

SYBASE®

Troubleshooting Guide

Replication Server®

15.0

DOCUMENT ID: DC35920-01-1500-01

LAST REVISED: August 2006

Copyright © 1992-2006 by Sybase, Inc. All rights reserved.

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

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

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

Sybase, SYBASE (logo), ADA Workbench, Adaptable Windowing Environment, Adaptive Component Architecture, Adaptive Server, Adaptive Server Anywhere, Adaptive Server Enterprise, Adaptive Server Enterprise Monitor, Adaptive Server Enterprise Replication, Adaptive Server Everywhere, Advantage Database Server, Afaia, Answers Anywhere, Applied Meta, Applied Metacomputing, AppModeler, APT Workbench, APT-Build, APT-Edit, APT-Execute, APT-Translator, APT-Library, ASEP, Avaki, Avaki (Arrow Design), Avaki Data Grid, AvantGo, Backup Server, BayCam, Beyond Connected, Bit-Wise, BizTracker, Certified PowerBuilder Developer, Certified SYBASE Professional, Certified SYBASE Professional Logo, ClearConnect, Client-Library, Client Services, CodeBank, Column Design, ComponentPack, Connection Manager, Convoy/DM, Copernicus, CSP, Data Pipeline, Data Workbench, DataArchitect, Database Analyzer, DataExpress, DataServer, DataWindow, DataWindow .NET, DB-Library, dbQueue, Dejima, Dejima Direct, Developers Workbench, DirectConnect Anywhere, DirectConnect, Distribution Director, Dynamic Mobility Model, e-ADK, E-Anywhere, e-Biz Integrator, E-Whatever, EC Gateway, ECMAP, ECRTIP, eFulfillment Accelerator, EII Plus, Electronic Case Management, Embedded SQL, EMS, Enterprise Application Studio, Enterprise Client/Server, Enterprise Connect, Enterprise Data Studio, Enterprise Manager, Enterprise Portal (logo), Enterprise SQL Server Manager, Enterprise Work Architecture, Enterprise Work Designer, Enterprise Work Modeler, eProcurement Accelerator, eremote, Everything Works Better When Everything Works Together, EWA, ExtendedAssist, Extended Systems, ExtendedView, Financial Fusion, Financial Fusion (and design), Financial Fusion Server, Formula One, Fusion Powered e-Finance, Fusion Powered Financial Destinations, Fusion Powered STP, Gateway Manager, GeoPoint, GlobalFIX, iAnywhere, iAnywhere Solutions, ImpactNow, Industry Warehouse Studio, InfoMaker, Information Anywhere, Information Everywhere, InformationConnect, InstaHelp, Intelligent Self-Care, InternetBuilder, iremote, iScript, Jaguar CTS, jConnect for JDBC, KnowledgeBase, Legion, Logical Memory Manager, lRLite, M2M Anywhere, Mach Desktop, Mail Anywhere Studio, Mainframe Connect, Maintenance Express, Manage Anywhere Studio, MAP, M-Business Anywhere, M-Business Channel, M-Business Network, M-Business Suite, MDI Access Server, MDI Database Gateway, media.splash, Message Anywhere Server, MetaWorks, MethodSet, mFolio, Mirror Activator, ML Query, MobiCATS, MobileQ, MySupport, Net-Gateway, Net-Library, New Era of Networks, Next Generation Learning, Next Generation Learning Studio, O DEVICE, OASIS, OASIS logo, ObjectConnect, ObjectCycle, OmniConnect, OmniQ, OmniSQL Access Module, OmniSQL Toolkit, OpenBridge, Open Biz, Open Business Interchange, Open Client, Open ClientConnect, Open Client/Server, Open Client/Server Interfaces, Open Gateway, Open Server, Open ServerConnect, Open Solutions, Optima++, Partnerships that Work, PB-Gen, PC APT Execute, PC DB-Net, PC Net Library, Pharma Anywhere, PhysicalArchitect, Pocket PowerBuilder, PocketBuilder, Power++, Power Through Knowledge, power.stop, PowerAMC, PowerBuilder, PowerBuilder Foundation Class Library, PowerDesigner, PowerDimensions, PowerDynamo, Powering the New Economy, PowerScript, PowerSite, PowerSocket, Powersoft, PowerStage, PowerStudio, PowerTips, Powersoft Portfolio, Powersoft Professional, PowerWare Desktop, PowerWare Enterprise, ProcessAnalyst, Pylon, Pylon Anywhere, Pylon Application Server, Pylon Conduit, Pylon PIM Server, Pylon Pro, QAnywhere, Rapport, Relational Beans, RemoteWare, RepConnector, Report Workbench, Report-Execute, Replication Agent, Replication Driver, Replication Server, Replication Server Manager, Replication Toolkit, Resource Manager, RFID Anywhere, RW-DisplayLib, RW-Library, SAFE, SAFE/PRO, Sales Anywhere, Search Anywhere, SDF, Search Anywhere, Secure SQL Server, Secure SQL Toolset, Security Guardian, ShareLink, ShareSpool, SKILS, smart.partners, smart.parts, smart.script, SOA Anywhere Trademark, SQL Advantage, SQL Anywhere, SQL Anywhere Studio, SQL Code Checker, SQL Debug, SQL Edit, SQL Edit/TPU, SQL Everywhere, SQL Modeler, SQL Remote, SQL Server, SQL Server Manager, SQL SMART, SQL Toolset, SQL Server/CFT, SQL Server/DBM, SQL Server SNMP SubAgent, SQL Station, SQLJ, Stage III Engineering, Startup.Com, STEP, SupportNow, S.W.I.F.T. Message Format Libraries, Sybase Central, Sybase Client/Server Interfaces, Sybase Development Framework, Sybase Financial Server, Sybase Gateways, Sybase Learning Connection, Sybase MPP, Sybase SQL Desktop, Sybase SQL Lifecycle, Sybase SQL Workgroup, Sybase Synergy Program, Sybase Virtual Server Architecture, Sybase User Workbench, SybaseWare, Syber Financial, SyberAssist, SybFlex, SybMD, SyBooks, System 10, System 11, System XI (logo), SystemTools, Tabular Data Stream, The Enterprise Client/Server Company, The Extensible Software Platform, The Future Is Wide Open, The Learning Connection, The Model For Client/Server Solutions, The Online Information Center, The Power of One, TotalFix, TradeForce, Transact-SQL, Translation Toolkit, Turning Imagination Into Reality, UltraLite, UltraLite.NET, UNIBOM, Unilib, Uninull, Unisep, Unistring, URK Runtime Kit for UniCode, Viafone, Viewer, VisualWriter, VQL, WarehouseArchitect, Warehouse Control Center, Warehouse Studio, Warehouse WORKS, Watcom, Watcom SQL, Watcom SQL Server, Web Deployment Kit, Web.PB, Web.SQL, WebSights, WebViewer, WorkGroup SQL Server, XA-Library, XA-Server, XcelleNet, XP Server, XTNDAccess and XTNDConnect are trademarks of Sybase, Inc. or its subsidiaries. 07/06

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

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

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

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

Contents

About This Book	ix	
CHAPTER 1	Identifying the Problem	1
	Overview	1
	Tasks or events that can cause replication system problems ...	2
	Reduced replication system performance	3
	Types of problems	5
	Analyzing error logs	9
	rs_init	10
	Replication Server error log	11
	Diagnostic tools	11
	Troubleshooting configuration problems	12
	Troubleshooting subscription failures	13
	Materialization	14
	Dematerialization	15
	Troubleshooting replication failures	16
	Identify the data that is failing to replicate	18
	Verify that Replication Server threads are up	19
	Replication Server is down	21
	Verify that it is not a queue problem	21
	Verify that all RepAgents are up	21
	Whether only text, unitext, and image columns are not replicated	22
	Check system tables	22
	Activate traces and dump queues	22
	Verify that no route problems exist	23
	Troubleshooting manual recovery problems	23
CHAPTER 2	Analyzing Error Messages	25
	Overview	25
	What is not covered	26
	Replication Server errors	26
	Error message format	27

	Analyzing Replication Server error messages	28
	RepAgent error messages	31
	RepAgent error message format	31
	Analyzing RepAgent error messages.....	32
CHAPTER 3	Common Errors	35
	Finding the error description	35
	Error description format.....	36
	rs_init error message descriptions	36
	Could not find entry for Adaptive Server	36
	Unknown host machine name	37
	Replication Server error message descriptions	37
	21: Open Server call to routine srv_spawn failed.....	38
	1028: data server errors.....	38
	7035: Replication Server out of memory.....	45
	11061: RSSD problems.....	46
	13045: replication suspended because RSSD restarted.....	48
	15020: replication definition could not be found	50
	15040: no connection to replicate database.....	51
	15052: primary and replicate database cannot be the same ..	51
	28028: connect source not executed	51
	29024: no matching function strings found.....	52
	37022: primary subscribe permission required	53
	37023: create object permission required	53
	Replication Server informational and warning messages	54
	A cached row for system table was swapped out	54
	Detecting loss for database.....	54
	DSI detected rs_update_lastcommit not marked as replicated	57
	Stable storage use is above 75 percent.....	58
	RepAgent error message descriptions	59
	9202: nested stored procedures not allowed	59
	9210: network problems	60
	9215: log truncated (ASE 624).....	61
CHAPTER 4	Route Problems	63
	Overview	63
	Using the rs_helproute stored procedure	64
	Problems with creating routes.....	65
	Common problems	65
	Messages in the error log at the source Replication Server....	66
	Output from rs_helproute at the source Replication Server	67
	Output from rs_helproute at the destination server	69
	Problems with altering routes.....	69

	Problems with dropping routes.....	70
	Dropping routes.....	70
	Dropping routes using the with nowait option.....	72
CHAPTER 5	Subscription Problems.....	75
	Overview.....	75
	Using check subscription.....	76
	Materialization problems.....	79
	Common problems.....	80
	Atomic materialization.....	82
	Nonatomic materialization.....	87
	Bulk materialization.....	92
	Dematerialization problems.....	95
	with purge dematerialization.....	96
	Bulk dematerialization.....	99
CHAPTER 6	Replication Server Interface Problems.....	103
	Overview.....	103
	Connection errors.....	104
	Incorrect RSI user login name or password.....	104
	Incorrect user permissions at replicate Replication Server ...	106
	Invalid RSI locator.....	106
CHAPTER 7	RepAgent Problems.....	107
	Overview.....	107
	Problems when starting the RepAgent.....	108
	Invalid login.....	108
	Invalid permissions.....	108
	Errors from the Replication Server.....	109
	Normalization errors.....	109
	Errors from the Adaptive Server.....	112
	Dropping primary objects.....	113
	Invalid truncation page.....	114
CHAPTER 8	Data Server Interface Problems.....	115
	Overview.....	115
	Listing databases and using admin who and admin who, dsi.....	116
	Listing databases controlled by a Replication Server.....	116
	Checking the DSI status for a database connection.....	117
	Troubleshooting procedure.....	118
	DSI down or suspended.....	119
	Connection failure to the database.....	120

	Data server errors	121
	Replication Server errors.....	123
	DSI active.....	124
	Incorrect duplicate transaction resolution.....	124
	Examining the exceptions log.....	125
CHAPTER 9	Adaptive Server Log Problems	127
	Overview	127
	Truncating an Adaptive Server log.....	128
	Verifying the state of the secondary truncation point	128
	Turning off the secondary truncation point in a database	129
	Setting the secondary truncation point.....	130
	Resetting the locator	130
	Log truncation problems.....	131
	Orphaned transactions	132
	Inbound queue requires more disk space	133
	Adaptive Server log corruption.....	133
CHAPTER 10	Replication Monitoring Services Problems.....	135
	Overview	135
	RMS tracing	136
	Common Problems	136
	Using the RMS with UAF version 2.0	137
	Using multiple UAF servers on the same computer	137
	Using RMS heartbeat with Replication Server 12.6	138
	Monitoring the ASA Replication Agent	139
APPENDIX A	Dumping Stable Queues	141
	Introduction	141
	Using trace	142
	Using sysadmin dump_queue.....	143
	Confirming suspected problems.....	146
	sysadmin dump_queue examples.....	147
	Interpreting the output of a dump queue	147
	Using sysadmin sqt_dump_queue	154
	Using sysadmin dump_file	156
	sysadmin purge commands	157
APPENDIX B	Language, Sort Order, and Character Set Issues.....	159
	Overview	159
	Message language.....	160
	Stored procedure messages	160

Other situations that produce mixed languages	160
Sort order.....	161
Sort order problems	161
Character sets	162
Index.....	163



About This Book

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

This book, *Replication Server Troubleshooting Guide*, describes Replication Server troubleshooting tasks, such as dumping stable queues, debugging failures with the Data Server Interface (DSI) and Replication Server Interface (RSI), and diagnosing and correcting problems with subscriptions. It also touches on the common Replication Monitoring Services (RMS) problems.

Note Use the information in this manual for system-level troubleshooting. For Replication Agent–specific troubleshooting information, see the Replication Agent™ documentation for your data server.

Audience

The *Replication Server Troubleshooting Guide* is for Replication Server system administrators and application developers.

How to use this book

This book contains the following chapters:

- Chapter 1, “Identifying the Problem” describes how to identify configuration, subscription, replication, and manual recovery problems.
- Chapter 2, “Analyzing Error Messages” describes how to analyze Replication Server and RepAgent error messages.
- Chapter 3, “Common Errors” describes the most common errors that can occur during configuration and operation of a replication system.
- Chapter 4, “Route Problems” explains how to troubleshoot routing problems.
- Chapter 5, “Subscription Problems” helps you resolve common problems with subscriptions.

-
- Chapter 6, “Replication Server Interface Problems” helps you resolve problems with the RSI.
 - Chapter 7, “RepAgent Problems” helps you troubleshoot problems with RepAgent.
 - Chapter 8, “Data Server Interface Problems” provides troubleshooting information for the DSI.
 - Chapter 9, “Adaptive Server Log Problems” describes how and when to truncate the Adaptive Server® transaction log and other log problems.

Note Since changing the name of Sybase® SQL Server® to Adaptive Server Enterprise, Sybase may use the names Adaptive Server and Adaptive Server Enterprise to refer collectively to all supported versions of Sybase SQL Server and Adaptive Server Enterprise. From this point forward, in this document, Adaptive Server Enterprise is referred to as Adaptive Server.

- Chapter 10, “Replication Monitoring Services Problems” provides troubleshooting information on RMS.
- Appendix A, “Dumping Stable Queues” explains how to dump Replication Server stable queues.
- Appendix B, “Language, Sort Order, and Character Set Issues” discusses problems associated with running a replication system in an international environment.

Related documents

The Sybase Replication Server documentation set consists of the following:

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

A more recent version of the release bulletin may be available on the World Wide Web. To check for critical product or document information that was added after the release of the product CD, use the Sybase Technical Library.

- *Installation Guide* for your platform – describes installation and upgrade procedures for all Replication Server and related products.
- *What’s New in Replication Server?* – describes the new features in Replication Server version 15.0 and the system changes added to support those features.

- *Administration Guide* – contains an introduction to replication systems. This manual includes information and guidelines for creating and managing a replication system, setting up security, recovering from system failures, and improving performance.
- *Configuration Guide* for your platform – describes configuration procedures for all Replication Server and related products, and explains how to use the `rs_init` configuration utility.
- *Design Guide* – contains information about designing a replication system and integrating heterogeneous data servers into a replication system.
- *Getting Started with Replication Server* – provides step-by-step instructions for installing and setting up a simple replication system.
- *Heterogeneous Replication Guide* – 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.
- *System Tables Diagram* – illustrates system tables and their entity relationships in a poster format. Available only in print version.
- *Troubleshooting Guide* (this book) – contains information to aid in diagnosing and correcting problems in the replication system.
- Replication Manager plug-in help, which contains information about using Sybase Central™ to manage Replication Server.

Other sources of information

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

- The Getting Started CD contains release bulletins and installation guides in PDF format, and may also contain other documents or updated information not included on the SyBooks CD. It is included with your software. 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.
- The SyBooks CD contains product manuals and is included with your software. The Eclipse-based SyBooks browser allows you to access the manuals in an easy-to-use, HTML-based format.

Some documentation may be provided in PDF format, which you can access through the PDF directory on the SyBooks CD. To read or print the PDF files, you need Adobe Acrobat Reader.

Refer to the *SyBooks Installation Guide* on the Getting Started CD, or the *README.txt* file on the SyBooks CD for instructions on installing and starting SyBooks.

- The Sybase Product Manuals Web site is an online version of the SyBooks CD that you can access using a standard Web browser. 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

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

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

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

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

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

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

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

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

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

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

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

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

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

`rs_default_function_class` is a default function-string class.

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

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

Obligatory choices

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

Optional choices

- One item in square brackets – choose it or omit it.
`[anchovies]`

- Square brackets and vertical bars – choose none or only one.

```
[beans | rice | sweet_potatoes]
```

- Square brackets and commas – choose none, one, or more options. If you choose more than one, separate your choices with commas.

```
[extra_cheese, avocados, sour_cream]
```

Repeating elements

An ellipsis (...) means that you may repeat the last unit as many times as you need. For the alter function replication definition command, for example, you can list one or more parameters and their datatypes for either the add clause or the add searchable parameters clause:

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

RCL command formatting

RCL commands are similar to Transact-SQL® commands. The following sections present the formatting rules.

