agent</code> , <code>trace</code>                                                                              |

## shutdown *server*

|             |                                                                                                                                                                        |
|-------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Description | Issues a shutdown command to a server.                                                                                                                                 |
| Syntax      | <code>shutdown {<i>server_name</i>   <i>group_name</i>} [<i>with nowait</i>]</code>                                                                                    |
| Parameters  | <p><i>server_name</i><br/>Specifies the server to be shut down.</p> <p><i>group_name</i><br/>Specifies the name of a group. Each server in the group is shut down.</p> |

|          |                                                                                                                       |
|----------|-----------------------------------------------------------------------------------------------------------------------|
|          | with nowait<br>Shut down the server immediately without waiting for the executing operation to complete.              |
| Examples | Issues the shutdown command to the server named INVENTORY_RS:<br><br><code>shutdown INVENTORY_RS</code>               |
| Usage    | The RMS allows the user to shut down <i>only</i> Replication Server, Replication Agent, and Mirror Replication Agent. |
| See also | add server, configure server, connect to server, disconnect server, drop server, get servers                          |

## start heartbeat

|             |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
|-------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Description | Sets up and starts a heartbeat process from a specified primary connection to a specified replicate connection.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| Syntax      | start heartbeat from <i>pds.pdb</i> to <i>rds.rdb</i><br>[set interval [to] <i>hb_interval</i> ]<br>[set maximum rows [to] <i>max_rows</i> ]<br>[do not load rs_ticket_report]                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| Parameters  | <i>pds.pdb</i><br>The name of the primary data server and database. The name must be associated with an existing primary connection.<br><br><i>rds.rdb</i><br>The name of the replicate data server and database. The name must be associated with an existing primary and replicate, or replicate-only connection.<br><br><i>hb_interval</i><br>The interval in seconds that the RMS executes the rs_ticket command. The default is 60 seconds.<br><br><i>max_rows</i><br>The maximum number of rows that the rms_ticket_history table can contain. The RMS tests the size of the table at every heartbeat interval. If the size is greater than <i>max_rows</i> , the RMS removes the oldest entries. The RMS deletes 10% of the <i>max_row</i> size rows in the table. The default is 5000 rows. |

do not load `rs_ticket_report`

If this flag is included, the RMS does not load the `rs_ticket_report` and you can provide a custom stored procedure instead. You must provide an `rs_ticket_report` procedure that loads the `rms_ticket_history` table with the required information.

#### Examples

Sets up and starts the heartbeat process, then executes the `rs_ticket` procedure every 60 seconds; limits the `rms_ticket_history` table to 5000 rows:

```
start heartbeat
 from inventory_pds.vendor to inventory_dss.vendor
```

#### Usage

- To set up the heartbeat, the RMS uses the user name that was provided when the server was added to the domain. The user names must have the correct permissions to create the table and stored procedure at the replicate database, configure the DSI at the replicate Replication Server, and execute the `rs_ticket` stored procedure at the primary database.
- The RMS can create only one heartbeat between a primary and replicate database. The RMS generates an error if a heartbeat already exists.
- The RMS does not delete an `rms_ticket_history` table if one already exists, but assumes that another heartbeat from a different primary database is already executing.
- The RMS assumes that the replicate database is set-up to receive data from the Replication Server and it neither checks for subscriptions nor generates a new one. Replication Server version must be at least 12.6.
- The Replication Server requires that the replicate database must have at least one subscription against a table, stored procedure, or database before the replicate Replication Server sends the `rs_ticket` information. The subscription does not have to be against any specific table or stored procedure. In case there is no subscription, `rs_ticket` functions in a warm-standby environment.

#### See also

`get heartbeat`, `get heartbeat tickets`, `stop heartbeat`

## stop heartbeat

#### Description

Stops the heartbeat process between the primary and replicate databases. Optionally, truncates the `rms_ticket_history` table.

#### Syntax

```
stop heartbeat from pds.pdb to rds.rdb
[delete history]
```

|            |                                                                                                                                                      |
|------------|------------------------------------------------------------------------------------------------------------------------------------------------------|
| Parameters | <i>pds.pdb</i><br>The name of the primary data server and database.                                                                                  |
|            | <i>rds.rdb</i><br>The name of the replicate data server and database.                                                                                |
|            | delete history<br>If included, the rms_ticket_history table is deleted when the heartbeat is stopped. By default, the table is not deleted.          |
| Examples   | Stops the heartbeat process:<br><br><pre>stop heartbeat from inventory_pds.vendor to inventory_dss.vendor</pre>                                      |
| Usage      | Optionally, you can delete the rms_ticket_history table when the heartbeat is stopped. This means you can no longer retrieve tickets from the table. |
| See also   | get heartbeat, get heartbeat tickets, start heartbeat                                                                                                |

## **suspend component**

|             |                                                                                                                                                                                                                                             |
|-------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Description | Suspends a component in a specified server. The command suspends a DSI thread, a route in a Replication Server, or a RepAgent thread in Adaptive Server Enterprise.                                                                         |
| Syntax      | <pre>suspend {dsi   rep agent   route} <i>component_name</i> for {<i>server_name</i>   <i>group_name</i>} [<i>with nowait</i>]</pre>                                                                                                        |
| Parameters  | dsi, rep agent, route<br>Specifies the component type to suspend.                                                                                                                                                                           |
|             | <i>component_name</i><br>Specifies the name of the component to suspend. The component is a database name if you are suspending a RepAgent thread in an Adaptive Server Enterprise. Otherwise, the component is a connection or route name. |
|             | <i>server_name</i><br>Specifies the name of either a Replication Server or an Adaptive Server Enterprise that contains the component.                                                                                                       |
|             | <i>group_name</i><br>Specifies the name of a group. Each component in the group is suspended.                                                                                                                                               |

with `nowait`

Suspends the component immediately without waiting for the executing operation to complete.

#### Examples

**Example 1** Suspends the DSI thread for the connection

“inventory\_pds.vendor” in the Replication Server INVENTORY\_RS, without waiting for the current operation to complete:

```
suspend dsi inventory_pds.vendor
for INVENTORY_RS with nowait
```

**Example 2** Suspends the Replication Agent thread for the connection

“inventory\_pds.vendor” in the Replication Server named INVENTORY\_RS:

```
suspend rep agent inventory_pds.vendor for INVENTORY_RS
```

**Example 3** Stops the RepAgent thread for the database vendor in the Adaptive Server Enterprise named INVENTORY\_PDS:

```
suspend rep agent vendor for INVENTORY_PDS
```

#### Usage

- The `rep agent` component type is used to suspend either a Replication Agent thread for a connection in a Replication Server, or a RepAgent thread in an Adaptive Server Enterprise.
- The `with nowait` option is valid with a Replication Server DSI connection or an Adaptive Server Enterprise RepAgent thread.
- `suspend component` issues the `sp_stop_rep_agent` stored procedure when suspending a RepAgent thread in an Adaptive Server Enterprise.
- `suspend component` returns the following result set:

**Table 9-23: Column descriptions for `suspend component`**

| Column | Description                  |
|--------|------------------------------|
| Action | The name of the action.      |
| Result | The result of the execution. |

#### See also

`configure component`, `get component`, `get threads`, `resume component`

## suspend Replication Agent

**Description** Suspends replication in a Replication Agent.

**Syntax** `suspend {server_name | group_name}`

|            |                                                                                                                                                                                                     |
|------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Parameters | <p><i>server_name</i><br/>Specifies the name of the Replication Agent to suspend.</p> <p><i>group_name</i><br/>Specifies the name of a group. Each Replication Agent in the group is suspended.</p> |
| Examples   | <p>Suspends the Replication Agent “sales_ra”:</p> <pre>suspend sales_ra</pre>                                                                                                                       |
| Usage      | None                                                                                                                                                                                                |
| See also   | configure RMS, get version, resume Replication Agent, trace                                                                                                                                         |

## trace

|             |                                                                                                                                                                                                                                                                                                      |
|-------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Description | Displays trace information in the RMS log file.                                                                                                                                                                                                                                                      |
| Syntax      | <code>trace [<i>flag</i>   all {on   off}]</code>                                                                                                                                                                                                                                                    |
| Parameters  | <p><i>flag</i><br/>Specifies the trace flag name for which you want to change settings.</p> <p>all<br/>A keyword that allows you to apply a switch value to all trace flags.</p> <p>on, off<br/>Indicates whether to enable or disable tracing for the trace point specified in the flag option.</p> |
| Examples    | <p><b>Example 1</b> Returns the current settings for all RMS trace flags:</p> <pre>trace</pre> <p><b>Example 2</b> Turns the RMS_Command trace flag on:</p> <pre>trace RMS_Command on</pre> <p><b>Example 3</b> Turns off all trace flags:</p> <pre>trace all off</pre>                              |
| Usage       | <ul style="list-style-type: none"> <li>The trace command should only be used by knowledgeable users to troubleshoot RMS.</li> <li>When trace is invoked with no options specified, it returns the current settings for all RMS trace flags.</li> </ul>                                               |

- When trace is invoked with the flag and on, off options, it changes the setting of the trace point specified in the flag option.
- Changes made with the trace command take effect immediately.
- These trace flags are supported by RMS:

**Table 9-24: Trace flags**

| Flag                     | Description                                                                                                                                                |
|--------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Add_Drop_Server          | Write a message to the log when a server is added or dropped.                                                                                              |
| Add_Drop_Trigger         | Write a message to the log when a trigger is added or dropped.                                                                                             |
| Client_Connection        | Display information about a connection when a client initially connects to the RMS.                                                                        |
| Configuration            | Write a trace message to the log every time an RMS configuration parameter is changed.                                                                     |
| Filter_Conn              | Writes a trace message to the log when a connection is filtered.                                                                                           |
| Monitoring               | Add trace messages to the RMS at each step of the monitoring cycle, and write a message before monitoring each server.                                     |
| Network_Connection       | Add trace messages to the RMS whenever a connection to a server is created. Include all connection information (except the password) in the trace message. |
| RMS_Command              | Write every command received by the RMS to the error log.                                                                                                  |
| Server_Command           | Write every command sent to a monitored server by the RMS to the error log.                                                                                |
| Shutdown_Server          | Write a message to the log when the server is shut down.                                                                                                   |
| Start_Stop_Heartbeat     | Write a message to the log when a heartbeat is started or stopped.                                                                                         |
| Startup                  | Add trace messages to the RMS at each step of the start-up process.                                                                                        |
| Status_Change            | Display server and component result description when status changes.                                                                                       |
| Suspend_Resume_Component | Write a message to the log when a component is suspended or resumed.                                                                                       |
| Trigger_Execution        | Display message stating that event trigger was executed.                                                                                                   |

See also

configure RMS, get version, resume Replication Agent, suspend Replication Agent



# Acronyms and Abbreviations

This appendix lists acronyms and abbreviations that are used in the Replication Server documentation or that you may encounter in Replication Server messages. You can find definitions for many terms in the glossary of the *Replication Server Administration Guide Volume 2*.

