SYBASE®

New Features Guide

Replication Server®

15.5

DOCUMENT ID: DC00783-01-1550-01

LAST REVISED: March 2010

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

Contents

About This Book ..... **ix**

CHAPTER 1 **New Features in Replication Server Version 15.5** **1**
Replication Server 15.5 licenses 1
Replication to Sybase IQ 2
Licensing 3
Database and platform support 4
Heterogeneous replication support 4
Heterogeneous parallel DSI 5
Heterogeneous warm standby for Oracle 6
Controlling trigger execution at the Oracle replicate database 7
Performance enhancements 7
Replication Server – Advanced Services Option 8
Dynamic SQL enhancements 12
Function-string efficiency improvements 13
Usability and processes enhancements 14
Replication definition change request process enhancements 15
Scheduling replication tasks 17
Delaying replication 17
Resynchronizing replicate databases 18
Row count validation enhancements 19
Implementing a reference replication environment 21
admin who enhancements 22
Resetting database generation numbers 23
Injecting rs_ticket markers 24
Changes to default settings and reserved words 24
Changes to parameter default values 24
Changes to the RSSD locking schema 26
New reserved words 26
Adaptive Server support 26
bigdatetime and bigtime datatypes replication 26
Deferred name resolution 27
SQL statement replication threshold setting 28
Incremental data transfer 28

In-memory and relaxed-durability databases 29
Mixed-version environments 31
Operating system and platform support 32
64-bit support 32

CHAPTER 2 New Features in Replication Manager 15.5 35
Support for bigdatetime and bigtime datatypes 35

CHAPTER 3 New Features in Replication Server Version 15.2 37
Support for DSI bulk copy-in 37
Setting up bulk copy-in 38
Changes to subscription materialization 39
New counters for bulk copy-in 39
Limitations 40
Non-blocking commit 42
Adaptive Server delayed commit feature 42
Configuring non-blocking commit 42
System functions that support non-blocking commit 43
Support for non-Adaptive Server databases 44
Product version requirements 44
Quoted identifiers 45
Enabling the quoted identifier feature 45
Marking identifiers as quoted 46
Forwarding quoted identifiers to data servers 47
Changes to rs_helprep 48
Product version requirements 51
Replication Server gateway 51
Cascading connection 51
Enabling the Replication Server gateway 52
Tracking connections 53
Dropping connections 53
Limitations 53
Product version requirements 53
Row count validation for non-SQL statement replication 54
Creating Replication Server error classes 54
Assigning error actions 55
Displaying Replication Server error classes 56
Replication Server System Database (RSSD) modifications 56
SQL statement replication 57
Modifications to system configuration 57
SQL statement replication configuration 59
Row count validation for SQL statement replication 62
Configuring warm standby database for SQL replication 64

Replication Server System Database (RSSD) modifications .. 64
Product and mixed-version requirements 64
Non-Adaptive Server error class support 65
Default non-ASE error classes 65
Creating error classes 65
Altering error classes 66
Creating and altering connections 66
Native error codes 66
Non-Adaptive Server replication support enhancements 67
Simplified installation and configuration 67
Connection profiles 68

CHAPTER 4 New Features in Replication Server Version 15.1 73
Dynamic SQL enhancements 73
Function replication enhancements 75
Adaptive Server shared-disk cluster support 76
Monitor and counter enhancements 76
Active object identification 77
New procedures interface 77
Improved stable queue management 78
Changes to sysadmin dump_queue 78
Changes to sysadmin sqt_dump_queue 79
Changes to resume connection 79
Changes to sysadmin log_first_tran 80
New sysadmin sqm_zap_tran command 80
New sysadmin sqm_unzap_tran command 80
New sysadmin dump_tran command 81
Changed locales directory 81
Extended password encryption support 82
rs_ticket stored procedure version 2 82
New Replication Server counters 83
Extended support for large-object (LOB) datatypes 84
Partial update of LOB datatypes 85
Extended timestamp support 86
New opaque datatype 86
Dump transaction enhancement 87
LTL dump subcommand enhancement 87
rs_dumptran enhancement 88
Distributor status recording 89
Enhanced text update 89
Adaptive Server integer identity support 90
Stable Queue Manager performance enhancements 90
Stable queue caching 90
Segment preallocation 92

Support for direct I/O file access 92

CHAPTER 5 New Features in Replication Manager 15.1 93
Enhanced support for dynamic SQL 93
Enhanced support for function replication definitions 94
Support for large-object datatypes 94
Sybase Central 6.0 95
Support for opaque datatypes 95
Support for timestamp datatypes 96

CHAPTER 6 New Features in Replication Server Version 15.0.1 97
Configuration and tuning enhancements 97
Monitor and counter enhancements 98
Dynamic SQL for enhanced Replication Server performance 100
Master database replication 102
rs_subcmp enhancement 103
Schema comparison 105
Manual data reconciliation 108

CHAPTER 7 New Features in Replication Manager 15.0.1 111
Support for dynamic configuration 111
Support for heterogeneous data servers 112
Non-Sybase data servers support 112
Replication Agents and Mirror Replication Agents support 113
Replication Manager plug-in enhancements 114

CHAPTER 8 New Features in Replication Server Version 15.0 115
Support for longer identifiers 115
New datatype: bigint 116
New unsigned integer datatypes 117
New Unicode datatype: unitext 118
Replicating computed columns 119
Replicating encrypted columns 120
Replicating partitioned tables 120
Larger disk partitions 122
Larger text and image size 123
Embedded Replication Server System Database (ERSSD) enhancement 123
New password encryption algorithm 124
Mixed-version enhanced support 125
New interface for monitors and counters 126
Support for isolation levels 127

Bidirectional replication support for DDL in MSA 129
Batching of commands for non-ASE servers 129
SySAM license management 130

CHAPTER 9 New Features in Replication Manager 15.0 133
New user interface features 133
Two-tier management solution 133
Three-tier management solution 134
Replication Manager plug-in replaces Replication Server plug-in 134
Online help 134
Visual monitoring of status 135
Event Log pane 135
Background processing 136
Replication Manager logging enhancement 136
Script editors 137
Replication Manager features supported 138
Support for new datatypes 138
Support for DirectConnect 138
Replication support 139
Routes 139
Troubleshooting tools 140
Connection status hide options 143
Warm standby wizards 144
Thread management 147

CHAPTER 10 Introducing Replication Monitoring Services 149
Introducing Replication Monitoring Services 149
Monitoring servers in the replication environment 150
Software requirements and compatibilities 151
Installation 151
Starting and stopping RMS 151
Connecting to RMS in Sybase Central 152
Monitoring a replication environment using RMS 153
Adding and dropping servers for monitoring in Sybase Central 153
Viewing monitored objects in Sybase Central 155
Setting configuration parameters for monitored replication objects 155
Monitoring a logical group of servers 156
Suspending or resuming components in the replication environment 157
Shutting down monitored servers 158
Generating rollup status for servers 158

Generating latency and heartbeat information 159
Adding event triggers 159

CHAPTER 11 New Features in Replication Server Version 12.6 161
MultiSite availability (MSA)..... 161
Support for symmetric multiprocessors (SMP)..... 162
The embedded RSSD (ERSSD) 164
Performance enhancements 164
Better management of empty transactions..... 164
Internal commit control for parallel processing..... 164
New Replication Server configuration parameters 165
New database configuration parameters..... 165
Changed database configuration parameters 166
date and time datatypes..... 167
Support for sending encrypted passwords..... 167
New bulk materialization method 168
Chinese character set (GB18030) support..... 169

Index 171

About This Book

Audience

This book is intended for customers who are installing and using Replication Server® 15.5 and the subsequent EBFs.

How to use this book

This book describes the new Sybase® Replication Server version 15.5 features.

This book also provides information about the features in Replication Server versions 15.2, 15.1, 15.0.1, 15.0, and 12.6.

- Chapter 1, “New Features in Replication Server Version 15.5,” describes the Replication Server version 15.5 features.
- Chapter 2, “New Features in Replication Manager 15.5,” describes the Replication Manager version 15.5 features.
- Chapter 3, “New Features in Replication Server Version 15.2,” describes the Replication Server version 15.2 features.
- Chapter 4, “New Features in Replication Server Version 15.1,” describes the Replication Server version 15.1 features.
- Chapter 5, “New Features in Replication Manager 15.1,” describes the Replication Manager version 15.1 features.
- Chapter 6, “New Features in Replication Server Version 15.0.1,” describes the Replication Server version 15.0.1 features.
- Chapter 7, “New Features in Replication Manager 15.0.1,” describes new features introduced in Replication Manager version 15.0.1.
- Chapter 8, “New Features in Replication Server Version 15.0,” describes the Replication Server version 15.0 features.
- Chapter 9, “New Features in Replication Manager 15.0,” describes the new features introduced in Replication Manager version 15.0.
- Chapter 10, “Introducing Replication Monitoring Services,” introduces the new component in Replication Server 15.0. The Replication Monitoring Services is a middle-management layer for monitoring large and complex replication environments.

- Chapter 11, “New Features in Replication Server Version 12.6,” describes the new and changed features in Replication Server version 12.6.

Related documents

The Sybase Replication Server documentation set consists of:

- 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.
- *Configuration Guide* for your platform – describes configuration procedures for Replication Server and related products.
- *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.
- *New Features Guide* (this book) – describes the new features in Replication Server.
- *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.
- *Design Guide* – contains information about designing a replication system and integrating heterogeneous data servers into a replication system.
- *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* – contains the syntax and detailed descriptions of Replication Server commands in the Replication Command Language (RCL); Replication Server system functions; Sybase Adaptive Server commands, system procedures, and stored procedures used with Replication Server; Replication Server executable programs; and Replication Server system tables.

- *Troubleshooting Guide* – contains information to aid in diagnosing and correcting problems in the replication system.
- *System Tables Diagram* – illustrates system tables and their entity relationships in a poster format. Available only in print version.
- 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 Certification Report.
3. In the Certification Report filter select a product, platform, and timeframe and then click Go.
4. Click a 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.

1 Point your Web browser to Technical Documents at http://www.sybase.com/support/techdocs/.

2 Click MySybase and create a MySybase profile.

Sybase EBFs and software maintenance

❖ **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

The following style conventions are used in this book:

- In a sample screen display, commands you should enter exactly as shown are in:

```
this font
```

- In a sample screen display, words that you should replace with the appropriate value for your installation are shown in:

  this font

- In the regular text of this document, the names of files and directories appear in italics:

  /usr/u/sybase

- The names of programs, utilities, procedures, and commands appear in this type:

  bcp

The conventions for syntax statements in this manual are as follows:

Table 1: Syntax conventions

Key	Definition
command	Command names, command option names, utility names, utility flags, and other keywords are in Arial.
variable	Variables, or words that stand for values that you fill in, are in italics.
{ }	Curly braces indicate that you choose at least one of the enclosed options. Do not include braces in your option.
[ ]	Brackets mean choosing one or more of the enclosed options is optional. Do not include brackets in your option.
( )	Parentheses are to be typed as part of the command.
|	The vertical bar means you can select only one of the options shown.
,	The comma means you can choose as many of the options shown as you like, separating your choices with commas to be typed as part of the command.

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.

CHAPTER 1

New Features in Replication Server Version 15.5

Topic	Page
Replication Server 15.5 licenses	2
Replication to Sybase IQ	2
Heterogeneous replication support	4
Performance enhancements	7
Usability and processes enhancements	14
Changes to default settings and reserved words	24
Adaptive Server support	26
Mixed-version environments	31
Operating system and platform support	32

Replication Server 15.5 licenses

Replication Server 15.5 is released as two separate product editions that bundle different base and optional features that require separate licences.

Table 1-1: Replication Server licenses

License	Description
REP_SERVER	Replication Server features, excluding the REP_HVAR_ASE, REP_EC_ORA, and REP_RTL_IQ features.
REP_HVAR_ASE	"Replication Server – Advanced Services Option" includes several Replication Server performance enhancements. See "Replication Server – Advanced Services Option" on page 8.
REP_EC_ORA	ExpressConnect for Oracle provides Replication Server with the capability to connect directly to Oracle. See the Replication Server Options 15.5 documentation.
REP_RTL_IQ	Real-Time Loading (RTL) feature for replicating to Sybase IQ. See "Replication to Sybase IQ" on page 2.

The licenses described in Table 1-1are bundled into product editions described in Table 1-2.

Table 1-2: Replication Server product editions

Edition	Base features	Optional features
Enterprise Edition	REP_SERVER	• REP_HVAR_ASE • REP_EC_ORA
Real-Time Loading Edition	• REP_SERVER • REP_RTL_IQ • REP_HVAR_ASE	None

Note You cannot use the "Replication Server – Real-Time Loading Edition" product edition to replicate to Adaptive Server.

Replication to Sybase IQ

In versions earlier than 15.5, Replication Server sends each replication operation to the replicate database directly, row-by-row and in log order in a continuous replication mode.

In version 15.5, you can replicate to Sybase IQ from Adaptive Server using real-time loading (RTL). When replicating into Sybase IQ replicate databases with identical database schema, Replication Server achieves better performance than with the continuous replication mode. RTL uses these processes, which result in data reduction:

- Compilation – rearranges replicate data, by clustering it by each table, and each insert, update, and delete operation, and then compiling the operations into net-row operations.
- Bulk apply – applies the net result of the compilation operations in bulk using the most efficient bulk interface for the net result. Replication Server uses an in-memory net-change database to store the net row changes which it applies to the replicate database.

Instead of sending every logged operation, compilation removes all the intermediate operations and sends only the final states of a replicated transaction. Depending on the application, this generally means a much smaller amount of data is processed.

As Replication Server compiles and combines a larger number of transactions into a group, bulk operation processing improves; therefore, replication throughput and performance also improves.You can control the amount of data that is grouped together for bulk apply by adjusting group sizes.

See Chapter 11, “Sybase IQ Replicate Data Server Issues” in the *Replication Server Heterogeneous Replication Guide* to connect and replicate to Sybase IQ.

Licensing

Replication to Sybase IQ using real-time loading (RTL) is available in the Replication Server – Real-Time Loading Edition. See “Replication Server 15.5 licenses” on page 1.

Download the “Replication Server – Real-Time Loading Edition license from the Sybase Product Download Center (SPDC) at https://sybase.subscribenet.com, to which you are automatically enrolled in when you purchase a Sybase product. Log in to SPDC by using the information in your SPDC welcome e-mail message. Contact your Sybase representative for more information.

Note If you have purchased your Sybase software from a Sybase reseller, you receive a Web key rather than an e-mail message.

For information about licensing and Sybase Software Asset Management (SySAM), see the *Sybase Software Asset Management Users Guide*.

Database and platform support

You can use RTL to replicate into Sybase IQ 12.7 ESD #3 and later. You can achieve optimal performance using 64-bit hardware platforms. See “64-bit support” on page 32.

Replication Server 15.5 supports replication to Sybase IQ only from Adaptive Server version 15.0.3 or version 15.5 and later as the primary database.

Heterogeneous replication support

Replication Server 15.5 extends support for heterogeneous databases with:

- Heterogeneous parallel DSI
- Heterogeneous warm standby for Oracle
- Controlling trigger execution at the Oracle replicate database

Heterogeneous parallel DSI

You can configure Replication Server to apply transactions to the replicate data server in a heterogeneous environment using parallel Data Server Interface (DSI) threads. Applying transactions in parallel increases the speed of replication, yet maintains the serial order of the transactions as they are applied at the primary site.

Replication Server 15.5 introduces a transaction serialization method to improve performance and data integrity with parallel DSI for heterogeneous replication. In the new method, wait_after_commit, each thread waits to begin its first batch until the previous thread has completely committed. Sybase recommends that you use wait_after_commit serialization method for databases that use multiversion concurrency control (MVCC) or optimistic concurrency control (OCC), such as an Oracle database. Otherwise, you can use wait_for_commit as the default method.

Table 1-3: Support for parallel DSI for non-ASE databases by Replication Server

Database	Internal commit control method	External commit control method
Oracle	Yes	No
Microsoft SQL Server	Yes	Yes
IBM DB2 UDB	Yes	Yes

See the *Replication Server Heterogeneous Guide* for detailed information about using parallel DSI for non-ASE databases.

New configuration parameter

Table 1-4 describes dsi_max_cmds_in_batch, which enables larger command batching.

Table 1-4: New configuration parameter

Parameter	Value	Default	Description
dsi_max_cmds_in_batch	Integer	100	Defines maximum number of source commands for which output commands can be batched. Range: 1 – 1000

Changes to existing configuration parameters

Table 1-5 describes the changes to the existing configuration parameters to enable larger transaction grouping.

Table 1-5: Existing configuration parameter changes

Parameter	Value	Default	Description
dsi_max_xacts_in_group	Integer	20	Specifies the maximum number of transactions in a group. Larger numbers may improve data latency at the replicate database. Range: 1 – 1000
dsi_serialization_method	no_wait wait_for_start wait_for_commit wait_after_commit	wait_for_commit	Specifies the method used to maintain serial consistency between parallel DSI threads when applying transactions to a replicate data server. In all cases, commit order is preserved.

Function-string changes for internal commit control for Adaptive Server

Replication Server uses the rs_dsi_check_thread_lock function to check whether the current DSI executor thread is blocking another replicate database process. In Replication Server 15.5, the rs_dsi_check_thread_lock function string has been modified to detect deadlocks.

See Chapter 4, “Replication Server System Functions” in the *Replication Server Reference Manual* for information about rs_dsi_check_thread_lock.

Heterogeneous warm standby for Oracle

In Replication Server 15.5, you can create and maintain warm standby applications for Oracle database. There are tasks that the Replication Server system administrator has to perform manually to create the warm standby setup for Oracle databases. See Chapter 12, “Managing Heterogeneous Warm Standby for Oracle” in the *Replication Server Heterogeneous Replication Guide* for complete information.

A new configuration parameter, ra_standby, has been added to support Oracle warm standby applications. This parameter identifies whether Replication Agent for Oracle works in a standby mode. See Chapter 2, “Configuration Parameters” in the *Replication Agent 15.5 Reference Manual*.

For Adaptive Server, administrators can continue using the rs_init utility to set warm standby environment. See the *Replication Server Administration Guide Volume 2* for detailed information.

Product compatibility

Table 1-6 describes the additional replication components required for supporting the Replication Server 15.5 heterogeneous warm standby feature for Oracle database.

Table 1-6: Product compatibility for Oracle warm standby support

Database server version	Replication Agent version	ECDA Option version	ExpressConnect version
Oracle 10g, 11g	Replication Agent for Oracle 15.5	ECDA 15.0 ESD #3	ExpressConnect for Oracle 15.5

Controlling trigger execution at the Oracle replicate database

You can now control trigger firing at the session level or connection level, each time a PL/SQL command is executed against an Oracle replicate database. Controlling trigger execution at the replicate database eliminates duplication and inaccuracy that were caused in earlier versions due to the absence of trigger control at the replicate database.

The RS_TRIGGER_CONTROL package supports this feature and is automatically installed when a connection to the replicate Oracle database is created through connection profiles. The rs_triggers_reset system function has also been modified to support this feature, and you can now set the dsi_keep _triggers connection parameter to off to disable triggers in Oracle. Re-create every trigger needed to be controlled at the replicate database, adding the trigger control statement at the beginning of your trigger action.

Databases supported: Oracle 10g and 11g

See “Settings for trigger firing,” in Chapter 10, “Oracle Replicate Data Server Issues” in the *Replication Server Heterogeneous Replication Guide*.

Performance enhancements

Replication Server 15.5 includes these performance enhancements:

- Replication Server – Advanced Services Option
- Dynamic SQL enhancements
- Function-string efficiency improvements

Replication Server – Advanced Services Option

Replication Server 15.5 includes the Replication Server – Advanced Services Option which is a separately licensed product option for Replication Server containing these enhancements:

- High Volume Adaptive Replication
- Enhanced DSI efficiency
- Enhanced RepAgent executor thread efficiency
- Enhanced distributor thread read efficiency
- Enhanced memory allocation
- Increasing queue block size

Licensing

The Replication Server – Advanced Services Option is:

- Available as an option in the Replication Server – Enterprise Edition product edition, as the separate REP_HVAR_ASE license.

  If you are using the Replication Server – Enterprise Edition, download the REP_HVAR_ASE license file from the Sybase Product Download Center (SPDC) at https://sybase.subscribenet.com to activate any of the enhancements in the Advanced Services Option.

- Bundled in the Replication Server – Real-Time Loading Edition product edition.

  To activate Replication Server – Real-Time Loading Edition, download the Replication Server – Real-Time Loading Edition product edition license from the SPDC.

See “Replication Server 15.5 licenses” on page 1.

You are automatically enrolled in the SPDC when you purchase a Sybase product. Log in to SPDC by using the information in your SPDC welcome e-mail message. Contact your Sybase representative for more information.

Note If you have purchased your Sybase software from a Sybase reseller, you will receive a Web key rather than an e-mail message.

For information about licensing and Sybase Software Asset Management (SySAM), see the *Sybase Software Asset Management Users Guide*.

High Volume Adaptive Replication

In versions earlier than 15.5, Replication Server sends each replication operation to the replicate database directly, row-by-row and in log order in the continuous replication mode.

Replication Server 15.5 includes High Volume Adaptive Replication (HVAR) which uses compilation and bulk apply processes that result in data reduction and achieves better performance compared to the continuous replication mode:

- Compilation – rearranges replicate data, by clustering it by each table, and each insert, update, and delete operation, and then compiling the operations into net-row operations.
- Bulk apply – applies the net result of the compilation operations in bulk using the most efficient bulk interface for the net result. Replication Server uses an in-memory net-change database to store the net row changes which it applies to the replicate database.

Instead of sending every logged operation, compilation removes all the intermediate operations and sends only the final states of a replicated transaction. Depending on the application, this generally means a much smaller amount of data is processed.

As Replication Server compiles and combines a larger number of transactions into a group, bulk operation processing improves; therefore, replication throughput and performance also improves. You can control the amount of data that is grouped together for bulk apply by adjusting group sizes.

HVAR is especially useful for creating online transaction processing (OLTP) archiving and reporting systems where the replicate databases have the same schemas as the primary databases.

See “High Volume Adaptive Replication to Adaptive Server,” in Chapter 4, “Performance Tuning” in the *Replication Server Administration Guide Volume 2*.

System table support

Replication Server uses the rs_tbconfig table to store support table-level configuration parameters, and the ref_objowner and ref_objname columns in the rs_columns table to support referential constraints.

See Chapter 8, “Replication Server System Tables” in the *Replication Server Reference Manual* for full table descriptions.

Database and platform support

HVAR supports replication into Adaptive Server 12.5 and later, and you can achieve optimal performance using 64-bit hardware platforms. See “64-bit support” on page 32.

Enhanced DSI efficiency

Enhanced Data Server Interface (DSI) efficiency improves performance by reducing data replication latency. which decreases the length of time that Replication Server waits for results from the replicate data server through the ct_results routine, and subsequently reduces the length of time the data server waits for Replication Server.

See “Enhanced DSI efficiency,” in Chapter 4, “Performance Tuning” in the *Replication Server Administration Guide Volume 2* for configuration information.

Enhanced RepAgent executor thread efficiency

Enhanced RepAgent executor thread efficiency improves performance by using the NRM thread 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.

See “Enhanced RepAgent Executor thread efficiency,” in Chapter 4, “Performance Tuning” in the *Replication Server Administration Guide Volume 2* to enable the NRM thread and specify the memory available to the RepAgent Executor thread.

Enhanced distributor thread read efficiency

With Replication Server 15.5, the distributor (DIST) thread reads SQL statements from the Stable Queue Transaction thread (SQT) cache directly. This reduces contention between inbound and outbound queues, and improves Replication Server performance.

See “Enhanced distributor thread read efficiency,” in Chapter 4, “Performance Tuning” in the *Replication Server Administration Guide Volume 2*.

Enhanced memory allocation

Enhanced memory allocation improve performance by allocating memory in larger chunks and therefore reducing the number of memory allocations.

See “Enhanced memory allocation,” in Chapter 4, “Performance Tuning” in the *Replication Server Administration Guide Volume 2*.

Increasing queue block size

In Replication Server version 15.5 and later, you can increase the block size to process more transactions in a single block.

The queue block size is the number of bytes in a contiguous block of memory used by stable queue structures. In earlier versions of Replication Server, the queue block size is fixed at 16KB.

Note You must suspend incoming data flow and routes, and quiesce Replication Server before modifying the queue block size. After executing the command to set the block size, Replication Server automatically shuts down, and you must restart Replication Server for the new block size to take effect.

See “Increasing queue block size,” in Chapter 4, “Performance Tuning” in the *Replication Server Administration Guide Volume 2* for recommendations, prerequisites, and instructions for configuring the queue block size.

Dynamic SQL enhancements

Dynamic SQL enhances performance by allowing the Replication Server Data Server Interface (DSI) module to prepare dynamic SQL statements at the replicate database and to run them repeatedly. See “Dynamic SQL for enhanced Replication Server performance,” in Chapter 4, “Performance Tuning” in the *Replication Server Administration Guide Volume 2*.

Replication Server 15.5 includes enhancements to dynamic SQL that:

- Optimize dynamic SQL statement execution.
- Utilize dynamic SQL even if the replicate minimal columns clause is used at the same time.
- Extend the replicate_minimal_columns parameter to connections.

Optimizing Dynamic SQL statement execution

In versions earlier the 15.5, Replication Server generates the language command, the prepared statement, and the execute statement each time DSI executes dynamic SQL statements, but Replication Server only uses the language command when the dynamic SQL command fails.

In version 15.5, Replication Server optimizes dynamic SQL statements by:

- Generating the language command only when the dynamic SQL command fails.
- Generating the prepared statement only once when the prepared statement is used for the first time.

Using *replicate minimal columns* with dynamic SQL

With version 15.5, replication processing does not skip Dynamic SQL even if the replicate minimal columns clause is enabled and Replication Server uses replicate minimal columns and Dynamic SQL effectively at the same time.

In Replication Server versions earlier than 15.5, if the replicate minimal columns clause is used in a replication definition, the columns that are not changed are not available to the DSI, and dynamic SQL is skipped.

Extending *replicate_minimal_columns* to connections

With version 15.5, Replication Server extends the replicate_minimal_columns parameter to connections in all situations, so that DSI can use the parameter to determine whether to use minimal columns when there is no replication definition, or when the replication definition does not contain the replicate minimal columns clause

In Replication Server versions earlier than 15.5, you could only use replicate_minimal_columns in warm standby situations.