Command format and command batches

- You can break a line anywhere except in the middle of a keyword or identifier. You can continue a character string on the next line by typing a backslash (\) at the end of the line.
- Extra space characters on a line are ignored, except after a backslash. Do not enter any spaces after a backslash.
- You can enter more than one command in a batch, unless otherwise noted.
- RCL commands are not transactional. Replication Server executes each command in a batch without regard for the completion status of other commands in the batch. Syntax errors in a command prevent Replication Server from parsing subsequent commands in a batch.

Case sensitivity

- Keywords in RCL commands are not case-sensitive. You can enter them with any combination of uppercase or lowercase letters.
- Identifiers and character data may be case-sensitive, depending on the sort order that is in effect.
 - If you are using a case-sensitive sort order, such as “binary,” you must enter identifiers and character data with the correct combination of uppercase and lowercase letters.

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

Identifiers

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

- Identifiers can be 1 – 255 bytes long (equivalent to 1 – 255 single-byte characters) and must begin with a letter, the @ sign, or the _ character. See “Identifiers,” in Chapter 2, “Topics” of the *Replication Server Reference Manual* for a list of identifiers that have been extended to 255 bytes.
- Replication Server function parameters are the only identifiers that can begin with the @ character. Function parameter names can include 255 characters *after* the @ character.
- After the first character, identifiers can include letters, digits, and the #, \$, or _ characters. Spaces are not allowed.

Parameters in function strings

- Parameters in function strings have the same rules as identifiers, except:
 - They are enclosed in question marks (?), allowing Replication Server to locate them in the function string. Use two consecutive question marks (??) to represent a literal question mark in a function string.
 - The exclamation point (!) introduces a parameter modifier that indicates the source of the data that will be substituted for a parameter at runtime. Refer to *Replication Server Reference Manual* for a complete list of modifiers.

Datatype support

Replication Server supports all Adaptive Server datatypes.

User-defined datatypes are not supported. The timestamp, double precision, nchar, and nvarchar datatypes are indirectly supported by mapping them to other datatypes. Columns using the timestamp datatype are mapped to varbinary(8).

For more information about the supported datatypes, including how to format them, see *Replication Server Reference Manual*.

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

Icons

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

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

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.

Identifying the Problem

This chapter describes the types of problems that can occur in Replication Server, procedures you can use to identify your specific Replication Server problem, and the diagnostic tools you can use to solve the problem.

Name	Page
Overview	1
Analyzing error logs	9
Diagnostic tools	11
Troubleshooting configuration problems	12
Troubleshooting subscription failures	13
Troubleshooting replication failures	16
Troubleshooting manual recovery problems	23

Overview

A correctly configured Replication Server system is designed to be fault-tolerant. However, in the event of a serious failure, it might be necessary to manually intervene and fix the problem. This guide will help you locate, identify, and fix the cause of the problem.

You can solve most Replication Server problems by following the procedures in this guide. The key to finding the cause of a replication system failure is to eliminate possible causes. Use the following troubleshooting methods to narrow down the possible causes:

- Identify recent tasks or events that might have adversely affected the replication system. You or another coworker could have performed a task that caused a replication failure or reduced the performance of the replication system. Many things can happen when a number of people are working on the same replication system.

An event such as a temporary network outage may also have caused a replication system problem.

- Analyze error logs.

- Determine the general problem category (configuration, subscription, replication, recovery).
- Use diagnostic tools, such as Replication Server programs or stored procedures, or isql, to analyze the replication system.

If you see an error message displayed in a Replication Server error log, you can identify the problem by reading the error log and referring to Chapter 2, “Analyzing Error Messages” to learn how to read error log files, Chapter 3, “Common Errors” to see if the error is a common one, or to one of the other chapters that is relevant to the symptoms of your problem.

If you do not see an error message, you must use diagnostic tools to further analyze the replication system.

The Replication Manager(RM) and the Embedded Replication Server System Database(ERSSD) problems are not covered in this guide. RM uses the Sybase Central message logging feature to provide a log of all commands sent by the RM to any server. It also has a view queue data feature that helps you to troubleshoot transactions in a queue. See the online help for the Replication Manager plug-in and the *Replication Server Administration Guide Volume 1* for more information on how to use these features. See also the *Replication Server Administration Guide Volume 1* for more information on ERSSD recovery procedure.

This guide may be able to help you to identify hardware, network, and operating system problems, but solving these problems is beyond the scope of this book. Any time a server or a network connection is down, you should also check for hardware, network, or operating system problems.

On Windows, you can usually see a hardware or operating system problem when stack traces randomly or frequently occur at the same time you get errors in the Replication Server error log.

Check the operating system error log for errors that indicate hardware or operating system problems. Such failures might only partially resolve the effects on the replication system. You may still need to resynchronize data between the primary and destination databases.

Tasks or events that can cause replication system problems

Before attempting to categorize your problem, see if you have performed any of the following tasks or if any of these events have occurred. These tasks or events could have caused a replication system problem:

- Binaries changed. You might have changed binaries and have made the Replication Server, RepAgent, or Adaptive Server incompatible with each other. Incompatible binaries can produce Open Server™ and Open Client™ error messages. Check the latest release bulletins for version compatibility between replication system components.
- Network went down and restarted. See “Verify that Replication Server threads are up” on page 19.
- Rebuilt queues. See “Troubleshooting manual recovery problems” on page 23.
- Recovered primary database. See “Troubleshooting manual recovery problems” on page 23.
- Added, altered, or deleted routes. The route operation might not be complete. See Chapter 4, “Route Problems”.
- Added, altered, or deleted subscriptions. The subscription operation might not have completed. See Chapter 5, “Subscription Problems”.
- Added, altered, or deleted Replication Server, RepAgent, primary or replicate database, data server, or table. You might have changed the replication system incorrectly. See “Troubleshooting configuration problems” on page 12.

Reduced replication system performance

Reduced performance usually occurs when the operating system or machines are overloaded as a result of increased demands on the system. Reduced performance can result from adding applications or Replication Server components, executing very large transactions, or even upgrading the operating system.

Performance problems can be critical or non-critical. You can eliminate non-critical performance problems by optimizing your replication system. For non-critical performance problems, see the *Replication Server Administration Guide Volume 2*.

If critical performance problems are left unresolved long enough, performance degradation can lead to fatal problems, such as full stable queues (see “Poor performance and stable queue size increasing” on page 4), in which replication stops. A replication, materialization, or dematerialization failure can be caused by a critical performance problem.

Critical performance problems can stem from:

- Introducing new components, such as Adaptive Servers, databases, Replication Servers, RepAgents, or Replication Agents may cause resource contention and overload any component.
- Changing the operating system—such as upgrading the operating system, applying patches, changing kernel parameters, or rebuilding the kernel—may adversely impact your replication system, memory allocation, and resources.
- Adding applications to the replication system may impact memory requirements and use resources.
- Replicating a very large database may produce a very high latency. Large transaction or an open transaction is also a possible cause.

Poor performance and stable queue size increasing

Sometimes poor Replication Server performance is caused by the Stable Queue Transaction (SQT) cache size being too small. If the SQT cache is not big enough to hold all open transactions, transactions that cannot fit completely into the cache are processed one command at a time.

The inbound stable queues grow (more than 1MB) because Replication Server cannot process the transactions fast enough.

To solve this problem:

- 1 Make sure that the problem is not caused by orphaned transactions. For more information about orphaned transactions, see “Log truncation problems” on page 131.
- 2 Find the optimal size for the `sqt_max_cache_size` parameter for your application. See the *Replication Server Administration Guide Volume 2* for more information on sizing the SQT cache.
- 3 If you need to adjust the `sqt_max_cache_size` parameter:
 - a Suspend the connection to the data server.
 - b Log in to Replication Server and run this command:

```
> configure replication server
    set sqt_max_cache_size to 'new_value'
> go
```
 - c To activate the new values, resume the connection. If you want to dump the SQT cache, use the `sysadmin sqt_dump_queue` command.

Types of problems

This section describes the types of problems that can occur with Replication Server. These types roughly correspond to the different stages of development of the replication system. A replication system consists of Replication Server components (Replication Servers, Replication Agents, data servers, routes, connections) connected together such that data is copied reliably from source tables to destination tables.

- Configuration problems – occur while you are setting up Replication Server such as adding Replication Servers, RepAgents, and data servers.
- Subscription problems – occur while you are materializing or dematerializing subscriptions.
- Replication problems – occur when updates to a primary database are being copied to a replicate database.
- Manual recovery problems – occur when you are recovering a failed partition, truncated primary database log, primary database failure, or Replication Server System Database (RSSD).

Configuration problems

Configuration errors occur when you are running `rs_init`, the Replication Server program used to add and configure Replication Servers, RepAgents, and data servers to the replication system.

Usually, symptoms and problems are identified by error messages in the `rs_init` log file. After using the procedure in Chapter 2, “Analyzing Error Messages”, you may be able to resolve the problem by looking for the solution in Chapter 3, “Common Errors”.

Some configuration problems cause subscription materialization failures, and their symptoms do not appear until you attempt to materialize a subscription.

For more information about Replication Server configuration problems, see “Troubleshooting configuration problems” on page 12.

Subscription problems

The replication process begins with subscription *materialization*, which is the process by which data is first copied to the destination database. When you no longer want a subscription replicated to a destination database, you dematerialize the subscription at the destination database. *Dematerialization* is the process by which data is deleted from the destination database. Subscription failures occur when materialization or dematerialization fails.

Usually, the person who is conducting the materialization or dematerialization monitors the operation and reports any problems.

Subscription problem symptoms are easily identified and include:

- Materialization failure – no data in the subscription’s replicate table at the destination database, the status for subscriptions at the primary and replicate Replication Servers is not Valid, or materialization has been taking longer than is reasonable.
- Dematerialization failure – data still exists in the subscription’s replicate table at the destination database, the status for subscriptions at the primary and replicate Replication Servers is not Invalid, or dematerialization has been taking longer than is reasonable.

Some subscription problem symptoms are displayed as error messages in the Replication Server error log. You might also need to use the diagnostic tools to identify subscription problem symptoms. For more information about subscription problems, see “Troubleshooting subscription failures” on page 13.

Using incorrect variables in function strings

If you are having problems with customized function strings, which can be used to replicate changes, you might have used incorrect variables. Here are the restrictions:

- Pre-11.0.2 release Replication Server function strings can use system-defined variables. Function-string class scope functions cannot use the defaults.
- Only function strings for `rs_insert` and `rs_update` can use new column values.
- Only function strings for `rs_delete` and `rs_update` can use old column values.
- Only function strings for `rs_select` and `rs_select_with_lock` can have input templates. Only they can use user-defined variables.

- Only function strings for user-defined functions can use parameter values of functions. The parameter value of a function consists of the parameters passed to a replicated stored procedure.

Replication problems

Replication consists of copying data operations, such as updating or deleting data, from a primary database to the destination database. Replication begins after a subscription has been successfully materialized. Replication problems occur when data is changed at a primary database but not at the destination database.

You can find that a replication problem exists by observing that data is not replicating to a destination database or by hearing from someone that their application is retrieving incorrect data from a destination database.

If you are monitoring the replication system, you might directly observe that data is not replicating to a destination database. Use `rs_subcmp` to determine which subscription is not being replicated.

You should consider that a replication problem may exist if someone reports that their client application retrieved incorrect data from a destination database. Compare the primary and destination tables; if they are the same, then data is being replicated correctly. There is probably a problem with the client application that is causing incorrect data to be displayed in the client application. If data is not the same at the primary and destination databases, replication is failing, and you must troubleshoot the replication system.

Some symptoms of a replication problem directly identify the cause; other symptoms require more investigation to find the underlying cause. These symptoms can be organized into the following types in order of the most common first:

- Data Server Interface (DSI) thread is down.
- Threads other than DSI are down.

You use `admin who_is_down` to display information about threads that are down.

- DIST thread
- RepAgent User thread
- RSI thread
- RSI User thread

- RS User thread
- SQM thread
- SQT thread
- Major replication system component is down.
Use `isql` to check to see if a server is down by logging in to each server.
- RepAgent
- Replication Server
- Data server
- A “detecting loss” message, which means that data replication messages were lost after queues were rebuilt. This message is shown in the Replication Server error log or in the `rs_oqid` system table.
- Inbound or outbound stable queues are growing larger.
Use `admin who`, `sqm` and `sysadmin dump_queue` to display information about inbound and outbound stable queues.
- Number of duplicate transactions is increasing.
Use `admin who`, `sqt` and `sysadmin dump_queue` to display information about inbound and outbound stable queues.
- Transactions remain open for longer than is reasonable. These transactions might be orphans or a very long transaction. Orphans are transactions that do not have an ending commit or rollback statement.
Use `admin who`, `sqt` and `sysadmin dump_queue` to display information about inbound and outbound stable queues.
- Primary and destination Replication Servers do not have the same locator.
Use `isql` to login to the RSSD and view the `rs_locator` system table.
- Replication is successful for other subscriptions on different data servers with connections to the same destination Replication Server.
Use `rs_subcmp` to compare a subscription’s tables in the primary and replicate databases to make sure the tables are the same.
- Replication is successful for other subscriptions in the same or different tables on the same data server while replication for a particular subscription is failing.

Use `rs_subcmp` to compare a subscription's tables in the primary and replicate databases to make sure the tables are the same.

Some symptoms are displayed as error messages in the Replication Server and Adaptive Server error logs. Use the diagnostic tools to identify replication problem symptoms. For more information about replication problems, see “Troubleshooting replication failures” on page 16.

Manual recovery problems

While Replication Server is designed to tolerate most failures and to recover from them automatically, some failures require you to intervene by manually executing recovery tasks. Sometimes after you have completed a recovery task, you run into replication problems or errors show up in the Replication Server error log. Although the *Replication Server Administration Guide Volume 1 and Volume 2* covers most recovery situations and you can refer to that manual to see if you missed a step in a recovery task, this guide covers the most common problems that you might run into after you think that you have successfully completed a recovery task.

For information about some common manual recovery problems, see “Troubleshooting manual recovery problems” on page 23.

Analyzing error logs

Always maintain open windows for all error logs, including:

- Replication Server error logs (including a window for Replication Server standard error output (*stderr*))
- Replication Agent logs (such as Replication Agent for DB2)
- Any data server error logs (such as Adaptive Server)
- Any gateway error logs (such as DirectConnect™ or OmniConnect™)

Usually an error message will be displayed in one of these logs. Before checking for failures throughout your replication system, check for error messages in the error logs.

❖ **Checking for error messages in the error logs**

- 1 Scan the Replication Agent, data server, and gateway error logs to see if you can immediately find an error message that describes the cause of the error. If you find an error, see the documentation for the Replication Agent, data server or gateway to solve the problem.
- 2 Check for error messages in the `rs_init`, Replication Server error logs following the instructions in Chapter 2, “How to Read Error Log Files.”

Note RepAgent error messages are displayed in the Adaptive Server error log.

- 3 Find the error message in Chapter 3, “Common Errors.”
- 4 Apply the corresponding repair procedure.
- 5 If you cannot find the error message in Chapter 1, “Identifying the Problem”, look in one of the other chapters that corresponds to the symptoms of your problem or in one of the following error message text files in the `$$SYBASE/$SYBASE_REP/doc` directory on UNIX platforms or the `%SYBASE%\%SYBASE_REP%\doc` directory on Windows. These text files contain the text of the error message, a brief description of the error, its cause, and, sometimes, how to fix it.

For Windows:

- `errmsg.rs` for Replication Server error messages
- `errmsg.sub` for `rs_subcmp` error messages

For UNIX:

- `error_messages_rs` for Replication Server error messages
- `error_messages_subcmp` for `rs_subcmp` error messages

rs_init

`rs_init` error messages are recorded in the `rs_init` error log. The `rs_init` error log is located in:

- The `$$SYBASE/$SYBASE_REP/init/logs` directory for UNIX
- The `%SYBASE%\%SYBASE_REP%\init\logs` directory for Windows.

The `rs_init` error log file name is determined by the following format:

`logmonthday.session#`

- *month* is a 2-digit integer that represents the current month.
- *day* is a 2-digit integer that represents the current day.
- *session#* is a 3-digit integer that represents the number of the session for that day. Each time a new *rs_init* session is started on the same day, the session number is increased by one.

An example of an *rs_init* error log file name is *log0108.001*.

- *month* is 01
- *day* is 08
- *session#* is 001

Replication Server error log

The default location for a Replication Server error log is in:

- The `$SYBASE/$SYBASE_REP/install` directory for UNIX
- The `%SYBASE%\%SYBASE_REP%\install` directory for Windows

The Replication Server error log file name has this format:

```
rs_name.log
```

rs_name is the Replication Server name.

Diagnostic tools

Diagnostic tools retrieve the status and statistics of Replication Server components which, depending on the type of problem, you can use to analyze the replication system. Check the troubleshooting section of the problem category for detailed information. This section summarizes the available diagnostic tools.

Use:

- `isql` to log in to a Replication Server or data server to see if servers are up. You can also use `isql` to execute SQL commands to see if data is the same in the primary and replicate databases, or if data has been materialized or dematerialized.