**Table A-1: List of acronyms**

| Acronym | Stands for                                  |
|---------|---------------------------------------------|
| APC     | Asynchronous Procedure Call                 |
| API     | Application Program Interface               |
| BM      | Bitmap                                      |
| C/SI    | Client/Server Interfaces                    |
| CM      | Connection Manager                          |
| dAIO    | Asynchronous I/O Daemon                     |
| dALARM  | Alarm Daemon                                |
| DBO     | Database Owner                              |
| dCM     | Connection Manager Daemon                   |
| DDL     | Data Definition Language                    |
| DIST    | Distributor                                 |
| DML     | Data Manipulation Language                  |
| dREC    | Recovery Daemon                             |
| DSI     | Data Server Interface                       |
| dSUB    | Subscription Retry Daemon                   |
| ELM     | Exceptions Log Manager                      |
| ERSSD   | Embedded Replication Server System Database |
| EXC     | Exception                                   |
| EXEC    | Executor                                    |
| FSTR    | Function String                             |
| HDS     | Heterogeneous datatype support              |
| HTS     | Hash Table                                  |
| LAN     | Local Area Network                          |
| LL      | Linked List                                 |
| LTI     | Log Transfer Interface                      |

---

| Acronym   | Stands for                                                 |
|-----------|------------------------------------------------------------|
| LTL       | Log Transfer Language                                      |
| MD        | Message Delivery                                           |
| MEM       | Memory Management                                          |
| MP        | Multiprocessor                                             |
| MSA       | Multi-site Availability                                    |
| NRM       | Normalization                                              |
| OQID      | Origin Queue ID                                            |
| PDS       | Primary Data Server                                        |
| PRS       | Primary Replication Server                                 |
| PRS       | Parser                                                     |
| QID       | Queue ID                                                   |
| RA        | Replication Agent                                          |
| RCL       | Replication Command Language                               |
| RDS       | Replicate Data Server                                      |
| REP AGENT | RepAgent thread, the Replication Agent for Adaptive Server |
| RM        | Replication Manager                                        |
| RMI       | Remote Method Invocation                                   |
| RMP       | Replication Manager plug-in                                |
| RMS       | Replication Monitoring Services                            |
| RPC       | Remote Procedure Call                                      |
| RRS       | Replicate Replication Server                               |
| RS        | Replication Server                                         |
| RSI       | Replication Server Interface                               |
| RSP       | Replicated Stored Procedure                                |
| RSA       | Replication System Administrator                           |
| RSI       | Replication Server Interface                               |
| RSSD      | Replication Server System Database                         |
| SA        | System Administrator                                       |
| SP        | Stored Procedure                                           |
| SQM       | Stable Queue Manager                                       |
| SQT       | Stable Queue Transaction Interface                         |
| SRE       | Subscription Resolution Engine                             |
| STS       | System Table Services                                      |
| SUB       | Subscription                                               |
| TD        | Transaction Delivery                                       |
| TDS       | Tabular Data Stream™                                       |
| WAN       | Wide Area Network                                          |

# Replication Server Design Limits

This appendix lists the maximum and minimum parameters and values for various replication system objects.

## Replication Server limits

The variable *For\_Life\_Of* refers to the total number of objects that you can create for a Replication Server, regardless of whether or not any of them are dropped. For example, if the limit is 100,000, when you create 100,000, you cannot create any more, even if you drop some or all of them. The *For\_Life\_Of* count and limit remain in effect as long as the replication software remains installed. You can restart the *For\_Life\_Of* count by deleting the entire server from a system and then reinstalling it.

**Table B-1: Replication Server limits**

| Type of object                                                | Number                            |
|---------------------------------------------------------------|-----------------------------------|
| Replication definitions <i>For_Life_Of</i> Replication Server | $2^{24}$ (16,777,216)             |
| Users <i>For_Life_Of</i> Replication Server                   | $2^{24}$ (16,777,216)             |
| Reject log commands <i>For_Life_Of</i> Replication Server     | $2^{32} - 2^{29}$ (3,758,096,384) |
| Reject log transactions <i>For_Life_Of</i> Replication Server | $2^{31}$ (2,147,483,648)          |
| Replication Servers per ID Server                             | $2^{24}$ (16,777,216)             |
| Databases per ID Server                                       | $2^{24}$ (16,777,216)             |
| Databases per Replication Server                              | $2^{24}$ (16,777,216)             |
| Partitions <i>For_Life_Of</i> Replication Server              | $2^{16}$ (65,536)                 |
| Minimum size for initial partition (to install RS)            | 20MB                              |
| Minimum size for additional partitions                        | 1MB                               |
| Maximum partition size                                        | 1TB                               |
| Stable queues per Replication Server                          | $2^{64}$                          |
| Subscriptions <i>For_Life_Of</i> Replication Server           | $2^{31}$                          |

| Type of object                                         | Number                |
|--------------------------------------------------------|-----------------------|
| Connections per Replication Server:                    |                       |
| • Incoming (Replication Agent, DIST, RS, user)         | $2^{24}$ minus 1      |
| • Outgoing (DSI, route)                                | $2^{32}$ minus 1      |
| Function strings <i>For_Life_Of</i> Replication Server | $2^{24}$ (16,777,216) |
| Error classes <i>For_Life_Of</i> Replication Server    | $2^{24}$ (16,777,216) |

## Platform-specific limits

Certain limits specific to platform operating systems, such as number of file descriptors per process, may affect Replication Server operation. For specific limits, see the release bulletin for your platform.

## Replication definition and subscription limits

**Table B-2: Replication definition limits**

| Type of object                                 | Number                                                     |
|------------------------------------------------|------------------------------------------------------------|
| Columns per replication definition             | Limited to 1024                                            |
| Primary key columns per replication definition | Limited to columns specified in the replication definition |
| Searchable columns per replication definition  | Limited to columns specified in the replication definition |
| Subscriptions per replication definition       | Unlimited                                                  |
| String width for subscription where clause     | Limited to 255 bytes                                       |

## Function string limits

**Table B-3: Function string limits**

| Type of object                                                                     | Number           |
|------------------------------------------------------------------------------------|------------------|
| Function strings per function-string class                                         | Unlimited        |
| Bytes per language-type function string template                                   | 64K              |
| Bytes per language-type function string template after variable value substitution | 64K minus 1 byte |
| Embedded variables per function string                                             | Unlimited        |

| Type of object                                   | Number |
|--------------------------------------------------|--------|
| User variables in function string input template | 1024   |

## Programming limits and parameters

*Table B-4: Programming limits and parameters*

| Type of object                                          | Number                    |
|---------------------------------------------------------|---------------------------|
| Number of terms in subscription where clause            | Unlimited                 |
| Transactions in a DSI transaction group                 | 20                        |
| Source commands in a DSI command batch                  | 50                        |
| Bytes for every command processed by Replication Server | 16K                       |
| Action assignments per error class                      | $2^{31}$ (2,1474,836,448) |
| Maximum message size written to stable queue            | Unlimited                 |



# RMS Server and Component States

This appendix provides information about Replication Monitoring Services (RMS) server and component states.

RMS monitors the servers and components in a replication environment, and provides information that helps you troubleshoot problems. You can monitor the replication environment either by actively viewing information about the state of servers and components, or by being notified when particular events occur.

The status of any server or component object consists of:

- An integer state value
- A list of strings that describe the reason for the current state

For example, a Replication Server can be in “Suspect” state because two different connections are “Suspended.”

The integer state value is different for each monitored object, and the descriptions can be localized.

## Server states

RMS monitors the following servers:

- Replication Server
- Adaptive Server Enterprise
- IQ
- DirectConnect
- Open Server
- Replication Agent

- RMS

Table C-1 provides a summary of server states. Details are in the sections that follow.

**Table C-1: Summary of server states**

| Server type                                        | Value | Description |
|----------------------------------------------------|-------|-------------|
| Replication Server                                 | 5     | ACTIVE      |
|                                                    | 3     | UNKNOWN     |
|                                                    | 0     | DOWN        |
|                                                    | 4     | SUSPECT     |
|                                                    | 6     | HIBERNATE   |
|                                                    | 7     | REBUILDING  |
|                                                    | 8     | RECOVERY    |
|                                                    | 9     | STANDALONE  |
|                                                    | 1     | TIMEOUT     |
|                                                    | 10    | QUIESCE     |
| Adaptive Server Enterprise (ASE)                   | 5     | ACTIVE      |
|                                                    | 3     | UNKNOWN     |
|                                                    | 0     | DOWN        |
|                                                    | 4     | SUSPECT     |
|                                                    | 1     | TIMEOUT     |
| IQ                                                 | 5     | ACTIVE      |
|                                                    | 3     | UNKNOWN     |
|                                                    | 0     | DOWN        |
|                                                    | 1     | TIMEOUT     |
| DirectConnect                                      | 5     | ACTIVE      |
|                                                    | 3     | UNKNOWN     |
|                                                    | 0     | DOWN        |
|                                                    | 1     | TIMEOUT     |
| Replication Agent / Mirror Replication Agent (MRA) | 5     | ACTIVE      |
|                                                    | 3     | UNKNOWN     |
|                                                    | 0     | DOWN        |
|                                                    | 1     | TIMEOUT     |
|                                                    | 6     | ADMIN       |
| RMS                                                | 5     | ACTIVE      |
|                                                    | 3     | UNKNOWN     |
|                                                    | 4     | SUSPECT     |
|                                                    | 0     | DOWN        |
|                                                    | 1     | TIMEOUT     |

| Server type | Value | Description |
|-------------|-------|-------------|
| Open Server | 5     | ACTIVE      |
|             | 3     | UNKNOWN     |
|             | 0     | DOWN        |
|             | 1     | TIMEOUT     |

Replication Server

The RMS determines the state of a Replication Server by:

- 1    Testing the connection to the Replication Server
- 2    Testing the connection to the server that contains the RSSD
- 3    Determining the health of the Replication Server
- 4    Determining the status of the server’s connections, routes, and queues

The Replication Server can be in more then one state, but the RMS returns only one state. For example, the status of the server can be both HIBERNATE and QUIESCE.

Table C-2 describes the Replication Server states.

Table C-2: Replication Server states

| State type | Value | Meaning    | Description                                                                                                                                          |
|------------|-------|------------|------------------------------------------------------------------------------------------------------------------------------------------------------|
| Normal     | 5     | ACTIVE     | The Replication Server is running and actively replicating data.                                                                                     |
|            | 10    | QUIESCE    | The Replication Server is running but is not currently replicating data.                                                                             |
| Warning    | 3     | UNKNOWN    | The initial value before the actual state has been determined. UNKNOWN can also indicate that the server is not part of the replication environment. |
|            | 4     | SUSPECT    | At least one of the Replication Server connections, routes, or queues is down.                                                                       |
|            | 6     | HIBERNATE  | The Replication Server is in hibernation mode. This state is returned by the admin health command.                                                   |
|            | 7     | REBUILDING | The Replication Server is rebuilding queues. This state is returned by the admin health command.                                                     |
|            | 8     | RECOVERY   | The Replication Server is in standalone mode and is rebuilding queues. This state is returned by the admin health command.                           |
|            | 9     | STANDALONE | The Replication Server is in standalone mode. This state is returned by admin health command.                                                        |

| State type | Value | Meaning   | Description                                                                                                                                                       |
|------------|-------|-----------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Error      | 0     | DOWN      | The RMS cannot connect to the Replication Server or the server that contains the RSSD. The server state is also set to DOWN if the user or password is incorrect. |
|            | 1     | TIMED OUT | The attempt to connect to the Replication Server, or the server that contains the RSSD, timed out. This indicates a server that has stopped responding.           |

## Adaptive Server Enterprise

The RMS determines the state of an Adaptive Server Enterprise by:

- 1 Testing the connection to the Adaptive Server
- 2 Determining the state of the Adaptive Server's RepAgent threads

RMS tests only the RepAgent thread of databases that participate in replication, and not all databases in Adaptive Server. Databases that are offline are not queried.

Table C-3 describes the Adaptive Server states.