For example, to enable replicate_minimal_columns for the connection to the pubs2 database in the SYDNEY_DS data server, enter:

```
alter connection to SYDNEY_DS.pubs2
set replicate_minimal_columns to 'on'
```

Use admin config to display replicate_minimal_columns configuration information to verify that version 15.5 extends the parameter to connections.

Note When you set dsi_compile_enable to on, Replication Server ignores what you set for replicate_minimal_columns.

Function-string efficiency improvements

Replication Server 15.5 includes enhancements to function string processing commands, stored procedures, and a system table to allow you to identify specific function strings that do not need to be applied to replicate databases.

In versions earlier than 15.5, Replication Server executes all function strings for all replicate databases although many of these function strings, such as those without output commands, do not apply to non-ASE databases. Preventing these function strings from being executed reduces processing overhead and simplifies the replication environment.

Changes to function string processing commands

Replication Server 15.5 extends the scope of the none parameter to apply to all functions, and provides the flexibility to identify which function strings Replication Server can avoid executing on replicate databases.

In versions earlier than 15.5, the none parameter of the alter function string and create function string commands applies only to the rs_writetext function, and instructs Replication Server not to replicate a text, unitext, or image column value.

Use the none parameter to identify class-level and table-level function strings that do not have output commands. Replication Server does not execute these function strings on replicate databases.

See "Using output templates," in Chapter 2, "Customizing Database Operations" in the *Replication Server Administration Guide Volume 2*.

There is no change in the syntax of the alter function string and create function string commands. See Chapter 3, "Replication Server Commands" in the *Replication Server Reference Manual*.

Changes to stored procedures

To support the enhancements to function string processing, Replication Server 15.5 extends the rs_helpfstring and rs_helpclassfstring stored procedures:

- rs_helpfstring – displays function strings for table-level functions and displays those without output commands
- rs_helpclassfstring – displays function strings for class-level functions and those without output commands

See Chapter 6, "Adaptive Server Stored Procedures" in the *Replication Server Reference Manual*.

Changes to *rs_funcstrings* system table

Replication Server 15.5 adds the 0x08 bit to the attributes column in the rs_funcstrings table to support the function string enhancements.

See "rs_funcstrings," in Chapter 8, "Replication Server System Tables" in the *Replication Server Reference Manual*.

Usability and processes enhancements

Replication Server 15.5 includes these enhancements to usability and processes:

- Replication definition change request process enhancements
- Scheduling replication tasks
- Delaying replication
- Resynchronizing replicate databases
- Row count validation enhancements
- Implementing a reference replication environment
- admin who enhancements
- Resetting database generation numbers
- Injecting rs_ticket markers

Replication definition change request process enhancements

Replication Server 15.5 includes enhancements for requesting changes to replication definitions that automatically coordinate the propagation of replication definition changes and data replication. You can request replication definition changes directly 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.

You can:

- Issue a replication definition command directly from a primary database.
- Use an alter replication definition command to instruct Replication Server to suspend the target DSIs after Replication Server applies all data for the old replication definition version at the target database. This provides a window for you to alter the target schema and alter customized function strings before the data for the new replication definition version arrives.
- Verify that Replication Server can execute a replication definition request successfully by executing the request without changing any data.
- Drop columns from a replication definition using alter replication definition.
- 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.

These enhancements make coordinating database schema changes and replication definition changes more convenient because there is no downtime for the primary database, and minimal or no downtime for the replicate database. When you issue a replication definition change request, Replication Server determines if there is a need to create a new replication definition version based on the type of change requested. If Replication Server creates a new replication definition version, primary updates before the replication definition change request automatically use the old replication definition version, while primary updates after the replication definition change request use the new replication definition version.

Without these enhancements, to coordinate schema changes and replication definition changes, you must quiesce primary updates, wait until all the data associated with a primary table or stored procedure is processed through the entire replication system, shut down Replication Agent, alter the primary schema, alter the replication definition, alter any customized function strings, wait for the changes to replicate, alter the replicate schema, and then restart Replication Agent and resume primary updates.

See "Replication definition change request process," in Chapter 9, "Managing Replicated Tables" in the *Replication Server Administration Guide Volume 1* for the commands, procedures, and a user scenario for the enhanced replication definition change request process.

System table changes

To support the enhanced replication definition change process, Replication Server includes changes to the rs_columns, rs_locater, and rs_objects system tables:

- rs_columns – version column added.
- rs_locater – C, F, and S values added to type column.
- rs_objects – active_inbound, attributes2, and version columns added.

See Chapter 8, "Replication Server System Tables" in the *Replication Server Reference Manual*.

Product compatibility

You can change replication definitions on primary databases for Adaptive Server, and for all versions of Microsoft SQL Server and Oracle that Replication Server supports. See the Replication Server Options documentation for supported versions.

Mixed-version support

Suppose you execute the alter replication definition with the drop *column name* clause and there is a subscription to the replication definition from a replicate site with a site version earlier than 1550, the primary Replication Server rejects the alter replication definition command.

Issuing any alter replication definition request with the with DSI_suspended parameter does not suspend any replicate DSI with site versions earlier than 1550.

Scheduling replication tasks

Replication Server 15.5 lets you schedule replication tasks.

For example, you can report on a specific state of the replicate database while the replicate database is 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 from the previous day. You can do this by creating schedules to suspend and resume connections to the replicate database at specific times of the day.

See “Scheduling replication tasks,” in Chapter 13, “Scheduling Replication Tasks” in the *Replication Server Administration Guide Volume 1*.

System tables changes

Replication Server 15.5 includes the rs_schedule and rs_scheduletxt system tables to store schedules you create.

See Chapter 8, “Replication Server System Tables” in the *Replication Server Reference Manual*.

Delaying replication

Replication Server 15.5 lets you delay replication by a fixed period of time.

You can use a replicate database as a failback system by delaying updates for a certain amount of time behind the primary database to recover from any human error committed on the primary database.

See “Delaying replication,” in Chapter 13, “Scheduling Replication Tasks” in the *Replication Server Administration Guide Volume 1*.

Resynchronizing replicate databases

Replication Server 15.5 introduces database resynchronization, which allows you to rematerialize your replicate database and resume further replication without data loss or inconsistency, and without forcing a quiesce of your primary database.

Database resynchronization is based on obtaining a dump of data from a trusted source and applying the dump to the target database you want to resynchronize. Database resynchronization includes these steps:

1 Stop replication processing by suspending Replication Agent.

2 Place Replication Server in resync mode. In resync mode, Replication Server skips transactions and purges replication data from replication queues in anticipation of the replicate database being repopulated from a dump taken from the primary database or trusted source.

3 Restart Replication Agent and send a resync database marker to Replication Server to indicate that a resynchronization effort is in progress.

4 Obtain a dump from the primary database.

5 When Replication Server detects a dump marker that indicates the completion of the primary database dump, Replication Server stops skipping transactions and can determine which transactions to apply to the replicate database.

6 Apply the dump to the replicate database.

7 Resume replication.

Replication Agent support

The full functionality of database resynchronization, such as automatic generation of the resync marker, requires Replication Agent support. Replication Agent 15.5 for Oracle supports the full functionality of database resynchronization. See Chapter 13, “Resynchronizing Oracle Replicate Databases” in the *Replication Server Heterogeneous Replication Guide* and the Replication Agent documentation.

RepAgent, the Replication Agent for Adaptive Server, is scheduled to support the full functionality of database resynchronization in a version later than Adaptive Server 15.5. See “Resynchronizing replicate databases,” in Chapter 7, “Replication System Recovery” in the *Replication Server Administration Guide Volume 2* to resynchronize Adaptive Server databases without support from RepAgent.

Product compatibility

Table 1-7 lists the versions of Oracle, Replication Agent for Oracle, ECDA Option for Oracle, and ExpressConnect for Oracle that support the resynchronization of Oracle databases. With Replication Server Options 15.5, ExpressConnect for Oracle replaces ECDA Option for Oracle.

See the Replication Server Options documentation and the *Replication Server Heterogeneous Replication Guide*.

Table 1-7: Product compatibility for resynchronizing Oracle databases

Database server version	Replication Agent version	ECDA Option version	ExpressConnect version
Oracle 10g, 11g	Replication Agent for Oracle 15.5	ECDA 15.0 ESD #3,	ExpressConnect for Oracle 15.5

System table support

In the rs_databases table, the datatype of dist_status and src_status columns has been changed from tinyint to cs_int, and the “0x100 – waiting for a resync marker” status has been added to dist_status.

Row count validation enhancements

Replication Server 15.5 changes the default error actions for the 5185 and 5187 error numbers from “warn” to “stop replication” and adds 5203 for Replication Server error classes.

Table 1-8: 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 '*dataserver.database*'. The command impacted *x* rows but it should impact *y* 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.
5187	Row count mismatch for the autocorrection delete command executed on '*dataserver.database*'. The command deleted *x* rows but it should delete *y* 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.
5203	Row count mismatch on '*dataserver.database*'. The delete command generated by dsi_command_convert deleted *x* rows, whereas it should delete *y* rows.	stop_replication	This message appears if the number of rows deleted is different from the expected number of rows to be deleted.

Use the assign action command at the primary site for the Replication Server error class to override the default error action.

With Replication Server 15.5, you can alter a Replication Server error class using alter error class.

See “Row count validation for non-SQL statement replication,” in Chapter 2 “New Features in Replication Server Version 15.2” in this book and Chapter 6, “Handling Errors and Exceptions” in the *Replication Server Administration Guide Volume 2*.

For details about commands, parameters, stored procedures, and system tables, see the *Replication Server Reference Manual*.

Implementing a reference replication environment

Replication Server 15.5 includes a toolset to enable you to quickly set up a reference implementation of Adaptive Server to Adaptive Server and Oracle to Oracle replication using the products available in your environment. A reference replication environment lets you collect statistics to identify performance issues, and demonstrates Replication Server features and functionalities.

Use the toolset to:

1 Build Replication Server and the primary and replicate databases.

2 Configure the replication environment.

3 Perform simple transactions on the primary database and replicate the changes by database-level replication.

4 Collect statistics and monitors counters from the replication processing in step 3.

5 Clean up the reference replication environment.

See Appendix D, "Implementing a Reference Replication Environment" in the *Replication Server Administration Guide Volume 2* to build, configure, and use a reference replication environment.

Note The reference implementation builds a replication environment containing a single Replication Server, primary database server, and replicate database server. You cannot configure the reference environment topology for multiple replication system components.

Platform support

Reference implementation is available for all platforms that Replication Server 15.5 supports. However, to set up the reference environment on any Microsoft Windows platform that Replication Server supports, you must use Cygwin to run the reference implementation scripts. See the Cygwin Web site at http://www.cygwin.com/.

Components for reference implementation

Before you can implement a reference environment, you must have supported versions of the required components.

Adaptive Server

You can build a reference implementation environment for Adaptive Server to Adaptive Server replication with the versions of Replication Server and Adaptive Server listed in Table 1-10.

Table 1-9: Supported product component versions for Adaptive Server reference implementation

Replication Server	Adaptive Server
15.5	15.0.3, 15.5

Oracle

You can also build a reference implementation environment for Oracle to Oracle replication with the versions of Replication Server, Oracle, Replication Agent for Oracle, and ECDA Option for Oracle listed in Table 1-10.

Table 1-10: Supported product component versions for Oracle reference implementation

Replication Server	Oracle	Replication Agent for Oracle	ECDA Option for Oracle
15.5	10.2	15.2	15.0 ESD #3

For example, you can build a reference implementation environment for Oracle with Replication Server 15.5, Oracle 10.2, Replication Agent 15.2, and ECDA Option for Oracle 15.0 ESD #3.

admin who enhancements

Replication Server 15.5 lets you specify connection identifiers when you execute admin who on any thread module. Instead of seeing information for all the connections of a thread module, you can display only the admin who execution results of a specific connection by specifying connection identifiers for these thread modules:

- DIST – Distributor
- DSI – Data Server Interface
- RSI – Replication Server Interface
- SQM – Stable Queue Manager
- SQT – Stable Queue Transaction

See “admin who,” in Chapter 3, “Replication Server Commands” in the *Replication Server Reference Manual* for the full syntax and examples.

Note You cannot use the no_trunc option if you specify connection identifiers.

If you specify connection identifiers and Replication Server cannot find information that fulfills the criteria, the output does not display any record.

The output for the DIST and DSI thread modules includes additional columns:

Table 1-11: Additional columns displayed for DIST and DSI thread modules

Thread	Column name	Description	Value
DIST	RSTicket	The number of rs_ticket subcommands that have been processed by a DIST thread, if the Replication Server stats_sampling parameter is on.	Minimum: 0 Maximum: 2^{63}-1 Default: 0
DIST	SqtMaxCache	Maximum SQT cache memory for the database connection, in bytes.	The default, 0, means that the current setting of sqt_max_cache_size is used as the maximum cache size for the connection. Default: 0
DSI	RSTicket	The number of rs_ticket subcommands that have been processed by a DSI queue manager, if the Replication Server stats_sampling parameter is on.	Minimum: 0 Maximum: 2^{63}-1 Default: 0
DSI	dsi_rs_ticket_report	Determines whether to call function string rs_ticket_report. rs_ticket_report is invoked when dsi_rs_ticket_report is set to on.	on or off Default: off

Resetting database generation numbers

Each primary database in a replication system includes a database generation number. This number is stored both in the database and in the RSSD of the Replication Server that manages the database.

Any time you load a primary database for recovery, you must change the database generation number, as instructed in the recovery procedure you are using.

The maximum value for the database generation number is 65,535. Sybase recommends that you do not increase the number to high values unless absolutely necessary. In Replication Server 15.5 and later, you can reset the database generation number to 0 before it reaches the maximum of 65,535. In Replication Server versions earlier than 15.5, you must rebuild the replication environment after resetting the database generation number.

See “Resetting database generation numbers,” in Chapter 7, “Replication System Recovery” in the *Replication Server Administration Guide Volume 2*.

Injecting *rs_ticket* markers

Replication Server 15.5 introduces a system command to identify performance issues in data replication. sysadmin issue_ticket injects an rs_ticket marker into the inbound queue, bypassing the need for the RepAgent on the primary database to process the ticket. rs_ticket processes from this point by appending the system time at the end of the marker as it passes through certain modules on the Replication Server. The information gathered by this marker is stored in the rs_ticket_history table in the replicated database.

See Chapter 3, “Replication Server Commands” in the *Replication Server Reference Manual*.

Changes to default settings and reserved words

Replication Server 15.5 includes changes to default settings and values to accommodate the enhancements in Replication Server and improve performance.

Changes to parameter default values

Except for memory_limit and smp_enable, if you upgrade to Replication Server 15.5, the only values that are set to the new defaults are those that used the defaults in the earlier version.

Table 1-12: Changes to parameter default values

Parameter	Old value	New value	Downgrading from version 15.5
exec_cmds_per_timeslice	5	2,147,483,647	Downgrading does not change the value you have set.
init_sqm_write_delay	1000 milliseconds	100 milliseconds for all platforms	Downgrading does not change the value you have set.
init_sqm_write_max_delay	10,000 milliseconds	1000 milliseconds for all platforms	Downgrading does not change the value you have set.
memory_limit	80MB	2,047MB Upgrading increases the value to the new default if the earlier value was less than 2,047MB.	If the value you set is larger than 2,047MB, downgrading resets the value to 2,047MB to protect against overflow.
smp_enable	off	on Upgrading does not change the setting to on if your original setting was off.	Downgrading does not change the value you have set.
sqt_max_cache_size	1,048,576 bytes (1MB)	• 32-bit platforms: 1,048,576 bytes (1MB) • 64-bit platforms: 20,971,520 bytes (20MB)	If the value set is larger than 2,147,483,647 bytes, downgrading resets the value to 2,147,483,647 bytes to protect against overflow.
sts_full_cache_*system_table_name* for these system tables: rs_columns, rs_functions, rs_objects	off – not fully cached	on – fully cached	Downgrading does not change your settings.

See Chapter 3, “Replication Server Commands” in the *Replication Server Reference Manual* for descriptions of the parameters, examples and usage information.

Changes to the RSSD locking schema

To reduce contention and improve performance, the default locking schema for the system tables in the Replication Server System Database (RSSD) is row-level locking when you install or upgrade to version 15.5. The locking schema does not change when you downgrade from version 15.5.

There is no change in the default locking schema for the Embedded Replication Server System Database (ERSSD), which is row-level locking.

New reserved words

See “Reserved words,” in Chapter 2, “Topics” in the *Replication Server Reference Manual* which is updated with new reserved Replication Server keywords.

Adaptive Server support

Replication Server 15.5 includes these enhancements to support Adaptive Server replication:

- bigdatetime and bigtime datatypes replication
- Deferred name resolution
- SQL statement replication threshold setting
- Incremental data transfer
- In-memory and relaxed-durability databases

bigdatetime and *bigtime* datatypes replication

Replication Server 15.5 supports replication of the Adaptive Server 15.5 bigdatetime and bigtime. Replicate these datatypes to replicate databases and warm standby databases by specifying the datatypes in replication definitions, function replication definitions, and subscriptions.

bigdatetime and bigtime allow Adaptive Server to store data and time data up to microsecond precision. bigdatetime corresponds to the TIMESTAMP datatype, and bigtime corresponds to the TIME datatype in Sybase IQ and Sybase SQL Anywhere.

See "Support for bigdatetime and bigtime datatypes," in Chapter 5, "Managing RepAgent and Supporting Adaptive Server" in the *Replication Server Administration Guide Volume 1* to use bigdatetime and bigtime.

Mixed-version information

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.

Deferred name resolution

Replication Server 15.5 supports the Adaptive Server 15.5 deferred name resolution feature.

Deferred name resolution lets you create stored procedures in Adaptive Server without resolving the objects used internally by these stored procedures. Adaptive Server postpones the object resolution phase until you first execute the stored procedure in Adaptive Server. Stored procedures execute normally when you execute them after the first time. See "Deferred Name Resolution for User-Defined Stored Procedures" in the *Adaptive Server Enterprise 15.5 New Features Summary*.

Replication Server issues

In Replication Server versions earlier than 15.5, you can set up a warm standby application and enable sp_reptostandby at the active database to allow replication of supported data definition language (DDL) commands to the standby database.

However on a standby database or a replicate database in a non-warm standby environment, you cannot create a stored procedure that references a temporary table, because Replication Server does not replicate temporary tables. The create stored procedure process must resolve the objects used internally by the stored procedure. However, there is no temporary table in the replicate or standby database, therefore, Replicate Server does not create the stored procedure in the replicate or standby database.

With support for deferred name resolution, Replication Server 15.5 allows the replication of stored procedures that refer to temporary tables, tables that do not exist, and procedures that do not exist, to replicate or standby databases.

See “Deferred name resolution,” in Chapter 5, “Managing RepAgent and Supporting Adaptive Server” in the *Replication Server Administration Guide Volume 1* to configure deferred name resolution in Replication Server.

SQL statement replication threshold setting

With Adaptive Server 15.0.3 ESD #1 and later, you can set the threshold at the database level or session level to trigger SQL statement replication without having to set the threshold on individual tables.

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. Earlier versions of Replication Server allowed you to set the threshold only at the table level.

See “Setting SQL statement replication threshold,” in Chapter 4, “Performance Tuning” in the *Replication Server Administration Guide Volume 2*.

Incremental data transfer

With Adaptive Server 15.5, you can transfer data incrementally from a table instead of having to transfer the whole table from one Adaptive Server to another. See Chapter 8, “Adding, Changing, Transferring, and Deleting Data,” in the *Adaptive Server Enterprise Transact-SQL Users Guide*.

Replication Server supports the data definition language related to the Adaptive Server 15.5 incremental data transfer feature. Replication proceeds normally for data modification operations performed on a replicate table marked for incremental transfer.

If you load a replicate table using the transfer table command, and the table has an unique index command and the data on the incremental transfer already exists on the table, Adaptive Server internally converts an insert command into an update command.

The transfer table command applies only to the data server and database where you initiated the transfer the first time.

If you mark tables for incremental transfer in the active database within a warm standby or multisite availability (MSA) environment, and then switch to the standby database if the active database terminates, incremental data transfer may not resume correctly at the standby database. Unlike the active database, the standby database does not have a record of the incremental data transfer activity. Therefore, you must also initialize the incremental data transfer on the standby database.

In-memory and relaxed-durability databases

Adaptive Server 15.5 introduces in-memory and relaxed-durability databases.

In-memory databases reside entirely in cache and do not use disk storage for data or logs, and therefore do not require disk I/O. This results in potentially better performance than a traditional disk-resident database, as well as other advantages. However, since an in-memory database exists only in cache, you cannot recover the database if the supporting host is shut down or the database fails.

With relaxed-durability databases, Adaptive Server extends the performance benefits of an in-memory database to disk-resident databases. Disk-resident databases perform writes to disk, and ensure that the transactional ACID (atomicity, consistency, integrity, and durability) properties are maintained. A traditional disk-resident database operates at full durability to guarantee transactional recovery from a server failure. Relaxed-durability databases trade the full durability of committed transactions for enhanced runtime performance for transactional workloads. A relaxed-durability database created with the no_recovery level is similar to an in-memory database: you cannot recover data or logs if the server terminates or is shut down. You can also create a relaxed-durability database with the at_shutdown level, where transactions are written to disk if there is a proper shutdown of the database.

See the *Adaptive Server Enterprise In-Memory Database Users Guide*.

Replication Server support

Replication Server supports as the replicate database:

- In-memory databases
- Relaxed-durability databases set with durability at no_recovery

The primary database must be a traditional full-durability, disk-resident database. For convenience, this document refers to relaxed-durability databases with durability set to non_recovery as "relaxed-durability databases."

You can initialize an in-memory and relaxed-durability database as a new replicate database by obtaining data, object schema, and configuration information from one of:

- A template database that retains basic information.
- A database dump from another database. Load the dump to the target in-memory database or relaxed-durability database.

  The dump source database can be another in-memory database, relaxed durability database, or a traditional disk-resident database.

In-memory and relaxed-durability databases lose their object definition, data, and RepAgent configuration once the host data server shuts down or restarts. You must reinitialize the in-memory or relaxed-durability database from the template or database dump from a source database.

See "Support for in-memory and relaxed-durability databases," in Chapter 5, "Managing RepAgent and Supporting Adaptive Server" in the *Replication Server Administration Guide Volume 1* to configure an in-memory or a relaxed-durability database as a replicate database.

Minimal DML logging and replication

To optimize the log records that are flushed to the transaction log on disk, Adaptive Server can perform minimal to no logging when executing some data manipulation language (DML) commands—insert, update, delete, and slow bcp—on all types of low-durability databases, such as in-memory databases and relaxed-durability databases set with durability of at_shutdown or no_recovery. You can perform minimal logging for DMLs on a per-database, per-table, and session-specific basis. See Chapter 3, "Minimally-logged DML" in the *Adaptive Server Enterprise In-Memory Database Users Guide*.

Note Minimal DML logging session-level settings take precedence over database-level settings and table-level settings.

Replication Server support

As replication uses full logging, replication and the minimal DML logging feature in Adaptive Server 15.5 are incompatible at the same level, such as the database level or table level. However, you can take advantage of the performance enhancements from minimal DML logging on some tables while allowing replication on others, as minimal DML logging and replication can coexist at different levels. See "Minimal DMLlogging and replication," in Chapter 5, "Managing RepAgent and Supporting Adaptive Server" in the *Replication Server Administration Guide Volume 1* for scenarios that result in incompatibility between replication and minimal DML logging.

Mixed-version environments

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

You must upgrade Replication Server to version 12.6 or later, set the site version to 12.6 or later, and upgrade routes to 12.6 or later, before you can upgrade to version 15.5.

See Chapter 3, "Upgrading or Downgrading Replication Server" in the *Replication Server Configuration Guide*.

Operating system and platform support

You can run Replication Server 15.5 on these operating systems:

- Microsoft Windows Server 2008 R2
- Microsoft Windows 7
- SuSe Linux Enterprise Server SLES 11

64-bit support

Replication Server 15.5 supports 64-bit computing platforms, which provide Replication Server with a large amount of virtual memory space and removes the maximum memory constraint of 2GB. In addition, all the available Replication Server counters are now defined as 64-bit, which allows high-precision computations in Replication Server.

Table 1-13 describes changes to certain configuration parameters that impact performance on 32-bit Replication Server and 64-bit Replication Server.

Table 1-13: Replication Server configuration parameters

Parameter	Description	Valid range for 32-bit (in bytes)	Valid range for 64-bit (in bytes)
dsi_sqt_max_cache_size	The maximum Stable Queue Transaction (SQT) cache size for the database connection. The default, 0, means the current setting of the sqt_max_cache_size parameter is used as the maximum cache size for the connection.	Min: 0 Max: 2147483647	Min: 0 Max: 2251799813685247
dist_sqt_max_cache_size	The maximum Stable Queue Transaction (SQT) cache size for the DIST connection. The default, 0, means the current setting of the sqt_max_cache_size parameter is used as the maximum cache size for the connection.	Min: 0 Max: 2147483647	Min: 0 Max: 2251799813685247
sqt_max_cache_size	Maximum SQT interface cache memory, in bytes.	Min: 0 Max: 2147483647	Min: 0 Max: 2251799813685247

Table 1-14 describes changes to the memory_limit configuration parameter that impacts performance on 32-bit Replication Server and 64-bit Replication Server.

Table 1-14: memory_limit configuration parameter

Parameter	Description	Valid range for 32-bit	Valid range for 64-bit
memory_limit	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 fstr_cachesize, md_source_memory_pool, queue_dump_buffer_size, sqt_max_cache_size, sre_reserve, and sts_cachesize.	Min: 0 Max: 2047	Min: 0 Max: 2147483647

See Chapter 3 “Upgrading or Downgrading Replication Server” in the *Replication Server Configuration Guide* to migrate to 64-bit platforms and Chapter 4, “Performance Tuning” in the *Replication Server Administration Guide Volume 2* to configure support for 64-bit platforms.

CHAPTER 2

New Features in Replication Manager 15.5

Support for *bigdatetime* and *bigtime* datatypes

With Replication Manager 15.5, you can enable replication of the bigdatetime and bigtime datatypes included with Adaptive Server 15.5. You can replicate these datatypes to replicate databases and warm standby databases by specifying the datatypes in replication definitions, function replication definitions, and subscriptions.