- `admin who_is_down` to find out which Replication Server threads are down.
- `admin who, sqm` to display information, such as the number of duplicate transactions or the size of stable queues, about stable queues at a Replication Server.
- `admin who, sqt` to display information, such as the number of open transactions, about stable queues at a Replication Server.
- `admin stats, md` to display information, such as the number of messages delivered, about messages delivered by a Replication Server.
- `sp_config_rep_agent` to display the current RepAgent configuration settings.
- `sp_help_rep_agent` to display static and dynamic information about a RepAgent thread.
- `sysadmin dump_queue` to dump stable queues and view them.
- `rs_helproute` to display the status of routes at a Replication Server.
- `rs_subcmp` to compare a subscription's tables in the primary and replicate databases to make sure the tables are the same.
- `check subscription` to display the status of subscriptions at a Replication Server.
- `rs_helppub` to display publications.
- `rs_helppubsub` to display publication subscriptions.
- `sp_setrepcol` to check the replication status of text, unitext, or image columns.

Troubleshooting configuration problems

Verify that a replication system is configured correctly by materializing subscription data. Some configuration problems do not appear until you attempt to materialize subscription data. The most common configuration problems that cause materialization failures are:

- Failure to log in to the primary Adaptive Server. The user who creates the subscription in the replicate Replication Server must have the same login name and password in the primary Adaptive Server and the primary Replication Server.
- Missing permissions in the primary database. The user who creates the subscription must be a user in the primary database and must have select permission in the primary table.
- Missing permissions in the replicate database. The maintenance user must have select, insert, update, and delete permissions on the tables in the replicate database.
- A Replication Server or Adaptive Server has stopped running. Try to log in to each server using isql. Restart any servers that are not running.

Other common configuration problems include:

- Host name resolution error.
- The Adaptive Server entry does not exist in the interfaces file.

See “rs_init error message descriptions” on page 36.

For more information about troubleshooting Replication Server configuration problems, see the *Replication Server Configuration Guide* for your platform.

Troubleshooting subscription failures

Subscription failures consist of materialization and dematerialization failures. This section describes procedures you can use to troubleshoot these types of problems.

Before troubleshooting a subscription problem, always verify that data has failed to materialize or dematerialize by:

- 1 Using isql to log in to the replicate database
- 2 Executing a select command that selects the materialized or dematerialized columns from the replicate table

Also make sure that all tasks required for subscription materialization have been completed. Subscription materialization can fail if you have not completed:

- Creating replication definitions or function replication definitions

- Marking tables or stored procedures for replication
- Creating connections to the destination databases
- Creating articles, if you are using them
- Creating and validating publications, if you are using them
- Marking text, unitext, or image columns for replication, if you are replicating text, unitext, and image columns
- Creating direct and indirect routes, if the destination database is connected to a Replication Server different from the primary database's Replication Server
- Creating logical connections, if you are using warm standby applications

Materialization

If a subscription has not materialized, follow the general procedure in this section. See Chapter 5, “Subscription Problems”, for specific instructions.

❖ Procedure if subscription has not materialized

- 1 If you are materializing a large amount of data, make sure the *num_threads* and *num_concurrent_subs* parameters are large enough.
- 2 Log in to the destination Replication Server and check subscription status using `check subscription`. Do the same for the primary Replication Server. `check subscription` returns information that diagnoses the problem. The potential problems that `check subscription` returns information about include:
 - Other subscriptions to the same replication definition and replicate database have not yet processed
 - No connection to the primary Replication Server because of an incorrect login
 - Primary Replication Server down or out of stable queues
 - Stable Queue Manager (SQM), Stable Queue Transaction interface (SQT), and Distributor (DIST) threads down
 - Primary data server down, incorrect login, out of stable queues, or rows selected with holdlock
 - RepAgent problem

- Route problem
 - Destination Replication Server— incorrect login or out of stable queues
 - Destination Replication Server DSI problem—use admin who, dsi or admin who, sqm to determine what the specific problem is
 - Incorrect user privileges on destination database
- 3 Use rs_helppub and rs_helppubsub to find the publications and articles that a subscription is using.
 - 4 If only some columns in the table are not being materialized:
 - Check replication status of text, unitext, and image columns.
 - Verify that the replication definition is correctly defined.
 - Verify that the publications and articles are correctly defined.
 - 5 Fix the problem.
 - 6 When you think you have solved the problem, run the replication system. If subscriptions are still not being materialized correctly:
 - Analyze errors (see “Analyzing error logs” on page 9), or
 - Complete any of this section’s procedures that you skipped.

Dematerialization

If a subscription has not dematerialized, follow the general procedure in this section. See Chapter 5, “Subscription Problems”, for specific instructions.

❖ Procedure if subscription has not dematerialized

- 1 Log in to the destination Replication Server and check subscription status using check subscription. Do the same for the primary Replication Server. check subscription returns information that diagnoses the problem. The potential problems include:
 - Other subscriptions to the same replication definition and replicate database have not yet processed
 - No connection to the primary Replication Server because of an incorrect login
 - Primary Replication Server down or out of stable queues

- Stable Queue Manager (SQM), Stable Queue Transaction interface (SQT), and Distributor (DIST) threads down
 - Primary data server down, incorrect login, out of stable queues, or rows selected with holdlock
 - RepAgent problem
 - Route problem
 - Destination Replication Server DSI problem
 - Incorrect user privileges on destination database
- 2 If some columns are not being dematerialized:
 - Check replication status of text, unitext, and image columns.
 - Verify that the replication definition is correctly defined.
 - 3 Fix the problem.
 - 4 When you think you have solved the problem, run the replication system. If subscriptions are still not being dematerialized correctly:
 - Analyze errors (see “Analyzing error logs” on page 9), or
 - Complete any of this section’s procedures that you skipped.

Troubleshooting replication failures

Replication failures may occur after subscriptions have successfully materialized. This section documents procedures you can use to isolate replication failures.

These procedures assume that the replication system has:

- Been installed and configured correctly,
- Successfully completed subscription materialization, and
- Correctly replicated data previously

If you are having installation and configuration problems, see the *Replication Server Configuration Guide* for your platform. If you are having subscription problems, see “Troubleshooting subscription failures” on page 13.

Before you troubleshoot a replication failure, verify that data is not replicating:

- 1 Use `isql` to log in to the primary and replicate databases.
- 2 Execute `select` commands that select the replicate columns from the replicate table and the columns to be replicated from the primary table.
- 3 Compare the data from both tables to see if it is the same.

Note If the subscription has many columns that are replicated, use `rs_subcmp` to compare data in the primary and replicate databases. See “Identify the data that is failing to replicate” on page 18.

After you have verified that data is not replicating, use the procedures and diagnostic tools in the rest of this section to investigate and identify the exact cause of your problem. These procedures can be executed separately and in any order; however, they are listed in the order in which they are most likely to solve the problem:

- “Identify the data that is failing to replicate” on page 18
- “Verify that Replication Server threads are up” on page 19
- “Replication Server is down” on page 21
- “Verify that it is not a queue problem” on page 21
- “Verify that all RepAgents are up” on page 21
- “Whether only text, unitext, and image columns are not replicated” on page 22
- “Check system tables” on page 22
- “Activate traces and dump queues” on page 22
- “Verify that no route problems exist” on page 23

When you think you have solved the problem, run the replication system. If replication is still not executing correctly:

- Analyze errors (see “Analyzing error logs” on page 9), or
- Complete any of this section’s procedures that you skipped.

Identify the data that is failing to replicate

Use this procedure to determine the specific subscription and columns that are failing to replicate. This procedure also checks to see if the primary and destination data servers and primary or destination Replication Server are running.

❖ Identifying data that is failing to replicate

- 1 To find out which data in a suspect subscription is failing to replicate, use `isql` to log in to the primary or destination Replication Server and run `rs_subcmp`.

`rs_subcmp` logs in to the primary and destination data servers and compares the subscription's data in the primary and destination tables.

`rs_subcmp` can compare tables at Adaptive Server data servers only. To compare tables at a non-Adaptive Server data server, you can use a program equivalent to `bcp out` on the non-Adaptive Server data server and `bcp out` on the Adaptive Server data servers, and compare the output using the UNIX `diff` command

- 2 If `rs_subcmp` successfully displays inconsistent rows, note the columns and rows that are not being replicated.
- 3 If no data exists for subscribed columns, then the subscription has not materialized. Go to “Materialization” on page 14.
- 4 If you cannot log in to a Replication Server, then that Replication Server is down. Go to “Replication Server is down” on page 21.
- 5 If `rs_subcmp` fails, then one or both of the data servers are down:
 - If the primary data server is down, the Adaptive Server log may be corrupt or full. The data server may also have an operating system or hardware error. See Chapter 9, “Adaptive Server Log Problems”, for more information about troubleshooting these problems.
 - If the destination data server is down, then the data server may have a DSI problem, or an operating system or hardware error. See Chapter 8, “Data Server Interface Problems”.
- 6 Use `rs_subcmp` to check if other subscriptions on the same data server are replicating:
 - If no other subscriptions are replicating, it is likely that a problem exists with that data server and not with a particular subscription.

- If all other subscriptions are replicating, then a problem may exist with that particular subscription.
- 7 Use `rs_subcmp` to check if other subscriptions on databases controlled by the same destination Replication Server are replicating. If replication is working for other databases controlled by the destination Replication Server, then the problem is a specific database, database connection, or RepAgent. Take the following actions:
 - Look for orphaned transactions in the primary Replication Server inbound queue for the database. See Chapter 9, “Adaptive Server Log Problems”.
 - Troubleshoot RepAgents. See Chapter 7, “RepAgent Problems”.
 - Troubleshoot database connections. See Chapter 8, “Data Server Interface Problems”.

Once you have identified the data that is failing to replicate, go to the next procedure.

Verify that Replication Server threads are up

Use `admin who_is_down` to determine if the primary and destination Replication Server threads are up. The procedure below verifies that the Replication Servers and the Replication Server threads are up.

- ❖ **Verifying that the Replication Server and Replication Server threads are up**
 - 1 Log in to the destination Replication Server and execute `admin who_is_down`. This command displays all the threads on this Replication Server that are down. If any threads are down, an error message should have been displayed in the Replication Server error log. Check the error log again. Also see the section that corresponds to the thread that is down:
 - DSI – see Chapter 8, “Data Server Interface Problems”. The problem could be duplicate keys or permissions failure.
 - DSI EXEC – see Chapter 8, “Data Server Interface Problems”.
 - RSI User – see Chapter 6, “Replication Server Interface Problems”.
 - RS User – see Chapter 5, “Subscription Problems”.
 - SQM – the SQM should not go down. Restart the Replication Server; you cannot resume the SQM thread.

- SQT – see “13045: replication suspended because RSSD restarted” on page 48
 - User – this should have no effect on replication.
- 2 Log in to the primary Replication Server and use `admin who_is_down` to display all the threads on this Replication Server that are down. See the section that corresponds to the thread that is down as follows:
 - DIST – See “7035: Replication Server out of memory” on page 45 and “13045: replication suspended because RSSD restarted” on page 48.
 - DSI – see Chapter 8, “Data Server Interface Problems”. The problem could be duplicate keys or permissions failure.
 - DSI EXEC – see Chapter 8, “Data Server Interface Problems”.
 - RepAgent User – see Chapter 7, “RepAgent Problems”.
 - RSI - see Chapter 6, “Replication Server Interface Problems”.
 - RSI User – see Chapter 6, “Replication Server Interface Problems”.
 - RS User – see Chapter 5, “Subscription Problems”.
 - SQM – the SQM should not go down. Restart the Replication Server; you cannot resume the SQM thread.
 - SQT – see “13045: replication suspended because RSSD restarted” on page 48
 - User – this should have no effect on replication.
 - 3 The *keepalive* value is set too low and the TCP connection was terminated and never restarted, if the following conditions exist:
 - The DSI is down,
 - The RepAgent is not connected to the Replication Server or Adaptive Server, and
 - The entire (or part of the) network went down and was restarted.
 - 4 If the DSI is up, check for data loss. Although data loss error messages show up in the Replication Server error log, these errors only show up once and may have occurred several days earlier. See Chapter 8, “Data Server Interface Problems”.
 - 5 If you cannot log in to a Replication Server, it means that Replication Server is down. See the next procedure.

Replication Server is down

If the primary, destination, or both Replication Servers are down, analyze the error messages in each Replication Server's error log.

If both Replication Servers are down, there might be different problems for the primary and replicate Replication Servers. If both Replication Servers have their RSSDs on the same Adaptive Server data server, then troubleshoot the Adaptive Server data server.

Verify that it is not a queue problem

❖ **To verify that it is not a queue problem:**

- 1 Run `admin who, sqm` to see if the duplicates count is rising.
- 2 Check outbound queues for stuck and open transactions.

See Chapter 8, "Data Server Interface Problems".

Verify that all RepAgents are up

❖ **Verifying that the RepAgent or the primary Adaptive Server data server is up**

- 1 To verify that the RepAgent is active, use `isql` to log in to the primary Adaptive Server data server and execute `sp_who`.
- 2 If the RepAgent is down:
 - The Adaptive Server log is corrupt.
 - The Adaptive Server log is full.
 - The RepAgent set an illegal truncation point.
 - The RepAgent used an incorrect primary Replication Server login.
 - `text`, `unitext`, or `image` columns have inconsistent replication status.
 - Incorrect replication system configuration.

See Chapter 7, "RepAgent Problems", for more information about solving RepAgent problems. See Chapter 9, "Adaptive Server Log Problems", for information about log problems.

Whether only *text*, *unitext*, and *image* columns are not replicated

If you run `rs_subcmp` and see that only *text*, *unitext*, and *image* columns at the destination table are not being replicated, then these columns may have inconsistent replication status. See “Replication status inconsistency of *text*, *unitext*, or *image* columns” on page 110 for more information about repairing this problem.

Check system tables

The procedure checks to see if there is any error information in the RSSD system tables.

❖ Checking system tables

- 1 See if there are any recoverable actions in the `rs_recovery` system table.
- 2 See if there are any “detecting losses” or “rejecting messages after loss detected” validation status messages in the `rs_oqid` system table. (You might have missed the message in the error log.)

Activate traces and dump queues

You specify trace flags when you run the diagnostic versions of the Replication Server (*repserver.diag*). By turning on traces and examining the output of `sysadmin dump_queue`, you can determine the component in which replication has stopped.

If replication stopped in one of the following components, see the corresponding chapter or section:

- Primary data server – Chapter 8, “Data Server Interface Problems”, and Chapter 9, “Adaptive Server Log Problems”.
- RepAgent – Chapter 7, “RepAgent Problems”.
- Outbound queue – Chapter 4, “Route Problems”, and Chapter 8, “Data Server Interface Problems”.
- DSI thread – Chapter 8, “Data Server Interface Problems”.
- Custom function-string class or customized function strings – Chapter 8, “Data Server Interface Problems”, or “Using trace” on page 142.

Verify that no route problems exist

This procedure checks for route problems. See Chapter 4, “Route Problems”, for more information about using `rs_helproute` and solving route problems.

❖ Verifying that no route problem exists

- 1 Make sure that no other databases are replicating through the same route as the non-replicating subscription. If other databases are replicating through that route, then problems within the primary Replication Server/database and replicate Replication Server/database are more likely. It is unusual for replication to have stopped for one database while replication continues through the same route for other databases. Run `admin stats, md` on the replicate Replication Server to see if “Messages Delivered” increases over time.
- 2 If no other databases are replicating through the same route as the non-replicating subscription, run `rs_helproute` at the primary or destination Replication Server in the RSSD to determine if there is a route problem.

Troubleshooting manual recovery problems

If you recently recovered a primary database and replication is failing, one of these problems may have occurred:

- A new generation ID for the primary database has not been set. See *Replication Server Administration Guide Volume 2*.
- If you loaded the primary and replicate databases from the same backup, make sure that the `rs_lastcommit` table for the replicate database has the correct entries. If the `origin_qid` in the `rs_lastcommit` table is incorrect (old or changed), the DSI may ignore certain transactions associated with that `origin_qid`. You should `bcp` out the `rs_lastcommit` table, load the data, and then `bcp` in the `rs_lastcommit` table. If DSI still ignores certain transactions, update the `origin_qid` value to “0” to force DSI to accept these transactions.
- If you used `dump` and `load` to synchronize the primary and replicate databases because of a failure, be sure to increase the generation number in the primary database. If you do not set the generation number correctly, replication from this database may stop because Replication Server perceives that it has already processed the messages.

Also see *Replication Server Administration Guide Volume 2*, which contains information for recovering primary databases.

Analyzing Error Messages

This chapter describes the format of Replication Server and RepAgent error messages, how to analyze these error messages, and provides example analyses of these error messages.

Name	Page
Overview	25
Replication Server errors	26
RepAgent error messages	31

Overview

Error messages provide important information for identifying the cause of replication system problems. When a Replication Server or RepAgent error occurs, an error message is recorded in an error log. Error messages are displayed in a standard format; once you learn this format, you will be able to read and understand all Replication Server and RepAgent error messages.

Replication Server error messages are recorded in text files called error logs or sent to the standard error output (*stderr*), which is usually a terminal window. In general, Replication Agent error messages are recorded in their own error logs, except for RepAgent error messages, which are recorded in the Adaptive Server data server error log. Data server error messages are recorded in the data server's error log.