**Table C-3: Adaptive Server states**

| State type | Value | Meaning   | Description                                                                                                                                                                                |
|------------|-------|-----------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Normal     | 5     | ACTIVE    | Successfully connected to the Adaptive Server and all RepAgent threads for connections within this environment are enabled and started.                                                    |
| Warning    | 3     | UNKNOWN   | The initial value before the actual state has been determined. This also indicates that the server is not part of the replication environment.                                             |
|            | 4     | SUSPECT   | Set when the state of the RepAgent threads is checked. If any of the threads for the connections within this environment are disabled or stopped, then the server state is set to SUSPECT. |
| Error      | 0     | DOWN      | The RMS cannot connect to the Adaptive Server. The server state is also set to DOWN if the user or password is incorrect.                                                                  |
|            | 1     | TIMED OUT | The attempt to connect to the Adaptive Server timed out. Indicates that a server has stopped responding.                                                                                   |

## IQ

IQ uses TDS to participate in a replication environment. The RMS uses jConnect to connect to the server. The IQ server contains internal RepAgent threads. The RMS tests the connection to the IQ server to determine its availability.

Table C-4 describes the IQ states.

**Table C-4: IQ server states**

| State type | Value | Meaning   | Description                                                                                                                               |
|------------|-------|-----------|-------------------------------------------------------------------------------------------------------------------------------------------|
| Normal     | 5     | ACTIVE    | Successfully connected to the IQ server.                                                                                                  |
| Warning    | 3     | UNKNOWN   | The initial value before the actual state has been determined. Also indicates that the server is not part of the replication environment. |
| Error      | 0     | DOWN      | The RMS cannot connect to the IQ server. The server state is also set to DOWN if the user or password is incorrect.                       |
|            | 1     | TIMED OUT | The attempt to connect to the IQ server timed out. Indicates that a server has stopped responding.                                        |

## DirectConnect

The RMS determines the state of DirectConnect by:

- 1 Testing the connection to DirectConnect
- 2 Testing the connection from DirectConnect to the back-end data server

Table C-5 describes the DirectConnect server states.

**Table C-5: DirectConnect server states**

| State type | Value | Meaning   | Description                                                                                                                                                                                                                     |
|------------|-------|-----------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Normal     | 5     | ACTIVE    | RMS successfully connected to the DirectConnect, and DirectConnect can connect to the back-end data server.                                                                                                                     |
| Warning    | 3     | UNKNOWN   | The initial value before the actual state has been determined. Also indicates that the agent is not part of the replication environment.                                                                                        |
| Error      | 0     | DOWN      | The RMS cannot connect to the DirectConnect. The server state is also set to DOWN if the user or password is incorrect. Additionally, the state is set to DOWN if the DirectConnect cannot connect to the back-end data server. |
|            | 1     | TIMED OUT | The attempt to connect to the DirectConnect timed out. Indicates a server that has stopped responding.                                                                                                                          |

## Open Server

RMS tests the connection to the Open Server. Table C-6 describes the Open Server states.

**Table C-6: Open Server states**

| State type | Value | Meaning   | Description                                                                                                                               |
|------------|-------|-----------|-------------------------------------------------------------------------------------------------------------------------------------------|
| Normal     | 5     | ACTIVE    | Successfully connected to the Open Server.                                                                                                |
| Warning    | 3     | UNKNOWN   | The initial value before the actual state has been determined. Also indicates that the server is not part of the replication environment. |
| Error      | 0     | DOWN      | The RMS is unable to connect to the Open Server. DOWN can also indicate that a user or password is incorrect.                             |
|            | 1     | TIMED OUT | The attempt to connect to the Open Server timed out. This indicates that the server has stopped responding.                               |

## Replication Agent

The RMS determines the state of a Replication Agent by:

- 1 Testing the connection to the Replication Agent
- 2 Determining if the agent is in “administration” or “replicating” mode

Table C-7 describes the Replication Agent—including MRA and MRO—states.

**Table C-7: Replication Agent (MRA/MRO) states**

| State type | Value | Meaning   | Description                                                                                                                                                                                    |
|------------|-------|-----------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Normal     | 5     | ACTIVE    | Successfully connected to the Replication Agent. The agent is in the replicating state. This state is returned by the <code>ra_status</code> command.                                          |
| Warning    | 3     | UNKNOWN   | The initial value before the actual state has been determined. Also indicates that the agent is not part of the replication environment.                                                       |
|            | 6     | ADMIN     | Successfully connected to the Replication Agent. The agent is in the administration state and is not currently replicating data. This state is returned by the <code>ra_status</code> command. |
| Error      | 0     | DOWN      | The RMS cannot connect to the Replication Agent. The agent state is also set to DOWN if the user or password is incorrect.                                                                     |
|            | 1     | TIMED OUT | The attempt to connect to the Replication Agent timed out. Indicates that an agent has stopped responding.                                                                                     |

## RMS

Central RMS tests the connection to the Remote RMS. Table C-8 describes the RMS states.

**Table C-8: RMS states**

| State type | Value | Meaning   | Description                                                                                                                                   |
|------------|-------|-----------|-----------------------------------------------------------------------------------------------------------------------------------------------|
| Normal     | 5     | ACTIVE    | Central RMS successfully connected to the Remote RMS.                                                                                         |
| Warning    | 3     | UNKNOWN   | The initial value before the actual state has been determined. Also indicates that the Remote RMS is not part of the replication environment. |
|            | 4     | SUSPECT   | Indicates that a Remote RMS server or component is DOWN or SUSPENDED.                                                                         |
| Error      | 0     | DOWN      | The Central RMS is unable to connect to the Remote RMS. DOWN can also indicate that a user or password is incorrect.                          |
|            | 1     | TIMED OUT | The attempt to connect to the Remote RMS timed out. This indicates that the server has stopped responding.                                    |

## Component states

RMS monitors the following components in a Replication Server:

- Connections
- Logical Connections
- Queues
- Routes
- Partitions
- RepAgent threads

Table C-9 provides a summary of component states. Details are in the sections that follow.

**Table C-9: Summary of component states**

| Component type         | Value | Description   |
|------------------------|-------|---------------|
| Connection             | 5     | ACTIVE        |
|                        | 2     | SUSPENDED     |
|                        | 3     | UNKNOWN       |
| Logical Connection     | 5     | ACTIVE        |
|                        | 2     | SUSPENDED     |
|                        | 3     | UNKNOWN       |
| Queue                  | 5     | ACTIVE        |
|                        | 2     | SUSPENDED     |
|                        | 6     | LOSS_DETECTED |
| Route                  | 5     | ACTIVE        |
|                        | 2     | SUSPENDED     |
|                        | 3     | UNKNOWN       |
| Partition              | 6     | ONLINE        |
|                        | 7     | OFFLINE       |
|                        | 8     | DROPPED       |
| RepAgent threads (ASE) | 6     | DISABLED      |
|                        | 7     | SUSPENDED     |
|                        | 8     | ACTIVE        |

## Connections

The RMS monitors the state of a Replication Server's database connections. Database connections include two parts, the RepAgent and the DSI. The state of the Replication Server threads determines the state of the connection. The RMS executes the admin who command to retrieve the state of the threads.

The RMS returns the state of the DSI and RepAgent separately. Client applications such as the Replication Manager Java plug-in may consolidate the state of the threads (and the state of the actual RepAgent) when displaying the status of the connection. Table C-10 describes the connection states.

**Table C-10: Connection states**

| State type | Value | Meaning   | Description                                                                  |
|------------|-------|-----------|------------------------------------------------------------------------------|
| Normal     | 5     | ACTIVE    | The Replication Server DSI or RepAgent thread is not DOWN and not SUSPENDED. |
| Error      | 2     | SUSPENDED | The Replication Server DSI or RepAgent thread is DOWN or SUSPENDED.          |

| State type | Value | Meaning | Description                                                                       |
|------------|-------|---------|-----------------------------------------------------------------------------------|
| Warning    | 3     | UNKNOWN | The RepAgent for a primary connection is not part of the replication environment. |

## Logical Connections

The RMS monitors the state of a Replication Server's logical connections. A logical connection consists of a pair of physical connections that are configured in a warm-standby environment. The source of the replication data is the active database while the target of replication is the standby database. Monitoring a logical connection requires the RMS to determine the state of the Replication Agent thread for the active connection and the state of the DSI for the standby connection.

RMS reports the status of the active connection's Replication Agent thread separately from the state of the standby connection's DSI thread. Each thread is reported in a separate row in the result set. Table C-11 describes the logical connection states.

**Table C-11: Logical Connection states**

| State type | Value | Meaning   | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
|------------|-------|-----------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Normal     | 5     | ACTIVE    | The Replication Agent for the active physical connection and the DSI thread for the standby physical connection are both active.                                                                                                                                                                                                                                                                                                                             |
| Error      | 2     | SUSPENDED | The logical connection can be suspended for the following reasons: <ul style="list-style-type: none"><li>• The active or standby physical connection is not defined for the logical connection.</li><li>• The Replication Agent thread for the active connection is suspended.</li><li>• The DSI thread for the standby connection is suspended.</li><li>• The logical connection is in the process of switching the active and standby databases.</li></ul> |
| Warning    | 3     | UNKNOWN   | The Replication Agent thread for the active connection is unknown, or the DSI thread for the standby connection is unknown.                                                                                                                                                                                                                                                                                                                                  |

## Queues

The RMS monitors the state of Replication Server queues. Queue states are stored in the RSSD. The stored procedure `rma_queue` returns the name of the queue, whether the queue is up or down, and if any data loss is detected. Table C-12 describes the queue states.

**Table C-12: Queue states**

| State type | Value | Meaning       | Description                                                                                          |
|------------|-------|---------------|------------------------------------------------------------------------------------------------------|
| Normal     | 5     | ACTIVE (UP)   | The queue is not suspended.                                                                          |
| Error      | 2     | SUSPENDED     | The queue is suspended.                                                                              |
| Warning    | 6     | LOSS_DETECTED | Data loss has been detected in the queue. The state is set to LOSS DETECTED only if the queue is UP. |

## Routes

The RMS monitors the state of Replication Server routes, and determines the state of a route by:

- 1 Checking the state of the route at both its origin and destination
- 2 Querying the RSSD

The RMS uses the information to identify whether the route is UP or DOWN, and to identify the reason. Table C-13 describes the route states.

**Table C-13: Route states**

| State type | Value | Meaning   | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
|------------|-------|-----------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Normal     | 5     | ACTIVE    | The route is open and data can pass from the origin to the destination Replication Server.                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
| Error      | 2     | SUSPENDED | The route is unavailable and data cannot pass between the Replication Servers. The description provides the reason for the suspended route; for example: <ul style="list-style-type: none"> <li>• The route encountered an internal error.</li> <li>• The route is being created.</li> <li>• The route is suspended.</li> <li>• The route encountered an error at the destination.</li> <li>• The route is being dropped.</li> <li>• The route is being dropped with NOWAIT.</li> <li>• An indirect route is being changed to a direct route.</li> </ul> |
| Warning    | 3     | UNKNOWN   | The destination Replication Server is not part of the replication environment.                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |

## Partitions

The RMS monitors Replication Server partitions. The Replication Server command `admin disk_space` returns the state of a partition. Table C-14 describes the partition states.

**Table C-14: Partition states**

| State type | Value | Meaning | Description                                                      |
|------------|-------|---------|------------------------------------------------------------------|
| Normal     | 6     | ONLINE  | The partition device is available and functioning normally.      |
| Error      | 7     | OFFLINE | The device cannot be found.                                      |
|            | 8     | DROPPED | The device has been dropped, but some queues are still using it. |

## RepAgent threads

The RMS monitors Adaptive Server Enterprise RepAgent threads. `sp_help_rep_agent` determines the state of RepAgent threads for each database that participates in replication. Table C-15 describes the RepAgent thread states.

**Table C-15: RepAgent thread states**

| State type | Value | Meaning   | Description                                 |
|------------|-------|-----------|---------------------------------------------|
| Normal     | 8     | ACTIVE    | The RepAgent thread is enabled and started. |
| Error      | 6     | DISABLED  | The RepAgent thread is not enabled.         |
|            | 7     | SUSPENDED | The RepAgent thread is enabled but stopped. |

## Event Trigger Arguments

This appendix provides information about Replication Monitoring Services (RMS) event trigger arguments. Event trigger arguments contain information about the execution of a certain event, such as event name, date and time the event occurred, and name of the RMS that executed the event script. RMS passes these arguments whenever an event trigger is executed.