On the Columns tab of the Add New Table Replication Definition dialog box, select bigdatetime or bigtime from the Replication Definition list in the Datatype area.

See “bigdatetime and bigtime datatypes replication” on page 26.

CHAPTER 3

New Features in Replication Server Version 15.2

This chapter describes the enhancements that have been made to Replication Server 15.2.

Topic	Page
Support for DSI bulk copy-in	37
Non-blocking commit	42
Quoted identifiers	45
Replication Server gateway	51
Row count validation for non-SQL statement replication	54
SQL statement replication	57
Non-Adaptive Server error class support	65
Non-Adaptive Server replication support enhancements	67

Support for DSI bulk copy-in

In versions 15.1 and earlier, when Replication Server replicates data to Adaptive Server, Replication Server forms a SQL insert command, sends the command to Adaptive Server, and waits for Adaptive Server to process the row and send back the result of the operation. This process affects Replication Server performance when large batches of data are replicated, such as in end-of-day batch processing or trade consolidation.

Replication Server version 15.2 introduces support for bulk copy-in to improve performance when replicating large batches of insert statements on the same table in Adaptive Server® Enterprise 12.0 and later. Replication Server implements bulk copy-in in Data Server Interface (DSI), the Replication Server module responsible for sending transactions to replicate databases, using the Open Client™ Open Server™ Bulk-Library.

Note Bulk copy-in is supported only for Adaptive Server databases. If you turn on DSI bulk copy-in and the replicate database is not Adaptive Server, DSI shuts down with an error.

For information about the Open Client Open Server Bulk-Library, see the *Open Client and Open Server Common Libraries Reference Manual*.

Setting up bulk copy-in

These database connection parameters control bulk operations in DSI:

Parameter	Description
dsi_bulk_copy	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. Default: off.
dsi_bulk_threshold	The number of insert commands 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. Minimum: 1 Default: 20

To set the values of dsi_bulk_copy and dsi_bulk_threshold, use:

- alter connection to change the bulk copy-in connection parameters at the connection level:

  ```
  alter connection to dataserver.database
  set {dsi_bulk_copy | dsi_bulk_threshold} to value
  ```

- configure replication server to change the server defaults:

```
configure replication server
set {dsi_bulk_copy | dsi_bulk_threshold} to value
```

To check the values of dsi_bulk_copy and dsi_bulk_threshold, use admin config.

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.

Changes to subscription materialization

Bulk copy-in also improves the performance of subscription materialization. When dsi_bulk_copy is on, Replication Server uses bulk copy-in to materialize the subscriptions if the number of insert commands in each transaction exceeds dsi_bulk_threshold.

Note In normal replication, bulk operation is disabled for a table if autocorrection is on. However, in materialization, bulk operation is applied even when autocorrection is enabled, if dsi_bulk_threshold is reached and the materialization is not a nonatomic subscription recovering from failure.

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

New counters for bulk copy-in

New counters have been added for bulk copy-in:

Counter	Description
DSINoBulkDatatype	The number of bulk operations skipped due to the data containing datatype is incompatible with bulk copy-in.
DSINoBulkFstr	The number of bulk operations skipped due to tables that have customized function strings for rs_insert or rs_writetext.
DSINoBulkAutoc	The number of bulk operations skipped due to tables that have autocorrection enabled.
DSIEBFBulkNext	The number of batch flushes that executed because the next command is a bulk copy.
DSIEBulkSucceed	The number of times the Data Server Interface executor (DSI/E) invoked blk_done(CS_BLK_ALL) at the target database.
DSIEBulkCancel	The number of times DSI/E invoked blk_done(CS_BLK_CANCEL) at the target database.
DSIEBulkRows	The number of rows that DSI/E sent to the replicate data server using bulk copy-in.
BulkTime	The amount of time, in milliseconds, that DSI/E spent in sending data to the replicate data server using bulk copy-in.

Limitations

The Replication Server DSI does not use bulk copy-in when:

- Autocorrection is on and the data is not part of subscription materialization.
- rs_insert has a user-defined function string.
- text column has a user-defined function string for rs_writetext with output `none` or `rpc`.
- The data row contains opaque datatype or a user-defined datatype (UDD) that has an rs_datatype.canonic_type value of 255.
- The data row contains an image or a Java datatype.

The bulk copy-in feature is not supported under the conditions listed below. In these instances, disable bulk copy-in.

- The replicate database is not Adaptive Server. In this case, if the DSI bulk copy-in is enabled, DSI terminates with an error message.

- The data size changes between Replication Server and the replicate Adaptive Server character sets, and the datarow contains text columns. In this case, if the DSI bulk copy-in is enabled, DSI terminates with this message:

  ```
  Bulk-Lib routine 'blk_textxfer' failed.
  Open Client Client-Library error: Error: 16843015,
  Severity 1 -- 'blk_textxfer(): blk layer: user
  error: The given buffer of xxx bytes exceeds the
  total length of  the value to be transferred.'
  ```

- The owner.tablename length is larger than 255 bytes and the replicate database is earlier than version Adaptive Server 15.0.3 Interim Release. If the DSI bulk copy-in is enabled, Replication Server terminates with this message:

  ```
  Bulk-Lib routine 'blk_init' failed.
  ```

  To specify not to use bulk copy-in when owner.tablename length is larger than 255 bytes:

  a Turn trace on:

  ```
  trace "on", rsfeature, ase_cr543639
  ```

  b Add this to the Replication Server configuration file:

  ```
  trace=rsfeature,ase_cr543639
  ```

Other limitations:

- Unlike the insert command, bulk copy-in does not generate timestamps; NULL values are inserted to the timestamp column if the timestamp column is not included in the replication. Either disable bulk copy-in, or set up your replication definition to include the timestamp column.
- Text and image columns are always logged, even if you change the writetext function string to no log.
- Bulk copy does not invoke insert trigger in Adaptive Server.
- The configuration parameter send_timestamp_to_standby has no effect on bulk copy-in. timestamp data is always replicated.

Non-blocking commit

Replication Server 15.2 includes non-blocking commit, which uses the delayed commit feature in Adaptive Server to improve replication performance.

Note To use non-blocking commit, you must use Sybase Enterprise Connect™ Data Access 15.0 ESD #3 (ECDA) or later.

Adaptive Server delayed commit feature

Adaptive Server 15.0 and later includes the delayed commit feature designed to improve performance by delaying the commit phase of a transaction. The commit phase includes writing log records of the transaction to disk and then notifying the client application of the transaction status. With delayed commit, Adaptive Server notifies the client application of a successful commit before writing the corresponding transaction log to disk. This delay in writing to disk reduces contention on the last and active log page and thus improving performance.

However, the last page of a transaction log can be lost, if Adaptive Server terminates or if you shut down Adaptive Server using shutdown with no wait.

See "Using delayed_commit to determine when log records are committed," in Chapter 11, "Developing a Backup and Recovery Plan" in the *Adaptive Server Enterprise 15.0 System Administration Guide: Volume 2* and the delayed_commit parameter of the set command in "set," in Chapter 1, "Commands" in the *Adaptive Server Enterprise 15.0 Reference Manual: Commands*.

Configuring non-blocking commit

The dsi_non_blocking_commit configuration parameter extends the period of time Replication Server saves messages after a commit.

Extending the save period requires a larger stable queue. See "Stable queues," in Chapter 2, "Replication Server Technical Overview" in the *Replication Server Administration Guide Volume 1*.

Use alter connection to configure dsi_non_blocking_commit for a database connection:

alter connection to *data_server.database*
set dsi_non_blocking_commit to '*value*'

Note You cannot use this parameter with alter connection to configure an active database connection in a standby environment.

Use configure replication server to configure dsi_non_blocking_commit as a server default:

configure replication server
set dsi_non_blocking_commit to '*value*'

where *value* is the number of minutes, to a maximum of 60, to extend the save period. The default is zero, which disables non-blocking commit.

Use admin config to check the current *value* of dsi_non_blocking_commit.

System functions that support non-blocking commit

rs_non_blocking_commit and rs_non_blocking_commit_flush support the non-blocking commit feature.

rs_non_blocking_commit

rs_non_blocking_commit executes every time DSI connects to the replicate data server, if 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 has function-string class scope.

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.

rs_non_blocking_commit_flush

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_blocking_commit_flush has function-string class scope.

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.

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'
```

Support for non-Adaptive Server databases

Replication Server 15.2 with non-blocking commit enabled supports replication into Oracle 10gV2 and 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 Heterogeneous Replication Guide*.

Product version requirements

You can use dsi_non_blocking_commit only with Adaptive Server 15.0 and later, and Oracle 10g v2 and later. Replication Server disables the dsi_non_blocking_commit configuration parameter for the connection for unsupported versions of Adaptive Server, Oracle, or other databases.

For details about the commands discussed in this section, see the *Replication Server Reference Manual*.

Quoted identifiers

Object names that contain special characters such as spaces and non-alphanumeric characters, start with a character other than an alphabetic character, or that correspond to a reserved word, must be enclosed in double quote characters to be parsed correctly. These object names are referred to as quoted identifiers. Although Replication Server version 15.1 and earlier can accept quoted identifiers, forwarding quoted identifiers to data servers is not supported in these versions.

Note To use quoted identifiers, you must use ECDA 15.0 ESD #3 or later.

As of Replication Server 15.2, quoted identifier support allows you to:

- Mark identifiers in a replication definition as quoted.
- Create a connection where you can forward quoted identifiers to data servers.

Embedded double quote characters in identifiers is not currently supported.

Data servers such as Adaptive Server, SQL Anywhere®, Microsoft SQL Server, Universal Database (UDB), and Oracle handle quoted identifiers differently in terms of supported length, special characters, and reserved words. In a heterogeneous environment, ensure that the quoted identifiers being replicated are valid on both the primary and replicate data servers.

Enabling the quoted identifier feature

The dsi_quoted_identifier configuration parameter enables or disables quoted identifier support in the Data Server Interface (DSI). Use the create connection or the alter connection command to set dsi_quoted_identifier on or off for a data server connection. The default value of dsi_quoted_identifier is off.

To check the value of dsi_quoted_identifier, use the admin config command.

Marking identifiers as quoted

create replication definition and alter replication definition commands allow you to mark quoted identifiers using the new parameter quoted. When an identifier is marked and the dsi_quoted_identifier is set to on, the replicate servers that subscribe to the replication definition 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.

Modified *create replication definition* syntax

The modified create replication definition syntax is:

```
create replication definition replication_definition
with primary at data_server.database
[with all tables named [table_owner.] 'table_name' [quoted] |
[with primary table named [table_owner.]'table_name']
 with replicate table named [table_owner.]'table_name'] [quoted]]
(column_name [as replicate_column_name] [datatype [null | not null]
      [map to published_datatype]] [quoted]
[, column_name [as replicate_column_name]
      [datatype [null | not null] computed]
      [map to published_datatype]] [quoted]...)
primary key (column_name [, column_name]...)
[searchable columns (column_name [, column_name]...)]
[send standby [{all | replication definition} columns]]
[replicate {minimal | all} columns]
[replicate_if_changed (column_name [, column_name]...)]
[always_replicate (column_name [, column_name]...)]
[with dynamic sql | without dynamic sql]
```

For example, to create a table foo with column foo_col1 as a quoted identifier, enter:

```
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")
```

Modified *alter replication definition* syntax

The modified alter replication definition syntax is:

```
alter replication definition replication_definition
{with replicate table named table_owner.]'table_name' |
add column_name [as replicate_column_name]
      [datatype [null | not null]]
      [map to published_datatype] [quoted],... |
alter columns with column_name
      [as replicate_column_name] [quoted | not quoted],...|
alter columns with column_name
       datatype [null | not null]
       [map to published_datatype],...|
alter columns column_name {quoted | not quoted}
```

add primary key *column_name* [, *column_name*]... |
drop primary key *column_name* [, *column_name*]... |
add searchable columns *column_name* [, *column_name*]... |
drop searchable columns *column_name* [, *column_name*]... |
send standby [off | {all | replication definition} columns] |
replicate {minimal | all} columns |
replicate_if_changed *column_name* [, *column_name*]... |
always_replicate *column_name* [, *column_name*]...} |
{with | without} dynamic sql
alter replicate table name {quoted | not quoted}

For example, to mark as quoted the table named foo, enter:

```
alter replication definition repdef
   alter replicate table name "foo" quoted
```

To unmark the column foo_col1, enter:

```
alter replication definition repdef
   with replicate table named "foo"
   alter columns "foo_col1" not quoted
```

Note 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, Replication Server sends both the primary table name and the replicate table name as quoted.

Forwarding quoted identifiers to data servers

Data servers receive quoted identifiers differently. Adaptive Server, SQL Anywhere, and Microsoft SQL Server do not expect quoted identifiers, and require a special command to configure the connection for quoted identifiers. Oracle and UDB do not require the connection to be configured to accept quoted identifiers.

Use rs_set_quoted_identifier function string to set the DSI connection appropriately for each data server type. 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 "".

rs_set_quoted_identifier has function-string-class scope.

Changes to rs_helprep

rs_helprep stored procedure displays quoted identifiers.

Example

Example 1 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                                                      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 2 Given the table and replication definition defined in example 1, 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                                                      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 3 Given the replication definition defined in example 2, 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_      rs_sqlserver_function_class   Class Default    rs_select_
with_lock                                                      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)
```

Product version requirements

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.

For details about the commands discussed in this section, see the *Replication Server Reference Manual*.

Replication Server gateway

In managing a replication system, the replication system administrator (RSA) must log in to multiple replication servers, ID servers, and the corresponding Replication Server System Database (RSSD). The RSA must also frequently switch logins between Replication Server and the RSSD. In version 15.2, Replication Server introduces Replication Server gateway, which minimizes explicit login into the different servers.

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 into ID Server, your remote server identification (RSI) to log into a remote Replication Server, and your maintenance user ID to log into the remote Adaptive Server. You do not need to supply this information more than once, when you access Replication Server itself.

Cascading connection

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. For example, you can connect to an ID Server, and then to the ID Server's RSSD. In this case, both the primary, controlling Replication Server and the ID server are gateways; commands pass through to the ID server's RSSD, and result sets pass back through to you.

Enabling the Replication Server gateway

The connect command has been added to turn Replication Server into a gateway to its RSSD, ID server, or to a remote Replication Server.

Note Issuing the connect command requires an sa role for the first login to Replication 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.
- 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. The Replication Server gateway uses RSI to log into the remote server, and requires a direct route to the remote server.

  Note Replication Server cannot directly connect to itself. However, you can work around this by using a cascading connection.

- *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 use the ECDA to connect the Replication Server and the remote data server.

Tracking connections

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. Use show connection and show server commands to manage your cascaded connections:

- show connection – lists the contents of the connection stack.
- show server – displays the current working server.

Dropping connections

Use disconnect command to terminate a connection to a server. disconnect exits the connection stack one at a time. To exit from all the connections, use disconnect all.

Syntax {disconnect | disc} [all]

Limitations

When using Replication Server gateway, the client and the server must use the same locale set because Replication Server cannot perform character set conversion.

Product version requirements

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.

For details about the commands discussed in this section, see the *Replication Server Reference Manual*.

Row count validation for non-SQL statement replication

To address errors in Replication Server, Replication Server 15.2 includes support for Replication Server error classes and error actions for row count verification errors not related to SQL statement replication. See “Row count validation for SQL statement replication” on page 62 for SQL statement error actions.

Note Replication Server ignores row count validation for those commands that are in a customized function string. See “SQL statement replication does not support autocorrection,” in Chapter 4, “Performance Tuning” in the *Replication Server Administration Guide Volume 2*.

Replication Server 15.2 introduces the Replication Server error class. Therefore, with version 15.2, a connection associates itself with two error class types—a data server error class and a Replication Server error class. You must associate a Replication Server error class with a connection before Replication Server can query the Replication Server error class for overrides to the default Replication Server error actions. You can associate a connection with only one Replication Server error class. However, you can associate one Replication Server error class with multiple connections. Use the set replication server error class parameter for the create connection and alter connection commands to associate a Replication Server error class with a connection.

When Replication Server responds to errors, it looks first for the Replication Server error class assigned to the connection. If Replication Server does not find the Replication Server error class, Replication Server uses the default rs_repserver_error_class error class assigned to the server.

Creating Replication Server error classes

With Replication Server 15.2, you can use create error class to create Replication Server error classes that you can use to assign error actions for errors that occur in Replication Server.

Syntax

create [replication server] error class *error_class*
[set template to *template_error_class*]

Parameters

- replication server – indicates that the new error class is a Replication Server error class and not a data server error class

- *error_class* – the name for the new error class. The name must be unique in the replication system and must conform to the rules for identifiers.

  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

To create my_rs_err_class based on the default rs_repserver_error_class:

```
create replication server error class my_rs_err_class
set template to rs_repserver_error_class
```

Usage

You can drop a Replication Server error class using drop error class, and you can change the primary Replication Server of a Replication Server error class using move primary.

Assigning error actions

Use the assign action command at the primary site for the Replication Server error class to specify an error action. See “assign action” and Table 3-17 , in Chapter 3, “Replication Server Commands” in the *Replication Server Reference Manual*.

Syntax

```
assign action
  {ignore | warn | retry_log | log | retry_stop | stop_replication}
  for error_class
  to server_error1 [, server_error2]...
```

Parameters

error_class is the error class name for which the action is being assigned, such as a Replication Server error class.

server_error is the error number. You can specify these error numbers for Replication Server for error actions not related to SQL statement replication:

Table 3-1: Error actions for Replication Server error classes

server_error	Error message	Default error action	Description
5185	`Row count mismatch for the command executed on 'dataserver.database'. The command impacted x rows but it should impact y rows.`	warn	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.
5187	`Row count mismatch for the autocorrection delete command executed on 'dataserver.database'. The command deleted x rows but it should delete y rows.`	warn	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.

Examples

To assign the ignore error action if Replication Server encounters error number 5185, enter:

```
assign action ignore for rs_repserver_error_class to 5185
```

Displaying Replication Server error classes

Use the rs_helpdb, rs_helpclass, and rs_helperror stored procedures to display information about Replication Server error classes. See Chapter 6, “Adaptive Server Stored Procedures” in the *Replication Server Reference Manual*.

Replication Server System Database (RSSD) modifications

To support Replication Server error handling, these system tables in the RSSD have been modified:

- rs_classes – classtype column includes a new “R” value for Replication Server error classes.
- rs_databases – rs_errorclassid is a new column for the Replication Server error class associated with the database.

SQL statement replication

Replication Server 15.2 supports SQL statement replication which complements log-based replication and addresses performance degradation caused by batch jobs.

In SQL statement replication, Replication Server receives the SQL statement that modified the primary data, rather than the individual row changes from the transaction log. Replication Server applies the SQL statement to the replicated site. RepAgent sends both the SQL data manipulation language (DML) and individual row changes. Depending on your configuration, Replication Server chooses either individual row change log replication or SQL statement replication.

SQL statement replication includes row count verification to ensure that the number of rows changed in the primary and replicate databases match after replication. If the number of rows do not match, you can specify how Replication Server handles this error.

To enable SQL statement replication:

- Configure the primary database to log SQLDML.
- Configure Replication Server to replicate SQLDML:
  a Create replications definitions with SQLDML for table and multisite availability (MSA) replication.
  b In Replication Server, set WS_SQLDML_REPLICATION parameter on for warm standby replication.

See the *Adaptive Server 15.0.3 New Features Guide* for more information on SQL statement replication.

Modifications to system configuration

Several Adaptive Server stored procedures support SQL statement replication.

Database level

sp_setrepdbmode has been added to support SQL statement replication.

sp_setrepdbmode

sp_setrepdbmode allows you to enable SQL statement replication for a specific DML operation.

The DML operations that apply to SQL statement replication include:

- 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.

For example, to replicate delete statements as SQL statement replication and also enable replication of select into, enter:

```
sp_setrepdbmode pdb, 'DS', 'on'
```

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.

See "sp_setrepdbmode" in the "System Changes" chapter in the *Adaptive Server Enterprise 15.0.3 New Features Guide*.

Table level

sp_setrepdefmode has been enhanced to support the SQL statement replication.

sp_setrepdefmode

sp_setrepdefmode includes options to:

- Enable or disable SQL statement replication for a specific DML operation
- Configure the threshold that must be reached to activate SQL statement replication

The DML operations that apply to SQL statement replication include:

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

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.

For example, to enable SQL statement replication for update, delete, and insert select operations on table t, enter:

```
sp_setrepdefmode t, 'UDI', 'on'
```

```
go
```

See “sp_setrepdefmode” in the “System Changes” chapter in the *Adaptive Server Enterprise 15.0.3 New Features Guide*.

Session level

Use session option set repmode to set replication mode to SQL statement replication. You can specify session-level settings either when you log in, or at the beginning of a batch job. Session-level settings override both database-level and object-level settings.

Use set repmode on to enable SQL statement replication for the DML operation specified, for the duration of the session. Use set repmode off to remove all SQL statement replication settings at the session level. For example, to replicate only select into and delete as SQL statements for the duration of the session, enter:

```
set repmode on 'DS'
```

See “set repmode” in the “System Changes” chapter in the *Adaptive Server Enterprise 15.0.3 New Features Guide.*

SQL statement replication configuration

You can change replication options at database and table levels.

Database replication definition

To replicate SQL statements in an MSA environment, you must include the replicate SQLDML clause with the create replication definition or alter replication definition commands.

This code segment displays the syntax for create and alter database replication definitions:

```
[[not] replicate setname [in (table list)] ]
```

where:

setname = DDL | tables | functions | transactions | system procedures | SQLDML | ‘options’.

The ‘options’ parameter is a combination of:

- U – update

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

The SQLDML parameter is also defined as a combination of U, D, I, and, S statements.

This example shows how to use the 'options' parameter to replicate SQLDML on tables tb1 and tb2:

```
replicate 'UDIS' in (tb1,tb2)
```

This example shows how to use the SQLDML parameter that produces the same result as the 'options' parameter in the previous example:

```
replicate SQLDML in (tb1,tb2)
```

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.

Examples

This example shows how to filter 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
```

This example 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
```

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
```

This example 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
```

Table replication definition

To support SQL statement replication, you must include the replicate SQLDML clause for a create table replication definition. This code segment displays the syntax for a create table replication definition:

[replicate {SQLDML ['off'] | '*options*'}]

The 'options' parameter is a combination of these statements:

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

Note If your replication definition has the `[replicate {minimal | all} columns]` clause, then the `[replicate {minimal | all} columns]` clause must always precede the `[replicate {SQLDML ['off'] | 'options'}]` clause.

This is a sample create replication definition for a table:

```
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
```

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.

SQL statement replication restrictions

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 or articles include where clauses.
- Updates include one or more text or image columns.
- Function strings rs_delete, rs_insert, and rs_update are customized.

SQL statement replication does not support autocorrection

SQL statement replication cannot perform autocorrection. If Data Server Interface (DSI) encounters a DML command for SQL statement replication and autocorrection is on, by default, DSI is suspended and stops replication. 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.

Row count validation for SQL statement replication

In Replication Server 15.2, you can specify how Replication Server responds to SQLDML row count errors that may occur during SQL statement replication. SQLDML row count errors occur when the number of rows changed in the primary and replicate databases do not match after SQL statement replication. The default error action is to stop replication.

Syntax

Use the assign action command at the primary site for the Replication Server error class to specify other error actions for SQLDML row count errors:

```
assign action
  {ignore | warn | retry_log | log | retry_stop | stop_replication}
```

for *error_class*
to *server_error1* [, *server_error2*]...

Parameters

error_class is the error class name for which the action is being assigned. With Replication Server 15.2, you can specify Replication Server error classes such as the default rs_repserver_error_class error class.

server_error is the error number. With Replication Server 15.2, you can specify error numbers for Replication Server:

Table 3-2: Error actions for SQL statement replication

server_error	Error message	Default error action	Description
5186	`Row count mismatch for the command executed on 'dataserver.database'. The command impacted x rows but it should impact y rows.`	stop_replication	Row count verification error for SQL statement replication if the affected number of rows is different from what is expected.
5193	`You cannot enable autocorrection if SQL Statement Replication is enabled. Either enable SQL Statement Replication only or disable SQL Statement Replication 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

Examples

To assign the warn error action if Replication Server encounters error number 5186, enter:

```
assign action warn for rs_repserver_error_class to 5186
```

If there is a row count error, this is an example of the error message that displays:

```
DSI_SQLDML_ROW_COUNT_INVALID 5186
Row count mismatch for SQLDML command executed on
'mydataserver.mydatabase'.
The command impacted 1000 rows but it should impact 1500
rows.
```

See “Row count validation for non-SQL statement replication” on page 54 for row count validation not related to SQL statement replication.

Configuring warm standby database for SQL replication

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 a 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.

Replication Server System Database (RSSD) modifications

These system tables in the RSSD support SQL statement replication:

- rs_dbreps – status column includes 4 new sets of 2-bit sets, each of which corresponds to a DML filter. The first bit of a set indicates if it is an empty filter and the second bit indicates if it is a negative statement set.
- rs_dbsubsets – type column includes four new types: U, L, I, and, S corresponding to the DML UDIS filters. In this case, L is used for delete instead of D.
- rs_objects – attributes column includes five new bits; one for each U, D, I, or S operation, and one to indicate if a table replication definition has fewer columns than the number of incoming data rows.

  A system function replication definition, rs_sqldml, also supports SQL statement replication.

Product and mixed-version requirements

SQL statement replication requires Adaptive Server version 15.0.3 and later, primary and replicate Replication Server version 15.2 and later, and route version 15.2 and later.

For details about the commands discussed in this section, see the *Replication Server Reference Manual*.

Non-Adaptive Server error class support

Replication Server 15.2 provides support for error classes and error action mapping for non-Adaptive Server Enterprise (non-ASE) replication databases. You can use the default non-ASE error classes included in Replication Server 15.2. You can also create and alter your own error classes for non-ASE replicate databases.

Default non-ASE error classes

The default non-ASE error classes provided with Replication Server 15.2 are listed in Table 3-3. You cannot modify these default error classes.

Table 3-3: Non-ASE error classes

Database	Class name
IBM DB2	rs_db2_error_class
IBM UDB	rs_udb_error_class
Microsoft SQL Server	rs_msss_error_class
Oracle	rs_oracle_error_class

Creating error classes

Replication Server 15.2 includes the set template to option for the create error class command. Use create error class, and set template to, and another error class as a template, to create your own error classes. create error class copies the error actions from the template error class to the new error class.