Multiple error messages can be generated by a single problem, and can be generated from any or all Replication Server components (including Adaptive Server, Replication Server, and RepAgent) that are adversely affected by the problem.

Replication Server and RepAgent can print messages to the error log and to clients in several languages. See Appendix B, "Language, Sort Order, and Character Set Issues" for problems that can result from output in mixed languages.

What is not covered

This guide does not explain how to analyze the error logs from networks, gateways, non-Adaptive Server data servers, and non-Adaptive Server Replication Agents. See the documentation for these products for information about how to analyze their error logs.

Replication Server errors

Replication Server errors are recorded in the Replication Server error log. A Replication Server error log contains informational, warning, thread-terminated, fatal, and internal error messages. Messages are appended to the error log while Replication Server is operating; each time you restart Replication Server, error messages are appended to the end of the existing error log.

Informational messages report the current status of a component; for example, a process or command has completed or a component has been shut down. Warning, thread-terminated, fatal, and internal error messages are reported when a component abnormally terminates, a process or command cannot be completed, or a fatal internal error occurs in a component.

When Adaptive Server errors cause a Replication Server error, you find references to the Adaptive Server errors in the text of the Replication Server error message. In these cases, you must fix the Adaptive Server problem, which should eliminate the Replication Server error.

Replication Server error logs contain error messages generated during the execution of asynchronous commands, such as create subscription and create route. When you are executing asynchronous commands, pay special attention to the error logs for the Replication Servers affected by the asynchronous command.

Dedicate a terminal window to display the standard error output from a Replication Server. If the Replication Server error log is unavailable, error messages are sent to *stderr*. Error messages that are sent only to *stderr* are extremely serious, because the only action available to the Replication Server is to dump the stack and exit.

Error message format

The format of Replication Server error messages is:

```
s. date time severity_text error_num thread  
(thread_context) module (line_num) error_text
```

The variables in the error message format are defined as:

- *s* – indicates the severity or seriousness of the error.
- *date* – date.
- *time* – military time.
- *severity_text* – brief phrase that describes the severity.
- *error_num* – integer that uniquely identifies the error.
- *thread* – name of the Replication Server thread that received the error.
- *thread_context* – more specific information about the thread, such as the database and data server if it is a DSI thread.
- *module* – name of the Replication Server source file where the error was reported. This name is only for use by a Sybase Technical Support Engineer.
- *line_num* – line in the Replication Server source file where the error was reported. This number is only for use by a Sybase Technical Support Engineer.
- *error_text* – error message text. If a Replication Server error was caused by an Adaptive Server error, the Adaptive Server error is displayed as part of the Replication Server error message text.

Error messages with a severity of “I” (informational) contain only the date and time the error occurred, and the full text of the message.

Error messages are displayed in the language specified in the *RS_language* configuration parameter.

Severity

Each error message begins with a letter that indicates the severity of the error. Table 2-1 lists the severities.

Table 2-1: Replication Server error severities

Severity Code	Description
I	An informational message.
W	A warning about a condition that has not yet caused an error, but may require attention, for example, running out of a resource.
E	An error that does not prevent further processing, such as a site that is unavailable.
H	A Replication Server thread has died, for example, a lost network connection.
F	Fatal. A serious error caused Replication Server to exit. For example, if you start the Replication Server using an incorrect configuration.
N	Internal error, caused by anomalies in the Replication Server software. Report these errors to Sybase Technical Support.

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

Error numbers

Error numbers from Replication Server are constructed using the following formula:

$$\text{module_number} * 1000 + \text{error_condition}$$

A *module_number* is assigned to each source code module in the server and the *error_condition* is a numbered error condition within the source code module.

Analyzing Replication Server error messages

To analyze a Replication Server error log, you must understand how multiple error messages are related. Multiple error messages can be generated by the same problem and usually have identical or very close dates and times. Error messages that occur close together constitute an *error block*.

Replication Server error messages may contain references to Adaptive Server errors. By examining the causes of the Adaptive Server error, you will be able to solve the problem.

The Replication Server error log may also contain Open Client/Server™ error messages, since Replication Server is an Open Server program with Open Client components. Any Open Server errors in the error log constitute internal errors.

❖ **Reading a Replication Server error log**

- 1 Look for the most recent Replication Server error by scanning from the end to the beginning of the error log. As Replication Server errors occur, Replication Server appends the error messages to the end of the error log. The most recent Replication Server error marks the end of the error block.

Find the first Replication Server error message in the error block.

Note If you see an Open Server/Client error such as “no more threads,” fix that error first. An Open Server error “no more threads” message means that you have run out of an Open Server resource (and Replication Server might have terminated)

- 2 Find and diagnose the last non–Replication Server error message that occurred after the first Replication Server error. If there are no non–Replication Server error messages after the first Replication Server error message, investigate the causes of the first Replication Server error message.

Note In diagnosing non–Replication Server errors, you may need to use other troubleshooting or error message guides—such as those for Adaptive Server or Open Client (particularly with network communication or other client problems).

- 3 In the error block, find any Replication Server error messages that tell you if a Replication Server component or thread has terminated. You must restart the component or thread that has terminated. Usually, subsequent Replication Server errors in the same error block were caused by the first Replication Server error.

Example

Examine the following error message block in the Replication Server log:

```
H. 2006/06/15 20:52:28. THREAD FATAL ERROR #5049 DSI
EXEC(104(3) WESTERND.S.westDB) - dsimint.c(3252) The
DSI thread for database 'WESTERND.S.westDB' is being
```

shutdown. DSI received data server error #102 which is mapped to STOP_REPLICATION. See logged data server errors for more information. The data server error was caused by output command #1 mapped from input command #2 of the failed transaction.

I. 2006/06/15 20:52:28. The DSI thread for database 'WESTERNDS.westDB' is shutdown.

Table 2-2 breaks this error message into its components.

Table 2-2: Replication Server error message example description

Error Message Text	Explanation
<p>H. 2006/06/15 20:52:28. THREAD FATAL ERROR #5049 DSI EXEC(104(3) WESTERNDS.westDB) - dsimint.c(3252)</p>	<p>This is the first error. The H identifies the error as a thread termination error. This message shows that the DSI thread terminated.</p> <p>The error message includes the dataserver.database name where the error occurred, the internal Replication Server module (<i>dsimint.c</i>), and the line of code (3252) where the error occurred.</p>
<p>The DSI received data server error # 102 which is mapped to STOP_REPLICATION. See the logged data server errors for more information.</p>	<p>This Adaptive Server error message identifies the cause of the problem.</p> <p>Find a description for Adaptive Server Error 102 by:</p> <ul style="list-style-type: none"> • Finding the error in the Adaptive Server error log (the error would occur at approximately the same time as it occurred in the Replication Server error log), • Performing a select * from sysmessages where error = 102 in the master database, or • Looking up the number in the <i>Adaptive Server Enterprise Troubleshooting and Error Messages Guide</i>. <p>The <i>Adaptive Server Enterprise Troubleshooting and Error Messages Guide</i> describes Error 102 as invalid syntax near %s for a Transact-SQL statement. It also gives suggestions about when this error occurs (a keyword is spelled incorrectly, a keyword or parameter is missing, or the order of the keyword is incorrect). Look for additional information in the data server error log, which displays the character string for the %s field in the error message.</p>
<p>The data server error was caused by RS output command #1 mapped from input command #1 of the failed transaction.</p>	<p>This text describes the command position in a grouped transaction sent by the Replication Server.</p>
<p>I. 2006/06/15 20:52:28. The DSI thread for database 'WESTERNDS.westDB' is shutdown.</p>	<p>This last error message is informational (I) and caused by the problem specified in a previous block. After fixing the Adaptive Server problem, you must restart the DSI thread for the specified database.</p>

RepAgent error messages

RepAgent records RepAgent errors such as “Could not spawn Rep Agent thread” and all error messages returned by Replication Server (including CT-Library error messages) in the Adaptive Server error log.

RepAgent error message format

RepAgent error messages are recorded in the Adaptive Server error message format and are identified by the string “RepAgent(*dbid*),” which appears in the first line of the message. *dbid* is the database identification number of the RepAgent that logged the error.

Errors that you can retry are logged only once in the Adaptive Server error log.

The RepAgent error message format is:

```
date time RepAgent (dbid): error_number, severity,
state, error_text
```

The RepAgent error format includes the following information:

- *date* – date that the error occurred.
- *time* – time that the error occurred.
- *dbid* – Adaptive Server identification number of the database that RepAgent is using. You can find this database ID by executing this command:


```
select x = db_id()
```
- *error_number* – RepAgent error message numbers range from 9200 to 9299.
- *severity* – severity can be:
 - EX_INFO: informational error message
 - EX_USER: user error
 - EX_RESOURCE: resource error in which an operating system resource or Replication Server resource is not available
 - EX_CMDFATAL: fatal error in which RepAgent cannot continue processing a transaction
- *state* – is for internal use only.

- *error_text* – description of the cause of the error.

Example

Here is an example of a RepAgent error message:

```
00:00000:00036:2006/01/13 13:08:16.39 server Error:
9209, Severity: 20, State: 0
00:00000:00036:2006/06/23 13:08:16.39 server
RepAgent(6): Missing datarow in TEXT/IMAGE insert log
record. Transaction log may be corrupt. Please contact
SYBASE Technical Support. (current marker = (107634,
10)).
```

Table 2-3 breaks this error message into its components.

Table 2-3: RepAgent error message example description

Error Message Text	Explanation
2006/01/13	Date
13:08:16.39	Time
(6)	Database ID
9209	Error number
20	Severity
0	State
Missing data row in TEXT/IMAGE insert log record. Transaction log may be corrupt. Please contact SYBASE Technical Support. (current marker = (%d, %d)).	Error message text

For more information about the Adaptive Server error message format, see the *Adaptive Server Enterprise System Administration Guide*.

Analyzing RepAgent error messages

Analyzing RepAgent errors is much easier than analyzing Replication Server errors.

Example

The following error message indicates that the RepAgent thread attempted to connect to a database that already has a RepAgent thread running.

```
06/12/2006 16:37:58.34 PM RepAgent (10) 9220, EX_User,  
1: The Rep Agent Thread context for database westernDB  
has already been reserved by another session.
```

Only one RepAgent thread can connect to a database at any one time. Because the RepAgent abnormally terminated, the RepAgent thread process still exists although it is in sleep mode. The RepAgent thread wakes after the time specified in the *scan_retry* parameter elapses. Since the RepAgent has already terminated, the RepAgent thread process terminates when it wakes.

Common Errors

This chapter describes some of the most common Replication Server errors and solutions.

Name	Page
Finding the error description	35
rs_init error message descriptions	36
Replication Server error message descriptions	37
Replication Server informational and warning messages	54
RepAgent error message descriptions	59

Note This chapter does not describe all errors. If you do not find the error you are looking for in this chapter, refer to Chapter 1, “Identifying the Problem”, for more troubleshooting information or refer to the chapter that corresponds to your error description.

Finding the error description

To find the error description in this chapter, go to the appropriate section and look for the error by:

- The error number of a Replication Server error, rs_init, or Replication Agent, or
- The text of the rs_init error, or Replication Server informational or warning message.

If there are multiple errors in an error log, use the procedure in Chapter 2, “Analyzing Error Messages”, to determine which error to look for in this chapter.

Error description format

Each error description includes these sections:

- Symptom – includes the text of any error messages that are displayed in an error log and error conditions, such as poor performance, replication failure, connection failure, and abnormal component termination.
- Explanation – describes the error messages and probable causes of the errors.
- Solution – describes procedures, workarounds, upgrades, and EBF information.

rs_init error message descriptions

Because rs_init error messages do not have error numbers, they are arranged alphabetically.

Could not find entry for Adaptive Server

Symptom

Error messages stating that the Adaptive Server entry does not exist in the *interfaces* file are displayed in the rs_init error log.

```
2006/06/13 10:20:48 There is no entry for server
'westdss' in the interfaces file. The server must have
an existing entry in the interfaces file.
2006/06/13 10:20:48 Attribute 'rs_ds_name' could not be
set because 'westdss' is an invalid value.
2006/06/13 10:20:48 Cannot execute configuration
because validation of input values failed.
2006/06/13 10:20:48 Exiting.
```

Explanation

When you execute rs_init with a resource file (rs_init -r), rs_init looks in the *interfaces* file for the Adaptive Server name that matches the one in the resource file.

The errors listed above occur because the Adaptive Server entry in the *interfaces* file does not match the resource file's Adaptive Server entry due to unprintable characters in the resource file.

Solution Check for unprintable characters (such as control characters) in the Adaptive Server name entry in the resource file. The Adaptive Server name entry is specified as the *rs.rs_ds_name* parameter.

Unknown host machine name

Symptom The following messages are displayed in the *rs_init* error log.

```
2006/06/13 10:34:17 Running task: check the SQL Server.
2006/06/13 10:34:18 Unable to connect to SQL Server
'ost_agate_9'. Please make sure that the SQL Server is
running, and the SA password is correct.
2006/06/13 10:34:18 Task failed: check the SQL Server.
Terminating configuration.
2006/06/13 10:34:18 Configuration failed.
2006/06/13 10:34:18 Exiting.
```

Explanation *rs_init* cannot find the RSSD machine's host name. Sometimes, directory services do not recognize the machine's host name and, instead, may recognize a machine only by its Internet Protocol (IP) address.

Solution Add an additional query line for the machine to the *interfaces* file and specify the host machine's IP address instead of its name. For example, for the *bss.bsslt.ch.corma.com* host name substitute its IP address, 2.41.100.35, as follows:

```
query tcp ether bss.bsslt.ch.corma.com 31440
query tcp ether 2.41.100.35 31440.
```

If you run *rs_init* again, the following messages may be displayed in the *rs_init* error log, but you can ignore them:

```
Could not obtain Hostname for Internet address '2.41.100.35'.
Could not obtain Hostname for Internet address '2.41.100.41'.
Could not obtain Hostname for Internet address '2.41.100.35'.
Could not obtain Hostname for Internet address '2.41.100.41'. Running task
to check the RSSD Adaptive Server.
```

Replication Server error message descriptions

These error messages, which are recorded in the Replication Server error log, are arranged numerically in ascending order.

21: Open Server call to routine *srv_spawn* failed.

Symptom	After a new connection is started the following message is displayed: <pre>ERROR #21 DSI(126 U10PDTE.u10pdte) - (1426) Open server call to routine 'srv_spawn' failed. Open Server error: Error: 16115, State: 0, Severity 10 -- 'Could not start thread'.</pre>
Explanation	Replication Server ran out of Open Server threads. <i>srv_spawn</i> allocates these threads. The maximum amount of Open Server threads that a Replication Server can use is equal to the value specified by the <i>num_threads</i> parameter.
Solution	Increase the number of Open Server threads that the Replication Server can use by increasing the value of the <i>num_threads</i> parameter. For example: <pre>configure replication server set num_threads to '70'</pre>

1028: data server errors

Symptom	Error 1028 is displayed in the Replication Server error log. An Adaptive Server error is reported in the text of error 1028. <pre>E. 2005/06/12 15:25:44. ERROR #1028 DSI EXEC(107(2) westernDS.westDB) - dsigmint.c(3027) Message from server: Message: ..., State ..., Severity... -- '...'</pre>
Explanation	In some cases, the DSI thread shuts down. Adaptive Server errors caused the connection to be suspended if: <ul style="list-style-type: none">• The Adaptive Server error is assigned the <i>retry_stop</i> or <i>stop_replication</i> error action.• You executed the <i>suspend connection</i> command with the <i>nowait</i> clause. Error 1028 occurs when a data server returns an error to Replication Server. Error 1028 is usually followed by several other errors. The most common error situations are described in these sections: <ul style="list-style-type: none">• DSI shuts down because of SQL error in transaction• Adaptive Server and DB2 table names do not match• Adaptive Server last-chance threshold passed• Accessing a database while in recovery

If the DSI thread shuts down, you cannot simply resume the connection. Resuming the connection without fixing the problem sends the same transaction to the Adaptive Server and causes the same error.

Solution

In general, you must identify and fix the Adaptive Server problem that is causing the error. See “Correcting Adaptive Server error” on page 39. If you cannot correct the cause of the problem, as a last resort, you can resume the connection and skip the transaction. Skipping a transaction, however, causes inconsistencies between the primary and replicate databases, which you must manually fix in the replicate database. See “Skipping the transaction” on page 40. Also see the most common situations for error 1028 for solutions to their particular problems.