### Connection status event arguments

Table D-1 describes the arguments of a connection status event. There are two types of connections—inbound and outbound. An inbound connection is a connection to a Replication Server from a database via a Replication Agent. An outbound connection is a connection from a Replication Server to a database.

**Table D-1: Connection status event trigger arguments**

| Argument           | Description                                                                                                                                                                                         |
|--------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>connection</i>  | Keyword identifying the event as a connection status event.                                                                                                                                         |
| <i>date_time</i>   | The date and time the event occurred.<br>Format: Month Day Year HH:MM:SS:TTTMeridian                                                                                                                |
| <i>rms</i>         | The name of the RMS that executed the event script.                                                                                                                                                 |
| <i>object_id</i>   | The server where the event occurred.                                                                                                                                                                |
| <i>source_type</i> | The type of server that raised the event. Values are: <ul style="list-style-type: none"><li>• repserver</li><li>• database</li></ul>                                                                |
| <i>source_name</i> | The name of the Replication Server or data server that raised the event.                                                                                                                            |
| <i>ra_type</i>     | Type of Replication Agent. Values are: <ul style="list-style-type: none"><li>• rep agent</li><li>• rep agent thread</li><li>• dbltm</li><li>• Empty string (") if connection is outbound.</li></ul> |
| <i>ra_name</i>     | Replication Agent name. Empty string (") if connection is outbound.                                                                                                                                 |
| <i>dest_type</i>   | The destination server type. Values are: <ul style="list-style-type: none"><li>• repserver</li><li>• database</li></ul>                                                                             |
| <i>dest_name</i>   | Destination server name.                                                                                                                                                                            |
| <i>state</i>       | The new connection status.                                                                                                                                                                          |

## Partition status event arguments

Table D-2 describes the arguments of a partition status event.

**Table D-2: Partition status event trigger arguments**

| Argument         | Description                                                                          |
|------------------|--------------------------------------------------------------------------------------|
| <i>partition</i> | The keyword that identifies the event as a partition status event.                   |
| <i>date_time</i> | The date and time the event occurred.<br>Format: Month Day Year HH:MM:SS:TTTMeridian |
| <i>rms</i>       | The name of the RMS that executed the event script.                                  |
| <i>object_id</i> | The name of the Replication Server that owns the partition.                          |
| <i>part_name</i> | The logical name of the stable device.                                               |
| <i>state</i>     | The new partition status.                                                            |

## Route status event arguments

Table D-3 describes the arguments of a route status event.

**Table D-3: Route status event trigger arguments**

| Argument           | Description                                                                                                                                                                                  |
|--------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>route</i>       | The keyword that identifies the event as a route status event.                                                                                                                               |
| <i>date_time</i>   | The date and time the event occurred.<br>Format: Month Day Year HH:MM:SS:TTTMeridian                                                                                                         |
| <i>rms</i>         | The name of the RMS that executed the event script.                                                                                                                                          |
| <i>object_id</i>   | The server where the event occurred.                                                                                                                                                         |
| <i>repserver</i>   | The keyword that identifies the origin server of the route as a Replication Server.                                                                                                          |
| <i>server_name</i> | The name of the origin Replication Server.                                                                                                                                                   |
| <i>thru_type</i>   | The type of intermediate server. Values are: <ul style="list-style-type: none"><li>• <i>repserver</i></li><li>• Empty string (") if there is no intermediate server for the route.</li></ul> |
| <i>thru_name</i>   | The name of the intermediate Replication Server. Empty string (") if there is no intermediate server for the route.                                                                          |
| <i>repserver</i>   | The keyword that identifies the destination server of the route as a Replication Server.                                                                                                     |
| <i>dest_name</i>   | The name of the destination Replication Server.                                                                                                                                              |
| <i>state</i>       | The new route status.                                                                                                                                                                        |

## Server status event arguments

Table D-4 describes the arguments of a server status event.

**Table D-4: Server status event trigger arguments**

| Argument         | Description                                                                          |
|------------------|--------------------------------------------------------------------------------------|
| <i>server</i>    | The keyword used to identify an event as a server status event.                      |
| <i>date_time</i> | The date and time the event occurred.<br>Format: Month Day Year HH:MM:SS:TTTMeridian |
| <i>rms</i>       | The name of the RMS that executed the event script.                                  |
| <i>object_id</i> | The server where the event occurred.                                                 |
| <i>old_state</i> | The server status before the event occurred.                                         |
| <i>new_state</i> | The server status after the event occurred.                                          |
| <i>reason</i>    | The reason the event occurred.                                                       |

## Database connection latency event arguments

Table D-5 describes the arguments of a database connection latency event.

**Table D-5: Database connection latency event arguments**

| Argument                      | Description                                                                                                                                                 |
|-------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>latency</i>                | The keyword that identifies the event as a latency event.                                                                                                   |
| <i>date_time</i>              | The date and time the event occurred.<br>Format: Month Day Year HH:MM:SS:TTTMeridian                                                                        |
| <i>rms</i>                    | The name of the RMS that executed the event script.                                                                                                         |
| <i>object_id</i>              | The name of the Replication Server for which you are monitoring latency.                                                                                    |
| <i>origin_dbname</i>          | Name of data server and database from which the transaction was sent.                                                                                       |
| <i>dest_dbname</i>            | Name of the data server and database to which the transaction was sent.                                                                                     |
| <i>delta_diff</i>             | The difference, in seconds, between the time the transaction was committed at the primary database and the time it was committed at the replicate database. |
| <i>last_commit_time</i>       | The date and time of the last commit at the destination database.<br>Format: Month Day Year HH:MM:SS:TTTMeridian                                            |
| <i>secs_since_last_commit</i> | The time elapsed, in seconds, since the last commit.                                                                                                        |
| <i>dest_type</i>              | The type of database connection. Values are: <ul style="list-style-type: none"><li>• Primary and Replicate</li><li>• Replicate Only</li></ul>               |
| <i>reason</i>                 | The reason the event occurred.                                                                                                                              |

## Queue latency event arguments

Table D-6 describes the arguments of a queue latency event.

**Table D-6: Queue latency event arguments**

| Argument               | Description                                                                          |
|------------------------|--------------------------------------------------------------------------------------|
| <i>queue_latency</i>   | The keyword that identifies the event as a queue latency status event.               |
| <i>date_time</i>       | The date and time the event occurred.<br>Format: Month Day Year HH:MM:SS:TTTMeridian |
| <i>rms</i>             | The name of the RMS that executed the event script.                                  |
| <i>object_id</i>       | The name of the Replication Server that owns the queue.                              |
| <i>log_name</i>        | The logical name of the queue.                                                       |
| <i>phys_name</i>       | The physical name of the queue.                                                      |
| <i>latency_in_secs</i> | The time, in seconds, the first block has remained in a queue.                       |

## Partition and queue size threshold event arguments

Table D-7 describes the arguments of a partition and a queue size threshold event.

**Table D-7: Partition and queue size threshold event arguments**

| Argument           | Description                                                                                                                     |
|--------------------|---------------------------------------------------------------------------------------------------------------------------------|
| <i>threshold</i>   | The keyword that identifies the event as a partition threshold or queue threshold event.                                        |
| <i>date_time</i>   | The date and time the event occurred.<br>Format: Month Day Year HH:MM:SS:TTTMeridian                                            |
| <i>rms</i>         | The name of the RMS that executed the event script.                                                                             |
| <i>object_id</i>   | The name of the Replication Server that owns the partition or queue.                                                            |
| <i>log_name</i>    | The logical name of the partition or queue.                                                                                     |
| <i>phys_name</i>   | The physical name of the partition or queue.                                                                                    |
| <i>size</i>        | Indicates the area, in percentage, used by the partition or the size, in megabytes, of the queue.                               |
| <i>object_type</i> | Identifies the threshold event type. Values are: <ul style="list-style-type: none"> <li>• Partition</li> <li>• Queue</li> </ul> |



# Index

## Symbols

@ xxii  
– xxii

## A

abbreviations, defined 773  
abort switch command 50  
acronyms, defined 773  
activate subscription command 51  
Adaptive Server  
    commands 523  
    RMS states 783  
    support for 40  
    system procedures 523  
add partition command 54  
add server command (RMS) 732  
add trigger command (RMS) 729  
**admin config** command 54  
admin disk\_space command 58  
admin echo command 59  
admin get\_generation command 60  
admin health command 61  
admin log\_name command 62  
admin logical\_status command 63  
admin pid command 65  
admin quiesce\_check command 66  
admin quiesce\_force\_rsi command 67  
admin rssid\_name command 68  
admin schedule command 69  
admin security\_property command 70  
admin security\_setting command 71  
admin set\_log\_name command 72  
**admin show\_connection\_profiles** command 72  
admin show\_connections command 76  
admin show\_function\_classes command 78  
admin show\_route\_versions command 79  
admin show\_site\_version command 80  
admin sqm\_readers command 81  
**admin stats** command 83–87  
    report usage 86  
    statistics collectors 85  
admin stats, backlog command 87  
**admin stats, backlog** command  
    report usage 88  
admin stats, bps command 91  
**admin stats, cancel** command 89  
admin stats, cps command 91  
**admin stats, md** command 89  
admin stats, mem command 89  
admin stats, mem\_in\_use command 89  
admin stats, reset command 90  
admin stats, status command 91  
admin stats, tps command 91  
admin time command 94  
admin translate command 95  
admin verify\_repserver\_cmd 97  
admin version command 99  
admin who command 100, 116  
**admin who** command  
    filters output 101  
    info column, increasing size of 100  
admin who, dsi command 100, 116  
admin who, rsi command 100, 116  
admin who, sqm command 100, 116  
admin who, sqt command 100, 116  
admin who\_is\_down command 117  
**admin who\_is\_down** command  
    info column, increasing size of 117  
admin who\_is\_up command 118  
**admin who\_is\_up** command  
    info column, increasing size of 118  
allow connections command 119  
**alter applied function replication definition** command  
    119–122  
alter connection command 123, 124, 145  
    changing ERSSD password 145  
**alter connector** command 146

- alter database replication definition** command 148
- alter error class** command 150
- alter function command 153
- alter function replication definition command 155
- alter function string class command 160
- alter function string command 159
- alter logical connection command 162
- alter partition command 166
- alter queue command 168
- alter replication definition** command 170–180
- alter request function replication definition command 180
- alter route command 184
- alter schedule command 192
- alter user command 193
- alter user command, for ERSSD 193
- always\_replicate clause 307
- approximate numeric (floating point) datatypes
  - float 24
  - real 24
- articles
  - commands for 5
  - dropping 343
- assign action command 194
- asynchronous command
  - cancelling 89
- asynchronous procedures 258
- atomic materialization
  - description of 7
  - summary of commands for 7
- autocorrection
  - and replicating minimal columns 306
  - example of 382
  - setting 382
  - system table for 704

## B

- batch configuration parameter 124
- batch\_begin configuration parameter 124
- bigdatetime datatype 26
- bigint datatype 22
- bigtime datatype 26
- binary datatypes
  - binary 29
  - image 30

- rawobject 30
- rawobject in row 30
- rawobject large in row 30
- varbinary 29
- bit datatype 31
- block\_size configuration parameter 206
- bulk copy-in support
  - Data Server Interface (DSI), implementation in 143
  - multi-statement transactions, support for 143
- bulk materialization
  - defining subscriptions 335
  - description of 7
  - setting subscription status to valid 458
  - summary of commands for 8