Syntax

create error class *error_class*
[set template to *template_error_class*]

Example

This example creates the my_error_class error class for an Oracle database based on rs_oracle_error_class as a template:

```
create error class my_error_class set template to
rs_oracle_error_class
```

Altering error classes

Use the alter error class command, and another error class as a template, to alter error classes. alter error class copies error actions from the template error class to the error class you want to alter and overwrites error actions that have the same error code

Syntax

alter error class *error_class*
set template to *template_error_class*

Example

For example, to alter my_error_class for an Oracle database based on rs_sqlserver_error_class as a template:

```
alter error class my_error_class set template to
rs_sqlserver_error_class
```

For more information about error classes and error handling, see the *Replication Server Administration Guide Volume 2*. For details about the commands discussed in this section, see the *Replication Server Reference Manual*.

Creating and altering connections

In Replication Server 15.2, you can use the create connection and alter connection commands to assign non-ASE error classes to specific connections on non-ASE replication databases.

Native error codes

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.

Non-Adaptive Server replication support enhancements

Replication Server 15.2 includes several enhancements to installation, configuration, and overall usability of setting up replication environments that include actively supported non-Adaptive Server Enterprise (non-ASE) data servers. These enhancements automate the installation and configuration process by providing a pre-configured Replication Server environment that enables replication involving actively supported non-ASE data servers to be up and running quickly.

Actively supported data servers are data servers for which Sybase provides all the required software, documentation, and support for the data servers to serve as both a primary or a replicate data server. See the *Replication Agent Release Bulletin* for your platform for the list of actively supported non-ASE data servers.

Simplified installation and configuration

With Replication Server 15.2, you do not need to edit and execute scripts to install datatype definitions, function strings, and class-level translations for heterogeneous (non-ASE) datatype support. The functions provided by the scripts are included as part of the Replication Server 15.2 installation, or are included in connection profiles that are installed with Replication Server 15.2. These enhancements simplify installation and configuration for non-ASE environments. Follow the simplified configuration instructions in Chapter 8, “Installing and Implementing Non-ASE Support Features” in the *Replication Server 15.2 Configuration Guide* for your platform.

In addition, Replication Server 15.2 provides error class support for non-ASE replicate databases. See “Non-Adaptive Server error class support” on page 65.

Connection profiles

With Replication Server 15.2, you can use connection profiles that contain connection configurations and replicate database object definitions relevant to each type of actively supported non-ASE data server. You can use these connection profiles and simple syntax to create connections between actively supported data servers such as Adaptive Server Enterprise, IBM DB2, Microsoft SQL Server, and Oracle. Replication Server uses the connection profile to configure the connection and create replicate database objects for you.

Connection profiles specify the function-string class, error class, and class-level translations to be installed. You can also use connection profile options to specify other actions such as whether commands should be batched and the command separator to use.

Note 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.

Using connection profiles

Use the using profile clause with the create connection command to create a connection between a non-ASE database and Adaptive Server using a connection profile.

Syntax

Here is a portion of the create connection syntax that shows the using profile and display_only clauses:

```
create connection to data_server.database
using profile connection_profile[;version]
set username [to] user
[other_create_connection_options]]
[display_only]
```

Parameters:

- *connection_profile* – provide the connection profile you want to use to configure a connection, modify the Replication Server System Database (RSSD), and build replicate database objects.
- *version* – specify the particular version of a connection profile.

- *other_create_connection_options* – use *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. See "create connection," in Chapter 3, "Replication Server Commands" in the *Replication Server Reference Manual* for all the parameters you can use with create connection.
- display_only – use with a connection profile to display but not execute commands and the servers where the commands would be executed. Use the client and Replication Server logs to see the results 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
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 another error class—my_msss_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
```

```
set username to ora_maint
set password to ora_maint_pwd
display_only
go
```

Listing available connection profiles

Use the admin show_connection_profiles command to list the profile name, version, and comments for each profile defined in Replication Server.

Syntax

admin show_connection_profiles[, "*match_string*"]

Use the *match_string* option to display only the connection profiles whose names contain the string you provide in the option.

Examples

Example 1 Lists the names of all connection profiles currently defined in Replication Server:

```
admin show_connection_profiles
go
```

Extract of output is:

```
Profile Name              Version     Comments
------------              -------     --------
rs_ase_to_db2             Standard    Standard ASE to DB2 replication
                                      connection profile.
rs_ase_to_udb             Standard    Standard ASE to DB2 replication
                                      connection profile.
...
rs_db2_to_ase             Standard    Standard DB2 to ASE replication
                                      connection profile.
rs_db2_to_msss            Standard    Standard DB2 to Microsoft SQLServer
                                      connection profile.
...
```

There is a connection profile for each primary and replicate database combination, such as Adaptive Server and Oracle, Oracle and Adaptive Server, and IBM DB2 and /Microsoft SQL Server, and so on.

See "admin show_connection_profiles," in Chapter 3, "Replication Server Commands" in the *Replication Server Reference Manual* for the list of connection profiles.

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
```

Output is:

```
Profile Name              Version     Comments
------------              -------     --------
rs_ase_to_oracle          Standard    Standard ASE to Oracle replication
                                      connection profile.
```

System tables for connection profiles

Two system tables in the RSSD support connection profiles:

- rs_profile – stores currently defined profiles in Replication Server.
- rs_profdetail – stores the profile details.

For more information on support for non-ASE data servers, see the *Replication Server Heterogeneous Replication Guide* and the *Replication Server Administration Guide Volume 1*. For details about the commands discussed in this section, see the *Replication Server Reference Manual*. For the updated configuration process, see the *Replication Server Configuration Guide* for your platform.

CHAPTER 4

New Features in Replication Server Version 15.1

This chapter describes the enhancements that have been made to Replication Server 15.1.

Topic	Page
Dynamic SQL enhancements	73
Function replication enhancements	75
Adaptive Server shared-disk cluster support	76
Monitor and counter enhancements	76
Improved stable queue management	78
Changed locales directory	81
Extended password encryption support	82
rs_ticket stored procedure version 2	82
New Replication Server counters	83
Extended support for large-object (LOB) datatypes	84
Partial update of LOB datatypes	85
Extended timestamp support	86
New opaque datatype	86
Dump transaction enhancement	87
Distributor status recording	89
Enhanced text update	89
Adaptive Server integer identity support	90
Stable Queue Manager performance enhancements	90

Dynamic SQL enhancements

Dynamic SQL in Replication server enhances replication performance by allowing Replication Server Data Server Interface (DSI) to prepare dynamic SQL statements at the target user database and to run them repeatedly.

Dynamic SQL now supports heterogeneous replicate databases including Oracle, Universal Database (UDB), DB2, and Microsoft SQL. create/alter replication definition commands allow you to control the application of dynamic SQL on each table through replication definition.

See the *Replication Server Reference Manual* for information about create/alter replication definition commands.

You can change the dynamic SQL execution at the table level for a specific replicate database by using:

```
set dynamic_sql {on | off}
for replication definition with replicate at
data_server.database
```

To check for dynamic SQL usage, turn on stats_sampling and run admin stats, dsi command and look for DSIEDsqlPrepared, DSIEDsqlExecuted and other dynamic SQL related counters.

Use stored procedures rs_helprep, rs_helpsub, and rs_helppubsub to display dynamic SQL setting for each replication definition.

See "rs_helprep", "rs_helpsub", and "rs_helppubsub," in Chapter 6, "Adaptive Server Stored Procedures" in the *Replication Server Reference Manual* for information about using these stored procedures.

Limitations

Dynamic SQL commands support the data within Sybase range. If data falls outside Sybase ranges that cause dynamic SQL to fail, DSI logs an error message and resends dynamic SQL using the language command. DSI shuts down only if the language command also fails.

If this condition happens frequently, disable dynamic SQL from the table replication definition or use the set dynamic_sql off command.

Use any of these commands to turn dynamic_sql off:

- alter connection... set dynamic_sql off – turns dynamic SQL off for all commands replicating to this connection.
- create/alter replication definition...without dynamic_sql – turns dynamic SQL off for all commands using this replication definition.
- set dynamic_sql off for replication definition with replicate at... – turns dynamic SQL off for all commands using this replication definition at this replicate connection.

Function replication enhancements

In Replication Server 15.1, you can create a function replication definition that has a different name than the function name.

Replication Server enforces different primary and replicate function names for the request function replication definition. The maint_user runs the transaction at the replicate database if the function is replicated through applied function replication definition. The origin_user runs the transaction if the function is replicated through request function replication definition at the replicate database.

These enhancements let you:

- Replicate multiple functions with the same name from different databases.
- Have multiple replication definitions for one primary function and each specifies a different replicate function for a different replicate site.

To manage function replication definition, use:

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

Mixed-version support

This enhancement supports mixed version environments. However, function replication definitions that have different primary function name and replication definition name are not replicated to Replication Servers earlier than 15.1.

Warning! If your system has an earlier version of a request function replication definition, drop the earlier-version definition before creating a 15.1 replication definition for the same primary function.

Warm standby and multisite availability (MSA) support

In a warm standby or MSA environment, there is only one parameter list for all the function replication definitions of the same primary function. If you alter one function replication definition to add a parameter, the new parameter is added to all the function replication definitions created for this function.

See Chapter 3, “Replication Server Commands” in the *Replication Server Reference Manual* for detailed information about these commands.

Limitations

Enhanced function replication has these limitations:

- All the function replication definitions created for the same function must have the same parameter list with the same name and datatype.
- If you created a function replication definition with differing primary function name and replication definition name in version 15.1, any earlier version of the request function replication definition for the same primary function is disabled.
- You cannot have both applied function replication definitions and request function replications for a primary function. The function replication definition created by using create function replication definition command is considered an applied function at the function primary site.
- For each applied and request function replication definition, you must create a corresponding subscription to replicate a function.

Adaptive Server shared-disk cluster support

Replication Server and RepAgent thread both support the Adaptive Server shared-disk cluster environment, which is where many Adaptive Servers share a single set of disks or databases. In a Sybase shared-disk cluster, a database can be either a replication source or a replication destination. You can perform all tasks, such as configuring RepAgent or marking tables for replication, from any instance in the cluster.

See Chapter 5 “Setting Up and Managing RepAgent” in the *Replication Server Administration Guide Volume 1*.

Monitor and counter enhancements

The enhanced monitors and counters feature allows you to collect the information of the most active tables, procedures and related statistics, and store this information into the Replication Server System Database (RSSD) rs_statdetail table and related tables. You can use this information to diagnose the replicate database performance issues such as missing indexes on the primary keys, and latency problems in the Replication Agent™ and Stable Queue Transaction interface (SQT)/Distributor (DIST) processing.

Active object identification

New counters have been added to count the statement execution time on tables and procedures:

- AOBJInsertCommand
- AOBJUpdateCommand
- AOBJDeleteCommand
- AOBJWritetextCommand
- AOBJExecuteCommand

To flush the active object counter metrics to the RSSD, run any of these commands:

- admin stats, “all”, save
- admin stats, dsi, save
- admin stats, sysmon, save

See “rs_helpcounter” in the *Replication Server Reference Manual* for details on displaying information about counters.

New procedures interface

To bring out the most active tables and procedures, and related statistics information, Replication Server 15.1 introduces the stored procedures rs_stat_populate and rs_stat_genreport. rs_stat_populate reads data from rs_statdetail, summarizes, augments, denormalizes, and saves result into rs_statreport, where rs_stat_genreport reads data and generates report.

Load this script into the RSSD after upgrading to Replication Server 15.1:

$SYBASE/$SYBASE_REP/scripts/rs_install_statreport_v1510_[ase|asa].sql

After loading the script, run the stored procedures rs_stat_populate and rs_stat_genreport. Running these stored procedures produces this information:

- Replication Server performance overview – overview information about your Replication Servers, such as DIST processing, DSI processing, and so on.
- Replication Server performance analysis – performance analysis and tuning suggestions based on critical Replication Server counters. The detailed description is available in the script file.

- Active object identification result – lists the active table and procedure names, owner names, execution times, and so on.

For more information about rs_stat_populate, rs_stat_genreport, rs_statreport, and rs_statdetail, see the script file, which contains syntax, examples, and so on.

Improved stable queue management

Replication Server 15.1 simplifies stable queue management. Enhanced queue dump commands provide flexibility in identifying the stable queues, controlling the stable queue contents to dump, and supporting additional output file options. Replication Server 15.1 also includes new commands that allow you to delete and restore specific transactions from the Stable Queue Manager (SQM).

For more information about the stable queue management, see the *Replication Server Administration Guide Volume 1*. For details about the commands discussed in this section, see the *Replication Server Reference Manual*.

Changes to *sysadmin dump_queue*

Enhancements to sysadmin dump_queue include:

- an option to specify the server or database name instead of the queue number when identifying the stable queue to dump.
- an option to specify the number of commands to dump.
- filtering options such as dumping only the begin and end commands of a transaction and dumping everything in the queue as comments except SQL statements.
- the option to direct the output to the Replication Server log or to a user-defined log file.
- an option to start the data dump from where the previous sysadmin dump_queue command stopped for that particular queue and session.

The modified sysadmin dump_queue syntax is:

```
sysadmin dump_queue {, q_number | server [,database]},
      {q_type,seg, blk, cnt
      [, num_cmds]
```

```
[, {L0 | L1 | L2 | L3}]
[, {RSSD | client | “log” | file_name}] |
“next” [, num_cmds]}
```

Changes to *sysadmin sqt_dump_queue*

Enhancements to sysadmin sqt_dump_queue include:

- an option to specify the server or database name instead of the queue number when identifying the stable queue to dump.
- an option to dump all committed transactions and read transactions found in the SQT cache.
- an option to specify the number of commands to dump.
- filtering options such as dumping only the begin and end commands of a transaction and dumping everything in the queue as comments except SQL statements.
- the option to direct the output to the Replication Server log or to a user-defined log file.

The modified sysadmin sqt_dump_queue syntax is:

```
sysadmin sqt_dump_queue {, q_number | server [,database]},
    q_type, reader
    [,{open | closed | read}]
    [, num_cmds]
    [, {L0 | L1 | L2 | L3}]
     [, {RSSD | client | “log” | file_name}]
```

Changes to *resume connection*

The resume connection skip transaction option has been enhanced to support skipping a specified number of transactions in the connection queue before resuming the connection. Skipped transactions are written to the database exception log, and to either the Replication Server log or the alternative log file specified by the sysadmin dump_file command. The maximum number of transactions that this command can skip is the number of transactions in the Data Server Interface (DSI) outbound queue.

The modified resume connection syntax is:

```
resume connection to data_server.database
    [skip [n] transaction |  execute transaction]
```

Changes to *sysadmin log_first_tran*

A new option *n* has been added to the sysadmin log_first_tran command. Use the new option to specify 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.

The modified sysadmin log_first_tran syntax is:

```
sysadmin log_first_tran [,n], data_server, database
```

New *sysadmin sqm_zap_tran* command

sysadmin sqm_zap_tran deletes a specific transaction from the stable queue and returns a message stating the number of deleted commands. The transaction is identified through the local queue ID (LQID).

The sysadmin sqm_zap_tran syntax is:

```
sysadmin sqm_zap_tran {, q_number, | server [,database]},
    q_type, lqid [, {L0 | L1 | L2 | L3}]
    [, {RSSD | client  | “log” | file_name}]
```

Note Replication Server must be in standalone mode to use this command.

New *sysadmin sqm_unzap_tran* command

sysadmin sqm_unzap_tran restores a transaction in the stable queue and returns a message stating the number of restored transaction commands. The transaction is identified through the LQID.

The sysadmin sqm_unzap_tran syntax is:

```
sysadmin sqm_unzap_tran {, q_number, | server [,database]},
    q_type, lqid [,  {L0 | L1 | L2 | L3}]
    [, {RSSD | client  | “log” | file_name}]
```

Note Replication Server must be in standalone mode to use this command.

New *sysadmin dump_tran* command

Use sysadmin dump_tran to dump the statements of a specific stable queue transaction into a log file. The transaction is identified through the LQID.

The sysadmin dump_tran syntax is:

```
sysadmin dump_tran {{, q_number, | server [,database]},
    q_type, lqid
    [, num_cmds]
    [, {L0 | L1 | L2 | L3}]
    [, {RSSD | client  | “log” | file_name}] |
    “next” [, num_cmds]}
```

Changed *locales* directory

The Replication Server release area and localization directory structure have been modified.

You can now install or uninstall multiple Sybase products, such as Replication Server and Adaptive Server, on the same computer and same directory. You can also install multiple versions of Replication Server in the same directory.

The changes to the *locales* directory include:

- Replication Server-specific *locales* files have been moved from *$SYBASE/locales* to a new directory *$SYBASE/$SYBASE_REP/locales*.
- All <*charset*> subdirectories have been consolidated into one <*utf8*> subdirectory for each language in the new directory *$SYBASE/$SYBASE_REP/locales*.

  The UTF-8 character set that is used for all supported languages instead of using different character sets. You can convert UTF-8 into other characters and vice versa.

  Replication Server reads messages from a localization file, and converts the messages to the specified character set format during runtime.
- rs_init locale files are now located in *$SYBASE/$SYBASE_REP/locales/<language>/utf8/sybinit*.

Extended password encryption support

Replication Server 15.1 uses Sybase Common Security Infrastructure (CSI) for server or client authentication, cryptography for encryption and decryption of passwords that are stored in the RSSD tables, and key-pair generation to support extended password encryption.

Extended password encryption uses asymmetric key encryption, which allows Open Client applications with connection property CS_SEC_EXTENDED_ENCRYPTION enabled to connect to Replication Server. It also allows Replication Server to enable CS_SEC_EXTENDED_ENCRYPTION when connecting to other servers.

Asymmetric key encryption uses a public key to encrypt the password and a private key to decrypt the password. The private key is not shared across the network, and is therefore secure.

Note To use the extended password encryption feature, you must have a server that supports extended password encryption, such as Adaptive Server 15.0.2 ESD #2 or later. Additionally, this feature is not supported in HP-Itanium platform in 15.1 release.

rs_ticket stored procedure version 2

rs_ticket is now on its version 2 and provides support for non-ASE databases. You can use rs_ticket stored procedure without additional configuration and administration. Tickets are automatically inserted into the rs_ticket_history table, which is located in the replicate database. Tickets are sharable among multiple applications, where they are issued without obstruction from tickets of other applications.

With rs_ticket version 2, more ticket information is provided for better usability, such as the Replication Server Interface (RSI) timestamp, which you can retrieve without writing complex queries. You can directly query the rs_ticket_history table for Replication Server performance. If the computer time or tickets are not synchronized across multiple time zones, you can change the timestamp columns to adjust the ticket date.

The earlier version of rs_ticket has been renamed rs_ticket_v1. To use the earlier version, alter rs_ticket_report function string with your old content or with the default, exec rs_ticket_report ?rs_ticket_param!param?.

Note If you previously disabled dsi_rs_ticket_report and upgraded to Replication Server 15.1, the dsi_rs_ticket_report setting is reenabled after the upgrade process has finished.

These are the format changes that have been made to rs_ticket:

- Changed version number to 2, V=2; if a ticket has version number equal to 1, Replication Server does not write date to tickets.
- The ticket size has been increased from 255 to 1024 bytes.
- The timestamp format has been changed to include the date. The new timestamp format is mm/dd/yy hh:mm:ss:mmm.
- A Replication Server Interface (RSI) module timestamp that forces the RSI sender to parse RSI messages has been added. Tickets pass more than one RSI modules. However, the rs_ticket_history table keeps only the last RSI timestamp.
- Primary and target Replication Server names have been added to identify where a ticket comes from and where it goes to.
- Primary and replicate database names have been added.
- Two DSI counters have been added:
  - DSI_T=xxx – total transactions that the Data Server Interface (DSI) reads.
  - DSI_C=xxx – total commands that the DSI reads.

See the *Replication Server Reference Manual* for detailed information about using rs_ticket version 2.

New Replication Server counters

New counters have been added for REPAGENT, RSIUSER, SQM, DSI, and DSIEXEC:

Counter	Description
RepAgentExecTime	The amount of time, in milliseconds, that the RepAgent user thread is scheduled by Open Client/Server™.
RSIUExecTime	The amount of time, in milliseconds that the RSI user thread is scheduled by Open Client/Server.
SQMWaitSegTime	The amount of time waiting for allocating segments.
DSINoDsqlNULL	Number of commands that cannot use dynamic SQL statements because of NULL value in where clauses.
DSINoDsqlDatatype	Number of commands that cannot use dynamic SQL statements because of text, image, java and ineligible UDDs
DSINoDsqlRepdef	Number of commands excluded from dynamic SQL by replication definition.
DSINoDsqlColumnCount	Number of commands excluded from dynamic SQL because the number of parameters exceeds 255.
DSINoDsqlMissingCols	Number of commands excluded from dynamic SQL because some columns are not available at DSI. This can be caused by minimal columns feature.
DSIEDsqlPrepared	Dynamic SQL statements prepared at target database by a Data Server Interface executor (DSI/E).
DSIEDsqlDealloc	Dynamic SQL statements deallocated at target database by a DSI/E.
DSIEDsqlExecuted	Dynamic SQL statements executed at target database by a DSI/E.
DSIEDsqlDeallocSchema	Dynamic SQL statements deallocated at replicate database by a DSI/E because of schema change.
DSIEDsqlDeallocExecFail	Dynamic SQL statements deallocated at replicate database.

See “rs_helpcounter” in Chapter 6 “Adaptive Server Stored Procedures” in the *Replication Server Reference Manual* for commands to retrieve information about counters.

Extended support for large-object (LOB) datatypes

Replication Server 15.1 supports the replication of Microsoft SQL Server 2005 datatypes varchar(max), nvarchar(max), and varbinary(max). These datatypes can each store up to 2,147,483,647 bytes of data.

Replication Server introduces LOB types as user-defined datatypes (UDDs) in the table-level replication environment. Replication Server also supports database-level replication for new LOB types. The new LOB types are directly mapped to text, unitext, and image datatypes.

The base type of UDDs is:

New LOB datatype	Base type
varchar(max)	text
nvarchar(max)	unitext
varbinary(max)	image

Limitations

The limitations of the new LOB datatypes are:

- You cannot define as a primary key a LOB column in the table replication definition.
- You cannot define as searchable a LOB column in the table replication definition or function replication definition.
- You cannot replicate stored procedures that include one of the new LOB datatypes as a parameter.
- You cannot use text pointers to manipulate the data of the new LOB datatypes.

In a mixed-version environment, the primary and replicate Replication Server must have a site version of 15.1 and an LTL version of 710.

For more information about the new LOB datatypes, see the *Replication Server Reference Manual*.

Partial update of LOB datatypes

A partial-update transaction allows you to directly insert a character string or overwrite an existing character string of a table column without issuing a delete and replace commands.

Replication Server 15.1 supports replication of partial-update transaction to supported LOB datatypes.

To implement partial update, use the new rs_updatetext LTL command:

```
{distribute|_ds} command_tags {applied|_ap} 'table'.rs_updatetext
{partialupd|_pu} [{first|_fi}] [last] [{changed|_ch}] [with log]
[{withouttp|_wo}] [{offset|_os}=offset {deletelen|_dln}=deletelength]
[{textlen|_tl}=length] text_image_column
```

Limitations

Partial update:

- Does not support multiple character set conversion.

- Support is restricted to Microsoft SQL Server 2005.

For more information about partial update, see the *Replication Server Design Guide*.

Extended *timestamp* support

A new datatype, timestamp has been added to Replication Server 15.1. timestamp is defined as varbinary(8) with a status bit indicator that differentiates it from varbinary. The timestamp datatype allows the replication of timestamp columns to replicate, standby, and MSA databases. You can also define timestamp as a primary key in a replication definition, and as a searchable column in a replication definition and a function replication definition.

The send_timestamp_to_standby configuration parameter has been added to support timestamp replication. When send_timestamp_to_standby is enabled and there are no replication definitions, timestamp columns are sent to the replicate database.

Note The replicate Adaptive Server must be version 15.0.2 or later to support timestamp in replication definition.

See the *Replication Server Reference Manual* for more information about the timestamp datatype. See the *Replication Server Administration Guide Volume 1* for information about replicating timestamp columns.

New *opaque* datatype

The opaque datatype handles replication of datatypes that Replication Server does not support. RepAgent provides formatting data that can be directly applied in the target database.

The opaque datatype handles replication of datatypes that can store unspecified or inconsistent values, such as anydata datatype and the Microsoft SQL Server sql_variant datatype.

Limitations

Limitations of the opaque datatypes include:

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

In a mixed-version environment, the primary and replicate Replication Server must have a site version of 15.1 and an LTL version of 710.

For more information about opaque datatypes, see the *Replication Server Reference Manual*.

Dump transaction enhancement

The Log Transfer Language (LTL) dump subcommand and rs_dumptran function string have been enhanced to support replication of dump transaction commands with the with standby_access parameter.

LTL *dump* subcommand enhancement

standby, stdb, and status parameters have been added to the dump subcommand syntax to support with standby_access:

```
{distribute|_ds} command_tags dump [ database |
    {transaction | tran | _tr}[standby | stdb | status]]
     database_name, 'dump_label' id
```

Table 4-1: New dump subcommand parameters

Parameter	Description
standby / stdb	An optional keyword that tells Replication Server that the command is a dump transaction command that uses with standy_access.
status	Dump status: • 0 – the dump transaction command does not contain with standby_access. • 1 – the dump transaction command contains with standby_access.

Example

RepAgent sends a dump transaction command to the Replication Server:

```
distribute @origin_time='Apr 15 1988 10:23:23.010PM',
@origin_qid=0x00000000000000000000000000000000000000000000
00000000000000000000377,@tran_id=0x0000000000000000
00000372 dump tran stdb devdb 'Apr 15 1988
10:23:23.011PM'0x00000000000000000000000000000000000000000
00000000000000000000000377
```

rs_dumptran enhancement

A system variable *rs_dump_status* has been added to the rs_dumptran function string to support with standby_access. Valid values for *rs_dump_status* are:

- 0 – the dump transaction command does not contain with standby_access.
- 1 – the dump transaction command contains the parameter with standby_access.

The changes to rs_dumptran are supported in Replication Server 15.1 or later. If you are using an earlier version of the Replication Server as the primary replication server, Replication Agent sends the dump transaction command without the new clause.

If you are using an earlier version of Replication Server as the replicate server, the primary Replication Server sends the dump transaction command without the new clause.

For more information about dump transaction enhancement, see the *Replication Server Design Guide*.