❖ **Correcting Adaptive Server error**

- 1 Log in to the RSSD and display the transaction ID by executing the `rs_helpexception` stored procedure.

```
1> rs_helpexception
2> go
```

```
Summary of Logged Transactions on 'westernRS'
Total # of Logged Transactions = 1
Xact ID Org          Site Org User Org Date Dest Site #      Recs/Xact
-----
107 mil01hprdss.eur eurian          Jun 13 2006  westernDS.eur      3
For Detailed Information on a Logged Xact., type 'rs_helpexception
{XactID}'
(return status = 0)
```

- 2 Using the transaction ID displayed in the previous step, display the full transaction text by executing the `rs_helpexception` stored procedure with the `v` option. The `v` option includes the text of the transaction in the output.

```
1> rs_helpexception 107, v
2> go
```

You see this information:

```
Detailed Summary of Logged Transaction # 107 on 'westernRS'
Origin Site          Origin User   Org. Commit Date   #Cmds in Xact
-----
westernDS.westDB    eurian       Jun 13 2006 12:24   3
Dest. Site          Dest. User   Date Logged
-----
westernDS.westDB    ...eurian   Jun 13 2006 12:27
This transaction was logged by the 'sysadmin log_first_tran' command.
Rejected Records
textval
```


An Adaptive Server error is included in the text of the Replication Server error:

```
E. 2006/06/13 12:31:29. ERROR #1028 DSI(western.west1) -
dsiqmint.c(3645) Message from server: Message: 229,
State: 1, Severity: 14-- 'DELETE permission denied on
object real_run, database west1, owner dbo'.
H. 2006/06/13 12:31:29. THREAD FATAL ERROR #5049
DSI(western.west1) - dsiqmint.c(3652) The DSI thread
for database 'western.west1' is being shutdown. DSI
received data server error #229 which is mapped to
STOP_REPLICATION. See logged data server errors for more
information. The data server error was caused by RS
output command #1 mapped from input command #2 of the
failed transaction.
I. 2006/06/13 12:31:29. The DSI thread for database
'western.west1' is shutdown. The Message from server:
text is the message you would get from the Adaptive
Server had you sent the command batch to the server with
isql. The text is taken from the table sysmessages. In
isql you would get: Msg 229, Level 14, State 1: DELETE
permission denied on object real_run, database west1,
owner dbo
```

Explanation

These errors occur when a Replication Server transaction that is sent to the replicate Adaptive Server has a SQL error. The replicate Adaptive Server detects the SQL error in the transaction and returns a message to the Replication Server.

The Replication Server responds by shutting down the connection and suspending replication; this prevents data inconsistencies between the primary and replicate sites. It allows the user to correct the problem at the replicate Adaptive Server and to maintain data consistency.

For example, when a SQL semantic error occurs in a referential integrity or duplicate keys situation, Adaptive Server sends its message back to Replication Server. In response, Replication Server shuts down the connection and records the Replication Server error in the Replication Server error log. The Adaptive Server error is referenced in the body of the Replication Server error.

Solution

Fix the problem in Adaptive Server and resume the connection as described in the Solutions section of 1028: data server errors. If the problem persists, verify that autocorrection for the subscription has been turned on in the `rs_repopjs.attributes` table of the RSSD. Also, check for triggers that enforce referential integrity rules.

Adaptive Server and DB2 table names do not match

Symptom These messages are displayed in the Replication Server error log:

```
E. 2006/06/13 12:31:29. ERROR #1028 DSI(DSMA1.apptst) - dsigmint.c(3668)
Message from server: Message: 208, State: 1, Severity: 16 -- 'TEST not found.
Specify owner.objectname or use sp_help to check whether the object exists
(sp_help may produce lots of output). `'.
H. 2006/06/13 12:31:29. THREAD FATAL ERROR #5049 DSI(DSMA1.apptst) -
dsigmint.c(3675) The DSI thread for database 'DSMA1.apptst' is being
shutdown. DSI received data server error #208 which is mapped to
STOP_REPLICATION.
See logged data server errors for more information. The data server error
was caused by RS output command #1 mapped from input command #2 of the failed
transaction.
E. 2006/06/13 12:31:29. ERROR #32032 LTM USER(TCPIP.ZD60) - /nrm/nrm.c(1658)
No table with name 'TEST' is defined at repserver with id TCP/IP.ZD6
```

Explanation These errors occur when replicating a DB2 table with an uppercase name to an Adaptive Server. DB2 object names are in uppercase while Adaptive Server object names can be in mixed case.

Solution Use any one of the following solutions:

- Replicate into VIEWS with uppercase object names.
- Create a table name in uppercase, and use custom function strings with column names that match the case of the columns in the replicate table.
- Create the Adaptive Server object names in uppercase to match the DB2 object names.
- If the Adaptive Server table uses lowercase names, use function strings to force the Replication Server to generate lowercase table and column names when transactions are sent to the Adaptive Server.

Adaptive Server last-chance threshold passed

Symptom The following errors, which include references to the Adaptive Server error 7415, are recorded in the Replication Server error log:

```
I. 2006/06/13 10:45:07. Message from server: Message: 7415, State: 1,
Severity: 10 - - 'The transaction log in database northDB is almost full.
Your transaction is being suspended until space is made available in the
log.'.
E. 2006/06/13 10:45:07. ERROR #5046 DSI(afx st.northDB) - /dsioqid.c(1638)
When executing the rs_get_lastcommit function in database 'afx st.northDB',
received data server errors. See logged data server errors for more
information.
```

Explanation By default, the replicate Adaptive Server suspends all transaction processing when the destination database log segment size becomes greater than the Adaptive Server's last-chance threshold. The last-chance threshold is a parameter that keeps the log from expanding to greater than the maximum size of the log.

If the database is the RSSD, other serious consequences can occur. See "RSSD log device full" on page 47.

This problem is related to "DSI shuts down because replicate database log is full" on page 43, which occurs when the Adaptive Server log for the database is almost full.

Note This problem can also apply to Replication Server version 11.0.x.

Solution Reduce the size of the log by dumping or truncating it as follows:

- 1 Manually dump the log using these commands in the RSSD:


```

> sp_helpdb northDB
> go
> dump tran northDB to ...
> go
> sp_helpdb northDB
> go
```
- 2 If this step fails, truncate the log by executing `dump tran` with the `truncate_only` or `no_log` option.

DSI shuts down because replicate database log is full

Symptom The DSI thread shuts down and the following error message is displayed in the Replication Server error log:

```

E. 2006/06/13 10:49:07. ERROR #1028 DSI EXEC(107(1) SYDNEY_DS.pubs2) -
dsiqmint.c( 2361) Message from server: Message: 1105, State 3, Severity 17
-- 'Can't allocate space for object 'syslogs' in database 'pubs2' because
the 'logsegment' segment is full. If you ran out of space in syslogs, dump
the transaction log. Otherwise, use ALTER DATABASE or sp_extendsegment to
increase the size of the segment.
H. 2006/06/13 10:49:07. THREAD FATAL ERROR #5049 DSI EXEC(107(1)
SYDNEY_DS.pubs2) - dsiqmint.c(2368) The DSI thread for database
'SYDNEY_DS.pubs2' is being shutdown.
DSI received data server error #1105 which is mapped to STOP_REPLICATION.
See logged data server errors for more information. The data server error
was caused by output command #1 mapped from input command #1 of the failed

```

transaction.

Explanation	<p>The DSI thread to the replicate data server has shut down because the Adaptive Server log for the database is full. A full Adaptive Server log could be caused by:</p> <ul style="list-style-type: none">• Problems at the replicate database, or• Problems at the replicate Replication Server. <p>This problem is related to “Adaptive Server last-chance threshold passed” on page 42, which occurs when the Adaptive Server log for the database is almost full.</p>
Solution	<p>Complete the following tasks:</p> <ol style="list-style-type: none">1 Add space to the Adaptive Server log, after verifying that there is space in the Replication Server stable device.2 If there is not enough space in the stable device to increase the size of the log, add space to the stable device.3 Make sure the DSI thread to the data server and the RSI thread to the Replication Server are up and running. <ul style="list-style-type: none">• Starts up, or• Resumes a connection, or• Replicates new transactions after a closed connection.

Accessing a database while in recovery

Symptom	<p>If a Replication Server attempts to use an Adaptive Server database that is in the process of recovery, you see these messages in the Replication Server error log:</p>
---------	--

```
E. 2006/06/13 10:53:36. ERROR #1028 DSI(western.west1) - /dsiexec.c(306)
Message from server: Message: 921, State: 1, Severity: 14 -- 'Database
'west1' has not been recovered yet - please wait and try again.'
I. 2006/06/13 10:53:36. Message from server: Message: 5701, State: 1,
Severity: 10 - - 'Changed database context to 'master'.'.
E. 2006/06/13 10:53:36. ERROR #5051 DSI(western.west1) - /dsiexec.c(314)
Received errors from database 'western.west1'. See logged ct-lib and data
server messages for more information.
```

Explanation	<p>A Replication Server cannot connect to an Adaptive Server database that is recovering. The database is not accessible until it has recovered. Replication Server automatically attempts to connect to databases (to which connections have been created) when Replication Server:</p> <ul style="list-style-type: none"> • Starts up, or • Resumes a connection, or • Replicates new transactions after a closed connection.
Solution	The connection resumes automatically after the database has recovered.

7035: Replication Server out of memory

Symptom	<p>After a connection is started, these error messages are displayed in the Replication Server error log:</p> <pre>I. 2006/06/13 10:58:42. The DSI thread for database 'westernDS.westDB' is started. E. 2006/06/13 10:58:42. ERROR #7035 DIST(westernDS.westDB) - m/memseg.c(771)Additional allocation would exceed the memory_limit of '20000' specified in the configuration. Increase the value of the memory_limit parameter in the rs_config table and restart the Repserver.</pre>
---------	---

A second attempt at resuming the connection displays only the following informational message:

```
I. 2006/06/13 11:08:06. Attempt to start a DSI thread
for database 'westernDS.westDB' that has already been
started.
```

Running `admin who_is_down` at the Replication Server indicates that threads are down.

Both DIST and DIST EXEC threads are down:

Spid	Name	State	Info
----	-----	-----	-----
	DIST	Down	westernDS.westDB
	DIST EXEC	Down	105:1 westernDS.westDB

Explanation	<p>Replication Server ran out of segment memory. The maximum amount of operating system memory that a Replication Server can use is equal to the value specified by the <i>memory_limit</i> parameter. Replication Server directly uses operating system memory.</p>
-------------	--

- Solution
- To fix this problem:
- 1 Increase the value of the *memory_limit* parameter.
 - 2 Restart the Replication Server.

11061: RSSD problems

Error 11061 is displayed in the Replication Server error log when an Adaptive Server error occurs. The most common error situations are:

- RSSD Deadlocks
- RSSD Log Device Full

RSSD deadlocks

Symptom

The RSSD deadlocks and these error messages are displayed in the Replication Server error log:

```
E. 2006/06/13 11:14:12. ERROR #11061 USER(rho_dbo) - s/stscol.c(1717) Check
the log for error messages from RSSD.
E. 2006/06/13 11:18:22. ERROR #1028 USER(rho_dbo) - s/stscol.c(1717) Message
from server: Message: 1205, State: 2, Severity: 13 -- 'Your server command
(process id #14) was deadlocked with another process and has been chosen as
deadlock victim. Re-run your command.'
```

Explanation

RSSD deadlocks usually occur when commands for the RSSD are issued faster than the server can process them. For example, RSSD deadlocks may occur even on a fast machine and network when you run scripts that create, alter, or delete many subscriptions or replication objects.

Note In a production environment, deadlock situations on the replicate database are automatically handled by the Replication Server.

RSSD deadlocks are known to occur when you are:

- Creating routes in parallel within a star configuration. A *star configuration* has one primary Replication Server with only direct routes to other destination Replication Servers and each destination Replication Server has only one direct route back to the primary Replication Server.
- Creating, activating, or validating subscriptions in one or more Replication Servers.

- Dropping replication definitions in parallel in different Replication Servers.

Solution

If routes are deadlocked, drop the routes and re-create them sequentially, allowing one minute between each creation.

If an RSSD deadlock occurs during the activation or validation of subscriptions:

- 1 Use `rs_helpsub` in the RSSD or check subscription in the Replication Server to check for subscriptions with status “active/activating” instead of “active/unknown.”
- 2 Drop the “active/activating” subscriptions using the `without purge` option and re-create subscriptions.

If deadlocks occur while you are dropping subscriptions, drop them again.

To prevent a large number of deadlocks, do not load several scripts into Replication Server simultaneously. In extreme situations, avoid loading scripts simultaneously in different Replication Servers; instead run scripts sequentially.

RSSD log device full**Symptom**

These error messages are displayed in the Replication Server error log:

```
E. 2006/06/13 10:35:15. ERROR #11061 USER(western_dbo)
- s/stscol.c(1717) Check the log for error messages from
RSSD.
I. 2006/06/13 10:35:15. Message from server: Message:
7412, State: 1, Severity: 10 - - 'Space available in the
log segment has fallen critically low in database
'rssd'.
All future modifications to this database will be
suspended until the log is successfully dumped and space
becomes available.'.
I. 2006/06/13 10:35:15. Message from server: Message:
7415, State: 1, Severity: 10 - - 'The transaction log
in database rssd is almost full.
Your transaction is being suspended until space is made
available in the log.'.
```

Explanation During subscription validation, the RSSD ran out of log space even though the truncate on checkpoint option had been set for the RSSD. Replication Server halted and Adaptive Server suspended modifying all transactions. After you resolved the log space issue, Replication Server continued to process the subscription but did not validate it correctly.

Solution To solve this problem, reduce the size of the log by dumping or truncating it as follows.

1 Manually dump the log using these commands in the RSSD:

```
> sp_helpdb RSSD
> go
> dump tran RSSD to ...
> go
> sp_helpdb RSSD
> go
```

2 If this step fails, truncate the log by executing dump tran with the truncate_only or no_log option.

To prevent this error, monitor the RSSD log. If the RSSD log becomes more than 80% full:

- 1 Suspend the operating system process that creates replicated objects (subscriptions).
- 2 Wait one minute to allow Replication Server to finish its transactions.
- 3 Dump the transaction log.
- 4 Resume the process.

13045: replication suspended because RSSD restarted

Symptom These error messages are displayed in the Replication Server error log:

```
E. 2006/06/13 14:50:16. ERROR #13045 SQT(101:1 DIST westss.eastlp) -
seful/cm.c(3914)
Failed to connect to server 'westss' as user 'westrs_rssd_prim'. See CT-Lib
and/or server error messages for more information.
I. 2006/06/13 14:50:17. Trying to connect to server 'westss' as user
'westrs_rssd_prim' .....
```

After the Adaptive Server with the RSSD has been started again, the following error messages are displayed in the Replication Server error log:

```
E. 2006/06/13 17:04:52. ERROR #1027 dSUB( ) -
```

```
seful/cm.c(3909)
Open Client Client-Library error: Error: 84083972,
Severity 5 -- 'ct_connect():
network packet layer: internal net library error: Net-
Lib protocol driver call to connect two endpoints
failed', Operating System error 0 -- 'Socket connect
failed - errno 146 Connection refused'.
E. 2006/06/13 17:04:52. ERROR #13045 dSUB( ) -
seful/cm.c(3914)
Failed to connect to server 'westss' as user 'amerttp'.
See CT-Lib and/or server error messages for more
information.
I. 2006/06/13 17:04:52. Trying to connect to server
'westss' as user 'westrs_rssd_prim' .....
E. 2006/06/13 17:04:57. ERROR #1027 dSUB( ) -
seful/cm.c(3909)
Open Client Client-Library error: Error: 84083972,
Severity 5 -- 'ct_connect():
network packet layer: internal net library error: Net-
Lib protocol driver call to connect two endpoints
failed', Operating System error 0 -- 'Socket
connectfailed - errno 146 Connection refused'.
E. 2006/06/13 17:05:56. ERROR #13043 USER(westss_ra) -
ul/cmapp.c(888)
Failed to execute the 'USE westss_rssd' command on
server 'westss'. See CT-Lib and SQL Server error
messages for more information.
E. 2006/06/13 17:05:56. ERROR #1028 USER(westss_ra) -
ul/cmapp.c(888)
Message from server: Message: 911, State 2, Severity 11
-- 'Attempt to locate entry in sysdatabases for database
'westss_rssd' by name failed - no entry found under that
name. Make sure that name is entered properly.'.
I. 2006/06/13 17:05:56. Message from server: Message:
5701, State 1, Severity 10 -- 'Changed database context
to 'master''.
E. 2006/06/13 17:05:56. ERROR #13045 USER(westss_ra) -
seful/cm.c(3318)
Failed to connect to server 'westss' as user
'westrs_rssd_prim'. See CT-Lib and/or server error
messages for more information.
E. 2006/06/13 17:05:56. ERROR #1028 USER(westss_ra) -
seful/cm.c(3318)
Message from server: Message: 911, State 2, Severity 11
-- 'Attempt to locate entry in sysdatabases for database
'westss_rssd' by name failed - no entry found under that
```

```

name. Make sure that name is entered properly.'.
I. 2006/06/13 17:05:56. Message from server: Message:
5701, State 1, Severity 10
-- 'Changed database context to 'master''.
E. 2006/06/13 17:05:56. ERROR #13043 dREC(dREC)--
ul/cmapp.c(888)
Failed to execute the 'USE westss_rssd' command on
server 'westss'. See CT-Lib and SQL Server error
messages for more information.
    
```

Explanation The Adaptive Server that controls the RSSD was shut down and restarted while the Replication Server was running. The DIST and SQT threads to the databases controlled by the Replication Server were terminated. Replication to those databases was terminated and will not resume even after the RSSD becomes available again.

Running the admin who_is_down command at the Replication Server shows that both DIST and SQT threads are down as follows:

Spid	Name	State	Info
-----	-----	-----	-----
	DIST	Down	westernDS.westDB
	SQT	Down	105:1 westernDS.westDB

Solution To solve the problem:

- 1 At the Replication Server, execute resume distributor for each database to resume SQT and DIST threads.
- 2 Run admin who_is_down at each database to verify that the SQT and DIST threads are up.