## C

- canonic\_type 673
- cascading connection
  - connection stack, list of 386
  - current server, display 387
  - terminating connection 342
- case in RCL commands xxi
- character datatypes
  - char 24
  - text 24
  - varchar 24
- character sets
  - conversion 41, 127, 656
  - Replication Server parameter 639
  - retrieval of 483
  - rs\_subcmp parameter 646
  - supported 40
- check publication command 199
- check subscription command 201
- class-level translations 142
- cluster instance name 543
- cm\_max\_connections configuration parameter 207
- column-level translations 171, 177, 298, 305
- columns, system table for 664
- column-size
  - supported 42
- command batching
  - rs\_batch\_end 465

- rs\_batch\_start 467
- use\_batch\_markers 140
- command\_retry configuration parameter 124
- commands
  - abort switch 50
  - active subscription 51
  - add partition 54
  - admin config** 54
  - admin disk\_space 58
  - admin echo 59
  - admin get\_generation 60
  - admin health 61
  - admin log\_name 62
  - admin logical\_status 63
  - admin pid 65
  - admin quiesce\_check 66
  - admin quiesce\_force\_rsi 67
  - admin rssid\_name 68
  - admin schedule 69
  - admin security\_property 70
  - admin security\_setting 71
  - admin set\_log\_name 72
  - admin show\_connection\_profiles** 72
  - admin show\_connections 76
  - admin show\_function\_classes 78
  - admin show\_route\_versions 79
  - admin show\_site\_version 80
  - admin sqm\_readers 81
  - admin stats** 83–87
    - admin stats, backlog 87
    - admin stats, bps 91
    - admin stats, cancel** 89
    - admin stats, cps 91
    - admin stats, md** 89
    - admin stats, mem 89
    - admin stats, mem\_in\_use 89
    - admin stats, reset 90
    - admin stats, status 91
    - admin stats, tps 91
    - admin time 94
    - admin translate 95
    - admin verify\_repserver\_cmd 97
    - admin version 99
    - admin who 100
    - admin who\_is\_down 117
    - admin who\_is\_up 118
- allow connections 119
- alter applied function replication definition** 119–122
- alter connection 123
- alter connector class** 146
- alter database replication definition 148
- alter error class** 150
- alter function 153
- alter function replication definition 155
- alter function string 159
- alter function string class 160
- alter logical connection 162
- alter partition 166
- alter queue 168
- alter replication definition** 170–180
- alter request function replication definition 180
- alter route 184
- alter schedule 192
- alter user 193
- alter user, for ERSSD 193
- assign action 194
- cancelling, asynchronous 89
- check publication 199
- check subscription 201
- configure connection 204
- configure logical connection 205
- configure replication server 206
- configure route 221
- connect** 14, 221–223, 342, 387
- create applied function replication definition** 224–230
- create article 231
- create connection** 235
- create connection using profile** clause 248
- create connection using profile** clause 241
- create connectionusing profile** clause 45
- create database replication definition 248
- create error class 254
- create function 257
- create function replication definition 259
- create function string 265
- create function string class 279
- create logical connection 282
- create partition 284
- create publication 286
- create replication definition** 296–312

**create request function replication definition** 290–295  
 create route 313  
 create schedule 318  
 create subscription 322  
 create user 334  
 define subscription 335  
**disconnect** 14, 342, 387  
 drop article 343  
 drop connection 345  
 drop database replication definition 345  
 drop error class 347  
 drop function 348  
 drop function replication definition 349  
 drop function string 350  
 drop function string class 352  
 drop logical connection 353  
 drop partition 354  
 drop publication 355  
 drop replication definition 357  
 drop route 359  
 drop schedule 361  
 drop subscription 362  
 drop user 367  
 grant 368  
 ignore loss 369  
 move primary 370  
 rebuild queues 372  
 resume connection 374  
 resume distributor 377  
 resume log transfer 378  
 resume queue 379  
 resume route 380  
 revoke 381  
**set autocorrection** 382–384  
 set log recovery 385  
 set proxy 386  
**show connection** 342, 386, 387  
**show server** 342, 387  
 shutdown 388  
 suspend connection 389  
 suspend distributor 390  
 suspend log transfer 391  
 suspend route 392  
 switch active 393  
 sysadmin apply\_truncate\_table 394

sysadmin cdb 396  
 sysadmin drop\_queue 406  
 sysadmin dropdb 404  
 sysadmin dropldb 405  
 sysadmin dropprs 407  
 sysadmin dump\_file 408  
 sysadmin dump\_queue 409  
**sysadmin dump\_thread\_stack** 414  
**sysadmin dump\_tran** 416–419  
 sysadmin erssd 419  
 sysadmin fast\_route\_upgrade 422  
 sysadmin hibernate\_off 423  
 sysadmin hibernate\_on 425  
 sysadmin issue\_tickets 426  
 sysadmin log\_first\_tran 428  
 sysadmin purge\_all\_open 429  
 sysadmin purge\_first\_open 431  
 sysadmin purge\_route\_at\_replicate 433  
 sysadmin restore\_dsi\_saved\_segments 434  
 sysadmin set\_dsi\_generation 435  
 sysadmin site\_version 436  
 sysadmin skip\_bad\_repserver\_cmd 439  
 sysadmin sqm\_purge\_queue 441  
 sysadmin sqm\_unzap\_command 442  
**sysadmin sqm\_unzap\_tran** 443–445  
 sysadmin sqm\_zap\_command 446  
**sysadmin sqm\_zap\_tran** 447–449  
 sysadmin sqt\_dump\_queue 450  
 sysadmin system\_version 454  
 validate publication 457  
 validate subscription 458  
 wait for create standby 461  
 wait for delay 462  
 wait for switch 463  
 wait for time 464  
 committed transactions, system table for 688  
 comparing primary with replicate tables 642, 646  
 components  
   configuring (RMS) 735  
   defined 735  
   getting (RMS) 748  
   getting status descriptions (RMS) 759  
   resuming (RMS) 763  
   states 786  
   suspending (RMS) 768  
 computed columns

- replication of 309, 504
- CONFIG\_charset configuration parameter 638
- configuration commands, summary of 17
- configuration file
  - Replication Server 638
  - rs\_subcmp program 651
- configuration parameters
  - dsi\_bulk\_copy** 126, 143
  - dsi\_bulk\_threshold** 127, 143
  - dsi\_max\_xacts\_in\_group 132
  - Replication Server 638
  - rs\_config system table 206, 213
  - rs\_subcmp program 651
  - summary of commands 17
  - system table for 667
- configure component command (RMS) 735
- configure connection command 204
- configure logical connection command 205
- configure replication server command 206, 661
- configure RMS command (RMS) 737
- configure route command 221
- configure server command (RMS) 740
- connect command (RMS) 742
- connection profile 72
  - creating connection 241–248
- connection status event arguments 791
- connection status, filtering (RMS) 747
- connections
  - altering 123
  - altering schedules. *See* schedules
  - creating between Replication Servers. *See* routes
  - creating schedules. *See* schedules
  - description of 10
  - displaying schedules. *See* schedules
  - dropping schedules. *See* schedules
  - resuming 374
  - security parameters 144, 240
  - security parameters for 140
  - summary of commands for 10
  - suspending 389
- connections, status codes 787, 788
- conventions
  - examples xix
  - syntax statements xx
- conversion of character sets 41
- coordinated database dump 476
- coordinated transaction dump 479
- create applied function replication definition**
  - command 224–230
- create article command 231
- create connection using profile** clause 241
- create connection command 241
- create connection** command 235
- create connection using profile** clause 45, 248
- create database replication definition command 248
- create database replication definition** command 248
- create error class command 254
- create function command 257, 258
- create function replication definition command 259
- create function string class command 279
- create function string command 265, 270, 277
- create groups command (RMS) 742
- create logical connection command 282
- create partition command 284
- create publication command 286
- create replication definition command
  - references table owner.table name and column name 172, 298
  - with primary table named 296
  - with replicate table named 297
- create replication definition** command 296–312
- create request function replication definition**
  - command 290–295
- create route command 313
- create schedule command 318
- create subscription command 322
  - examples of 332, 341
  - reducing initialization time 324
  - selecting primary data without a holdlock 323
  - truncate table replication 324, 336
  - without materialization 324
- create user command 334
- creating
  - direct routes 313
  - indirect routes 313
  - routes 313
  - schedules 318
- current\_rssd\_version configuration parameter 207

**D**

- data comparison 642
- data manipulation failures, autocorrection 382
- data replication commands, summary of 2
- Data Server Interface 143
- Data Server Interface (DSI)
  - maximum number of source commands 777
  - maximum number of transactions 777
- data server name
  - primary 439
- data servers
  - assigning error-handling actions 194
  - open architecture and Replication Server 10
- database connection latency event arguments 795
- database context, changing 519
- database interface, summary of commands for 10
- database name
  - primary 439
- database replication definition 4–5
  - commands for 4
  - overview 4
  - subscription 6, 7, 8
- databases
  - configuring Replication Server interface to 204
  - displaying information about 590, 619
  - system table 668, 703
- datatype classes
  - rs\_asa\_udd\_class 35
  - rs\_db2\_udd\_class 35
  - rs\_mssql\_udd\_class 35
  - rs\_oracle\_udd\_class 35
  - rs\_sqlserver\_udd\_class 35
- datatype definitions 34, 305
- datatypes
  - bigdatetime 26
  - bigint 22
  - bigtime 26
  - binary 29
  - binary entry format 30
  - bit 31
  - char 24
  - character entry format 24
  - date 26
  - date/time entry format 26
  - datetime 25
  - decimal 23
  - float 24
  - image 30
  - image entry format 30
  - in replication definitions 304
  - int 22
  - Java 33
  - large object. *See* LOB datatypes
  - money 25
  - money entry format 25
  - numeric 23
  - opaque. *See* opaque datatype
  - rawobject in row 30
  - rawobject large in row 30
  - real 24
  - rs\_address 23, 305, 331, 654
  - rs\_id 661
  - smalldatetime 25
  - smallint 23
  - smallmoney 25
  - smallmoney entry format 25
  - supported 21
  - text 24
  - time. *See* time datatype
  - timestamp. *See* timestamp datatype
  - tinyint 23
  - unichar 31
  - Unicode 31
  - unitext 31
  - univarchar 31
  - unsigned bigint 23
  - unsigned int 23
  - unsigned smallint 23
  - unsupported 22
  - user-defined 22
  - varbinary 29
  - varbinary entry format 30
  - varchar 24
- date datatype 26
- date/time datatypes
  - bigdatetime 26
  - bigtime 26
  - datetime 25
  - smalldatetime 25
- db\_packet\_size configuration parameter 125
- DB2\_function\_class, described 280
- dbcc dbrepair Adaptive Server command 524