Distributor status recording

A distributor (DIST) thread reads transactions from the inbound queue and writes replicated transactions into the outbound queue. A DIST thread is created when Replication Server connects to the primary database, and can be suspended or resumed manually, or through a Replication Server configuration. Resuming and suspending a DIST thread modifies the DIST status of the thread.

With Replication Server 15.1, you can now save the DIST status of a distributor thread in the RSSD. This allows the DIST thread to retain its status even after a Replication Server is shut down.

For more information about DIST status recording, see the *Replication Server Reference Manual*.

Enhanced text update

Replication Server supports the replication of large objects such as text and image to non-ASE servers by passing a writetext command to DirectConnect Anywhere™, where it is converted to an update statement. The writetext command includes large-object pointers that an update statement uses to search and propagate the replicate database. Most data servers have their own unique implementation of updating large objects. Therefore, large-object replication to these servers becomes slow and inefficient, often requiring a full table scan of the replicate database for a single update.

Replication Server 15.1 provides an option to include primary keys with writetext commands sent to DirectConnect Anywhere. With the primary keys, DirectConnect Anywhere can create update statements that can efficiently search and replicate the replicate database.

Replication Server 15.1 introduces the Data Server Interface (DSI) configuration parameter dsi_alt_writetext. Use dsi_alt_writetext to instruct the Replication Server to include a text pointer or a set of primary keys with the writetext command.

Note You need ECDA 15.0 ESD #2 to use this feature.

For more information, see the *Replication Server Reference Manual*.

Adaptive Server integer *identity* support

The Adaptive Server 15.0 allows you to use these datatypes as identity values:

- bigint
- int
- numeric
- smallint
- tinyint
- unsigned bigint
- unsigned int
- unsigned smallint

Replication Server 15.1 supports replication of the above datatypes. When you create a replication definition for a table that contains an identity column, specify identity as the datatype for the column.

Stable Queue Manager performance enhancements

The Stable Queue Manager (SQM) performance has been enhanced to include:

- Stable queue caching
- Segment preallocation
- Support for direct I/O file access

Stable queue caching

Replication Server uses a simple caching mechanism to optimize I/O. This mechanism reduces write latency and improves reader speed, since data can usually be read quickly from the cache.

A cache is made up of multiple pages and each page is made up of multiple adjoining blocks. A cache is allocated for each queue at start-up time. Changing the page size changes the size of I/O in the stable queue devices. When a page is full, the entire page is written in one single write operation.

In stable queue caching, the page pointer moves forward and rotates back at the end of the cache. SQM flushes the current page if the writer has filled the message queue and is blocked when waiting for messages. Only blocks with data are written to a disk when flushing a page that is not full.

Configuring stable queue cache parameters

Set the server-wide caching default value using:

```
configure replication server set sqm_cache_enable to
"on|off"
```

Enable or disable the caching for a queue and override the server-level setting using:

```
alter queue q_number, q_type, set sqm_cache_enable to
"on|off"
```

When sqm_cache_enable parameter is disabled, SQM module returns back to the earlier mechanism, which maintains a fixed 16K; one-block buffer.

Set the server-wide page size default value using:

```
configure replication server set sqm_page_size to
"num_of_blocks"
```

Set the page size for a specified queue using:

```
alter queue q_number, q_type, set sqm_page_size to
"num_of_blocks"
```

num_of_blocks specifies the number of 16K blocks in a page. Configuring the page size also sets the I/O size of Replication Server. For example, if you set the page size to 4, this instructs the Replication Server to write to stable queue in 64K chunks.

Set the server-wide cache size default value using:

```
configure replication server set sqm_cache_size to
"num_pages"
```

Set the cache size for a specified queue using:

```
alter queue q_number, q_type, set sqm_cache_size to
"num_pages"
```

num_pages specifies the number pages in the cache.

All SQM configuration commands are static, thus you must restart the server for these commands to take effect.

See the *Replication Server Reference Manual* for detailed information about these configuration parameters.

Segment preallocation

Replication Server 15.1 preallocates segments in the background to reduce segment allocation latency. Segment allocation imposes significant latency to writer threads especially when the RSSD is on a remote Adaptive Server.

When a writer thread needs a new segment, it checks whether the preallocated segment is available, if it is not, the thread requests to allocate the segment. Once the writer thread gets the new segment, a preallocation request is made so that the segment is allocated in the background. By the time the writer thread needs a new segment, it is already available.

Enable or disable segment preallocation using:

```
configure replication server set sqm_seg_prealloc to
"on|off"
```

This command is static, which means you must restart the server for it to take effect. It supports only the server-level configuration.

Support for direct I/O file access

For file system partitions, direct I/O reduces the I/O latency as compared to the synchronous I/O, DSYNC.

Note Direct I/O is supported only on Sun Solaris SPARC.

Adjust the stable queue cache properly. A proper cache size ensures that most read transactions are completed within the cache. Configure direct I/O using:

```
configure replication server set sqm_write_flush to
"dio"
```

This command enables direct /IO and is effective only when the stable queue is on the file system. The direct I/O method allows the Replication Server to read or write directly to the disk without the buffering the file system.

This command is static, which means you must restart the server for it to take effect.

CHAPTER 5

New Features in Replication Manager 15.1

This chapter describes the Replication Server 15.1 features supported by Replication Manager.

Topic	Page
Enhanced support for dynamic SQL	93
Enhanced support for function replication definitions	94
Support for large-object datatypes	94
Sybase Central 6.0	95
Support for opaque datatypes	95
Support for timestamp datatypes	96

Enhanced support for dynamic SQL

With Replication Manager 15.1, you can enable dynamic SQL replication in table replication definitions in mixed-version replication environments where the Replication Server version is 15.0.1 or later.

The Replication Manager GUI has been modified. The General tab of the Create/Alter Replication Definition dialog box now includes a Replicate dynamic SQL option. This option is selected by default. If you do not require dynamic SQL replication, unselect the option. If you are working with a version of Replication Server for which Replication Manager does not support dynamic SQL, the Replicate dynamic SQL option is disabled.

Note If you modify the dynamic SQL replication property of a replication definition, the changes apply only to the modified replication definition. Other replication definitions for the same table remain unchanged.

See “Dynamic SQL for enhanced Replication Server performance” in the *Replication Server Administration Guide Volume 2* for detailed information about dynamic SQL replication in Replication Server.

Enhanced support for function replication definitions

With Replication Manager 15.1, you can create multiple function replication definitions with names that differ from the corresponding primary function names. To support this feature, the following changes have been incorporated in the Replication Manager GUI:

- On the General tab of the Add New Function Replication Definition dialog box, after you create a function replication definition, the Add New Function Replication Definition icon is not replaced by the Function Replication Definitions folder. The Add New Function Replication Definition icon and the Function Replication Definitions folder coexist so that you can create multiple function replication definitions.
- On the General tab of the Add New Function Replication Definition dialog box, the Replication Definition text field is now available for you to create your own function replication definition name. You can create a function replication definition name that differs from the selected stored procedure in the Replication Definition field.
- The Replicate Stored Procedure option available on the General tab instead of the Advanced tab.

See the *Replication Server Administration Guide Volume 1* for detailed information about working with function replication.

Support for large-object datatypes

With Replication Manager 15.1, you can manage large-object (LOB) datatypes that are defined in your replication environment. Replication Manager supports Microsoft SQL Server LOB datatypes of varchar(max), nvarchar(max), and varbinary(max). However, if a column contains any of these LOB datatypes, the column cannot be a primary key, or marked as searchable.

The following changes have been made in the Replication Manager GUI:

- If you have LOB datatypes in your replication environment, the Replication Definition Datatype and Published Datatype lists in the Create Replication Definition dialog box display LOB.

- On the Columns tab in the Add New Table Replication Definition dialog, when you select a column that contains varchar(max), nvarchar(max), and varbinary(max) datatypes, the Primary Key and Searchable options are disabled.

See the *Replication Server Administration Guide Volume 1* for detailed information about working with LOB datatypes.

Sybase Central 6.0

Replication Manager 15.1 runs in Sybase Central 6.0.

Support for opaque datatypes

In Replication Manager 15.1, you can replicate opaque data. Opaque data can store unspecified or inconsistent values such as the anydata datatype of Oracle or the sql_variant datatype of Microsoft SQL Server. This is how Replication Manager supports opaque data:

- In a mixed-version replication environment, you can replicate opaque data when the Replication Server is version 15.0.1 or later.
- The Replication Manager GUI has been modified. On the Columns tab of the Add New Table Replication Definition window, a new option "opaque" appears on the Replication Definition list in the Datatypes area.
- You cannot make columns with opaque data searchable.
- The opaque datatype is supported both in function and table replication definitions.

For detailed information about replication of supported datatypes, see the *Replication Server Reference Manual*.

Support for timestamp datatypes

In Replication Manager 15.1, you can replicate timestamps in your data. This is how Replication Manager supports timestamp replication:

- In a mixed-version replication environment, you can replicate timestamps where the Replication Server version is 15.0.1 or later.
- The Replication Manager GUI has been modified. On the Columns tab of the Add New Table Replication Definition window, a new option “time stamp” appears on the Replication Definition list in the Datatypes area.
- Timestamp datatype is supported both in function and table replication definitions.

For detailed information about replication of supported datatypes, see the *Replication Server Reference Manual*.

CHAPTER 6

New Features in Replication Server Version 15.0.1

This chapter describes the new features introduced in Replication Server 15.0.1. Several enhancements have been made to enable more efficient configuration and tuning of Replication Server.

Topic	Page
Configuration and tuning enhancements	97
Monitor and counter enhancements	98
Dynamic SQL for enhanced Replication Server performance	100
Master database replication	102
rs_subcmp enhancement	103
Schema comparison	105
Manual data reconciliation	108

Configuration and tuning enhancements

Replication Server 15.0.1 now supports dynamic configuration, which simplifies Replication Server customization and performance tuning by:

- Providing the ability to some of the Replication Server configuration parameters to handle dynamic modification of values.
- Introducing admin config, a Replication Command Language (RCL) that retrieves the server, connection, logical connection, and route parameters and their runtime values.

Configuring dynamic parameters

Several Replication Server configuration parameters are changed to dynamic, allowing you to change their values using the configure replication server command. You no longer need to restart the Replication Server for the new values to take effect. Table 6-1 lists the dynamic configuration parameters.

Table 6-1: Dynamic configuration parameters

init_sqm_write_delay	init_sqm_write_max_delay
memory_limit	num_concurrent_subs
queue_dump_buffer_size	sqm_recover_segs
sqm_warning_thr_ind	sqm_warning_thr1
sqm_warning_thr2	sqt_max_cache_size
sqt_init_read_delay	sqt_max_read_delay
sts_cachesize	sts_full_cache_*system_table_name*

Use the new admin config command to retrieve the values of these parameters.

admin config

admin config lets you retrieve the values of the configuration parameters used to customize and tune the Replication Server. In Replication Server 15.0.1, these parameters are categorized as:

- Server parameters
- Connection parameters
- Logical connection parameters
- Route parameters

Syntax

The admin config syntax is:

admin config [,"connection" |,"logical_connection" |,"route"] [,*server* [,*database*]] [,*configuration_name*]

See Chapter 3, "Replication Server Commands" of the *Replication Server Reference Manual*, for detailed information in using admin config command.

Monitor and counter enhancements

The process of monitoring the counters that provide performance information has been improved in Replication Server 15.0.1. Sampling and saving statistics to the RSSD now runs asynchronously, allowing you to execute other Replication Server commands or terminate the session while the sampling continues in the background.

You can check the progress of asynchronous commands to prevent overlaps. When a command is denied due to overlapping, you can cancel the commands in progress to execute another command.

Enhanced *admin stats* command

The admin stats command has been modified. The modified command syntax is:

```
admin { stats | statistics } [, sysmon | "all"
    | module_name [, inbound | outbound ] [, display_name] ]
    [, server[, database ]     | instance_id ]
    [, display |, save [, obs_interval ] ]
    [, sample_period]
```

The behavior for the save parameter with a *sample_ period* has also changed. The admin stats returns the session immediately. Its original *num_obs* option has been changed to *obs_interval*, which can be a numeric value in seconds, or a quoted time format string hh:mm[:ss].

Example

To start sampling and saving statistics to the RSSD for one hour and thirty minutes at 20-second intervals, enter:

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

cancel command

The new command syntax also supports the cancel option, which stops the currently running command.

```
admin { stats | statistics } , cancel
```

Note For multiple observation intervals (*obs_interval*), the data that is saved at the time of cancellation is not deleted.

Replication Server checks the unfinished command to avoid corrupted sampling. If there is any command in progress, running either in the background or in other user sessions, the following commands cannot be executed:

- admin stats with any variation of save. For example:

  ```
  admin stats, "all", save, 30
  ```

- admin stats with a sampling period. For example:

  ```
  admin stats, "all", display, 120
  ```

- Commands that resets counters. For example:

  ```
  admin stats, reset
  ```

When a command is denied, the following error message is returned, and admin stats with the cancel option is required to stop the unfinished command before executing another command.

```
Admin command failed. Existing command is currently in
progress
```

Commands that display current statistics of your session are allowed to be executed even if there is another command in progress. For example:

```
admin stats, sqm, inbound, display
```

Checking the status of asynchronous commands

The output of the existing admin stats, status command has been modified to include the status of the sampling command:

```
1> admin stats, status
2> go
Command in progress, sampling period 00:30:00, time
elapsed 00:02:32


Sybase Replication Server Statistics Configuration
================================================
Configuration              Default            Current
--------------------       ----------         --------
stats_sampling             off                on
stats_show_zero_counters   off                off
stats_reset_rssd           on                 on
```

Keeping previously saved data in RSSD

Replication Server 15.0.1 has a new configuration parameter, stats_reset_rssd, which allows you to optionally keep the old sampling data in RSSD by setting its value to off. By default, stats_reset_rssd is turned on.

```
configure replication server
set stats_reset_rssd to {"on", "off"}
```

See Chapter 3, “Replication Server Commands” of the *Replication Server Reference Manual*, for detailed information in using admin stats command.

Dynamic SQL for enhanced Replication Server performance

Dynamic SQL in Replication Server enhances replication performance by allowing Replication Server Data Server Interface (DSI) to prepare dynamic SQL statements at the target user database and to execute them repeatedly. Instead of sending SQL language commands to the target database, only the literals are sent on each execution, thereby eliminating the overheads brought by SQL statement syntax checks and optimized query plan builds.

You can use dynamic SQL in a user database connection for a language command if:

- The command is insert, update, or delete.
- There are no text, image, unitext, or java columns in the command.
- There are no NULL values in the where clause of an update or delete command.
- There are no more than 255 parameters in the command:
  - insert commands can have no more than 255 columns.
  - update commands can have no more than 255 columns in the set clause and where clause combined.
  - delete commands can have no more than 255 columns in the where clause.
- The command does not use user-defined function strings.

Setting up the configuration parameters to use dynamic SQL

Configure dynamic SQL at a server or a connection level by issuing the following commands:

```
configure replication server
set { dynamic_sql |
      dynamic_sql_cache_size |
      dynamic_sql_cache_management }
to value

alter connection to server.db
set { dynamic_sql |
      dynamic_sql_cache_size |
      dynamic_sql_cache_management }
to value
```

The server-level configurations provide the default values for the connections created or started in the future. For database level configurations:

- dynamic_sql – turns dynamic SQL on or off for a connection. Other dynamic SQL related configuration parameters take effect only if this parameter is set to on.
- dynamic_sql_cache_size – tells the Replication Server how many database objects may use the dynamic SQL for a connection. This parameter is provided to limit the resource demand on the data server.

- dynamic_sql_cache_management – manages the dynamic SQL cache for a connection. Once the dynamic SQL statements reaches dynamic_sql_cache_size for a connection, it either stops allocating new dynamic SQL statements if the value is fixed, or keeps the most recently used statements and deallocates the rest to make room for the new statements if the value is mru.

Limitations

Dynamic SQL has these limitations:

- If a table is replicated to a standby or MSA connection using an 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, the existing prepared statements may be invalidated, and may require the standby or MSA connection to be restarted.
- Dynamic SQL requires ASE or DirectConnect™ 12.6.1 ESD#2 for UDB as target database.

Master database replication

The Adaptive Server RepAgent thread now supports master database replication in Adaptive Server 15.0, ESD #2 and later. The master database can be replicated using warm standby with any version of Replication Server that supports warm standby, including Replication Server version 12.0 and later.

In addition, the master database can be replicated using multisite availability (MSA) with any version of Replication Server that supports MSA.

Replication of the master database is limited to DDL commands and system procedures used to manage logins and roles.

The supported DDL commands and system procedures are:

- alter role
- create role
- drop role
- grant role
- revoke role
- sp_addlogin

- sp_displaylevel
- sp_droplogin
- sp_locklogin
- sp_modifylogin
- sp_password
- sp_passwordpolicy
- sp_role

rs_subcmp enhancement

In Replication Server 15.0.1, rs_subcmp uses a hash algorithm to improve its performance. Hash algorithm compresses the data in primary and replicated tables. The compressed data is then fetched by rs_subcmp.

Instead of taking the entire row of data during comparison between the primary table and replicated table, rs_subcmp now transfers only the compressed data of each data row from the primary or replicated tables, and then verifies or reconciles inconsistencies between them.

With the hash algorithm, the amount of data to be transferred from the primary and replicated tables to rs_subcmp is reduced, decreasing the total running time of rs_subcmp.

New command line and configuration file parameters

Table 6-2 and Table 6-3 describe parameters that are new to rs_subcmp that support the use of the hash algorithm.

Table 6-2: New command line parameters for hash algorithm

Parameter	Description
-h	Perform faster comparison.
-H	Normalize data when using hash algorithm across different platforms or character sets.

Table 6-3: New configuration file parameters for hash algorithm

Parameter	Description
FASTCMP	The same as command line parameter -h. The syntax is: FASTCMP = Y|N Values are: Y – perform fast comparison using compressed data. N (default) – perform normal comparison as before.
HASH_OPTION	The same as command line parameter -H. The syntax is: HASH_OPTION = *lsb* | *msb* | *unicode* | *unicode_lsb* | *unicode_msb* If this parameter does not exist in the configuration file, by default, rs_subcmp uses the native byte order and character set.

Limitation

The enhanced rs_subcmp requires ASE 15.0.2 or later and cannot handle case sensitive comparison. It also cannot handle text, unitext, or image datatypes and does not allow you to specify the precision for the float datatype (maximum precision is used).

Note Sybase suggests to set the ASE parameter default data cache to 128M or higher to get a better comparison performance.

Example

Example 1 To instruct rs_subcmp to get the compressed data and reconcile any inconsistencies between the primary and replicated tables, enter:

```
rs_subcmp [-h] [...]
```

Example 2 To instruct rs_subcmp to normalize data when using hash algorithm, enter the following command, where options can be *lsb*, *msb*, *unicode*, *unicode_lsb*, and *unicode_msb*:

```
rs_subcmp [-h -H options] [...]
```

Table 6-4 describes the string options to use with rs_subcmp.

Table 6-4: rs_subcmp string options

Option	Description
lsb	All byte-order dependent data is normalized to lsb-first (little-endian) before data compression.
msb	All byte-order dependent data is normalized to msb-first (big-endian) byte order before data compression.
unicode	Character data is normalized to unicode (UTF-16) before data compression.

Note UTF-16 string looks very much like an array of short integers and is byte-order dependent, thus Sybase suggests that you use *lsb* and *msb* in conjunction with unicode for platform independence. For example, use *unicode_lsb* or *unicode_msb*.

See Chapter 7, "Executable Programs" of the *Replication Server Reference Manual*, for more information on using these parameters with rs_subcmp.

Schema comparison

In Replication Server 15.0.1, the rs_subcmp function has been expanded to include schema comparison between tables and databases. This is useful in comparing schema between two databases that may have the same data but different schemas.

Table 6-5 and Table 6-6 enumerate the schema types and subtypes supported by rs_subcmp.

Table 6-5: 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 6-6: Schema sub-types 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

The command line and configuration file parameter set has been expanded to support the rs_subcmp schema comparison. Table 6-7 describes the new rs_subcmp command line parameters.

Table 6-7: Command line parameters for schema comparison

Parameter name	Description	Valid values
-x	comparison flag	0 (default) – data comparison. 1 – database schema comparison. 2 – table schema comparison.
-X	filter flag	If the value starts with "+", only the schema types are selected for comparison, and subtypes are ignored. Otherwise, the schema types and sub types are both unselected and not used for comparison.
-I	interface file	Interface file location.

Table 6-8 describes the new configuration file parameters.

Table 6-8: Configuration file parameters for schema comparison

Item name	Description	Valid values
SCHEMAFLAG	comparison flag	0 (default) – data comparison. 1 – database schema comparison. 2 – table schema comparison.
FILTER	filter flag	If the value starts with "+", only the schema types are selected for comparison, and sub-types are ignored. Otherwise, the schema types and subtypes are both unselected and not used for comparison.
IFILE	interface file	Interface file location.

Examples

Example 1 Compares all schemas between two databases using the *config.cfg* file:

```
rs_subcmp -f config.cfg
```

The configuration file contains the following:

```
PDS = PASE
RDS = R2ASE
PDB = pubs2
PTABLE = authors
RTABLE = authors
PUSER = sa
RUSER = sa
PPWD =
RPWD =
SCHEMAFLAG = 1
```

Example 2 Compares schema between two databases without a configuration file:

```
rs_subcmp -Spds -srds -Dpdb -drdb -Usa -usa -Psa_pwd
          -psa_pwd -x1
```

Example 3 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 4 Compares all table schemas and user schemas:

```
rs_subcmp -Spds -srds -Dpdb -drdb -Usa -usa -Psa_pwd
          -psa_pwd -x1 -X+TU
```

Report and reconciliation files

A report file which details the comparison result between two tables or two databases is created after every schema comparison. 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 where you issued the rs_subcmp.

Manual data reconciliation

The enhanced rs_subcmp command now supports manual reconciliation of inconsistent data through the creation of a reconciliation file. This allows you to verify the reconciliation statements before execution.

New *rs_subcmp* parameters

Replication Server 15.0.1 introduces two new parameters for this feature. Table 6-9 describes these parameters.

Table 6-9: New parameters for rs_subcmp manual reconciliation

Parameter	Description
-g	A command line parameter indicating to rs_subcmp to create a reconciliation file.
RECONCILE_FILE	A configuration file parameter. Values are: Y – create a reconciliation file. N (default) – do not create a reconcilition file.

Limitation

The reconciliation file's SQL statements cannot contain text, unitext, or image, data.

See Chapter 7, “Executable Programs” of the *Replication Server Reference Manual*, for more information on using these parameters with rs_subcmp.

CHAPTER 7

New Features in Replication Manager 15.0.1

This chapter describes the Replication Server 15.0.1 features supported by Replication Manager.

Topic	Page
Support for dynamic configuration	111
Support for heterogeneous data servers	112

Support for dynamic configuration

The Replication Manager (RM) plug-in has been modified to support the dynamic configuration feature of Replication Server 15.0.1. Versions of Replication Server earlier than 15.0.1 do not support dynamic configuration.

This feature allows the RM to dynamically retrieve all parameters that can be configured for the Replication Server, including the parameters for connections and routes.

Enhanced properties dialog box

To support dynamic configuration, the Parameters tab of the Properties dialog box in RM has been enhanced to accurately display the parameter status information returned by the Replication Server. It includes:

- The Status field replaces the Restart Required field in RM 15.0. Its associated text field now has a wider range of information. The remaining fields and functions of the properties dialog box remain the same.

- The parameter information in the Run Value column may now contain the value “<server default>”. This may be used for parameters in the connections, logical connections, and routes to indicate that a parameter at that component level has not been set, and the Replication Server default value will be used. These types of parameters can have a default setting at the server level, or can have the parameter set for an individual connection, logical connection, or route.
- There may be some parameters for which the legal values are considered nonstandard so the Legal values field may be left blank. This is usually due to a parameter that may contain a textual setting and a numerical setting. In these cases, the Explanation field may contain further information regarding legal values for the parameter.

Support for heterogeneous data servers

The RM plug-in now supports heterogeneous data servers, which provides you the ability to create connections, replication definitions, and subscriptions for non-Sybase data servers. Non-Sybase data servers include Oracle, Microsoft SQL Server, and IBM UDB.

Note This version of RM does not support IBM DB2 data servers or Replication Agents, although Sybase provides a replication solution for IBM DB2 on mainframe.

Non-Sybase data servers support

To provide support for heterogeneous data servers, the RM displays the databases, tables, and stored procedures for non-Sybase data servers in a two-tier environment, the same as in the ASE data servers.

See Chapter 3, “Managing Replication Server with Sybase Central” of the *Replication Server Administration Guide Volume 1*, for more information about two- and three-tier management solutions.

The RM allows you to create, update, and delete both primary and replicate connections. It also allows you to create, update, and delete databases, tables, and function replication definitions and subscriptions. The RM user interface for non-Sybase data servers is identical to the current ASE server interface.

Note RM supports table, and function replication definition, and subscription, but does not support database replication definition, and subscription for non-Sybase data servers. RM also supports all connections for non-Sybase data servers except for logical connection.

For non-Sybase data servers, the RM uses Replication Agents and DirectConnect to communicate with the data servers and to act as an interface for the RM. The RM uses an interface server to retrieve the metadata from the data servers. The Replication Agent serves as the interface server for the primary data servers, while DirectConnect serves as the interface server for the replicate data servers.

The interface server is not necessarily the server used to replicate data in the environment. For example, when defining an Oracle server that will be both a primary and a replicate data server, you only identify a Replication Agent as the interface server. This Replication Agent can be used to extract transactions from the Oracle server but a DirectConnect may also be needed to replicate data to the Oracle server. Also, if you are replicating from a Microsoft SQL Server, you can identify a Replication Agent as the interface server, but may require several Replication Agents to replicate data from several databases.

Replication Agents and Mirror Replication Agents support

In version 15.0.1, the RM provides the ability to initialize the Sybase Replication Agent and displays its performance statistics in a two-tier environment. Additionally, the connection associated with the Replication Agent is displayed in the tree view hierarchy upon clicking the Replication Agent.

The RM also provides a feature that tests the connection between the primary data server and the Replication Agent, and the connection between the Replication Agent and the Replication Server.

The context menu for the Replication Agent now provides commands to test connections, display statistics, and initialize the Replication Agent. In the Sybase Central object tree, right-click the Replication Agent icon to display the context menu.