15020: replication definition could not be found

Symptom The following error messages are displayed in the Replication Server error log:

```
Error #15020: "'%s' doesn't exist."
```

Explanation The replication definition for which you want to create a subscription could not be found.

Solution

- Verify that a replication definition has been created for the primary version of the table that is to be replicated.
- If the primary Replication Server is not the same as the replicate Replication Server, verify that:

- A route has been created from the primary Replication Server to the replicate Replication Server, or
- The replication definition has been replicated.
- Verify that subscription DDL commands were entered at the replicate Replication Server.

15040: no connection to replicate database

Symptom	The following error message is displayed in the Replication Server error log: <code>Error #15040: "This database '%s.%s' is not controlled by this site."</code>
Explanation	A connection from the replicate Replication Server to the database has not been created.
Solution	Use the create connection command at the replicate Replication Server to make a connection to the replicate database. To create a connection to an Adaptive Server replicate database, use <code>rs_init</code> .

15052: primary and replicate database cannot be the same

Symptom	The following error message is displayed in the Replication Server error log: <code>Error #15052: "The replicate database '%s.%s' for subscription '%s' is the same as the primary database for the replication definition '%s'. This is not allowed."</code>
Explanation	You cannot create a subscription in which the replicate database is the same as the primary database.
Solution	To correct the problem: <ol style="list-style-type: none"> 1 Create the replicate table on a different database. 2 Execute the create subscription command again.

28028: connect source not executed

Symptom	The following error message is displayed in the Replication Server error log: <code>Error #28028: "Connect source as user '%s' failed at PRS</code>
---------	--

```
'%s' for subscription '%s', for replication definition
'%RS_RSID' with replicate at '%RS_SITEID'."
```

Explanation The replicate Replication Server failed to execute the connect source command at the primary Replication Server.

Solution The subscription recovery daemon should recover the subscription.

29024: no matching function strings found

Symptom The error message is similar to the following:

```
Jan 26 11:27:59.300 2006: Error: 29204, Line: 864, File:
'generic/dsi/fstrmap.c' - Cannot find a matching
function string for function 'stocks.rs_insert' and
function class 'rs_sqlserver_function_class'.
```

Explanation This error occurs when Replication Server cannot find a function string to match a function. The DSI thread is suspended.

Solution To correct the problem:

- 1 Query the RSSD at the primary site of the function string to determine if the function string exists there. Use this query for strings with replication definition scope:

```
select name from rs_funcstrings, rs_functions,
rs_classes, rs_objects
where rs_funcstrings.classid = rs_classes.classid
and rs_funcstrings.funcid = rs_functions.funcid
and classname = function_class_name
and classtype = 'F'
and rs_functions.objid = rs_objects.objid
and rs_objects.objname = replication_definition
and objtype = 'R' and funcname = function_name
```

If the function string does not exist at the primary site, continue with step 2.

If the function string does exist at the primary site, go to step 3.

- 2 If the function string does not exist at the primary site, create it there. If you omit the output clause from the string, Replication Server generates the default function string.

- 3 If the function string does exist at the primary site, it hasn't been replicated to the Replication Server at the replicate site. Verify that the RSSD at the primary site is being replicated to the replicate site. Refer to Chapter 1, "Identifying the Problem", for details. If changes at the primary site are being replicated to the replicate site, then you may need to wait for the function strings to arrive. Then restart the DSI.
- 4 For `rs_select` and `rs_select_with_lock`, there must be a function string whose input template matches the `where` clause of the `select` command. Extend the above query with a `join` to `rs_systext` to retrieve the input templates of function strings.

If the DSI receiving the error is communicating with a server using a function-string class other than the `rs_sqlserver_function_class`, `rs_default_function_class`, or `rs_db2_function_class`, you may still need to create the function string mentioned in the preceding paragraph.

37022: *primary subscribe permission required*

Symptom	The following error message is displayed in the Replication Server error log: <code>Error #37022: "PRIMARY SUBSCRIBE permission is required to execute the subscription command."</code>
Explanation	The user does not have the permission required to create a subscription. You must have <code>primary subscribe</code> , <code>create object</code> , or <code>sa</code> permission at the primary Replication Server.
Solution	Use the <code>grant</code> command to change the user's permissions.

37023: *create object permission required*

Symptom	The following error message is displayed in the Replication Server error log: <code>Error #37023: "CREATE OBJECT permission is required to execute command."</code>
Explanation	The user does not have the required permission.
Solution	Use the <code>grant</code> command to change the user's permissions to <code>primary subscribe</code> .

Replication Server informational and warning messages

This section contains informational and warning messages displayed in the Replication Server error log. Because informational and warning messages do not have error numbers, they are organized alphabetically.

A cached row for system table was swapped out

Symptom	This Replication Server informational message is displayed many times in the Replication Server error log: <pre>I. 2006/06/13 15:39:53. A cached row for system table `rs_columns' was swapped out of the cache in order to accomodate another row.</pre>
Explanation	When there is no space available in the cache for any more rows in a specific RSSD system table, that system table's oldest rows are swapped out of the cache. The <i>sts_cachesize</i> parameter specifies the maximum number of rows that are cached for a single RSSD system table.
Solution	Increase the value of the <i>sts_cachesize</i> parameter for the RSSD.

Detecting loss for database

Symptom	The message below is displayed in the Replication Server error log after you executed a rebuild queues command (which is typically issued after a stable device failed).
---------	--

You should check for data loss detection messages in the Replication Server error log only after you have issued the `rebuild queues` command and the data loss detection process has completed. Data loss detection should only be performed when there are primary transactions to be replicated. The RMS heartbeat feature replicates a primary transaction to the destination data server. You can activate the heartbeat feature for the connection to force data loss detection to be performed.

Note Sometimes these messages are not displayed at the end of the error log file, so you must search back through the log for them. If you restart the Replication Server, these messages will be displayed again.

```
I. 2006/06/13 15:48:32. Rebuild Queues: Starting
I. 2006/06/13 15:48:33. Disconnecting Replication Agent
for westss.westrss_rssd. Replication Agent will shutdown
I. 2006/06/13 15:48:33. Disconnecting Replication Agent
for westss.westrss_rssd to Rebuild
I. 2006/06/13 15:48:33. Resetting Replication Agent
starting log position for westss.westrss_rssd
I. 2006/06/13 15:48:33. Resetting Replication Agent
starting log position for westss.eastlp
I. 2006/06/13 15:48:33. Shutting down distributor for
101.
I. 2006/06/13 15:48:33. A request to shutdown/suspend
the distributor for 101 has been received.
I. 2006/06/13 15:48:33. The distributor for
'westss.westrss_rssd' is shutting down
I. 2006/06/13 15:48:33. Shutting down distributor for
102.
I. 2006/06/13 15:48:33. A request to shutdown/suspend
the distributor for 102 has been received.
I. 2006/06/13 15:48:34. The distributor for
'westss.eastlp' is shutting down
I. 2006/06/13 15:48:34. Shutting down the DSI thread for
'westss.westrss_rssd'.
I. 2006/06/13 15:48:34. The DSI thread for database
'westss.westrss_rssd' is shutdown.
I. 2006/06/13 15:48:34. DSI: enabled loss detection for
'westss.westrss_rssd'.
I. 2006/06/13 15:48:34. Shutting down the DSI thread for
'westss.eastlp'.
I. 2006/06/13 15:48:34. The DSI thread for database
'westss.eastlp' is shutdown.
I. 2006/06/13 15:48:34. DSI: enabled loss detection for
```

```
'westss.eastlp'.  
I. 2006/06/13 15:48:34. Shutting down the DSI thread for  
'westss.westlp'.  
I. 2006/06/13 15:48:35. The DSI thread for database  
'westss.westlp' is shutdown.  
I. 2006/06/13 15:48:35. DSI: enabled loss detection for  
'westss.westlp'.  
I. 2006/06/13 15:48:35. Rebuild queues: deleting queue  
103:0  
I. 2006/06/13 15:48:35. SQM stopping: 103:0  
westss.westlp  
I. 2006/06/13 15:48:35. Rebuild queues: done rebuilding  
queue 103:0. Restarting.  
I. 2006/06/13 15:48:35. Rebuild queues: deleting queue  
102:1  
I. 2006/06/13 15:48:35. SQM stopping: 102:1  
westss.eastlp  
I. 2006/06/13 15:48:35. SQM starting: 103:0  
westss.westlp  
I. 2006/06/13 15:48:35. Rebuild queues: done rebuilding  
queue 102:1. Restarting.  
I. 2006/06/13 15:48:35. Rebuild queues: deleting queue  
102:0  
I. 2006/06/13 15:48:36. SQM stopping: 102:0  
westss.eastlp  
I. 2006/06/13 15:48:36. SQM starting: 102:1  
westss.eastlp  
I. 2006/06/13 15:48:36. Rebuild queues: done rebuilding  
queue 102:0. Restarting.  
I. 2006/06/13 15:48:36. Rebuild queues: deleting queue  
101:1  
I. 2006/06/13 15:48:36. SQM stopping: 101:1  
westss.westrs_rssd  
I. 2006/06/13 15:48:36. SQM starting: 102:0  
westss.eastlp  
I. 2006/06/13 15:48:36. Rebuild queues: done rebuilding  
queue 101:1. Restarting.  
I. 2006/06/13 15:48:36. Rebuild queues: deleting queue  
101:0  
I. 2006/06/13 15:48:36. SQM stopping: 101:0  
westss.westrs_rssd  
I. 2006/06/13 15:48:36. SQM starting: 101:1  
westss.westrs_rssd  
I. 2006/06/13 15:48:37. Rebuild queues: done rebuilding  
queue 101:0. Restarting.  
I. 2006/06/13 15:48:37. SQM starting: 101:0
```

```

westss.westrss_rssd
I. 2006/06/13 15:48:37. Starting DIST for 101:1.
I. 2006/06/13 15:48:37. Starting DIST for 102:1.
I. 2006/06/13 15:48:37. DIST for 'westss.westrss_rssd'
is Starting
I. 2006/06/13 15:48:37. DIST for 'westss.pdb' is
Starting
I. 2006/06/13 15:48:37. Starting the DSI thread for
'westss.westrss_rssd'.
I. 2006/06/13 15:48:38. Starting the DSI thread for
'westss.westlp'.
I. 2006/06/13 15:48:38. The DSI thread for database
'westss.westrss_rssd' is started.
I. 2006/06/13 15:48:39. Starting the DSI thread for
'westss.eastlp'.
I. 2006/06/13 15:48:40. The DSI thread for database
'westss.eastlp' is started.
I. 2006/06/13 15:48:41. The DSI thread for database
'westss.eastlp' is started.
I. 2006/06/13 15:48:41. Rebuild Queues: Complete
I. 2006/06/13 15:48:44. DSI: detecting loss for database
'westss.eastlp' from origin 'westss.westlp' date =
'Jun 13 2006 2:36:49:783PM', qid=0000000000001d
240000054b00090000054b0007000097df00f0d41700000000000
0001.

```

Explanation	Replication Server detected loss for a primary Replication Server or database. User transactions from the primary Replication Server or database are no longer accepted. If the replication system has more than one Replication Server, then the detected data loss could be either a DSI loss or an RSI loss. If the replication system has only one Replication Server, the detected data loss is a DSI loss. A DSI loss means that data was not replicated from the Replication Server to the replicate database. An RSI loss means that data was not replicated from one Replication Server to another Replication Server.
Solution	<p>For information on correcting data losses, see <i>Replication Server Administration Guide Volume 2</i>.</p> <p>To prevent data loss, use DSI or RSI save intervals to create a backup strategy with backlogs large enough to keep all messages.</p>

DSI detected *rs_update_lastcommit* not marked as replicated

Symptom	A warm standby database disconnects and these error messages are displayed in the Replication Server error log:
---------	---

```
I. 2006/06/13 15:39:53. DSI for %RS_SITEID detected that
stored procedure rs_update_lastcommit is not marked as
replicated. Please execute sp_setreplicate and resume
connection
```

When active DSI comes up in presence of the Warm Standby it checks whether rs_update_lastcommit stored procedure is marked as replicated. This DSI detected a problem with replication status of this stored procedure. Please, correct the problem and resume connection

Explanation This error occurs during a warm standby installation. You have used isql to execute the create connection command and did not mark rs_update_lastcommit as replicated.

Solution

- 1 Set replication status on rs_update_lastcommit by marking it with the sp_setrepproc system procedure.
- 2 Resume connection.

Instead of using the create connection command, you can use rs_init to create a connection to a warm standby database. This message does not occur when you use rs_init to set up the warm standby.

When you use rs_init to set up your active and standby database, the rs_update_lastcommit stored procedure is automatically marked for replication.

Stable storage use is above 75 percent

Symptom The total size of all the queues on a Replication Server has passed the set threshold. This warning message is displayed in the Replication Server error log:

```
W. 2006/06/13 18:41:12. WARNING #6089 SQM(129:0
TTTdss.eanp) - qm/sqmsp.c(1317) WARNING: Stable Storage
Use is Above 75 percent
```

Explanation A Replication Server has two default thresholds (75% and 90%) for stable devices. A warning message is issued each time these thresholds are exceeded.

Solution You should follow these procedures when the 75% and 90% thresholds have been exceeded:

- 1 Verify that the replicate database is running.
- 2 If all other components are functional and the inbound queues are growing, shut down the corresponding Adaptive Server and check for an open transaction.

- 3 If the outbound queues are growing, check the DSI save interval in conjunction with low block usage.

RepAgent error message descriptions

This section contains RepAgent error messages that are contained in the Adaptive Server error log.

9202: nested stored procedures not allowed

Symptom	<p>These error messages are displayed in the Adaptive Server error log.</p> <pre>00:00000:00011:2006/06/13 16:26:05.12 server Error: 9202, Severity: 20, State: 0 00:00000:00011:2006/06/13 16:26:05.12 server RepAgent(5): Nested replicated stored procedure detected. Transaction log may be corrupt. Please contact SYBASE Technical Support. (current marker = 1372, 12) 00:00000:00011:2006/06/13 16:26:05.12 server Rep Agent Thread for database 'pdb' (dbid = 5) terminated abnormally with error. (major 92, minor 2)</pre>
Explanation	<p>A nested stored procedure is called from within another stored procedure. The stored procedure that calls the nested stored procedure is called the outer stored procedure.</p> <p>When stored procedures with nested stored procedures are marked for replication with <code>sp_setrepproc</code>:</p> <ul style="list-style-type: none"> • The RepAgent shuts down. • The RepAgent forwards only the outer stored procedure call to the Replication Server. • An error message is displayed in the Adaptive Server error log.
Solution	<p>Do not use nested replicated stored procedures.</p> <p>❖ Working around the nested stored procedure</p> <ol style="list-style-type: none"> 1 Skip the nested stored procedure transaction: <ol style="list-style-type: none"> a Find the page of the secondary truncation point, by executing:

```
dbcc gettrunc
```

- b Find a valid page after the nested stored procedure transaction by executing the following command:

```
dbcc traceon(3604)
dbcc pagelinkage(dbid, pageid, 0,2,0,1)
```

where *pageid* is the ID for the page you received by executing `dbcc gettrunc` in step a.

- c Set a new secondary truncation point on the valid page after the nested stored procedure transaction by executing:

```
dbcc settrunc ('ltm', 'pageid', pageid)
```

where *pageid* is the id for the page *after* the current page you retrieved using `dbcc pagelinkage` in step b.

- d Reset the locator by executing:

```
rs_zeroltm
```

- 2 Reapply only the nested stored procedure transaction.
- 3 Restart RepAgent.

Note This procedure can also cause data loss because manually setting the secondary truncation point to a later page in the log skips any begin transaction statements on skipped pages. Those transactions do not replicate.

9210: network problems

Symptom

These error messages are displayed in the Adaptive Server error log:

```
2006/09/07 09:41:26.34 RepAgent (10) Error: 9210,
Severity: EX_CMDFATAL, State: 1
Fatal error from CT-Lib.
Open Client Client-Library error: Error: 84083974,
Severity: 5 -- ct_results(): unable to get layer message
string: unable to get origin message string: Net-Library
operation terminated due to disconnect.
```

Explanation

You see these error messages when there are network problems; for example, when a RepAgent could not initialize a CT-Lib connection, or a Replication Server returned an unexpected error. Network problems can also cause RepAgent error 9212 (which is caused by a CT-Lib error).

After you fix the network problems, execute the `admin who_is_down` command on the Replication Server. You see that the REP AGENT USER threads are down.

In RMS, “suspect status” (yellow) is displayed for the Replication Server, while “normal” status (green) is displayed for the RepAgent.

Solution Fix the network problems and restart RepAgent.

9215: log truncated (ASE 624)

Symptom This error message is displayed in the Adaptive Server error log:

```
2006/10/17 11:57:19.34 RepAgent (10) Error: 9215,
Severity: EX_USER, State: 1
The Secondary Truncation Point is not valid. Please make
sure that the Secondary Truncation Point is valid, and
then restart the Rep Agent Thread for this database.
```

Explanation This error occurs when the log is truncated past the secondary truncation point and the RepAgent tries to reconnect to the database. RepAgent requests the log page number from the Replication Server and then accesses that page in the database. Because the log was truncated, the RepAgent will not start.