- dbcc gettrunc Adaptive Server command 525
- dbcc settrunc Adaptive Server command 527
- deadlock detection, system table for 721
- decimal datatype 23
- declared datatype 178, 305
- deferred\_name\_resolution configuration parameter 125
- deferred\_queue\_size configuration parameter 207
- define subscription command 335
- definition
  - identifiers xxii
- delete group command (RMS) 743
- destination Replication Server, altering 184
- direct routes, creating 313
- DirectConnect
  - RMS states 784
- disconnect command (RMS) 744
- disk partitions. *See* partitions
- disk\_affinity configuration parameter 125, 184
- disk\_direct\_cache\_read configuration parameter 207
- DIST thread
  - suspended 668
- dist\_stop\_unsupported\_cmd configuration parameter 125, 162
- distributor thread, enabling or disabling 162
- distributor thread. *See* DIST thread 668
- do\_not\_replicate clause 307
- double precision datatype 21
- drop article command 343
- drop connection command 345
- drop database replication definition** command 345
- drop error class command 347
- drop function command 348
- drop function replication definition command 349
- drop function string class command 352
- drop function string command 350
- drop logical connection command 353
- drop partition command 354
- drop publication command 355
- drop replication definition command 357
- drop route command 359
- drop schedule command 361
- drop server command (RMS) 746
- drop subscription command 362
- drop trigger command (RMS) 744
- drop user command 367
- dropping
  - schedules 361
- dropping routes 359
- DSI 143
- DSI bulk copy-in
  - autocorrection, and 384
- dsi\_alt\_writetext configuration parameter 126
- dsi\_bulk\_copy** connection parameter 126, 143
- dsi\_bulk\_threshold** connection parameter 127, 143
- dsi\_charset\_convert configuration parameter 127
- dsi\_cmd\_batch\_size configuration parameter 127
- dsi\_cmd\_prefetch configuration parameter 128
- dsi\_cmd\_separator configuration parameter 128
- dsi\_command\_convert configuration parameter 128
- dsi\_commit\_check\_locks\_intrvl configuration parameter 129
- dsi\_commit\_check\_locks\_max configuration parameter 129
- dsi\_commit\_control configuration parameter 129
- dsi\_compile\_enable configuration parameter 130
- dsi\_compile\_max\_cmds configuration parameter 130
- dsi\_connector\_type configuration parameter 131
- dsi\_dataserver\_make configuration parameter 131
- dsi\_exec\_request\_sproc configuration parameter 131
- dsi\_fadeout\_time configuration parameter 131
- dsi\_ignore\_underscore\_name configuration parameter 131
- dsi\_isolation\_level configuration parameter 132
- dsi\_keep\_triggers configuration parameter 132
- dsi\_large\_xact\_size configuration parameter 132
- dsi\_max\_cmds\_to\_log configuration parameter 132
- dsi\_max\_text\_to\_log configuration parameter 132
- dsi\_max\_xacts\_in\_group configuration parameter 132
- dsi\_non\_blocking\_commit** configuration parameter 133
- dsi\_num\_large\_xact\_threads configuration parameter 133
- dsi\_num\_threads configuration parameter 133
- dsi\_partitioning\_rule configuration parameter 133
- dsi\_quoted\_identifier** 133
- dsi\_replication configuration parameter 133
- dsi\_replication\_ddl configuration parameter 134
- dsi\_rs\_ticket\_report configuration parameter 134
- dsi\_serialization\_method configuration parameter 135
- dsi\_sqt\_max\_cache\_size configuration parameter 125, 136

- dsi\_text\_convert\_multiplier configuration parameter 136
- dsi\_timer configuration parameter 136
- dsi\_xact\_group\_size configuration parameter 137
- dump transaction
  - status indicator 275, 482
- dump\_load configuration parameter 137
- dumps, system table for 696, 699
- dynamic SQL
  - example of 382
- dynamic\_sql
  - setting 382
- dynamic\_sql configuration parameter 137
- dynamic\_sql\_cache\_management configuration parameter 137
- dynamic\_sql\_cache\_size configuration parameter 137

## E

- error actions
  - displaying 600
  - grouping 254
  - system table 678
- error classes
  - changing primary Replication Server for 370
  - description of 11
  - displaying 590
  - initializing 628
  - maximum number of actions assignments 777
  - summary of commands for 11
  - system table 663
- error messages, system table for 691
- error-handling actions, assigning to data server errors 194
- ERSSD
  - changing passwords 145
- ERSSD configuration parameters 218
- erssd\_backup\_dir configuration parameter 638
- erssd\_backup\_interval configuration parameter 218
- erssd\_backup\_path configuration parameter 218
- erssd\_backup\_start\_date configuration parameter 218
- erssd\_backup\_start\_time configuration parameter 218
- erssd\_dbfile configuration parameter 638
- erssd\_errorlog configuration parameter 638
- erssd\_logmirror configuration parameter 638
- erssd\_ping\_cmd configuration parameter 638
- erssd\_port configuration parameter 638
- erssd\_ra configuration parameter 218
- erssd\_ra\_release\_dir configuration parameter 638
- erssd\_ra\_start\_cmd configuration parameter 639
- erssd\_release\_dir configuration parameter 638
- erssd\_start\_cmd configuration parameter 639
- erssd\_translog configuration parameter 639
- event arguments 791–797
  - connection status 791
  - database connection latency 795
  - partition and queue size 797
  - partition status 792
  - queue latency 796
  - route status 793
  - server status 794
- event trigger arguments. <ix\_italics>See event arguments
- event triggers
  - adding (RMS) 729
  - dropping (RMS) 744
- exact numeric (decimal) datatypes
  - decimal 23
  - numeric 23
- exact numeric (integer) datatypes 22
  - bigint 22
  - int 22
  - smallint 23
  - tinyint 23
  - unsigned bigint 23
  - unsigned int 23
  - unsigned smallint 23
- examples
  - style conventions xix
- exceptions log
  - deleting transactions 583
  - displaying transactions in 601
  - system table 679, 680, 682
- exec\_cmds\_timeslice configuration parameter 138
- exec\_nrm\_request\_limit configuration parameter 138
- exec\_sqm\_write\_request\_limit configuration parameter 138
- executable programs
  - repserver 636
  - rs\_subcmp 642
- extended page
  - supported 42

**F**

- failed transactions, autocorrection for 382
- failover
  - automatically start RepAgent after 541
  - enabling Sybase Failover support in Replication Server 207, 217
- filter connection status command (RMS) 747
- float datatype 24
- fragments, system table for 715
- function replication definitions
  - altering 155
  - commands for 3
  - data distribution and 3
  - dropping 349
  - searchable parameters, adding of 120, 181, 225
  - searchable parameters, adding to 291
  - specifying parameters to send to standby database 260
  - specifying primary table location for 119, 224, 259, 290
  - specifying searchable columns for 225, 260, 291
  - specifying table name at primary and replicate databases 224, 259, 290
  - standby database, sending parameters to 120
  - standby database, sending to 225, 291
  - standby database, suspending DSI 121, 174, 181
  - warm stanby, sending to 181
- function strings
  - altering 159
  - delimiters in templates 272
  - description of 12
  - displaying for a function-string class 592
  - displaying for a replication definition 602
  - grouping 160, 279
  - input templates of 273
  - limits 777
  - output templates of 273
  - replacing 159
  - RPC output templates 267
  - summary of commands for 12
  - system table 683
  - system-defined variables for output templates in 275
  - templates, size of, in 272
- functions
  - altering 153

- description of 12
  - displaying for a replication definition 604
  - displaying for a Replication Server 604
  - summary of commands for 13
  - system table for 685
- function-string classes 238
  - changing primary Replication Server for 370
  - description of 12
  - displaying 590
  - dropping 352
  - summary of commands for 12
  - system table 663

**G**

- get components command (RMS) 748
- get description command (RMS) 759
- get groups command (RMS) 751
- get heartbeat command (RMS) 753
- get network specifications command (RMS) 756
- get RMI address command (RMS) 754, 757
- get servers command (RMS) 757
- get threads command (RMS) 760
- get triggers command (RMS) 760
- get version command (RMS) 762
- grant command 368
  - examples of 368
  - permissions for 368
- groups
  - creating (RMS) 742
  - deleting (RMS) 743
  - getting (RMS) 751

**H**

- ha\_failover. See failover
- HDS, verifying translations 95
- heartbeat
  - defined 753
  - getting (RMS) 753
  - starting (RMS) 766, 767
- hibernation
  - turning off 423
  - turning on 425

**I**

## icons

- Adaptive Server xxiii
- client application xxiii
- Replication Agent xxiii
- Replication Manager xxiii
- Replication Server xxiii

ID Server, system table for 686

- id\_msg\_confidentiality configuration parameter 215
- id\_msg\_integrity configuration parameter 215
- id\_msg\_origin\_check configuration parameter 215
- id\_msg\_replay\_detection configuration parameter 216
- id\_msg\_sequence\_check configuration parameter 216
- id\_mutual\_auth configuration parameter 216
- ID\_pw configuration parameter 639
- ID\_pw\_enc configuration parameter 639
- id\_security\_mech configuration parameter 216
- ID\_server configuration parameter 639
- id\_server configuration parameter 207
- id\_unified\_login configuration parameter 216
- ID\_user configuration parameter 639

## identifiers

- definition of xxii
- described 35
- format xxii
- function parameters xxii
- length xxii
- name space for 37
- types of xxii

IDENTITY columns 23

- in replication definitions 304

ignore loss command 369

## image column

- retrieving description for 487

## image datatype

- changing replication for 560
- defining replication for 560
- description of 30
- executing replication for 509, 520
- logging updates for 521, 522

indirect routes, creating 313

init\_sqm\_write\_delay configuration parameter 207

init\_sqm\_write\_max\_delay configuration parameter 208

int datatype 22

## intermediate Replication Server

- altering 184

removing from a route 189

international environments, support for 40, 42, 656

## IQ

RMS states 784

**J**

Java datatypes 33

**K**

keywords 38

**L**

## languages

- Replication Server 639
- rs\_msgs system table 691
- rs\_subcmp program 646
- supported 42

large object datatypes. *See* LOB datatypes

limitations for Replication Server 775

LOB datatypes 24, 30

- conversion of 24, 30

## locator

- system table 689

## locator value

- resetting 634

## log

- exceptions 583

## log file

- displaying path to 62

## Log Transfer Manager (LTM)

- executable 635
- locator value 634

## logging

- updates to text or image data 521

## logical connections

- changing attributes of 205
- creating for warm standby 282
- displaying status of 63
- dropping for warm standby 353
- enabling or disabling Distributor thread 162

login names. See users  
ltm program 635

## M

maintenance users  
    system table 690  
map to option 171, 298  
materialization  
    atomic 7  
    bulk 7  
    non-atomic 7  
    non-materialization 7  
    rs\_marker system function 493  
    status of 201  
    summary of commands for 6  
materialization\_save\_interval configuration parameter  
    for logical connections 162  
md\_sqm\_write\_request\_limit configuration parameter 139  
mem\_reduce\_malloc configuration parameter 208  
memory\_limit configuration parameter 208  
message language  
    supported 42  
messages  
    abbreviations used in 773  
    acronyms used in 773  
    maximum size written to stable queue 777  
    storing in system tables 711  
minimal columns  
    replicating 306  
minimum\_rssd\_version configuration parameter 208  
mixed-version replication system 437, 455  
modifiers for function string variables 267  
money datatypes  
    money 25  
    smallmoney 25  
move primary command 370  
msg\_confidentiality configuration parameter 214, 313  
msg\_integrity configuration parameter 214, 314  
msg\_origin\_check configuration parameter 214, 314  
msg\_replay\_detection configuration parameter 214, 314

msg\_sequence\_check configuration parameter 214, 314  
multibyte data  
    replicating 22  
mutual\_auth configuration parameter 215, 314

## N

name space  
    for identifiers 37  
nchar datatype 22  
    replicating 22  
network specifications  
    getting (RMS) 756  
network-based security  
    setting parameters for 215  
network-based security for RepAgent 540  
non-Adaptive Server error class  
    **create error class** example 254  
    **create error class** option 254  
non-atomic materialization  
    description of 7  
    and replicating minimal columns 306  
    summary of commands for 7  
non-binary sort orders  
    supported 41  
non-blocking commit  
    **rs\_non\_blocking\_commit** 494  
    **rs\_non\_blocking\_commit\_flush** 495  
    **rs\_set\_non\_blocking\_commit\_flush** 495  
**not quoted** parameter 171  
nrm\_thread configuration parameter 209  
num\_client\_connections configuration parameter 209  
num\_concurrent\_subs configuration parameter 209  
num\_msgqueues configuration parameter 209  
num\_msgs configuration parameter 209  
num\_mutexes configuration parameter 209  
num\_stable\_queues configuration parameter 209  
num\_threads configuration parameter 210  
numeric datatype 23  
    in replication definitions 304  
nvarchar datatype 22  
    replicating 22