Replication Manager plug-in enhancements

To provide heterogeneous support, the following items have been modified in the RM:

- The Add Server wizard to add non-Sybase data servers to a two-tier environment.
- The Add Connection wizard to create connections to non-Sybase databases.
- The Replication Agent context menu provides commands to initialize the agent, displays performance statistics, and tests network connections.
- The tree view hierarchy displays the connection associated with the Replication Agent under the actual agent.
- The Create Replication Definition dialog to create replication definitions for tables in a non-Sybase database. You can use user-defined datatypes to define non-Sybase datatypes.

CHAPTER 8

New Features in Replication Server Version 15.0

This chapter describes the new features introduced with Replication Server 15.0.

Topic	Page
Support for longer identifiers	115
New datatype: bigint	116
New unsigned integer datatypes	117
New Unicode datatype: unitext	118
Replicating computed columns	119
Replicating encrypted columns	120
Replicating partitioned tables	120
Larger disk partitions	122
Larger text and image size	123
Embedded Replication Server System Database (ERSSD) enhancement	123
New password encryption algorithm	124
Mixed-version enhanced support	125
New interface for monitors and counters	126
Support for isolation levels	127
Bidirectional replication support for DDL in MSA	129
Batching of commands for non-ASE servers	129
SySAM license management	130

Support for longer identifiers

The limit of 30 bytes for selected database and replication object names (identifiers) is extended to 255 bytes for Replication Server version 15.0. Longer identifiers are supported for these objects:

- Tables.

- Columns.
- Stored procedures.
- Parameters – for Replication Server functions and Adaptive Server stored procedures.
- Functions – for function replication definitions.
- Function strings.
- Replication definitions – including table replication definitions, function replication definitions, and database replication definitions.
- Publications.
- Articles.

All other database and replication object identifiers, login names and passwords, transaction names, subscription names, database names, and server names retain the existing 30-byte limit.

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 as many as 256 bytes including the @ character.
- Adaptive Server stored procedure parameter names can be as many as 255 bytes including the @ character.

Note rs_subcmp supports long identifiers for both table and column names.

Limitation

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.

New datatype: *bigint*

Replication Server 15.0 adds support for the bigint datatype. bigint is a fixed-width, 8-byte datatype. Like int, smallint, and tinyint, bigint is a signed integer datatype. bigint can hold whole numbers between -2^{63} and $+(2^{63} - 1)$.

Table 8-1 describes the range and storage size for all integer datatypes.

Table 8-1: Signed integer datatypes

Integer datatypes	Range	Storage (bytes)
bigint	Whole numbers between -2^{63} and $+(2^{63} - 1)$, inclusive (–9,233,372,036,854,775,808 and +9,233,372,036,854,775,807)	8
int	Whole numbers between -2^{31} and $+(2^{31} - 1)$, inclusive (–2,147,483,648, and +2,147,483,647)	4
smallint	Whole numbers between -2^{15} and $+(2^{15} - 1)$, inclusive (–32,768 and +32,767)	2
tinyint	Positive whole numbers between 0 and 255, inclusive	1

Mixed-version issues

To fully support bigint, the primary and replicate Replication Server must have a site version of 15.0, and the LTL version must be 700. If the LTL version is less than 700 at connect-source time, RepAgent converts bigint columns to numeric.

Note rs_subcmp supports the new bigint datatype.

New unsigned integer datatypes

Replication Server version 15.0 supports four new unsigned integer datatypes. You can use these unsigned datatypes in the same way as their signed equivalents:

- unsigned tinyint – tinyint
- unsigned smallint – smallint
- unsigned int – int
- unsigned bigint – bigint

Signed integers are whole positive or negative numbers. Unsigned integers are only whole positive numbers. The storage sizes of the signed and unsigned integers datatypes are the same. Table 8-2 shows the range of positive numbers supported for each unsigned datatype.

Table 8-2: Unsigned integer datatypes

Integer datatypes	Range	Storage (bytes)
unsigned bigint	Whole numbers between 0 and 18,446,744, 073, 709,551,615, inclusive	8
unsigned int	Whole numbers between 0 and 4,294,967,295, inclusive	4

Integer datatypes	Range	Storage (bytes)
unsigned smallint	Whole numbers between 0 and 65535, inclusive	2
unsigned tinyint	Positive whole numbers between 0 and 255, inclusive	1

Mixed-version issues

To fully support the unsigned integer datatypes, the primary and replicate Replication Server must have a site version of 15.0, and the LTL version must be 700.

If the LTL version is less than 700 at connect-source time, RepAgent makes these datatype conversions:

- unsigned bigint –> numeric
- unsigned int –> numeric
- unsigned smallint –> int
- unsigned tinyint –> tinyint

Replication definitions created with unsigned integer datatypes are not sent to Replication Servers version 12.5.x and earlier. If a replication definition is subscribed by Replication Server version 12.5.x, it cannot be altered to add unsigned integer columns.

Note rs_subcmp supports the new unsigned integer datatypes.

New Unicode datatype: *unitext*

Replication Server version 15.0 adds support for the unitext datatype. unitext is a variable-width, nullable Unicode datatype. Although independent of the text datatype, unitext mirrors its behavior, and can be used wherever text is used.

Like the data in the other Unicode datatypes—unichar and univarchar—unitext data is encoded in UTF-16, which is essentially a 2-byte, fixed-width encoding of Unicode. unitext can hold as many as 1,073,741,823 Unicode characters, or the equivalent of 2,147,483,647 bytes. unitext has no connection to the default character set ID or the default sort order.

The main advantage of Unicode datatypes is efficiency. The UTF-16 character types are approximately 33% more space efficient than UTF-8 for Asian characters.

Mixed-version issues

To fully support unitext, the primary and replicate Replication Server must have a site version of 15.0, the route version must be 15.0, and the LTL version must be 700. If the LTL version is less than 700 at connect-source time, RepAgent converts unitext columns to image.

Note rs_subcmp supports the new unitext datatype.

Replicating computed columns

Computed columns allow you to create an expression and place the result of the expression in a table column. A computed column is:

- Materialized – when its value is computed for each insert or update. Materialized computed columns are stored in the same way as regular columns.
- Virtual – when its value is computed only when referenced in a query. Virtual computed columns are not stored in the table or index page.

A computed column expression is:

- Deterministic – when its value is the same each time it is evaluated.
- Nondeterministic – when its value may be different each time it is evaluated (for example, a date stamp).

Replication Server replicates materialized computed columns in DML statements in the same way it replicates other columns; it does not replicate virtual computed columns.

The replication of computed columns is supported by function strings. In Replication Server version 15.0, the class-level function string rs_set_dml_on_computed is applied at the replicate database DSI when a connection is established. It issues set dml_on_computed “on” after the use database statement. If the replicate Adaptive Server is version 12.5.x or earlier, the command is ignored.

Since Replication Server does not distinguish between computed and regular columns, there are no changes to the syntax for creating or altering replication definitions.

When creating or altering replication definitions for tables containing:

- Deterministic columns – you can choose whether to include those columns in the replication definition. Since deterministic columns always realize the same value, you can create the replication definition without them and allow each replicated insert and update to compute values at the replicate database.
- Nondeterministic columns – you must include nondeterministic computed columns in the replication definition to ensure that the primary and replicate databases remain synchronized.

rs_subcmp support for computed columns

rs_subcmp supports comparison and reconciliation of materialized computed columns; it does not support virtual columns.

If the supported Adaptive Server provides the set dml_on_computed "on" command, rs_subcmp inserts and updates deterministic and nondeterministic materialized columns as regular columns.

Replicating encrypted columns

Replication Server 15.0 supports replication of encrypted columns in Adaptive Server 15.0. Similar to Adaptive Server, Replication Server does not support encrypted columns that contain text and image data.

Replication Server replicates encrypted data as well as the encryption keys. For more information about this feature, see the *New Features Adaptive Server Enterprise 15.0 with Encrypted Columns*.

Note rs_subcmp supports replication of encrypted columns in Adaptive Server.

Replicating partitioned tables

In Replication Server 15.0, partitioned tables introduced in Adaptive Server® Enterprise 15.0 are replicated in a way that is similar to nonpartitioned tables. The rs_truncate system function in LTL, and the rs_truncate function and function string in Replication Server have been extended to support partitioned tables.

rs_truncate changes in LTL

In LTL, the rs_truncate function has been extended to include partition names, as shown in the following syntax:

```
distribute command_tags applied [owner=owner_name]
table.rs_truncate [partition_name,
[partition_name]...] yielding
```

You must assign a *partition_name* for each partition specified in the truncate table partition command.

rs_truncate function changes in Replication Server

In Replication Server, the rs_truncate function has been extended to accept parameters to support a new truncate table paritition command, shown as follows:

```
truncate table table_name partition partition_name
```

When a truncate partition command is issued, the RepAgent sends the following LTL:

```
applied [owner=owner_name] table_name.rs_truncate,
partition_name _yd
```

rs_truncate function string changes in Replication Server

The partition names are passed as parameters to the rs_truncate function. The rs_truncate function string accepts position-based function-string parameters. The following is a position-based variable that specifies the parameter position in the function, in the LTL command:

```
?n!param?
```

A sample function string for rs_truncate with the position-based variable is as follows:

```
truncate table publishers partition ?1!param?,
?2!param?
```

Examples

Example 1 To replicate truncate table partition as a delete command, alter the function string in the following way:

```
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 publishers where c1 < 1000
if (?1!param? = ''B'')
delete publishers where c1 >= 1000
commit transaction'
```

Example 2 : To not to truncate table partitions at the replicate server, alter the function string to do nothing if there is a parameter, in the following way:

```
alter function string publisher.rs_truncate
for rs_sqlserver_function_class
output language
'if (?1!param? = '''') delete publisher'
```

Mixed-version issues

For the RepAgent to send an rs_truncate applied subcommand with parameters, the site version must be 15.0 and the LTL version must be 700. If the LTL version is below 700, RepAgent skips the rs_truncate portion of the distribute command.

To replicate the rs_truncate function with a parameter to the replicate Replication Server, the route version must be 1500. If the route version is lower than 1500, the rs_truncate command with the parameter is skipped.

Larger disk partitions

Replication Server version 15.0 extends the maximum disk partition size from 2GB to 1TB. The new limit is applicable after the Replication Server site version is updated to 15.0.

Two new RCL commands support larger disk partitions:

- create partition – makes a partition available to Replication Server. This command replaces the existing add partition command.
  add partition is still supported, for backward compatibility with earlier versions. The syntax and usage of the two commands are identical. The command name has been changed to be consistent with other Replication Server command names.
- alter partition – increases the size of a partition. The syntax is:

  alter partition *logical_partition_name* [expand [size = *size*]]

  For example, to increase the size of the logical partition p2 by 50MB, enter:

  ```
  alter partition p2 expand size = 50
  ```

Larger text and image size

In Replication Server 15.0, the rs_subcmp utility extends the maximum text and image size from 32KB to 2GB. You can specify the new value as length in kilobytes, doing either of:

- Using the -L *text_image_length_in_kilobytes* parameter to set the new text and image length in kilobytes. For example, to set the new length to 64KB, enter:

  ```
  rs_subcmp -L 64 -f subcmp.cfg
  ```

- Providing the new value for text and image length to the parameter TXT_IMG_LEN in the configuration file. Start rs_subcmp using the modified configuration file:

  ```
  rs_subcmp -f subcmp.cfg
  ```

  subcmp.cfg file includes the following configuration parameter and its value:

  ```
  TXT_IMG_LEN = 64
  ```

Note If the value is specified both in the command line and the configuration file, the value in the command line overwrites the value in the configuration file.

Embedded Replication Server System Database (ERSSD) enhancement

ERSSDs were introduced in Replication Server version 12.6. For more information, see Chapter 11, “New Features in Replication Server Version 12.6.”

You can create a route from a Replication Server with an ERSSD, as long as both the source and the destination servers are version 15.0 or later.

To create a route from Replication Server with an ERSSD, use the create route command. Verify that the Replication Agent name is in the Replication Server interfaces file; an ERSSD Replication Agent is started as an open server during create route. If the Replication Agent name does not appear in the interfaces file, the command fails.

The default ERSSD Replication Agent name is *erssd_name*_ra. To replace the default name with that of your Replication Agent server, enter:

```
configure replication server
set erssd_ra to <value>
```

Note Sybase provides ERSSD in SQL Anywhere (SA) as an option, as well as continues to support the traditional RSSD in Adaptive Server Enterprise.

New password encryption algorithm

Replication Server 15.0 uses the FIPS-certified Advanced Encryption Standard (AES) algorithm to encrypt new Replication Server user passwords. The AES alogrithm uses the 128-bit encryption key and can be obtained from the Certicom Security Builder library.

Migrating existing encrypted passwords

Use the information in Table 8-3 to migrate existing encrypted passwords in the Replication Server configuration file, and the rs_users and rs_maintusers tables.

Table 8-3: Commands to encrypt passwords in new algorithm

To	Command
Migrate existing user passwords to the new alogrithm	`alter user user set password password` where: • *user* is the login name of the existing user. • *password* is the existing password you want to encrypt using the new alogorithm.
Migrate existing database maintenance user passwords to the new algorithm	`alter connection to data_server.database set password to password` where *password* is the existing password you want to encrypt using the new alogorithm.
Migrate existing route user passwords to the new algorithm	`alter route to dest_replication_server set password to passwd` where: • *dest_replication_server* is the name of the kdestination Replication Server. • *passwd* is the existing password you want to encrypt using the new alogorithm.
Migrate existing user passwords in the configuration file to the new algorithm	Use rs_init to encrypt the passwords using the new algorithm.

Mixed-version issues

To fully support the new password encryption algorithm, both the Replication Server and the rs_init utility must have a site version of 15.0. If the site version is lower than 15.0, an error message displays and encryption is disabled.

Mixed-version enhanced support

In mixed-version environments, interaction between Replication Servers of different versions is restricted to the capabilities of the oldest version. Information associated with new features may not be available to Replication Servers of earlier versions.

Features in the new version, when sent to downstream Replication Servers of earlier versions, can break the multisite availability (MSA) replication and must be filtered out. In Replication Server 15.0, mixed-version support has been enhanced to provide a mechanism in which the later-version feature data is filtered out by the primary Replication Server before sending it out to the downstream Replication Servers of earlier versions.

New configuration parameter

To enable the primary Replication Server to block certain new feature commands not supported by earlier versions of Replication Server, a new Replication Server configuration parameter, dist_stop_unsupported_cmd, is available for all server and connection-level commands.

Replication Server commands that include the new configuration parameter are described in Table 8-4.

Table 8-4: Modified commands for Mixed-version support

Replication Server command	Syntax
configure replication server	`configure replication server set dist_stop_unsupported_cmd to [on | off]`
alter connection	`alter connection srv.db set dist_stop_unsupported_cmd to [on | off]`
alter logical connection	`alter logical connection lsrv.ldb set dist_stop_unsupported_cmd to [on | off]`

By default, dist_stop_unsupported_cmd is set to off. When this parameter is set to on, DIST suspends itself if a command cannot be sent to some destination Replication Servers. You have to resume the DIST by skipping the entire transaction or by resetting this parameter to off.

When the parameter is set to off, the distributor (DIST) thread skips the newer commands to earlier version Replication Servers.

New interface for monitors and counters

Replication Server 15.0 provides a new, simpler interface for monitoring the counters that provide performance information. By default, all counters are inactive until you turn them on using either:

- The admin stats (admin statistics) command, which activates counters for a specific time period, or
- The Replication Server configuration parameter stats_sampling, which is a toggle that activates or deactivates counters.

Using admin stats, you can specify which counter statistics to report, whether to display those statistics on the screen or save them to the RSSD, and how many seconds to collect the statistics. admin stats lets you specify statistics for:

- Individual counters
- Individual modules
- The sysmon counters, which are a set of counters Sybase has identified as those most valuable for monitoring performance
- All counters

Viewing statistics on screen provides a point-in-time benchmark. Saving statistics to the RSSD lets you accumulate data, so you can see changes in statistics over time, and perform averages and other calculated values. You can specify a sampling period and the number of observations during that sampling period.

When you turn on counter activity using stats_sampling, the counters stay active until you turn them off.

New admin stats options are:

- admin stats, backlog – reports the current backlog in the inbound and outbound stable queues.
- admin stats, { tps | cps | bps } – reports throughput in transactions per second, commands per second, or bytes per second.
- admin stats, status – reports configuration information for counters.

Support for isolation levels

Isolation levels let you control the degree to which data can be accessed by other users during a transaction. With version 15.0, Replication Server decouples isolation levels and serialization methods for the replicate data server, and enables all isolation levels for replicate data servers that Replication Server supports. In versions earlier than 15.0, Replication Server supported only isolation level 3.

Through the use of custom function strings, Replication Server supports all isolation levels the replicate data servers may use. Support is not limited to the ANSI standard only.

Each isolation level specifies the types of actions that are not permitted while concurrent transactions are processing. Higher levels include the restrictions imposed by lower levels.

You can set the isolation level with the database configuration parameter dsi_isolation_level. The ANSI standard levels supported by Adaptive Server are:

- 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.

The default value is the current transaction isolation level for the target data server.

For example, to select isolation level 2 for the Replication Server connection to the TOKYO_DS data server and pubs2 database, enter:

```
alter connection to TOKYO_DS.pubs2
set dsi_isolation_level to '2'
```

In versions earlier than 15.0, you set isolation level 3 and the serialization method at the same time:

```
alter connection to TOKYO_DS.pubs2
set dsi_serialization_method to 'isolation_level_3'
```

With version 15.0, the equivalent is:

```
alter connection to TOKYO_DS.pubs2
set dsi_serializaiton_method to 'wait_for_start'


alter connection to TOKYO_DS.pubs2
set dsi_isolation_level to '3'
```

Bidirectional replication support for DDL in MSA

You can configure multisite availability (MSA) to set up a two-way replication of data definition language (DDL) transactions between two Adaptive Server databases.

In version 15.0, bidirectional DDL replication support in MSA environments is supported only for non-warm standby databases.

Replication Server 15.0 supports this bidirectional replication using a new configuration parameter dsi_replication_ddl. When dsi_replication_ddl is set to on, DSI sends set replication off to the replicate database, which instructs it to mark the succedent 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 the DDL transactions replication in bidirectional MSA replication environment.

❖ **Setting up bidirectional replication**

1. Create a bidirectional MSA replication environment. For steps, see Chapter 12, "Managing Replicated Objects Using MultiSite Availability," in the *Replication Server Administration Guide Volume 1*.
2. Grant "set session authorization" privilege to a maintenance user on the destination database, as shown in the following example:

   ```
   grant set session authorization to maint_user
   ```

3. In the destination database, set dsi_replication_ddl to on to enable bidirectional DDL replication, as shown in the following example:

```
alter connection to dataserver.database set dsi_replication_ddl on
```

4. Replicate DDL transactions.

Batching of commands for non-ASE servers

Replication Server 15.0 allows you to batch commands for non-ASE database servers. By batching commands, you may be able to achieve improved performance in Replication Server.

New function strings

Support for command batching to non-ASE servers is achieved through the use of two function strings, rs_batch_start and rs_batch_end. These function strings store the SQL translation needed for marking the beginning and end of command batches. Use of these function strings is not necessary for Adaptive Server Enterprise or any other data server where the function strings rs_begin and rs_commit already support the needed functionality.

New DSI connection parameter

A DSI connection parameter, use_batch_markers, is used to control the processing of the two function strings, rs_batch_start and rs_batch_end. You can set use_batch_markers with alter connection and configure connection commands. If use_batch_markers is set to on the function strings rs_batch_start and rs_batch_end are executed. The default is off.

New DSI configuration parameters

DSI must be correctly configured for batching of commands to the replicate data server. There are three DSI configuration parameters to consider for each connection that will be batching commands:

- batch
- batch_begin
- use_batch_markers

For more information about configuring DSI for non-ASE data servers, see Chapter 5, “Replication Server Issues,” in the *Replication Server Heterogeneous Replication Guide*.

SySAM license management

The Sybase Software Asset Management (SySAM) implementation has changed for this version of Replication Server. The changes include:

- Asset management and reporting tools are provided with SySAM version 2.0. These tools allow you to monitor license usage and compliance.
- A single installation method supports all editions of Replication Server.
- SySAM configuration is no longer optional.
- Flexible SySAM configuration options.
- SySAM licenses are no longer shipped along with order fulfillment. You must obtain license certificates from the Sybase Product Download Center (SPDC).

- SySAM license keys include information about the support plan you purchased. You must update these licenses whenever you renew your support plan.
- Licensing policies are strictly and consistently enforced.
- Replication Server can function under grace periods if cannot obtain a license. These grace periods allow you reasonable time to respond to the issues causing license failure. During the grace period, the Replication Server continues to function normally. However, the Replication Server or any of its features, will shut down at the end of the grace period if licensing issues remain unresolved.
- You can receive real-time e-mail notifications about licensing events.
- Licenses issued from SPDC include information about the host machine where the licenses will be deployed. These licenses cannot be used on another machine without being reissued from SPDC.

Note Replication Server 15.0 does not require the REP_SSL license, as SSL now comes as a part of the basic REP_SERVER license.

These changes affect the Replication Server installation and configuration process. For pre-installation and SySAM installation information, see the *Replication Server Installation Guide* for your platform.

Plan your SySAM deployment before installing Replication Server.

CHAPTER 9

New Features in Replication Manager 15.0

This chapter describes new features in the 15.0 version of the Replication Manager, a plug-in to Sybase Central.

Topic	Page
New user interface features	133
Replication Manager features supported	138

New user interface features

The Replication Manager supports several new interface features that promote ease-of-use and productivity.

Two-tier management solution

Replication Manager can manage replication environments by connecting directly to servers in the environment without communicating through a management server layer. This two-tier management solution lets you manage small, simple replication environments with fewer than ten servers.

To set up a replication environment in Sybase Central, you select the servers contained in the environment from a drop-down list, which is derived from the local interfaces file, then provide a user name and password that the Replication Manager uses to connect to these servers. These passwords are encrypted when stored in the Sybase Central repository.

Three-tier management solution

If you are managing a large or complex replication environment, you might want to install the Replication Monitoring Services (RMS) server. In a three-tier management solution, RMS server is a middle-management layer that monitors the status of the servers and other components in the replication environment. Replication Manager connects to the servers in the environment through RMS.

The Replication Manager provides the client interface that displays the information provided by RMS.

For more information about RMS and its functionality, see Chapter 10, "Introducing Replication Monitoring Services."

Replication Manager plug-in replaces Replication Server plug-in

With Replication Server 15.0, the Replication Manager plug-in replaces the Replication Server plug-in as the complete management tool for developing, managing, and monitoring a Sybase Replication Server environment for a two-tier management solution.

In earlier versions, the Sybase Central Replication Server Manager included a Replication Server plug-in and a Replication Server Manager Server (RSM Server) as the software tools to monitor, analyze, troubleshoot, and administer a replication system.

The Replication Server plug-in included a graphical user interface (GUI) integrated with Sybase Central. The Replication Manager plug-in has an interface that is similar to the Replication Server plug-in, but that runs within the Sybase Central, Java Edition framework.

In addition, Replication Manager does not require the RSM Server to manage servers in a replication environment. It communicates directly with the Replication Server and Adaptive Servers in a two-tier management solution.

Online help

Online help for Replication Manager is now available. The online help contains extensive topic-level help that provides a quick reference for all Replication Manager concepts and tasks.

Select Help from the Sybase Central main menu and then select Replication Manager online help.

Visual monitoring of status

The state of each object displays on the object icon, in the parent object Details list, and on the Properties dialog box for that object. You can monitor the status of servers, connections, routes, and queues.

Using the Details list

When you select an object in the left pane that contains subcomponents or function components, one or more tabs display in the right pane of the Sybase Central window with lists of information. For most objects, a single tab called "Details" displays, which contains a list of general information about the object.

The Details list displays:

- Subcomponents, which are other replication or database objects that are contained in another object.
- Function components, which are components that invoke a wizard when double-clicked. For example, the Add Connection object is a function component that invokes the Add Connection wizard.

Event Log pane

Replication Manager displays an event log in a pane at the bottom of the Sybase Central window. The event log displays:

- Component state changes for connections, routes, and queues
- Server availability changes
- Background thread completion
- RMS event trigger execution

Note For more information about RMS and event triggers, see Chapter 10, "Introducing Replication Monitoring Services."

To display or hide the event log, select Event Log from the View menu.

Background processing

Several tasks performed by Replication Manager can be very time-consuming, such as creating a subscription that also materializes the table. These tasks are now performed in the background, allowing Sybase Central to continue to function. When you start a time-consuming task, Replication Manager displays a message window indicating that a process is running. You can click Stop Process in this window to stop the process.

Note The process continues even if the Background Process window is closed.

When a background task completes, the Replication Manager places an event entry in the event log.

Using the Background Processes dialog box

To see the status of a background process, you can open the Background Processes dialog box, which displays a list of all of the currently running background threads.

To access the Background Processes dialog box, select Search | Background Processes. The Background Processes dialog box opens, displaying the following:

- Process – the name of the process.
- Start time – the start time of the process.
- Status – the status of the process.

Replication Manager logging enhancement

Replication Manager now uses the Sybase Central message logging feature to provide a log of all commands sent by the Replication Manager to any server. Because the log may contain passwords (for example, passwords are needed to create a connection, and these passwords are saved in the log), the Replication Manager provides the ability to turn off command logging.

❖ **Turning command logging on and off**

1 Right-click the Replication Manager object (at the top of the tree).

2 Select Properties.

3 In the Replication Manager properties dialog box, select Write SQL Commands to Log.

To turn logging off, return to the Replication Manager properties dialog box and unselect the Write SQL Commands to Log check box.

Warning! Turning on command logging can fill up the Sybase Central log, causing it to crash. If you turn command logging on, monitor the log closely.

Script editors

Replication Manager provides two script editors, the Replication Command Language (RCL) script editor and the Structured Query Language (SQL) script editor. These editors operate the same way, except the RCL script editor highlights RCL keywords while the SQL script editor highlights SQL keywords.

You can use the script editor to view generated RCL commands, which include syntax to create connection and configuration parameters that can be used to create connections outside the Replication Manager.

With the script editors, you can:

- Select several Replication Server objects and generate RCL for all objects selected.
- Edit and save the generated RCL script.
- Load an RCL script from a file and add it to the current script.
- Resubmit an RCL script to Replication Server.