Attempts to retrieve rows from the page via RID fail because the requested RID is a higher value than the last RID on the page. For example, truncating the log frees only those log pages that the RepAgent has sent to the Replication Server with confirmation.

Solution Because you truncated the log, you must reset the information about the database log in the `rs_locator` table within the RSSD as follows.

- 1 Execute the following stored procedure in the RSSD:


```
> rs_zeroltm primss, primdb
> go
```
- 2 Activate the secondary truncation point again in the primary database.


```
> dbcc settrunc(ltm, valid)
> go
```
- 3 Restart RepAgent.

Check for open transactions by comparing the output of dbcc gettrunc with the content of the rs_locator tables attribute locator. Truncating the transaction log may have caused orphaned transactions. Orphaned transactions occur when the transaction's begin statement is transferred, but the commit or rollback statement is accidentally deleted during log truncation.

Route Problems

This chapter describes problems that may occur when you are creating, altering, or dropping routes and how to troubleshoot these problems.

Name	Page
Overview	63
Problems with creating routes	65
Problems with altering routes	69
Problems with dropping routes	70

If you cannot find the solution in this chapter

If an error message is displayed in an error log and you cannot find it in this chapter, refer to:

- “Analyzing error logs” on page 9, or
- Chapter 3, “Common Errors”

If an error message is not displayed, refer to Chapter 1, “Identifying the Problem”.

Overview

Use the procedures in this chapter to successfully solve problems that occur after executing a create route, alter route, or drop route command. Each procedure requires you to obtain and analyze information from the Replication Server error logs, execute the `rs_helproute` stored procedure, or both.

To learn how to use the `rs_helproute` stored procedure and interpret its results, see “Using the `rs_helproute` stored procedure” on page 64 and the *Replication Server Reference Manual*.

A route is one-way message stream from a source Replication Server to a destination Replication Server. Routes carry data modification commands (including those for RSSDs) and replicated functions or stored procedures between Replication Servers.

There are two types of routes—direct and indirect. A direct route sends messages directly from source to destination Replication Servers, with no intermediate Replication Servers. An indirect route sends messages through one or more intermediate Replication Servers.

The `rs_marker` stored procedure contains the full create route, alter route, or drop route command. Routes are created, altered, and dropped according to the following general procedure:

- The source Replication Server executes the `rs_marker` stored procedure against its RSSD.
- The RepAgent of the RSSD picks up the stored procedure execution and forwards it to the source Replication Server.
- The source Replication Server appends the stored procedure execution to its inbound queue and forwards it to the destination Replication Server. The identity of the destination Replication Server is obtained from parameters in the stored procedure.
- The destination Replication Server processes the stored procedure execution, which materializes subscriptions to the primary Replication Server RSSD `rs_routes`, `rs_subscriptions`, `rs_rules`, and `rs_locator` system tables.

Using the `rs_helproute` stored procedure

Routes go through various states while being created, altered, or dropped. Execute the `rs_helproute` stored procedure on the RSSD at the source or the destination Replication Server to see the current state of the route:

```
rs_helproute [replication_server]
```

```
replication_server rs_helproute
```

is an optional parameter indicating the name of a Replication Server. If you enter this parameter, information is given for routes to and from the specified Replication Server. Otherwise, information is provided for all routes to and from the current Replication Server. For each route, returns:

- Route status

Status identifies the state of the route protocol. The status of a route is different at the source and destination Replication Servers. You analyze the route status at the source and destination Replication Servers to determine the problem.

- A list of system table subscriptions that are currently being processed

Incomplete materialization or dematerialization of system table subscriptions is one of the most common problems.

Because creating, altering, and dropping routes includes processing system table subscriptions, the list that `rs_helproute` returns helps you determine which subscriptions prevent you from proceeding to the next step in the process.

If you are creating a route, the list includes the system table subscriptions that are being created. If you are dropping a route, the list includes the system table subscriptions that are being dropped.

If no system table subscriptions are listed for a route, there are no problems with the system table subscriptions.

Problems with creating routes

This section explains how to troubleshoot problems that can occur when you create a route.

When a route is created:

- The source Replication Server connects to the destination Replication Server. The source Replication Server needs the correct user name and password to connect to the destination Replication Server.
- The source Replication Server sends a create route message to the destination Replication Server.
- The destination Replication Server receives this message and starts creating subscriptions to the source Replication Server system tables.
- When the system table subscriptions are created, the process is complete.

Common problems

Table 4-1 provides solutions to common problems that can occur while creating a route.

Table 4-1: Creating a route—problems and suggested actions

Problem	Suggested Action
The destination Replication Server has no entry in its interfaces file for the source Replication Server.	Include this name in the interfaces file.
You are trying to create a route that has just been dropped.	Wait until the destination Replication Server has cleaned up its route to the source Replication Server. Then the destination can accept the new route. You may need to resume this route from the source Replication Server—the source may have suspended this route after being refused connection by the destination.
You attempt to create an indirect route but the log at the intermediate site contains the message “route is needed to reach the destination.”	Drop the route from the source to the destination using the drop route command with the with nowait clause. Next, create a route from the intermediate site to the destination site. Then, use the create route command with the set next site clause to create an indirect route from the source to the destination.

Messages in the error log at the source Replication Server

Table 4-2 provides the solutions to the problems described by error messages in the source Replication Server error log.

Table 4-2: Messages at the source Replication Server

Message	What Does It Mean?	Suggested Actions
RSI <i>destination name</i> : Login incorrect RSI <i>destination name</i> : Trying to connect	The source Replication Server tries to connect to the destination Replication Server with the username/password supplied with the create route command, but the destination Replication Server does not recognize this username/password.	<ol style="list-style-type: none"> 1. Perform one of these tasks: <ul style="list-style-type: none"> • Use the create user or alter user command to create or alter this username/password at the destination Replication Server. • Use the alter route command with the set username clause to alter the username/password for this route. • Drop the route using the drop route command with the with nowait clause. Then re-create the route. 2. Use the resume route command to resume this route at the source Replication Server.
cm: Could not find interfaces entry for <i>destination name</i>	The destination Replication Server name is not found in the interfaces file.	<p>Perform one of these tasks:</p> <ul style="list-style-type: none"> • Add the Replication Server name in the interfaces file, or • If the user did not mean to create a route to this destination, use the drop route command with the with nowait option to drop the route.

Output from *rs_helproute* at the source Replication Server

Table 4-3 describes the output when *rs_helproute* is executed at the source Replication Server.

Table 4-3: rs_helproute at the source Replication Server

Message	What Does It Mean?	Suggested Actions
Route is being created. Source RS has not yet attempted to send the protocol message to the destination RS.	The source Replication Server could not execute the rs_marker stored procedure against its RSSD.	Restart the Replication Server. If restarting does not fix the route, check the RSSD and any related messages in the error logs. There may be a message in the error logs indicating some problem with the RSSD.
Route is being created. Either (1) RS is waiting for a route protocol message from the RSSD Replication Agent or (2) the RSSD Replication Agent inbound queue is not being processed.	A message may be stuck between the RSSD log and the Replication Server inbound queue. The sender Replication Server executed the stored procedure, but the RSSD Replication Agent cannot forward it to the Replication Server.	Check if the RSSD Replication Agent is experiencing problems sending messages to the Replication Server. If not, restart the RSSD Replication Agent.
	A message may be stuck between the inbound and outbound queues in the Replication Server.	Check if the Replication Server's inbound queues are full or the distributor for the RSSD is not running. If neither of these problems exist, restart the RSSD Replication Agent.
Route is being created. Create route protocol message is waiting to be delivered to the destination.	The sending Replication Server is not processing its outbound queue.	Suggested Action: Resume the route, if: <ul style="list-style-type: none"> • The RSI experienced problems connecting to the destination Replication Server. • The route was suspended. Check to see if: <ul style="list-style-type: none"> • The destination Replication Server is up. • The RSI for the destination Replication Server running at the source Replication Server.

Message	What Does It Mean?	Suggested Actions
Route is being created. Destination has not yet finished creating system table subscriptions.	The destination Replication Server is still creating subscriptions to system tables.	<p>Make sure the destination Replication Server is running. Resume the route, if:</p> <ul style="list-style-type: none"> • The RSI experienced problems connecting to the destination Replication Server. • The source could not send messages to the destination • The system suspends the route due to an error or the user explicitly suspends it. <p>Check the list of system table subscriptions that still need to be created.</p>

Output from *rs_helproute* at the destination server

Table 4-4 describes the output when *rs_helproute* is executed at the destination Replication Server.

Table 4-4: *rs_helproute* output at the destination server

Message	What Does It Mean?	Suggested Actions
This site received the create route protocol message from the source RS and is in the process of creating system table subscriptions.	The destination Replication Server is completing its system table subscriptions.	Check the list of system table subscriptions that still need to be created. If the destination Replication Server is suspended in this state, it could mean there is a problem with the system table subscriptions.

Problems with altering routes

If you are experiencing problems using the alter route command:

- Quiesce the replicated data system prior to executing the alter route command. For more information, see the *Replication Server Administration Guide Volume 1*.

- Create a route from the new intermediate site to the destination site if you are changing a direct route to an indirect route, or if you are changing the intermediate site in an indirect route.
- Execute `rs_helproute` at the destination Replication Server to determine the success of an `alter route` command. Make sure that the output reflects the route change. If the route change did not succeed, follow the troubleshooting procedures in “Output from `rs_helproute` at the source Replication Server” on page 67.

Problems with dropping routes

This section explains how to troubleshoot problems that can occur when you drop a route. By default, the source Replication Server waits until the destination Replication Server has cleaned up its part of the route. You can use the `nowait` option to drop a route without waiting for the destination Replication Server to clean up its part of the route.

Dropping routes

When a route is dropped:

- The source Replication Server drops all the system table subscriptions created by the destination.
- The source Replication Server sends a drop route protocol message to the destination Replication Server.
- After this message is delivered to the destination Replication Server (or to the next site, if the route is an indirect route), the source Replication Server drops the route by deleting the entry from its `rs_routes` system table and, for direct routes, deleting the outbound queue.
- The destination Replication Server receives the drop route message and cleans up its part of the route by deleting the locator for the source Replication Server from its `rs_locator` system table.

Output from *rs_helproute* at the source Replication Server

Table 4-5 describes the output when *rs_helproute* is executed at the source Replication Server.

Table 4-5: *rs_helproute* output at the source Replication Server

Message	What Does It Mean?	Suggested Actions
Route is being dropped. System table subscriptions are being dropped.	The destination Replication Server is trying to drop the system table subscriptions.	Check the list of system table subscriptions that still need to be dropped.
Route is being dropped. Waiting for the system table subscriptions to be dropped.	The system is in this state until the subscriptions are dropped.	The Replication Server should not be in this state for a long period of time. If it is, it may indicate a subscription problem. Refer to Chapter 5, "Subscription Problems".
Route is being dropped. Source RS has not yet attempted to send the drop route protocol message to the destination RS.	The sender Replication Server could not execute the <i>rs_marker</i> stored procedure against its RSSD.	Restart the Replication Server. Check the RSSD and any related messages in the error logs. There may be a message in the error logs indicating some problem with the RSSD.
Route is being dropped. Drop route protocol message should be either in RSSD Replication Agent queue or in the inbound queue.	A message may be stuck between the RSSD log and the Replication Server inbound queue. The sender Replication Server executed the stored procedure, but the RSSD Replication Agent cannot forward it to the Replication Server.	Check if the RSSD Replication Agent is experiencing problems sending messages to the Replication Server. If not, restart the RSSD Replication Agent.
	A message may be stuck between the inbound and outbound queues in the Replication Server.	Check if the Replication Server inbound queues are full or the distributor for the RSSD is not running. If neither of these problems exist, restart the RSSD Replication Agent.
Route is being dropped. Drop route protocol message is waiting to be delivered to the destination.	The sender Replication Server is not processing its outbound queue.	Is the destination Replication Server operating? Did the RSI (Replication Server Interface) experience problems trying to connect to the destination Replication Server? If so, it may be necessary to resume this route.

Output from *rs_helproute* at the destination server

Table 4-6 describes the output when *rs_helproute* is executed at the destination Replication Server.

Table 4-6: *rs_helproute* output at the destination server

Message	What Does It Mean?	Suggested Actions
Route is being dropped.	The route is being dropped by the source Replication Server. The destination Replication Server is not very active while a route is being dropped. All of the state changes are seen at the source Replication Server only.	Look at the output from <i>rs_helproute</i> at the source Replication Server.

Dropping routes using the *with nowait* option

Warning! Use the *with nowait* clause only when the destination is temporarily irrecoverable and you must drop the route without waiting for the destination Replication Server to become available.

This command instructs the source Replication Server to drop its route to the destination without informing the destination Replication Server. You must then clean up the destination Replication Server when it becomes available again.

When you execute *drop route* with the *with nowait* option:

- The source Replication Server drops all the system table subscriptions created by the destination.
- The source Replication Server cleans up the route by deleting the route entry from its *rs_routes* system table and, for direct routes, dropping the outbound queue.

Output from *rs_helproute* at the source Replication Server

The output is the same as the *drop route* command, except that the route status reads:

```
Route is being dropped with nowait. ...
```

instead of:

```
Route is being dropped. ...
```

❖ Cleaning up the destination Replication Server

- 1 Verify that all the messages from the source Replication Server are gone from the DSI queues.
- 2 From the `rs_subscriptions` system table, delete all the system table subscriptions entered by this site when the route was created. Delete rows from the `rs_rules` system table for these subscriptions.
- 3 Clean up user subscription entries from the `rs_subscriptions` and `rs_rules` system tables
- 4 Execute this command against each system table:

```
delete from system_table
      where prsid = sourceRS_siteid
```

where:
 - *system_table* is the name of the system table (`rs_subscriptions` or `rs_rules`).
 - *sourceRS_siteid* is the site ID of the source Replication Server. For more information about the delete command, see the *Adaptive Server Enterprise Reference Manual: Commands*.
- 5 Delete the entry for this route from the `rs_routes` system table.
- 6 Delete the entry for the source Replication Server from the `rs_locator` system table.

Subscription Problems

This chapter describes problems that occur when subscriptions are created, altered, or dropped.

Name	Page
Overview	75
Using check subscription	76
Materialization problems	79
Dematerialization problems	95

If you cannot find the solution in this chapter

If an error message is displayed in an error log and you cannot find it in this chapter, refer to:

- “Analyzing error logs” on page 9, or
- Chapter 3, “Common Errors”

If an error message is not displayed, refer to Chapter 1, “Identifying the Problem”

Overview

This chapter explains how to use the check subscription command to display the status of a subscription and how to solve some common problems when using atomic, nonatomic, and bulk materialization. Use `rs_helppub` and `rs_helppubsub` to find the publications and articles that a subscription is using. For more information about `rs_helppub` and `rs_helppubsub`, see the *Replication Server Administration Guide Volume 1*. The general procedure for troubleshooting subscription problems is in “Troubleshooting subscription failures” on page 13.

The replication process begins with subscription *materialization*, which is the process by which data is first copied to the destination database. When you no longer want a subscription replicated to a destination database, you dematerialize the subscription at the destination database. *Dematerialization* is the process by which data is deleted from the destination database.

Note If you are using a Replication Agent, your subscription materialization process may differ from the process described in this chapter. See your Replication Agent documentation for the Replication Agent-specific subscription materialization process.

Subscription failures occur when materialization or dematerialization fails. Usually, the person who is conducting the materialization or dematerialization monitors the operation and reports any problems.

Subscription problem symptoms are easily identified and include:

- Materialization failure – no data in the subscription’s replicate table at the destination database, the status for subscriptions at the primary and replicate Replication Servers is not valid, or materialization is taking longer than is reasonable.
- Dematerialization failure – data still exists in the subscription’s replicate table at the destination database, subscriptions still exist at the primary or replicate Replication Servers, or dematerialization is taking longer than is reasonable.

If a subscription problem caused the DSI thread for the replicate database to abnormally terminate, you can restart it using the resume connection command. For more information, see “DSI down or suspended” on page 119.

Using *check subscription*

If all of the servers and Replication Server threads are running, but the subscription is not being created or dropped, use the *check subscription* command, which returns messages that describe the status of the materialization process.

Execute the check subscription at the primary and replicate Replication Servers. If the primary and replicate Replication Servers are the same Replication Server, you need to execute check subscription only once; check subscription displays separate status messages.

Use check subscription for publication to find out the materialization status of publication subscriptions. For atomic, nonatomic, or incremental publication subscriptions, check subscription for publication shows the status of only the current article subscription being processed; whereas for bulk or no materialization, the status of all article subscriptions is shown because all article subscriptions are processed together.

Table 5-1 and Table 5-2 list the messages that can be returned by executing check subscription at the primary and replicate Replication Servers.