## O

- object IDs
  - system table 687
- objects
  - system table 692
- opaque datatype 21, 34
  - limitations 34
  - mixed-version support 34
- open architecture
  - and heterogeneous data servers 10
- oserver configuration parameter 210
- output templates
  - system-defined variables for 275

## P

- parallel DSI
  - configuring 204, 205, 221
  - rs\_get\_thread\_seq system function 489
  - rs\_get\_thread\_seq\_noholdlock system function 490
  - rs\_initialize\_threads system function 491
  - rs\_set\_isolation\_level 505
  - rs\_threads system table 721
- parallel\_dsi configuration parameter 139
- parameters
  - adding to user-defined functions 153
- partition and queue size event arguments 797
- partition status event arguments 792
- partitions
  - adding 54
  - altering 166
  - creating 284
  - displaying 606
  - dropping 354
  - recovering 372
  - removing from Replication Server 354
  - Replication Server storage and 17
  - status codes 790
  - summary of commands 17
  - system table for storing 677
- password\_encryption configuration parameter 210
- passwords
  - altering for a user 193
- permissions
  - assigning 368

- revoking 381
  - server, for RMS commands 728
  - summary of commands for 9
- prev\_min\_rssd\_version configuration parameter 210
- prev\_rssd\_version configuration parameter 210
- primary data server name 439
- primary database name 439
- primary tables
  - comparing to replicates 657, 658
- process ID
  - displaying for local Replication Server 65
- publications
  - commands for 5
  - dropping 355
  - status of 199
  - subscription commands for 8
  - validating 457
- published datatype 178
- published datatypes 305

## Q

- queue block size, setting 206
- queue latency event arguments 796
- queue\_dump\_buffer\_size configuration parameter 210, 412, 418
- queues, status codes 789
- quiesce
  - changing Replication Server state 67, 378, 391
  - checking Replication Server state 15, 43, 61, 66, 67, 632, 781, 782
- quotation marks
  - in character datatypes 24
- quoted** parameter 171, 296
- quoted identifiers
  - dsi\_quoted\_identifier** 133
  - embedded double quote characters 310
  - example 301
  - forwarding to data servers 506
  - marking identifiers as quoted 171, 296, 309, 310
  - removing markings 171
  - usage 309–310

## R

- raw disk partitions. See partitions
- rawobject datatype 30
- rawobject in row datatype 30
- rawobject large in row datatype 30
- real datatype 24
- rebuild queues command 61, 372
- rec\_daemon\_sleep\_time configuration parameter 210
- reconciliation
  - rs\_subcmp program 642
- recovery
  - system table for 702
- recovery commands
  - summary 20
- Recovery mode 119
- references option 172, 298
- Referential constraints, handling tables with 179, 311
- rep\_as\_standby configuration parameter 139
- RepAgent
  - automatic start-up, delaying 542
  - cluster instance name 543
  - configuring 537
  - recovery mode, starting in 577
  - starting 577
- RepAgent, status codes 790
- repeating groups
  - system table for 719
- replicate minimal columns option 306
- replicate tables
  - comparing to primary 657, 658
- replicate\_if\_changed clause 307
- replicate\_minimal\_columns configuration parameter
  - for logical connection 179
  - for database connection 140
- replicating computed columns 309
- replicating tables
  - sp\_setreptable Adaptive Server system procedure 574
- Replication Agent
  - cluster instance name 543
  - resuming (RMS) 765
  - RMS states 785
  - suspending (RMS) 769
- replication definitions
  - altering 170
  - commands for 3, 4
  - creating 296
  - data distribution and 2
  - datatypes in 304
  - description of 2
  - displaying 612
  - displaying information about versions 620
  - dropping 357
  - executing change requests directly at the primary database 629
  - limits 776
  - replicating minimal columns 298
  - replicating text and image columns 299
  - specifying columns for standby database 298
  - specifying primary keys for 298
  - specifying primary table location for 296
  - specifying searchable columns for 298
  - specifying table name at primary and replicate databases 296
  - system table for 664, 692
  - using rs\_address datatype in 305
- Replication Server
  - mixed-version 437, 455
  - RMS states 782
  - status of, displaying 61
- Replication Server error class 195
  - create connection** example 237
  - create connection** option 235
  - create error class** example 254
  - create error class** option 151, 254
  - drop error class** example 347
  - drop error class** option 347
  - error actions 196–198
  - move primary** example 370
  - move primary** option 370
  - parameter 194, 195
  - supported Replication Server errors 196
  - usage 151, 238, 255, 371
- Replication Server Gateway
  - connection stack, list of 386
  - current server, display 387
  - show connection** 386
  - show server** 387
- Replication Server gateway 14
  - connect** command 221
  - disconnect** 342

- summary of commands for 14
- terminating connection 342
- Replication Server System Database (RSSD)
  - description of 661
- Replication System Administrator
  - role of xv
- repserver executable program 636
- repserver program 637
- reserved words 38
- resume component command (RMS) 763
- resume connection command 374
  - example of 375
  - skip transaction option 374
- resume distributor command 377
- resume log transfer command 378
- resume queue command 379
- resume replication agent command (RMS) 765
- resume route command 380
- revoke command 381
  - examples of 381
- RMI address
  - getting (RMS) 754, 757
- RMS
  - component states 786
  - configuring 737
  - server states 779
  - states 785, 786
- rollback
  - assigning error-handling actions and 194
- route status event arguments 793
- route versions
  - system table for 706
- routes
  - altering 184
  - creating 313
  - displaying status of 622
  - dropping 359
  - removing intermediate Replication Servers 189
  - resuming 380
  - summary of commands for 14
  - suspending 392
  - system table for 705
- routes, status codes 789
- row count verification
  - example 195
- RPCs
  - replicating text or image data 521
- rs\_address datatype 23, 305, 331, 654
  - in replication definitions 304
- rs\_articles system table 661, 662
- rs\_batch\_end system function 465
- rs\_batch\_start system function 467
- rs\_begin system function 468
- rs\_capacity stored procedure 582
- rs\_captable table 582, 587
- RS\_charset configuration parameter 639
- rs\_check\_repl system function 469
- rs\_classes system table 663
- rs\_columns system table 664
- rs\_config system table 667
  - configuration parameters 206, 213
- rs\_databases system table 668
- rs\_datarow\_for\_writetext system function 472
- rs\_datatype system table 669
- rs\_dbreps system table 673
- rs\_dbsubsets system table 674
- rs\_default\_fs system variable
  - and minimal columns 272, 306
- rs\_default\_function\_class
  - described 280
- rs\_delete system function 474
- rs\_delexception stored procedure 583
- rs\_diskaffinity system table 676
- rs\_diskpartitions system table 677
- rs\_dsi\_check\_thread\_lock system function 474
- rs\_dumpdb system function 239, 476
- rs\_dumptran system function 239, 479
- rs\_erroractions system table 678
- rs\_exceptscmd system table 679
- rs\_exceptshdr system table 680
- rs\_exceptslast system table 682
- rs\_fillcapable stored procedure 587
- rs\_funcstrings system table 683
- rs\_functions system table 685
- rs\_get\_charset system function 483
- rs\_get\_errormode system function 484
- rs\_get\_lastcommit system function 485
- rs\_get\_sortorder system function 487
- rs\_get\_textptr system function 488
- rs\_get\_thread\_seq system function 489
- rs\_get\_thread\_seq\_noholdlock system function 490
- rs\_helpclass stored procedure 590

- rs\_helpclassfstring stored procedure 592
- rs\_helpcounter stored procedure 593
- rs\_helpdb stored procedure 596
- rs\_helpdbrep stored procedure 597
- rs\_helpdbsub stored procedure 599
- rs\_helperror stored procedure 600
- rs\_helpexception stored procedure 601
- rs\_helpfstring stored procedure 602
- rs\_helpfunc stored procedure 604
- rs\_helppartition stored procedure 606
- rs\_helpprep stored procedure 612
- rs\_helprepdb stored procedure 619
- rs\_helpreptable stored procedure 627
- rs\_helpreversion stored procedure 620
- rs\_helproute stored procedure 622
- rs\_helpsub stored procedure 624
- rs\_helpuser stored procedure 626
- rs\_id datatype 661
- rs\_idnames system table 686
- rs\_ids system table 687
- rs\_init installation program 235
- rs\_init\_erroractions stored procedure 628
- rs\_initialize\_threads system function 491
- rs\_insert system function 492
- RS\_language configuration parameter 639
- rs\_lastcommit system table 485, 688
- rs\_locator system table 689
- rs\_maintusers system table 690
- rs\_marker system function 493
- rs\_msgs system table 691
- rs\_non\_blocking\_commit** system function 494
- rs\_non\_blocking\_commit\_flush** system function 495
- rs\_objects system table 692
- rs\_oqid system table 696
- rs\_profile system table 697
- rs\_publications system table 698
- rs\_queuemsg system table 699
- rs\_queuemsgtxt system table 700
- rs\_queues system table 701
- rs\_raw\_object\_serialization system function 496
- rs\_recovery system table 702
- rs\_repdb system table 703
- rs\_repl\_off system function 497
- rs\_repl\_on system function 498
- rs\_repobjs system table 704
- rs\_rollback system function 499
- rs\_routes system table 705
- rs\_routeversions system table 706
- rs\_rules system table 707
- rs\_schedule system table 709
- rs\_scheduledtxt system table 710
- rs\_segments system table 707
- rs\_select system function 500
- rs\_select\_with\_lock system function 502
- RS\_send\_enc\_pw configuration parameter 639
- rs\_send\_repserver\_cmd stored procedure 629
- rs\_set\_ciphertext system function 503
- rs\_set\_dml\_on\_computed system function 504
- rs\_set\_isolation\_level system function 505
- rs\_set\_non\_blocking\_commit\_flush** system function 495
- rs\_set\_quoted\_identifiers** 506
- rs\_sites system table 712
- RS\_sortorder configuration parameter 639
- rs\_sqldml** system function 507–508
- rs\_sqlserver\_function\_class described 280
- RS\_ssl\_identity configuration parameter 640
- RS\_ssl\_pw configuration parameter 641
- RS\_ssl\_pw\_enc configuration parameter 641
- rs\_statcounters system table 712
- rs\_statdetail system table 713
- rs\_statrun system table 713
- rs\_subcmp executable program 642
- rs\_subcmp program configuration file 651 configuration parameters 651
- rs\_subscriptions system table 715
- rs\_systabgroup group 638, 661
- rs\_systext system table 719
- rs\_tbconfig system table 720
- rs\_textptr\_init system function 509
- rs\_threads system table 721
- rs\_ticket stored procedure 632
- rs\_ticket\_history system table 722
- rs\_ticket\_history table 510
- rs\_ticket\_report system function 510
- rs\_ticket\_v1 stored procedure 632
- rs\_translation system table 723
- rs\_triggers\_reset system function 511
- rs\_truncate system function 513