❖ **Accessing the script editor**

1 Select the Replication Server object for which you want to generate RCL.

2 Right-click that object.

3 Select Generate RCL from the context menu. The RCL Script Editor opens and contains the commands needed to create the object.

Replication Manager features supported

This section describes the Replication Server 15.0 features that the Replication Manager also supports.

Support for new datatypes

Replication Manager supports the following new datatypes supported by Replication Server 15.0:

- bigint
- unsigned integer datatypes
  - unsigned tinyint
  - unsigned smallint
  - unsigned int
  - unsigned bigint
- unitext

For more details of the datatypes supported by Replication Server 15.0, see Chapter 8, “New Features in Replication Server Version 15.0.”

Support for DirectConnect

In version 15.0, the Replication Manager manages a component that represents a Sybase DirectConnect data access server. The DirectConnect server acts as an Open Server gateway by converting the Open Client/Server protocol used by Replication Server to the native communication protocol used by the non-Sybase replicate database.

In Sybase Central, the DirectConnect data access server is managed as any other object in the replication environment. The Replication Manager displays the state on the icon of the DirectConnect data access server and on the parent object’s Detail list. The state of the server reflects whether the server is available and the state of the back-end data servers.

Limitations

In version 15.0, the Replication Manager neither displays the error log nor sets configuration parameters for DirectConnect.

Replication support

Replication Server provides replication at the database, table, and stored procedure levels. Replication Manager allows you to create a replication definition for a database, a table, or a stored procedure. Note that a replication definition for a stored procedure is called a function replication definition. You can create, alter, and delete function replication definitions and function subscriptions.

For more information about creating, altering, and deleting replication definitions and subscriptions, see the Replication Manager online help.

Routes

A route is a one-way message stream from a source Replication Server to a destination Replication Server. Routes carry replication data. You can use Replication Manager to manage routes.

For more information on managing routes using Replication Manager, see the Replication Manager online help.

Upgrading routes

With Replication Server 15.0, you can upgrade routes using the Replication Manager. In earlier versions, the RSM Server provided the route upgrade capability.

The Replication Manager displays a set of all routes in the Replication Server that are eligible to be upgraded. You can select one single route at a time for upgrade. Upon selection of a route, the Replication Manager runs the upgrade process in the background and reports any errors or warnings in the event log.

The new route upgrade implementation provides the following additional functionalities:

- Canceling a route upgrade process – the ability to cancel a route upgrade process. However, canceling can leave the replicate Replication Server in an unstable state.
- Recovering from an unsuccessful upgrade – a recovery procedure to reset the replicate RSSD. An unsuccessful upgrade can result from either an abnormal termination of the upgrade process or a cancellation of the upgrade process.

❖ **Upgrading a route using Replication Manager**

1 In the object tree, select a Replication Server object.

2 Right-click the Replication Server object and select Upgrade Route. A dialog box lists all the routes that can be upgraded.

3 From the list, select a route and click the Upgrade button.

A background process starts, which on completion indicates whether the upgrade is successful or not.

Limitations

The route upgrade implementation has these limitations:

- You cannot currently upgrade multiple routes at the same time.
- Route upgrade is supported only in an environment domain using the two-tier solution.

Troubleshooting tools

Occasionally, an environment stops replicating data. This can happen when a transaction is not formatted correctly, or when a server generates an error. To troubleshoot the situation, you can view the Replication Server queue data and exceptions log.

Accessing the exception log

Use the exceptions log to troubleshoot a problem with replication.

❖ **Using the exceptions log**

When a transaction is not processed because of a SQL error, you can:

1 Issue a resume command with a skip transaction clause to the queue.

2 Right-click the connection and select View Exceptions.

The View Exceptions dialog box displays the exceptions log in the Exceptions Log table.

3 View the skipped transaction and the erroneous SQL in the exceptions log.

4 Filter the display of the exceptions log by selecting the columns in the table: Origin Data Server, Origin Database, and Previous Hours.

5 To see the command language associated with a transaction, select it from the Exceptions Log table. The Command Editor text box displays the SQL commands in the transaction.

6 Edit the transaction in the text box.

7 To resubmit the transaction to the replicate data server, select Resubmit.

Accessing queue data

Data that is passed between servers (Adaptive Server, Replication Server, and so on) is stored in stable queues within Replication Server. The Replication Manager displays the statistics of queue usage and displays the content of the queues.

Using the View Queue Data dialog box

The View Queue Data dialog box lets you filter and sort the data from a queue as an aid in troubleshooting transactions in the queue. You can also edit, delete, or undelete a given command, or purge the first transaction in the queue.

The View Queue Data dialog box contains the following fields:

- Filter fields, which let you select the type of filters that the Replication Manager uses to display data from the queue. These filters include:
  - Column
  - Column value
  - Segment
  - Starting block
  - Number of blocks displayed
  - Number of rows displayed
  - Whether to start at the first active segment or not
  - Whether to include all data to the end of the segment
  - Whether to include all rows or not
  - Whether you want to show deleted data
  - Whether to view all data to the end of the queue
- General buttons, which let you:
  - Display the queue data with the current filters
  - Close the dialog box
  - Purge the first transaction from the queue

  - Edit transactions
  - Delete transactions
  - Undelete transactions
  - Group transactions, which returns the Queue Data scrolling list display back to grouped transactions
- Queue Data scrolling list, which contains rows of data from the current queue. Each column contains specific information about the command and transaction contained in each row. For example, to sort the queue data by a specific column, select that column name. The Queue Data scrolling list refreshes, sorting the data according to that column. An arrow displays next to the column name to show that you have sorted the data by that column. The columns you can sort by include:
  - Segment
  - Transaction Name
  - Command
  - Origin Site
  - Origin Commit Time
  - Origin User
  - Transaction ID
  - Origin QID

Note You can only delete, undelete, or purge queue transactions when Replication Server is in standalone mode. For more information, see Chapter 3, "Managing Replication Server with Sybase Central," in the *Replication Server Administration Guide Volume 1*.

❖ **Viewing queue data**

1 Right-click the queue whose data you want to view.

2 Select View Data. The View Queue Data dialog box opens.

3 To filter data shown, select one of the filter fields.

For more information, see "Using the View Queue Data dialog box" on page 141.

4 To sort the data, select segment, transaction, origin, size, status, commit time, or user.

Connection status hide options

You can hide (or filter out) the status of connections if you do not want to see the connection status either on the individual connection icon or as part of the rollup status for Replication Server.

Because the filtering state of the connection status is stored locally by the Replication Manager, different instances of the Replication Manager do not share filtering states. For example, if you create a connection using one instance of the Replication Manager, and then set the Replication Agent status to "hide" for that connection, another Sybase Central plug-in instance monitoring the same environment does not filter the connection status because the filtering information is available only to the original Replication Manager instance.

In addition, any connection created outside of Sybase Central (by rs_init or from the command line) is not filtered automatically by the Replication Manager. You must set the filtering manually from within Sybase Central.

Filtering connection status in warm standby environments

If you are creating a warm standby environment, the Replication Manager automatically sets the filtering state for the active Data Server Interface (DSI) thread and standby RepAgent thread connections. You must set filtering for the physical connection manually by selecting one of the connection status hide options from the context menu.

Using connection status hide options

The options for hiding connection status are as follows:

- Hide the State of the Replication Agent – hides the state of the Replication Agent thread in the Details list, on the Connection Properties dialog box, and in the rollup status for the Replication Server to which that Replication Agent thread is connected.
- Hide the State of the DSI Thread – hides the state of the DSI thread in the Details list, on the Connection Properties dialog box, and in the rollup status for the Replication Server to which the DSI thread is associated.

❖ **Hiding connection status**

1 Right-click the connection whose status you want to hide.

2 Select Hide Connection Status from the drop-down menu.

A dialog box shows options for hiding the connection status.

3 Select an option.

The state for that connection now reads “Hidden.” The state on the Connection Properties dialog box and in the rollup status for the Replication Server is also hidden. The Event Log records this change.

Warm standby wizards

To create a warm standby environment, you must create the following components in order:

- A logical connection
- A connection to the active database
- A connection to the standby database

Previously, these steps were only part of the Configure Replication Environment wizard, which enables you to build a warm standby environment in one step. With this version of Replication Manager, you can use a separate wizard for each step in the process, which lets you drop and re-create connections as needed.

The three wizards are:

- Add Logical Connection – add a logical connection whether you are basing this connection on an existing physical connection or not.
- Add Active Database – add an active database connection to an existing logical connection.
- Add Standby Database – add a standby database to an existing logical connection that already has an active database connection.

Using the Add Logical Connection wizard

Using the Add Logical Connection wizard, you can add a logical connection whether you are basing this connection on an existing physical connection or not.

❖ **Creating a logical connection**

1 In the tree view, select the Logical Connection folder under the Replication Server object.

2 In the Details view, double-click Add Logical Connection. The Add Logical Connection wizard starts.

3 On the Convert Existing Connection wizard page, verify that the Use an Existing Connection as the Active Connection check box is not selected.

4 Enter the database name and data server name for the logical connection.

5 Review the summary information for the logical connection.

6 If everything looks correct, click Finish. Otherwise, click Back to return to an earlier page in the wizard and change the logical connection information. Then return to the final wizard page and click Finish.

Replication Manager creates the logical connection object.

Note You must create the connections to the active and standby databases before you have a working warm standby environment. For more information, see "Using the Add Active Database wizard" on page 145 and "Using the Add Standby Database wizard" on page 146.

Using the Add Active Database wizard

Using the Add Active Database wizard, you can add an active database connection to an existing logical connection.

❖ **Creating an active database connection**

1 In the Details view, right-click the logical connection object you created using the Add Logical Connection wizard.

2 Select Add Active Database. The Add Active Database wizard starts.

3 Select the active server and active database.

4 Select the Replication Server that will manage the database connections.

5 Enter the user name and password of the maintenance user.

6 Select the user name and password that the RepAgent will use to connect to Replication Server.

If the RepAgent user does not exist, the wizard creates one for you and gives it a default name and password. Accept the defaults, or enter your own values.

7 Select the Enable all objects in the active database check box to enable database objects.

8 Review the summary information about the replication environment.

9 If everything looks correct, click Finish. Otherwise, click Back to return to an earlier page in the wizard and change the replication environment information. Then return to the final wizard page and click Finish.

Replication Manager creates the active database connection.

Using the Add Standby Database wizard

Using the Add Standby Database wizard, you can add a standby database to an existing logical connection which already has an active database connection.

❖ **Creating a standby database connection**

1 In the Details view, right-click the logical connection object you created using the Add Logical Connection wizard.

2 Select Add Standby Database. The Add Standby Database wizard starts.

3 Select the standby server and standby database.

4 Select the Replication Server that will manage the database connections.

5 Enter the user name and password of the maintenance user.

6 Select the user name and password that the RepAgent will use to connect to Replication Server.

If the RepAgent user does not exist, the wizard creates one for you and gives it a default name and password. Accept the defaults, or enter your own values.

7 Select the materialization method.

8 Select one of the following options:

- Initialize Standby Database with Dump Load, which initializes the standby database with the current data dump from the active database.
- Use Dump Marker in Transaction Log, which replicates transactions that are executed between the time the active database is enabled and the time the data is dumped.

9 Review the summary information about the replication environment.

10 If everything looks correct, click Finish. Otherwise, click Back to return to an earlier page in the wizard and change the replication environment information. Then return to the final wizard page and click Finish.

Replication Manager creates the standby database connection.

Note Before you use your warm standby environment, resume the DSI thread on the replicate database connection, if necessary.

Thread management

Replication Manager displays the state of the threads in the Replication Server. If the thread is directly related to another Replication Server component, such as a connection, route, or queue, then the thread component is represented in Sybase Central by the related component and its features. For example, the Replication Agent thread (RepAgent) or DSI threads are represented by the associated connection component. The DSI EXEC thread, however, is not represented by another component.

Using thread context menus

The Thread context menus give you access to the menus of a related component, if a related component exists for a particular thread. For example, if you right-click a DIST thread, the context menu displays Connection and Copy. If you select Connection, a context submenu displays, showing you the commands from the connection context menu.

Note Some thread objects are not related to a component and, therefore, do not give access to a submenu. For example, if you right-click a DSI EXEC thread, the context menu displays only the Copy command.

The following table shows the mapping of thread objects to related components:

Thread	Related component
DSI	Connection
DIST	
REP AGENT	

Thread	Related component
SQM	Queue
SQT	
RSI	Route
USER	User

Viewing the Details list and other thread information

When you select the Threads folder in the left pane, the Details list and tab display, as well as several other tabs. These other tabs give you access to additional information about each type of thread, as follows:

- DIST – displays information about distributor threads.
- DSI – displays information about DSI threads.
- RSI – displays information about RSI threads.
- SQM – displays information about SQM threads.
- SQT – displays information about SQT threads.

See the *Replication Server Administration Guide Volume 1* for more information about Replication Server threads.

Using thread information

You can copy information about a thread from any of the thread information lists to the system clipboard. The columns of data in each list are separated by tabs so that you can paste the columns directly into a columnar format such as a spreadsheet.

❖ **Copying thread information**

1 Right-click the thread in the right pane.

2 Select Copy.

3 Go to the destination file and paste the information as you would any text.

For example, if you select the DIST thread number 19 for a Replication Server called “myRepServer,” then select Copy from the menu and paste that information into a file, the results look similar to the following:

```
19 Awaiting Wakeup 102 myRepServer.emb2 102 P Normal 0 1 0 21787 43856 0 0 0 14
```

CHAPTER 10

Introducing Replication Monitoring Services

This chapter introduces a new Replication Server 15.0 component called the Replication Monitoring Services (RMS). RMS is the new middle-management monitoring layer that replaces the existing Replication Server Manager and provides monitoring services for large and complex replication environments.

Name	Page
Introducing Replication Monitoring Services	149
Monitoring a replication environment using RMS	153

Introducing Replication Monitoring Services

Replication Monitoring Services (RMS) replaces the functionality of the existing Replication Server Manager Server (RSM Server). RMS monitors the servers and components in a replication environment, provides the ability to control the flow of data in the replication environment, and sets the configuration parameters.

RMS functionality is available through the Replication Manager plug-in to Sybase Central and the command line API. Replication Manager provides commands to create, modify, or delete replication objects, while RMS provides an API to monitor, manage, and configure the replication environment.

RMS is applicable only for a three-tier management solution.

Three-tier management solution

A three-tier management solution is for large and complex replication environments consisting of ten or more Replication Servers. The Replication Manager connects to the servers in the environment through RMS. RMS provides the monitoring capabilities for the replication environment.

RMS monitors the status of servers and other components in the replication environment, and the Replication Manager plug-in provides the client interface that displays the status information provided by the RMS.

Table 10-1 lists the features supported in the Replication Server Manager, a component in the earlier versions, and in the Replication Monitoring Services, the new component in version 15.0.

Table 10-1: Difference in features supported in RSM and RMS

Feature	Replication Server Manager	Replication Monitoring Services
Allows you to manage, monitor, and configure replication system components	X	X
Monitors the availability of servers and the state of all connections and routes	X	X
Manages a warm standby environment	X	X
Supports multisite availability		X
Supports the Embedded RSSD	X	X
Provides support for SSL and network-based security (DCE/Kerberos) security between servers in a replication environment	X	
Runs on Microsoft Windows platforms as well as all UNIX platforms supported by Replication Server		X
Provides a server-centric view of the replication environment	X	X
Enables administration of a logical group of servers		X
Sets configuration parameters of Replication Servers, Replication Agents, the Adaptive Server Enterprise Replication Agent thread (RepAgent), connections, and routes	X	X
Monitors the latency and the state and performance of a replication path	X	X
Provides commands to create, alter, and delete replication objects	X	
Provides logging and tracing of server commands	X	X
Generates a rollup status for Replication Servers and Adaptive Servers	X	X
Executes user-defined scripts for events in the replication environment	X	X

Monitoring servers in the replication environment

Using RMS, you can monitor the following servers in your replication environment:

- Adaptive Server Enterprise
- SQL Anywhere and IQ

- Replication Agent
- Mirror Replication Agent
- DirectConnect
- Open Server
- Replication Server
- Remote RMS servers

Software requirements and compatibilities

Replication Monitoring Services requires the following software:

- JRE version 1.4.2.03
- Sybase Unified Agent Framework (UAF) version 1.5.0.97
- jConnect™ for JDBC™ version 6.0

Installation

You can use InstallShield to install the Replication Monitoring Services component at the same time you install Replication Server.

If you specify a Typical or Full installation when you install Replication Server 15.0, InstallShield automatically installs the Replication Monitoring Services component along with Sybase Central and Replication Manager.

Starting and stopping RMS

RMS is a Java application built using the Sybase Unified Agent Framework (UAF). To start or stop RMS you must start or stop UAF.

❖ **Starting RMS**

1 Navigate to the Replication Server installation directory, *%SYBASE%* on Windows, or *$SYBASE* on UNIX.

2 Set the environment variables by executing SYBASE.bat (Windows) or by sourcing SYBASE.csh (UNIX).

3 Change to the *%SYBASE_UA%\bin* directory (Windows) or the *$SYBASE_UA/bin* directory (UNIX).

4 In Windows, execute agent.bat or add this command to run Unified Agent (UA) in the background. Similarly on UNIX, execute agent or add this command to run UA in the background.

❖ **Stopping RMS**

1 Navigate to the Replication Server installation directory, *%SYBASE%* on Windows, or *$SYBASE* on UNIX.

2 Set the environment variables by executing SYBASE.bat (Windows) or by sourcing SYBASE.csh (UNIX).

3 Stop RMS:

- If an Adaptive Server Unified Agent is running, enter one of these commands:

  ```
  shutdown [-port rmi-port] [-U username]
  [-P password]
  ```

  Or

  ```
  shutdown [-port rmi-port] [-user username]
  [-password password]
  ```

  where:

  - *rmi-port* value is 9999.
  - *username* and *password* are the values configured for the UA.
- If RMS is the only Unified Agent running, enter shutdown.bat (Windows) or shutdown (UNIX).

Connecting to RMS in Sybase Central

To connect to RMS in Sybase Central:

1 Select the Connect icon from the toolbar. The Connect to a Replication Domain window opens.

2 Select the RMS Server radio button.

3 Enter the user name and password required to connect to RMS.

4 Select RMS from the list of servers in the drop-down list or click the Options button to provide the connection information for the RMS.

5 Enter a server name, host, and port number.

6 Click OK.

The RMS server is added to the object tree in Sybase Central.

Monitoring a replication environment using RMS

To monitor a replication environment, you must set up a RMS domain. This domain consists of the servers in your environment you want to monitor using the RMS. In a three-tier management solution, the Replication Manager connects to the RMS, which then connects to the various servers in your replication environment.

To set up the RMS domain and monitor servers in this domain, you can either:

- Use the Replication Manager plug-in graphical interface, or
- Use command line API

In a three-tier replication environment, you can perform some monitoring through Replication Manager, and additional monitoring through the RMS API at the command line.

The following sections provide information on how to monitor RMS using the Replication Manager plug-in and the API. For those tasks for which you can use both interfaces, only the Replication Manager plug-in is described in detail, while the name of API is identified. For those tasks that can be performed using only the API, detailed API information is given.

Adding and dropping servers for monitoring in Sybase Central

Add and drop servers in a three-tier environment in the same way as in a two-tier environment.

Adding a server

The servers that you add for monitoring in a three-tier solution can be from anywhere in your network. This allows you to monitor replication systems that are distributed worldwide.

You can use Replication Manager wizard to add a server to a RMS domain. Depending on the type of server you are adding (Adaptive Server, Replication Server, RepAgent, Open Server, Sybase Replication Agents, DirectConnect), the wizard prompts for different information.

Before you add a server, have this information available:

- The server's name.
- The type of server you are adding.
- The user name and password used to administer the server. The login must have System Administrator privileges on the server.
- Replication Server only – the user name and password of the RSSD primary user.

To add servers to the RMS server folder in Sybase Central:

1 Select the RMS.

2 Select File | New | Servers, or double-click the Add Servers icon in the right pane of the main window.

3 Enter the required information in the wizard dialog boxes.

The servers are added to the Sybase Central viewer under the RMS server in the object tree.

API command to add server to monitor

Use add *server* to add a server to be monitored by RMS. For detailed information about this API, see Chapter 9, "Replication Monitoring Services API," in the *Replication Server Reference Manual*.

Dropping a server

To drop a server from the RMS domain:

1 Select the server you want to drop.

2 Do one of the following:

- Click the Delete icon from the toolbar.

- Right-click the selected server and select Delete from the context menu.

Note Although Sybase Central removes the server from the RMS server folder, the server is not actually removed from your replication system. Therefore, the server name may still appear in the dialog boxes because there are routes or database connections associated with it.

API command to drop server from being monitored

Use drop *server* to drop a server that is being monitored by RMS. For detailed information about this API, see Chapter 9, "Replication Monitoring Services API," in the *Replication Server Reference Manual*.

Viewing monitored objects in Sybase Central

Viewing monitored objects in Replication Manager is the same as viewing objects in a two-tier environment.

In the object tree, double-click or expand the RMS icon to view the replication objects managed by RMS. Under RMS, you can view the monitored servers and its components such as connections, routes, queues, and threads. When you select a particular replication object such as the Routes folder, you can view the list of created routes. You can manage these replication objects using Replication Manager.

Setting configuration parameters for monitored replication objects

You can set the configuration parameters for the following replication objects monitored by RMS:

- Replication Server
- Replication Agent thread
- Remote RMS server
- Database connections and logical connections
- Routes

To set the configuration properties of an object, in Replication Manager

1 Select the object and choose File | Properties. The property sheet for the object opens.

2 Select the parameters tab in the Properties dialog box. The Parameters page displays a list of all configuration parameters for the selected server or component.

3 Select the parameter you want to modify and click the Edit Parameter button.

4 In the Edit dialog, enter the new parameter and click OK.

5 You can modify any other parameter in the list. When you are finished, click OK in the Properties dialog box.

API commands to set configuration parameters

RMS API commands to set configuration parameters for the replication objects are:

- configure server – returns configuration parameter information for a Replication Server or Replication Agent, or sets the value of a specified configuration parameter. configure server also retrieves and sets RMS-specific parameters.
- configure component – returns configuration parameters for a component or sets the value of the configuration parameter specified.
- configure *RMS* – returns the configuration parameter information for the RMS, or sets the value of a specified RMS configuration parameter

For detailed information about these APIs, see Chapter 9, "Replication Monitoring Services API," in the *Replication Server Reference Manual*.

Monitoring a logical group of servers

RMS enables you to define a set of servers as a logical group and monitor the group as a single entity. You can have many logical groups in your replication environment. The servers in a group can belong to different logical groups. However, all servers in one logical group must be of the same type. For example, you can have one logical group of all Replication Servers and another for all Adaptive Servers. When you issue any command to a group, it affects all the servers contained in this logical group.

You can delete logical groups. When you delete a logical group, only the group is deleted, not the servers within the group.

RMS also returns a rollup status for each group or each server in a 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.

API commands to create/delete/list logical groups

RMS API commands to manage logical groups are:

- create group – defines a logical group of servers.
- delete group – deletes a logical group that was added using the create group command.
- get group – returns a result set that contains either a list of the groups and a rollup status for each group or for each server.

For detailed information about these APIs, see Chapter 9, "Replication Monitoring Services API," in the *Replication Server Reference Manual*.

Suspending or resuming components in the replication environment

You can suspend or resume the following monitored components in the replication environment:

- RepAgent in Adaptive Server
- Replication Agent and DSI connection in Replication Server
- Routes
- Queues (resume only)
- Replication in a Replication Agent

In Sybase Central, select the monitored connection or route from the RMS Server folder in the object tree:

- To suspend a monitored connection or route, select the connection or route object from the RMS Server folder and select Suspend from the context menu.
- To resume a monitored connection or route, select the connection or route object from the RMS Server folder and select Resume from the context menu.

API commands to suspend/resume replication components

RMS API commands to suspend or resume components in the replication environment are:

- suspend *component* – suspends a component in a specified server.
- suspend *replication* – suspends replication in a Replication Agent.
- resume *component* – resumes a component in a specified server.
- resume *replication* – resumes replication in a Replication Agent.

For detailed information about these APIs, see Chapter 9, "Replication Monitoring Services API," in the *Replication Server Reference Manual*.

Shutting down monitored servers

RMS enables you to shut down any monitored Replication Server, Replication Agent, or Mirror Replication Agents.

In Sybase Central, select the monitored server or RMS in the object tree and select Shutdown from the context menu. To shut down a group of servers, select the logical group folder and select Shutdown All from the context menu.

API command to shut down Replication Server, group, or RMS

Use shutdown *server* to shut down one or more, or group of monitored Replication Servers or RMS. For detailed information about this API, see Chapter 9, "Replication Monitoring Services API," in the *Replication Server Reference Manual*.

Generating rollup status for servers

The RMS status 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.

For example, the status of the Adaptive Server reflects the status of its Replication Agent threads. If any of the monitored Replication Agent threads in the Adaptive Server are not UP, then the state of the Adaptive Server is set to SUSPECT.

For more information about RMS server and component states, see Appendix C, "RMS Server and Component States," in the *Replication Server Reference Manual*.

API commands to view rollup status information

RMS API commands to view rollup status information and the various server states are:

- get servers – returns a list of servers that are monitored by the RMS, and the status of the RMS environment.
- get status description – retrieves the list of status descriptions for a server or component.

For detailed information about these APIs, see Chapter 9, "Replication Monitoring Services API," in the *Replication Server Reference Manual*.

Generating latency and heartbeat information

The RMS heartbeat feature uses the stored procedure rs_ticket to generate latency information, which is the amount of time it takes for a transaction to move from the primary database to the replicate database. You can use this information to monitor Replication Server performance, module heartbeat, replication health, and table-level quiesce.

At a specified interval, RMS executes rs_ticket at the primary database. The generated latency information is stored in a table in the replicate database. You can use RMS to set up the heartbeat process to retrieve the latency information from the replicate database.

For more information about rs_ticket and its functionality, see Chapter 6, "Adaptive Server Stored Procedures," in the *Replication Server Reference Manual*.

API commands to monitor heartbeat

RMS API commands to monitor heartbeat are:

- get heartbeat – retrieves a list of the heartbeat processes that have been defined in the RMS.
- get heartbeat ticket – retrieves a set of tickets from the rms_ticket_history table, for the heartbeat process and date and time range specified.
- 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 rms_ticket_history table.