- RS\_unicode\_sortorder configuration parameter 639
  - rs\_update system function 516
  - rs\_update\_threads system function 518
  - rs\_usedb system function 519
  - rs\_users system table 724
  - rs\_version system table 725
  - rs\_whereclauses system table 726
  - rs\_writetext system function 520
  - rs\_zeroltm stored procedure 634
  - rsi\_batch\_size configuration parameter 184
  - rsi\_fadeout\_time configuration parameter 184
  - rsi\_packet\_size configuration parameter 185
  - rsi\_sync\_interval configuration parameter 185
  - rsi\_xact\_with\_large\_msg configuration parameter 185
  - RSSD
    - stored procedures 581
  - RSSD\_database configuration parameter 639
  - RSSD\_embedded configuration parameter 639
  - rssd\_error\_class configuration parameter 210
  - RSSD\_ha\_failover configuration parameter 639
  - RSSD\_maint\_pw configuration parameter 639
  - RSSD\_maint\_pw\_enc configuration parameter 639
  - RSSD\_maint\_user configuration parameter 640
  - RSSD\_msg\_confidentiality configuration parameter 640
  - RSSD\_msg\_integrity configuration parameter 640
  - RSSD\_msg\_origin\_check configuration parameter 640
  - RSSD\_msg\_replay\_detection configuration parameter 640
  - RSSD\_msg\_sequence\_check configuration parameter 640
  - RSSD\_mutual\_auth configuration parameter 640
  - RSSD\_primary\_pw configuration parameter 640
  - RSSD\_primary\_pw\_enc configuration parameter 640
  - RSSD\_primary\_user configuration parameter 640
  - RSSD\_sec\_mechanism configuration parameter 640
  - RSSD\_server configuration parameter 640
  - RSSD\_unified\_login configuration parameter 641
  - RTL and HVAR
    - rs\_tbconfig system table 720
- S**
- sa permission xv
  - save\_interval configuration parameter
    - for database connection 140
    - for logical connection 163
    - for route 185
  - schedule
    - switchinon and off 192
  - schedule, displaying 69
  - schedule, enabling or disabling 192
  - schedules
    - altering 192
    - creating 318
    - disabling 192
    - dropping 361
    - enabling 192
    - storing in system tables 709
    - storing schedule commands in system tables 710
    - turning on and off 192
  - schema comparison 642
  - security. See permissions
  - security\_mechanism configuration parameter 215
  - send\_enc\_password configuration parameter 215
  - send\_enc\_password configuration parameters 211
  - send\_timestamp\_to\_standby configuration parameters 211
  - server status event arguments 794
  - servers
    - adding (RMS) 732
    - configuring (RMS) 740
    - connecting to (RMS) 742
    - disconnecting from (RMS) 744
    - dropping (RMS) 746
    - getting (RMS) 757
    - getting status descriptions (RMS) 759
    - shutting down (RMS) 765
  - set command** 382–384
  - set log recovery command 385
  - set proxy command 386
  - set replication Adaptive Server command 530
  - set repmode Adaptive Server command 531
  - set repthreshold Adaptive Server command 532
  - show connection** command 386
  - show server** command 387
  - shutdown command 388
  - shutdown server command (RMS) 765
  - site ID, system table for 712
  - site version number 436
  - smalldatetime datatype 25
  - smallint datatype 23

- smallmoney datatype 25
- smp\_enable configuration parameter 211
- sort orders
  - expected 487
  - Replication Server 639
  - rs\_subcmp 646, 657, 658
- sp\_config\_rep\_agent Adaptive Server system procedure 537
- sp\_configure enable rep agent threads Adaptive Server system procedure 536
- sp\_help\_rep\_agent Adaptive Server system procedure 547
- sp\_reptostandby Adaptive Server system procedure 553
- sp\_setrepcol Adaptive Server system procedure 560
- sp\_setrepdbmode** 563–565
- sp\_setrepdefmode Adaptive Server system procedure 566
- sp\_setreplicate Adaptive Server system procedure 569
- sp\_setrepproc Adaptive Server system procedure 571
- sp\_setreptable Adaptive Server system procedure 574
- sp\_start\_rep\_agent Adaptive Server system procedure 577
- sp\_stop\_rep\_agent Adaptive Server system procedure 580
- SQL statement replication
  - alter replication definition option** 173, 175
  - create database replication definition** example 250, 251
  - create database replication definition option** 249
  - create replication definition** example 301
  - create replication definition** option 299
  - sp\_setrepdbmode** 563–565
  - usage 252–253, 310–311
- sqm\_cache\_enable configuration parameter 211
- sqm\_cache\_size configuration parameter 211
- sqm\_page\_size configuration parameter 211
- sqm\_recover\_segs configuration parameter 211
- sqm\_seg\_prealloc configuration parameter 212
- sqm\_warning\_thr\_ind configuration parameter 212
- sqm\_warning\_thr1 configuration parameter 212
- sqm\_warning\_thr2 configuration parameter 212
- sqm\_write\_flush configuration parameter 212
- sqt\_init\_read\_delay configuration parameter 212
- sqt\_max\_cache\_size configuration parameter 212
- sqt\_max\_read\_delay configuration parameter 213
- sre\_reserve configuration parameter 213
- stable queues
  - deleting a message 446
  - deleting transactions of 447
  - estimating size requirements 582
  - maximum message size 777
  - rebuilding 372
  - restoring transactions of 443
  - storing messages in 700
  - system table 676, 677, 700, 701
  - undeleting a message 442
- stand-alone mode 61
- standalone mode 637, 702
- start heartbeat command (RMS) 766, 767
- starting Replication Agent 543
- statistics collectors 85
  - counter 85
  - monitor 85
  - observer 85
- stats\_reset\_rssd configuration parameter 213
- stats\_sampling configuration parameter 213
- stats\_show\_zero\_counters configuration parameter 213
- sts\_cachesize configuration parameter 213
- sts\_full\_cache\_system\_table\_name configuration parameter 213
- sub\_daemon\_sleep\_time configuration parameter 213
- sub\_sqm\_write\_request\_limit configuration parameter 140
- subcmp program. See rs\_subcmp program
- subscription materialization. See materialization
- subscriptions
  - activating 51
  - creating 322
  - defining 335
  - description of 6
  - displaying information about 624
  - dropping 362
  - limits of 776
  - system table for 715
  - using rs\_address datatype in 331
  - validating 458

- where clause and 6
- with purge 365
- without materialization option 7
- without purge 365
- suspend component command (RMS) 768
- suspend connection command 389
- suspend distributor command 390
- suspend log transfer command 391
- suspend replication agent command (RMS) 769
- suspend route command 392
- switch active command 393
- syntax conventions
  - identifiers xxii
- syntax statements, conventions xx
- sysadmin apply\_truncate\_table command 394
- sysadmin cdb command 396
- sysadmin drop\_queue command 406
- sysadmin dropdb command 404
- sysadmin dropldb command 405
- sysadmin droprs command 407
- sysadmin dump\_file command 408
- sysadmin dump\_queue command 409
- sysadmin dump\_thread\_stack** command 414
- sysadmin dump\_tran** command 416–419
- sysadmin erssd, command 419
- sysadmin fast\_route\_upgrade command 422
- sysadmin hibernate\_off command 423
- sysadmin hibernate\_on command 425
- sysadmin issue\_ticket command 426
- sysadmin log\_first\_tran command 428
- sysadmin purge\_all\_open command 429
- sysadmin purge\_first\_open command 431
- sysadmin purge\_route\_at\_replicate command 433
- sysadmin restore\_dsi\_saved\_segments command 434
- sysadmin set\_dsi\_generation command 435
- sysadmin site\_version command 436
- sysadmin skip\_bad\_repserver\_cmd 439
- sysadmin sqm\_purge\_queue command 441
- sysadmin sqm\_unzap\_command command 442
- sysadmin sqm\_unzap\_tran** command 443–445
- sysadmin sqm\_zap\_command command 446
- sysadmin sqm\_zap\_tran** command 447–449
- sysadmin sqt\_dump\_queue command 450
- sysadmin system\_version command 454
- system administration commands, summary of 18
- system information, summary of commands 15
- system parameters for configuration 667
- system tables
  - access restrictions 661
  - autocorrection flags for replication definitions 704
  - database IDs 686
  - database information 703
  - database names 668, 686
  - error actions 678
  - error classes 663
  - event parameters 664
  - exceptions log 679
  - fragment information 715
  - function strings 683
  - functions 685
  - function-string classes 663
  - function-string text 719
  - localized error messages 691
  - locator fields 689
  - logged transaction information 680
  - maintenance user login names 690
  - maintenance user passwords 690
  - object IDs 687
  - object information 692
  - output command text 719
  - parallel DSI threads 721
  - partitions 677
  - queue dumps 699
  - queue IDs for last logged transaction 682
  - queue IDs from origin sites 696
  - queue information 701
  - raw disk partitions 677
  - recovery actions 702
  - replication definition columns 664
  - Replication Server IDs 686, 712
  - Replication Server names 686, 712
  - route version information 706
  - routing information 705
  - rs\_articles 662
  - rs\_classes 663
  - rs\_columns 664
  - rs\_config 667
  - rs\_databases 668
  - rs\_datatype 669
  - rs\_dbreps 673
  - rs\_dbsubsets 674
  - rs\_diskaffinity 676

- rs\_diskpartitions 677
- rs\_erroractions 678
- rs\_statcounters 712
- rs\_statdetail 713
- rs\_statrun 713
- rs\_systext 719
- rs\_tbconfig 720
- rs\_threads 721
- rs\_ticket\_history 722
- rs\_translation 723
- rs\_user 724
- rs\_version 725
- rs\_whereclauses 726
- RTL and HVAR 720
- schedules to execute commands 709, 710
- segment allocation for raw disk space 711
- source command text 719
- text of stable queue messages 700
- subscription information 715
- subscription rules 707, 715
- trigger information 715
- user information 724
- system-defined variables 275
- system-wide version number 454, 725

## T

- table replication definitions
  - commands for 3
  - data distribution and 3
  - description of 2
  - dynamic SQL, application of 173, 299
  - set properties of 382
- table-level configuration parameters, system table for 720
- tables
  - comparing replicate to primary 657, 658
  - system table descriptions 661
- text column, retrieving description for 488
- text datatype 24
  - changing replication for 560
  - defining replication for 299, 560
  - description of 24
  - executing replication for 509, 520
  - logging updates for 521, 522

- text pointers, text or image data 509
- threads
  - getting (RMS) 760
- ticket 510
- time datatype 21, 26
- timestamp datatype 21, 26, 176
  - attribute mask 693
  - column declaration in replication definition 666
  - function string variables, formatting for 274
  - in replication definitions 304
  - table replication definition, in 304
- tinyint datatype 23
- trace configuration parameter 641
- trace\_file configuration parameter 641
- transaction rates, for replication definitions 587
- transactions
  - displaying in exceptions log 601
  - number in DSI transaction group 777
  - restoring 443
  - system table 679, 680, 682, 688
- triggers
  - adding (RMS) 729
  - defined 729
  - dropping (RMS) 744
  - getting (RMS) 760
- triggers, system table for 715

## U

- UDD
  - conversion 673
- unified\_login configuration parameter 215, 314
- unsigned bigint datatype 23
- unsigned int datatype 23
- unsigned smallint datatype 23
- use\_batch\_markers configuration parameter 140
- use\_security\_services configuration parameter 215
- use\_ssl configuration parameter 215
- user administration, summary of commands for 9
- user-defined datatypes in replication definitions 304
- user-defined datatypes. *See* UDD 673
- users
  - altering passwords 193
  - assigning permissions to 368
  - displaying information about 626

## *Index*

dropping 367  
system table 690, 724

## **V**

validate publication command 457  
validate subscription command 458  
varbinary datatype 29  
varchar datatype 24  
varchar\_truncation configuration parameter 214  
variables in function strings 267, 275  
version  
    getting (RMS) 762  
version number  
    site 436  
    system-wide 454, 725

## **W**

wait for create standby command 461  
wait for delay command 462  
wait for switch command 463  
wait for time command 464  
warm standby applications  
    abort switch command 50  
    admin logical\_status command 63  
    alter logical connection command 162  
    configure logical connection command 205  
    create logical connection command 282  
    drop logical connection command 353  
    summary of commands 13  
    switch active command 393  
with primary table named 296  
with replicate table named 297  
writetext logging options 521, 522