For detailed information about these APIs, see Chapter 9, "Replication Monitoring Services API," in the *Replication Server Reference Manual*.

Adding event triggers

Replication Monitoring Services is designed to monitor the replication environment. When something happens in your environment, server and component status changes. These changes are displayed in the event log. RMS allows you to create event triggers to monitor these changes.

Event triggers notify you when some events occur in the replication environment. RMS executes the script when the specified event occurs. For example, a user can set up a script to be notified with an e-mail message when a connection suspends. You can create an event trigger for any server or component that the RMS monitors.

❖ **Creating an event trigger for a Replication Server**

1 In the object tree, select the Replication Server.

2 On the right side of the desktop, select the event log pane.

3 Double-click the Add Server Event Trigger icon.

4 Select the status change that will trigger the event.

5 As an option, enter a "Wait before executing" value. This notifies RMS to wait for the event to change before executing the trigger.

6 To execute a trigger at each monitoring interval, rather than only once, select "Execute at Each Interval."

7 Enter the name of the script for RMS to execute when the event occurs.

8 Click OK. The new event displays in the Event Log pane.

API commands to add/drop/get triggers

RMS API commands for trigger-related tasks are:

- 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.
- drop *event* trigger – removes a trigger that the RMS is monitoring.
- get triggers – displays information about the triggers that are monitored by the RMS.

For detailed information about these APIs, see Chapter 9, "Replication Monitoring Services API," in the *Replication Server Reference Manual*.

CHAPTER 11

New Features in Replication Server Version 12.6

This chapter describes the new features introduced with Sybase Replication Server version 12.6 and 12.5 EBF.

Topic	Page
MultiSite availability (MSA)	161
Support for symmetric multiprocessors (SMP)	162
The embedded RSSD (ERSSD)	164
Performance enhancements	164
date and time datatypes	167
Support for sending encrypted passwords	167
New bulk materialization method	168
Chinese character set (GB18030) support	169

MultiSite availability (MSA)

MSA extends Replication Server replication capabilities and can make the process of setting up a replication system both faster and easier.

Some of the features that MSA provides include:

- A simple replication methodology that requires only one replication definition for the primary database and only one subscription for each subscribing database.
- A replication filtering strategy that lets you choose whether or not to replicate individual tables, transactions, functions, system stored procedures, and data definition language (DDL).
- Replication of DDL to any replicate database—including non–warm standby databases.
- Replication to multiple replicate sites—for standby as well as nonstandby databases.

You can overlay MSA scenarios onto your existing replication structure. The procedures for implementing MSA are similar to those you already use to replicate to warm standby or replicate databases.

Database replication

When you use table and function replication, you describe each piece of data that is to be replicated using individual table and function replication definitions and subscriptions. This methodology allows you to transform data and provides fine-grained control over the information being entered in the replicate database. However, you must mark each table or function to be replicated, create a replication definition for each replicated table or function, and create subscriptions for each replication definition at each replicate database.

MSA lets you identify specific database objects: tables, functions, transactions, DDL, and system stored procedures in a single replication definition. You can choose to replicate the entire database; or you can choose to replicate—or not replicate—specific tables, functions, transactions, DDL, and system stored procedures in that database. If you do not need to replicate partial tables, MSA can provide replication while affording the advantages of simple setup and maintenance.

When the replicate is a warm standby database

In the non-MSA warm standby scenario, changes to the primary database are copied directly to the warm standby database without alteration. This methodology allows replication of DDL. To change or qualify the data sent, you must add table and function replication definitions. Each primary database can have one, and only one, standby database.

MSA provides all the features of traditional Sybase warm standby. In addition, MSA:

- Enables replication to multiple standby databases
- Provides the option to replicate or not replicate specific database objects

MSA supports the use of logical connections.

Support for symmetric multiprocessors (SMP)

Replication Server 12.6 lets you run Replication Server on either symmetric multiprocessor (SMP) or single-processor platforms. Replication Server multithreaded architecture supports both hardware configurations.

On a single processor platform, Replication Server threads run serially. On a multiprocessor platform, Replication Server threads can run in parallel, thereby improving performance and efficiency.

Replication Server support for multiple processors is based on Open Server support for multiple processors. Both servers use the POSIX thread library on UNIX platforms and the WIN32 thread library on Windows NT platforms. For detailed information about Open Server support for multiple processing machines, see the *Open Server Server-Library/C Reference Manual*.

When Replication Server is in single-processor mode, a server-wide mutual exclusion lock (mutex) enforces serial thread execution. Serial thread execution safeguards global data, server code, and system routines, ensuring that they remain thread-safe.

When Replication Server is in multiprocessor mode, the server-wide mutex is disengaged and individual threads use a combination of thread management techniques to ensure that global data, server code, and system routines remain secure.

Configuring SMP

To enable SMP on a multiprocessor machine, use configure replication server with the smp_enable option. For example:

```
configure replication server set smp_enable to 'on'
```

Monitoring thread status

You can verify Replication Server thread status using these commands and procedures:

- admin who – provides information on all Replication Server threads
- admin who_is_up or admin who_is_down – lists Replication Server threads that are running, or not running.
- sp_help_rep_agent – provides information on the RepAgent thread and the RepAgent User thread. This is an Adaptive Server stored procedure.

Monitoring performance

Replication Server provides monitors and counters for monitoring performance. See Chapter 15, “Using Counters to Monitor Performance,” in the *Replication Server Administration Guide Volume 2*.

See the *Replication Server Administration Guide* for more information about increasing Replication Server performance using SMP.

The embedded RSSD (ERSSD)

Replication Server can run either on an Adaptive Server Replication Server System Database (RSSD) or on an embedded RSSD (ERSSD). ERSSD is designed for users who do not want to manage the Replication Server RSSD in Adaptive Server. Replication Server is easier to install and manage with ERSSD. If you select embedded RSSD when you install Replication Server, ERSSD is automatically installed, configured, and started in the background. It is self-maintained. Backup procedures are automatic and pre-configured.

Limitations

Currently, you cannot create a route originating from Replication Server with ERSSD. Nor can you migrate between RSSD and ERSSD.

To use the ERSSD, you must select it when you install Replication Server. For more details, see the *Replication Server Installation Guide* and the *Replication Server Administration Guide*.

Performance enhancements

Replication Server 12.6 includes new performance enhancements. See Chapter 16, "Performance Tuning," in the *Replication Server Administration Guide Volume 1* for detailed information about these enhancements.

Better management of empty transactions

Transactions that contain only a begin and a commit statement can degrade the performance of warm standby connections. To enhancement performance, these transactions are now deleted from the inbound queue as they are read.

To further enhance performance, Sybase recommends that you also tune your application to eliminate as many of these empty transactions as possible.

Internal commit control for parallel processing

To resolve conflicting updates when using parallel processing, Replication Server must maintain transaction commit order and resolve commit consistency deadlocks.

Replication Server introduces a new method to maintain commit control using the function string rs_dsi_check_thread_lock. Replication Server uses rs_dsi_check_thread_lock to check whether the current DSI executor thread is blocking another replicate database process. This new method handles commit control within Replication Server thus requiring less network I/O than other methods, and may result in the rollback of only one transaction instead of many.

New database connection parameters used with internal commit control are:

- dsi_commit_check_locks_intrvl
- dsi_commit_check_locks_max
- dsi_commit_control

New Replication Server configuration parameters

You can use new Replication Server configuration parameter to fine-tune Replication Server performance.

- sqt_init_read_delay – the amount of time an SQT thread sleeps while waiting for an SQM read before checking for new instructions in its command queue.
- sqt_max_read_delay – the maximum amount of time an SQT thread sleeps while waiting for an SQM read before checking for new instructions in its command queue.

New database configuration parameters

You can use new database connection configuration parameters to fine-tune Replication Server performance.

- dsi_commit_check_locks_intrvl – specifies the number of milliseconds (ms) the DSI executor thread waits between executions of the rs_dsi_check_thread_lock function string. Used with parallel DSI.
- dsi_commit_check_locks_max – specifies 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.

- dsi_commit_control – specifies whether commit control processing is handled internally by Replication Server using internal tables (on) or externally using the rs_threads system table (off). Used with parallel DSI.

Changed database configuration parameters

The database connection parameters dsi_serialization_method and dsi_partitioning_rule have changed.

dsi_serialization_method specifies how and when parallel DSI threads can start. It includes these new or changed options:

- 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. Replaces the none option. When used with the origin partitioning parameter, this option replaces the single_transaction_per_origin serialization method.
- isolation_level_3 – is the same as wait_for_start, except that DSIs will specify isolation level 3 when connecting to the replicate database.
- wait_for_commit – specifies that a transaction can start only when the previous transaction is ready to commit.
- none – maintained for backward compatibility. Replaced by wait_for_start.
- single_transaction_per_origin – maintained for backward compatibility. Replaced by dsi_serialization_method set to wait_for_start and dsi_partitioning_rule set to origin.

dsi_partitioning_rule specifies the partitioning rules the DSI uses to partition transactions among available parallel DSI threads. It includes these new options:

- origin – specifies that transactions with the same origin must be serialized when applied to the replicate database.
- origin_sessid – specifies that transactions with the same origin *and* the same process ID (SPID in Adaptive Server) must be serialized when applied to the replicate database. The LTL version must be 600 or later.

  Sybase recommends that you try setting dsi_partitioning_rule to origin_sessid, time as this setting may provide the most efficient partitioning.

date and time datatypes

There are two new datatypes, date and time, in Replication Server. These datatypes extend the existing datetime and smalldatetime datatypes, providing date and time columns to replicate and standby databases. Both are fixed-width 4-byte datatypes that support rs_subcmp, and mixed-version environments.

Replication Server version 12.6 with date and time datatype support is backward-compatible with earlier versions of Adaptive Server. However, earlier versions of Adaptive Server do not recognize date and time, and thus can send only datetime and smalldatetime data.

The new columns generated by date and time datatypes allow you to replicate date and time data to both standby and replicate databases. These columns can be part of the primary key in a replication definition, and are searchable columns in a replication definition. You can use date and time columns in the where clause of define subscription, create subscription, or create article. In the same way, the date and time columns are searchable parameters in a function replication definition, again used in the where clause of define subscription, create subscription, or create article.

Table 11-1: Range and storage needs for date/time datatypes

Datatype	Range	Storage needed
date	January 1, 0001 to December 31, 9999	4
time	12:00:00 AM to 11:59:59.999 PM	4
smalldatetime	January 1, 1900 to June 6, 2079; 12:00:00AM to 11:59:59:999 PM	4
datetime	January 1, 1753 to December 31, 9999; 12:00:00AM to 11:59:59.999 PM	8

Support for sending encrypted passwords

Replication Server 12.6 supports the -X option in isql that sends encrypted passwords through the network when making a client connection.

To ensure that all Replication Server client connections—except the first connection to the RSSD—send encrypted passwords, set the Replication Server configuration parameter send_enc_password to on. For example, enter:

```
configure replication server
 set send_enc_password to 'on'
```

To ensure that all Replication Server client connections, *including* the first connection to the RSSD, send encrypted passwords, set the configuration parameter RS_enc_pw to on in the *rs_name.cfg* file using a text editor.

If RS_enc_pw is on, all Replication Server connections to the RSSD send encrypted passwords, even if send_enc_password is off.

New bulk materialization method

Replication Server 12.6 supports a new bulk materialization method for copying or moving a database from a source Adaptive Server to a destination Adaptive Server without shutting down the source Adaptive Server. The Adaptive Server quiesce database ... to *manifest_file* and mount commands let you quiesce the server and copy or move the database.

To use this bulk materialization method, both the source and destination database servers must be Adaptive Server version 12.5.1 or later.

You can use quiesce database ... to *manifest_file* to generate all the data storage information and then use mount to mount the data to a new database—see Chapter 10, "Managing Subscriptions," in the *Replication Server Administration Guide*. You can use quiesce database ... to *manifest_file* and mount when you add a warm standby database—see Chapter 13, "Managing Warm Standby," in the *Replication Server Administration Guide Volume 1*.

For information about mount and unmount, see Chapter 22, "Database Mount and Unmount," in the *Adaptive Server Enterprise System Administration Guide*.

Chinese character set (GB18030) support

Replication Server supports all character sets supported by Adaptive Server Enterprise. Accordingly, Replication Server 12.6 supports the Chinese character set (GB18030).

Index

Numerics

64-bit support 32

A

Adaptive Server support, in Replication Server 15.5 26
Adaptive Servers
  communication through Replication Manager 134
  shared-disk cluster support 76
adding event triggers in Replication Manager 159
adding servers to RMS server 153
admin config 98
admin config command 39
admin who
  enhancement 22
Advanced Services Option 8
alter connection command 38
alter partition command 122

B

background processes
  running in background 136
  stopping a background process 136
  viewing in Background Processes dialog box 136
Background Processes dialog box 136
batching of commands
  for non-ASE servers 129
  using DSI connection and configuration parameters 130
  using function strings 130
bigdatetime and bigtime datatype support in Replication Manager 35
bigdatetime, replication support 26
bigtime, replication support 26
bulk copy-in support 37–41
  commands for 38–39
  connection parameters 38
  connection parameters, checking value of 39
  connection parameters, setting value of 38
  Data Server Interface (DSI), implementation in 38, 39
  multi-statement transactions, support for 39
  subscription materialization, changes to 39
bulk insert.*See* bulk copy-in support
bulk materialization 168

C

cascading connection, in Replication Server gateway 51
changing replication definitions, enhancements to process 15
character set
  Chinese (GB18030) 169
Chinese character set support 169
commands
  admin config 39, 45
  alter connection 38, 42, 45
  alter replication definition 46
  configure replication server 38, 43
  connect 52, 53
  create connection 45
  create replication definition 46
  create route 123
  disconnect 53
  show connection 53
  show server 53
  sysadmin dump_tran 81
  sysadmin issue_ticket 24
  sysadmin sqm_unzap_tran 80
  sysadmin sqm_zap_tran 80
commit control
  internal 164

computed columns
materialized 119
replicating 119
virtual 119
configuration
stable queue cache parameters 91
configuration parameters
dist_sqt_max_cache_size 32
dsi_bulk_copy 38, 39
dsi_bulk_threshold 38, 39
dsi_non_blocking_commit 42
dsi_sqt_max_cache_size 32
dynamic_sql 101
dynamic_sql_cache_management 102
dynamic_sql_cache_size 101
exec_cmds_per_timeslice 25
init_sqm_write_delay 25
init_sqm_write_max_delay 25
memory_limit 25, 33
setting with Replication Manager 155
smp_enable 25
sqt_max_cache_size 25, 32
stats_reset_rssd 100
stats_sampling 127
sts_full_cache 25
configure replication server command 38, 43
connection profiles 68
connection status
filtering in warm standby environments 143
hide options 143
conventions, document style xii
counter statistics
displaying on screen 127
optionally keeping in RSSD 100
reporting 127
saving to RSSD 127
counters 83
create partition command 122

D

Data Server Interface 38, 39
data transfer, incremental 28
database generation numbers, resetting 23
database resynchronization 18
database support 4
datatype
bigdatetime 26
bigint 116
bigtime 26
date, time 167
unitext 118
unsigned 117
datatypes
opaque 86, 95
supported by Replication Manager 138
timestamp 96
date, datatype 167
default parameter values, changes 24
deferred name resolution, replication support 27
delaying replication 17
Details list 135
dialog boxes
Background Processes 136
Connection Properties 143
View Queue Data 141
direct I/O file access 92
DIST status recording 89
dist_sqt_max_cache_size 32
distributor thread read efficiency, enhanced 11
dropping servers from RMS server 154
DSI 38, 39
DSI efficiency enhanced 10
dsi_bulk_copy connection parameter 38, 39
checking value of 39
See also bulk copy-in support
setting value of 38
dsi_bulk_threshold connection parameter 38, 39
checking value of 39
See also bulk copy-in support
setting value of 38
dsi_max_cmds_in_batch configuration parameter 5
dsi_max_xacts_in_group configuration parameter 6
dsi_quoted_identifier 45
dsi_serialization_method configuration method 6
dsi_sqt_max_cache 32
dynamic parameters
configuring 97
dynamic SQL
configuration parameters setup 101
enhancement 73, 93

dynamic SQL parameters 98
dynamic SQL, extending replicate_minimal_columns parameter to connections 13
dynamic SQL, optimizing statement execution 12
dynamic SQL, using with replicate minimal columns clause 12

E

editions, types of 1
editors
  RCL script editor 137
  SQL script editor 137
Embedded Replication Server System Database 164
  create route 123
  limitations 164
  routing 123
empty transactions 164
encrypted passwords
  extended support 82
  sending 168
enhanced distributor thread read efficiency 11
enhanced DSI efficiency 10
enhanced memory allocation 11
enhanced RepAgent executor thread efficiency 10
enhanced text update 89
enhancements
  admin who 22
  configuration and tuning 97
  dump transaction 87
  dynamic SQL 73, 93
  error handling 19
  for Replication Server performance 38, 100
  function replication 75, 94
  locales directory 81
  log_first_tran 80
  monitor and counter 76, 98
  non-ASE error class support 65–66
  non-ASE replicate support 67–71
  non-blocking commit 42–44
  quoted identifiers 45–51
  release area 81
  Replication Manager plug-in 114
  Replication Server gateway 51–53
  resume connection 79
  rs_subcmp 103
  SQL statement replication 57–64
  SQM performance 90
  stable queue management 78
  sysadmin dump_queue 78
  sysadmin sqt_dump_queue 79
error handling
  enhancement 19
ERSSD
  See Embedded Replication Server System Database
Event Log pane 135
  displaying or hiding 135
  viewing events 135
event triggers
  adding in Replication Manager 159
  creating in RMS 159
exceptions log
  troubleshooting 140
  viewing 140
exec_cmds_per_timeslice 25

F

failback system, by delaying replication 17
fstr_cache_size 33
function replication
  enhancement 75
function string
  rs_dsi_check_thread_lock 6, 165
  rs_non_blocking_commit 43
  rs_non_blocking_commit_flush 43
  rs_set_dml_on_computed 119
  rs_set_quoted_identifier 47
function string efficiency improvements, extending none parameter of function string commands 13

H

hash algorithm
  command line parameters 104
  configuration file parameters 104
heterogeneous parallel DSI 5
heterogeneous replication support 4
High Volume Adaptive Replication 9

HVAR 9

I

IMDB 29, 31
increasing queue block size 11, 12
incremental data transfer, support for 28
init_sqm_write_delay 25
init_sqm_write_max_delay 25
injecting rs_ticket markers 24
in-memory databases 29
internal commit control 164
isolation level
  kinds of 128
  setting 128

J

Java JRE
  compatibility with Replication Monitoring Services 151
jConnect for JDBC
  compatibility with Replication Monitoring Services 151

L

license management
  with SySAM 130
licenses, types of 1
limitations
  dynamic SQL 74
  function replication 75
  LOB datatypes 85
  opaque datatype 86
  partial update 85
list
  details 135
LOB datatypes
  partial update 85
  support 94
locales directory
  changes 81
locking schema, RSSD 26
log
  event 135
longer identifiers 115

M

manual data reconciliation 108
marking identifiers as quoted 46–47
master database
  supported DDL commands 102
md_source__memory_pool 33
memory allocation, enhanced 11
memory_limit 25, 33
message logging
  disabling 136
  enabling 136
minimal DML logging 31
mixed-version
  enhanced support for 125
mixed-version environment restrictions, with version 15.5 and later 31
monitoring of status 135
monitoring replication environment
  for three-tier environment 149
  using heartbeat and latency information 159
  using Replication Manager graphical interface 153
  using RMS 153
  using RMS API 153
monitoring replication environment using RMS 153
monitoring status
  in Details list 135
  replication objects 135
  visual display 135
mount command 168
multiprocessors
  enabling 163
  monitoring 163
multisite availability (MSA) 161
multithreaded architecture 162

N

new features

Replication Manager 15.0 133
Replication Manager 15.0.1 111
Replication Manager 15.1 93
Replication Manager 15.5 35
Replication Server 15.0 115
Replication Server 15.0.1 97
Replication Server 15.1 73
Replication Server 15.2 37
Replication Server 15.5 1
non-ASE error class support 65–66
altering error classes 66
creating error classes 65
default non-ASE error classes 65
native error codes 66
non-ASE replicate support 67–71
connection profiles 68
listing connection profiles 70
simplified installation, configuration 67
using connection profiles 68
non-blocking commit 42–44
ASE delayed commit feature 42
configuring 42
dsi_non_blocking_commit 42
non-ASE databases, support for 44
Oracle, support for 44
rs_non_blocking_commit 43
rs_non_blocking_commit_flush 43
system functions for 43
none parameter, extending scope in alter function string, create function string 13

O

online help
in Replication Manager plug-in 134
invoking 135
opaque datatypes 95
operating system support, in Replication Server 15.5 32
Oracle, trigger execution 7

P

parallel DSI parameter
dsi_max_cmds_in_batch 5
dsi_max_xacts_in_group 6
dsi_serialization_method 6
parallel processing 162
empty transactions 164
internal commit control 164
parameters, changes to default values 24
password encryption
Advanced Encryption Standard algorithm 124
extended support 82
FIPS-certified 124
for maintenance user passwords 125
for route user passwords 125
for user passwords 125
for user passwords in configuration file 125
performance enhancements, in Replication Server 15.5 7
platform support 4
POSIX thread library 163
product editions, types of 1

Q

queue block size, increasing 11, 12
queue data
troubleshooting 141
viewing 141
queue_dump_buffer_size 33
quiesce database command 168
quoted identifiers 45–51
alter replication definition, changes to 46
create replication definition, changes to 46
dsi_quoted_identifier 45
embedded double quote characters 45
enabling in DSI 45
forwarding to data servers 47
marking identifiers as quoted 46–47
product version requirements 51
quoted parameter 46
rs_helprep changes 48–50
rs_set_quoted_identifier 47

R

RCL command
admin stats 99
admin stats, status 100
real-time loading 4
real-time loading (RTL) replication to Sybase IQ 2
Real-Time Loading Edition 3
reference implementation 21
relaxed-durability databases 29
RepAgent executor thread efficiency, enhanced 10
replicate minimal columns clause, using with dynamic SQL 12
replicate_minimal_columns parameter, extending to connections using dynamic SQL 13
replicating
data, large batch of 38
database objects 162
encrypted columns 120
master database 102
partitioned tables 120
to standby databases 162
replication definition change request process enhancements 15
Replication features
supported by Replication Manager 138
Replication Manager
adding event triggers 159
new features 15.0 133
new features 15.0.1 111
new features 15.1 93
new features 15.5 35
support of Replication Server/Replication features 138
using to set configuration parameters 155
Replication Monitoring Services
accessing using Replication Manager or API 153
adding servers 153
compatibilities 151
connecting to 152
description 149
dropping servers 154
in three-tier management solution 149
installing 151
monitoring logical group of servers 156
monitoring replication environment 153
setting configuration parameters 155
shutting down monitored servers 158
software requirements 151
starting and stopping 151
suspending/resuming components 157
using Unified Agent Framework 151
viewing monitored objects 155
Replication Server
communication through Replication Manager 134
configuring using Replication Manager 155
dynamic configuration 97
new features 15.2 37
new features 15.5 1
Replication Server features
supported by Replication Manager 138
Replication Server gateway 51–53
cascading connection 51
connection dropping 53
connections, tracking 53
enable, to 52
limitations 53
product version requirements 53
Replication Server System Database (RSSD) 164
replication threshold setting, in SQL statement replication 28
replication, scheduling tasks 17
reserved words, new 26
resetting database generation numbers 23
resynchronizing databases 18
RMS
See Replication Monitoring Services
routes
upgrading 139
row count validation enhancements 19
row count verification, in SQL statement replication 62
rs_dsi_check_thread_lock function string 165
rs_helprep 48–50
rs_subcmp
parameters for manual reconciliation 108
schema sub-types 106
schema types 106
string options 105
rs_ticket
version 2 82
rs_ticket markers, injecting 24
RSSD locking schema, changes 26
RTL 2

S

sampling data
  optionally keeping in RSSD 100
scheduling replication tasks 17
schema comparison 105
  command line parameters 107
  configuration file parameters 107
segment preallocation 92
send_enc_password configuration parameter 167
serialization method
  wait_after_commit 5
setting configuration parameters
  using Replication Manager 155
smp_enable 25
software compatibilities of Replication Monitoring Services 151
software requirements for Replication Monitoring Services 151
SQL statement replication 57–64
  autocorrection 62
  configuring warm standby 64
  database level 57
  database replication definition 59
  product and mixed version requirements 64
  replicate SQLDML clause 59
  restrictions 62
  row count verification 62
  RSSD modifications 64
  session level 59
  set repmode 59
  setting replication threshold 28
  sp_setrepdbmode 57
  sp_setrepdefmode 58
  table level 58
  table replication definition 61
  WS_SQLDML_REPLICATION parameter 64
sqt_max_cache_size 25, 32, 33
sre_reserve 33
stable queue caching 90
starting and stopping RMS 151
stats_reset_rssd 100
status monitoring using RMS 135
sts_cachesize 33
sts_full_cache 25
subscription materialization 39
support
  Adaptive Server integer identity 90
  bulk copy-in. *See* bulk copy-in support
  contacting Sybase Technical Support xiv
  direct I/O 92
  dynamic configuration 111
  for Adaptive Server shared-disk cluster 76
  for DirectConnect 138
  for heterogeneous data servers 112
  for non-Sybase data servers 112
  for Replication Agents and Mirror Replication Agents 113
  LOB datatypes 84, 94
  opaque datatypes 95
  timestamp datatypes 86, 96
Sybase IQ, replicating to 2
symmetric multiprocessors (SMP) 162
syntax conventions xiii

T

threads
  in Replication Server 147
  using context menu 147
  viewing details 148
three-tier management solution 149
time, datatype 167
timestamp datatypes 96
transactions
  empty 164
trigger execution, for Oracle 7
two-tier management solution 133, 134

U

unsigned datatypes 117
  unsigned bigint 117
  unsigned int 117
  unsigned smallint 118
  unsigned tinyint 118
usability and process enhancements, in Replication Server 15.5 14
user documentation, for Replication Server x
user interface features, Replication Manager 133

V

visual monitoring of status 135

W

warm standby
adding active database connection 145
adding logical connection 144
adding standby database connection 146
filtering connection status 143
heterogeneous 6
Oracle 6
wizards in 144
WIN32 thread library 163
wizards
Add Active Database 145
Add Logical Connection 144
Add Standby Database 146