Table 5-1: check subscription status at replicate Replication Server

Status	Message
Invalid	<i>subscription name</i> doesn't exist.
Removing	REMOVING subscription <i>subscription name</i> from system tables at the Replicate.
Dematerializing	Subscription <i>subscription name</i> is DEMATERIALIZING at the Replicate.
Valid	Subscription <i>subscription name</i> is VALID at the Replicate.
Validating	Subscription <i>subscription name</i> is VALIDATING at the Replicate.
Materialized	Subscription <i>subscription name</i> has been MATERIALIZED at the Replicate.
Active	Subscription <i>subscription name</i> is ACTIVE at the Replicate.
Activating	Subscription <i>subscription name</i> is ACTIVATING at the Replicate.
Qcomplete and Active	Subscription <i>subscription name</i> is ACTIVE at the Replicate and Materialization Queue has been completed.
Qcomplete	Materialization Queue for Subscription <i>subscription name</i> has been completed.
Active and Qcomplete	Subscription <i>subscription name</i> is ACTIVE at the Replicate, but Materialization Queue for it has not been completed.
Defined	Subscription <i>subscription name</i> has been defined at the Replicate.
Error	Subscription <i>subscription name</i> has experienced an unrecoverable error during Materialization or Dematerialization. Please consult the error log for more details.
Pending	Other subscriptions are being created or dropped for the same replication definition/database. Subscription <i>subscription name</i> will be processed when previous requests are completed.
Recovering	Subscription <i>subscription name</i> has experienced a recoverable error during Materialization or Dematerialization. It will be recovered by Subscription Daemon (dSub).

Table 5-2: check subscription status at primary Replication Server

Status	Message
Invalid	<i>subscription name</i> doesn't exist.
Dematerializing	Subscription <i>subscription name</i> is DEMATERIALIZING at the Primary.
Valid	Subscription <i>subscription name</i> is VALID at the PRIMARY.
Active	Subscription <i>subscription name</i> is ACTIVE at the PRIMARY.
Activating	Subscription <i>subscription name</i> is ACTIVATING at the PRIMARY.
Defined	Subscription <i>subscription name</i> has been defined at the PRIMARY.

Materialization problems

This section provides information for troubleshooting materialization problems. It provides information on common materialization problems as well as detailed information about each step in the materialization process. Because the materialization process is slightly different for atomic, nonatomic, and bulk materialization methods, they are covered in separate sections.

When publication subscriptions are materialized atomically, nonatomically, or incrementally, only one article subscription is processed at a time. When publication subscriptions are materialized using the bulk or no materialization methods, all article subscriptions are processed together.

If, when you are materializing a new subscription, there are other materializing or dematerializing subscriptions for the same replication definition and replicate database, the new subscription is assigned a Pending status (the recovering column in `rs_subscriptions` is set to 2). If you execute `check subscription`, the subscription's status is returned as Pending at the replicate Replication Server and Invalid at the primary Replication Server.

If the primary RSSD is unavailable or any other problems occur, the recovering column in `rs_subscriptions` is set to 1 to denote that the subscription requires recovery.

Subscriptions with a pending status are processed one at a time, in the order entered. The `request_date` column in the `rs_subscriptions` table indicates the time the subscription request was entered.

Common problems

Problems described in this section are due to:

- Incorrect or missing login account and permissions
- Schema inconsistency
- Missing interfaces file entries

Error messages that identify these problems are displayed in the replicate Replication Server error log.

Incorrect or missing login account and permissions

Many problems with materialization are due to incorrect or missing permissions in primary or replicate databases.

User requirements for creating subscriptions

Verify that the login accounts for the user creating a subscription meet the following requirements:

- The user's login name and password are the same at the replicate Replication Server, the primary Replication Server, and the primary data server.
- The user has been added to the primary database.
- The user has select permission on the primary table.
- The user has execute permission on the rs_marker stored procedure.

This requirement applies to a configuration that uses RepAgent. If you are using a Replication Agent for a foreign data server, you may want to check your Replication Agent documentation for a similar requirement

The default installation scripts grant execute permission on rs_marker to "public," so this should not be an issue unless you did not use the installation scripts.

- The user has at least create object permission at the replicate Replication Server.
- The user has at least primary subscribe permission at the primary Replication Server.

If you cannot give the user the required accounts and permissions, drop the subscription without purge and have a different user create it.

Maintenance user permissions

Make sure that the maintenance user for the replicate database has update, delete, insert, and select permission on the replicate table. Lack of update permission causes errors in the replicate database when the DSI attempts to apply transactions in the materialization queue.

Depending on error action assignments, some transactions may be rejected and recorded in the exceptions log.

Schema inconsistency

The problems in this section result from schema inconsistencies at the primary database, Replication Server, and the replicate database.

Conflicting table or view in the primary database

A table or view with the same name and columns as the replication definition has been created at the primary database. If you have created custom function strings, make sure they match the replicate table at the replicate database. This can cause errors at the primary database as a result of executing select during subscription materialization.

Conflicting table or view in the replicate database

A table or view with the same name and columns as the replication definition has been created in the replicate database. If you create custom function strings, make sure they match the replicate table at the replicate database.

This would cause errors returned by the replicate database when the data server interface attempts to apply transactions in the materialization queue. Some transactions may be rejected and placed into the exceptions log, depending on the error action assignments.

Missing *rs_select* function string

If the primary database does not use one of the system-provided function-string classes (*rs_sqlserver_function_class*, *rs_default_function_class*, *rs_db2_function_class*) or a function-string class that is a child of *rs_default_function_class* or *rs_db2_function_class*, make sure there are function strings for the *rs_select* function that match the where clause of the create subscription or define subscription command.

Missing interfaces file entries

Login attempts may fail due to missing interface file entries. The interfaces file used by the replicate Replication Server should contain:

- An entry for the primary Replication Server
- An entry for the primary data server

Atomic materialization

The atomic materialization method is invoked using the default version of the create subscription command. If there are no other subscriptions for the same replication definition and replicate database, the subscription is defined at the primary Replication Server. If the primary Replication Server is unavailable or any other problems occur, the recovering column in the rs_subscriptions system table is set to 1 to denote that the subscription requires recovery.

After the definition stage is complete, the replicate Replication Server builds the materialization queue for the subscription. You can monitor this activity with the admin who, sqm command.

Replication Server executes the rs_select_with_lock function to select subscription rows from the primary site. After the materialization queue is built, the replicate Replication Server sends an activation request to the primary Replication Server. This request is passed through the primary database via the rs_marker system function. When the primary Replication Server receives the activation request, the subscription is marked valid. All updates following the request are sent to the replicate database if they match the subscription.

The primary Replication Server sends the activation request back to the replicate Replication Server, where it is inserted into the DSI queue for the replicate database. When the DSI queue processes the request, the subscription status is changed to active.

The DSI thread also switches over to the materialization queue from its regular outbound queue for the site. The admin who, dsi command indicates which queue the DSI thread is processing. The materialization queue is applied to the replicate database. After the materialization queue is processed, the subscription is marked Valid and the materialization is complete.

If the replicate Replication Server runs out of stable queue segments:

- 1 Add a new partition to the replicate Replication Server.

- 2 If no partition is available, drop the subscription without purge.
- 3 Use nonatomic materialization.

Note To estimate how much space a subscription needs, refer to the *Replication Server Design Guide*.

If the replicate database log cannot hold all of the data for the subscription in a single transaction:

- 1 Drop the subscription without purge.
- 2 Use the incremental version of atomic materialization.

Table 5-3 describes solutions for atomic materialization problems based on the status returned by check subscription. The Replicate Status column shows the subscription status returned by check subscription at the replicate Replication Server. The Primary Status column shows the subscription status returned by check subscription at the primary Replication Server. The Subscription State column provides a detailed description of the subscription status. The Suggested Actions column provides the solutions.

Table 5-3: Atomic materialization problems

Replicate Status	Primary Status	Subscription State	Suggested Actions
Defined/ Pending	Invalid	Waiting for other subscriptions for the same replication definition and replicate database to be processed.	Check for other subscriptions being created and dropped for the same replication definition and replicate database. If there are no other subscriptions, wait five minutes.
Defined/ Recovering	Invalid	Cannot connect to the primary Replication Server to define the subscription.	Check the replicate Replication Server error log for messages. Make sure the user creating the subscription has the same login name and password at the primary Replication Server and the replicate Replication Server. Make sure the user has at least primary subscribe permission at the primary Replication Server.

Replicate Status	Primary Status	Subscription State	Suggested Actions
Defined/ Recovering	Defined	Cannot build the materialization queue.	<p>Verify that the primary data server is running.</p> <p>Make sure the user creating the subscription has the same login name and password at the primary data server and replicate Replication Server, select permission on the primary table, and execute permission for rs_marker.</p> <p>Investigate whether the replicate Replication Server is out of stable queue segments. Use the admin disk_space command.</p> <p>Use admin who, dsi and admin who, sqm to monitor the queues.</p>
Defined	Defined	Building materialization queue.	<p>Building the queue may take some time. You may need to wait until this process is complete.</p> <p>Use admin who, sqm to monitor materialization.</p> <p>Investigate whether the primary Replication Server is out of stable queue segments. Use the admin disk_space command.</p> <p>Check if rows are being selected using a holdlock at the primary database.</p>

Replicate Status	Primary Status	Subscription State	Suggested Actions
Qcomplete	Defined	Waiting for the activation request to get to the primary Replication Server. Materialization queue has been built.	Verify that the RepAgent for the primary database is running. Investigate whether the primary Replication Server is out of stable queue segments.
Qcomplete/ Recovering	Defined/ Valid	The replicate Replication Server stopped after the materialization queue was completed.	Wait. The status should return to Qcomplete at the replicate Replication Server.
Qcomplete	Activating	The activation request is being processed at the primary Replication Server.	Wait. The status at the primary Replication Server should change to valid.
Qcomplete	Valid	The subscription is complete at the primary Replication Server. Waiting for activation request or applying the materialization queue at the replicate Replication Server.	Check the route between the primary and the replicate Replication Server. Check the replicate Replication Server DSI thread for the replicate database. Investigate whether the replicate Replication Server ran out of queue segments. Use admin who, dsi and admin who, sqm to monitor the queues. If admin who, dsi shows that the DSI is processing the materialization queue, wait until the queue has been processed. The length of the wait depends upon the size of the queue.
Valid	Valid	Complete.	None.

Nonatomic materialization

To create a subscription using the nonatomic materialization method, execute the `create subscription` command with the `without holdlock` option at the replicate Replication Server. The subscription is saved in the replicate Replication Server RSSD. If there are no other subscription requests for the same replication definition and replicate database, the subscription is defined at the primary Replication Server.

After the definition stage is complete, the replicate Replication Server sends an activation request to the primary Replication Server. The replicate Replication Server immediately starts to build the materialization queue for the subscription. After the materialization queue is built, the subscription status becomes `Qcomplete`. The replicate Replication Server sends a validation request to the primary Replication Server through the primary database. The `admin who` command can be used to monitor this queue.

When the activation request arrives at the primary Replication Server, the subscription status becomes `Active`. All updates following the request are sent to the subscription.

The primary Replication Server sends the activation request back to the replicate Replication Server. When the `DSI` at the replicate Replication Server receives the request, the subscription status becomes `Active` and the transactions in the materialization queue are applied to the replicate database. If the materialization queue has not been built yet, the status returned by the `check subscription` command is `Active` and not `Qcomplete`. If the materialization queue has been built, the status is `Qcomplete` and `Active`. The `DSI` thread switches over to the materialization queue from its regular outbound queue for the site. The `admin who, dsi` command shows which queue the `DSI` thread is processing.

After the contents of the materialization queue are applied to the replicate database, the subscription status becomes `Materialized`.

While the replicate Replication Server is applying inserts from the materialization queue, the validation request is moving from the primary database log, through the `RepAgent`, to the primary Replication Server.

Once the validation request arrives at the primary Replication Server, the subscription status becomes Valid at the primary Replication Server and the request is forwarded to the replicate Replication Server. The subscription status becomes Valid at the replicate Replication Server after the materialization queue is applied and the validation request reaches the beginning of the DSI queue.

Warning! The subscription data may be inconsistent at the replicate database from the time the DSI thread starts applying the materialization queue until the subscription is validated at the replicate Replication Server. This is the result of not using a holdlock while selecting the subscription data from the primary database. Once the subscription status becomes valid, however, the replicate data is consistent with the primary data.

Table 5-4 describes solutions for nonatomic materialization problems based on the status returned by check subscription. The Replicate Status column shows the subscription status returned by check subscription at the replicate Replication Server. The Primary Status column shows the subscription status returned by check subscription at the primary Replication Server. The Subscription State column provides a detailed description of the subscription status. The Suggested Actions column provides the solutions.

Table 5-4: Nonatomic materialization problems

Replicate Status	Primary Status	Subscription State	Suggested Actions
Defined/ Pending	Invalid	The replicate Replication Server is waiting for other subscriptions for the same replication definition and replicate database to be created or dropped.	Look for other subscriptions being created or dropped for the same replication definition and replicate database. If there are no other subscriptions, wait for five minutes.
Defined/ Recovering	Invalid	The replicate Replication Server cannot connect to the primary Replication Server to define the subscription.	Check the replicate Replication Server error log for messages. Make sure the user creating the subscription has the same login name and password at the primary Replication Server and the replicate Replication Server. The user should have at least primary subscribe permission on the primary Replication Server.

Replicate Status	Primary Status	Subscription State	Suggested Actions
Defined	Defined	The replicate Replication Server is building the materialization queue. The primary Replication Server is waiting for the activation request.	Verify that the materialization queue is being built. Verify that the primary Replication Server is up and that the SQM, SQT, and DIST threads for the primary database are running.
Defined/ Recovering	Defined	The replicate Replication Server cannot build the materialization queue or cannot send the activation request to the primary Replication Server.	Verify that the primary data server is running. Make sure the user creating the subscription has the same login name and password at the primary data server, select permission on the primary table, and execute permission for rs_marker. Investigate whether the replicate Replication Server is out of stable queue segments. Verify that the primary Replication Server is running and that the SQM, SQT, and DIST threads for the primary database are running. Investigate whether the primary Replication Server is out of segments.
Defined	Active	The replicate Replication Server is building the materialization queue. The primary Replication Server has received the activation request and sent it to the replicate Replication Server.	Verify that the materialization queue is being built. Verify that the connection between the primary and replicate Replication Server is up. Verify that the DSI thread for the replicate database is running.
Qcomplete	Defined	Waiting for the activation request to be processed by the distributor at the primary Replication Server.	Investigate whether the primary Replication Server is out of stable queue segments. Verify that the primary Replication Server is up and the SQM, SQT, and DIST threads for the primary database are running.

Replicate Status	Primary Status	Subscription State	Suggested Actions
Qcomplete/ Recovering	Defined	The replicate Replication Server was recycled. Waiting for the activation request to be processed by the distributor at the primary Replication Server.	The state at the replicate Replication Server should soon change to Qcomplete.
Qcomplete	Active	The primary Replication Server received and processed the activation request. Now, this Replication Server is waiting for the validation request. The replicate Replication Server is waiting for the activation request.	Check the route from the primary Replication Server to the replicate Replication Server. Verify that the DSI thread for the replicate database is running. Investigate whether the replicate Replication Server has run out of queue segments.
Qcomplete	Valid	The subscription is done at primary Replication Server. The replicate Replication Server is waiting for the activation request.	Check the route between the primary Replication Server and the replicate Replication Server. Check the DSI to the replicate database. Investigate whether the replicate Replication Server has run out of queue segments.
Active and not Qcomplete	Active	The replicate Replication Server is applying and building the materialization queue at same time. The primary Replication Server is waiting for the validation request. This request is not sent until the queue is complete at the replicate Replication Server.	Verify that the DSI thread is processing the materialization queue. Investigate whether the replicate Replication Server is out of stable queue segments. You can monitor the materialization queue activity using <code>admin who</code> , <code>sqm</code> .

Replicate Status	Primary Status	Subscription State	Suggested Actions
Qcomplete and Active	Active	<p>The primary Replication Server is waiting for the validation request.</p> <p>The replicate Replication Server is applying the materialization queue.</p>	<p>Verify that the DSI is processing the materialization queue.</p> <p>You can monitor the materialization queue activity using <code>admin who</code>, <code>sqm</code>.</p> <p>Verify that the primary data server, the RepAgent for the primary database and the RepAgent User thread in the primary Replication Server are running.</p> <p>Verify that the primary Replication Server is running and that the SQM, SQT, and DIST threads for the primary database are running.</p> <p>Investigate whether the primary Replication Server is out of segments.</p>
Qcomplete and Active	Valid	<p>The primary Replication Server received the validation request.</p> <p>The replicate Replication Server is applying the materialization queue.</p>	<p>Verify that the DSI thread is processing the materialization queue.</p> <p>You can monitor the materialization queue activity using <code>admin who</code>, <code>sqm</code>.</p> <p>Check the route between the primary Replication Server and the replicate Replication Server.</p>
Materialized	Active	<p>The primary Replication Server is waiting for the validation request.</p> <p>The replicate Replication Server has finished applying the materialization queue.</p>	<p>Investigate whether the primary Replication Server is out of segments.</p> <p>Verify that the primary data server, the RepAgent for the primary database, and the RepAgent User thread at the primary Replication Server are running.</p> <p>Verify that the SQM, SQT, and DIST threads for the primary database are running.</p>

Replicate Status	Primary Status	Subscription State	Suggested Actions
Materialized	Valid	The primary Replication Server received the validation request. The replicate Replication Server has finished applying the materialization queue.	Check the route between the primary Replication Server and the replicate Replication Server. Verify that the DSI for the replicate database is running. Investigate whether the replicate Replication Server has run out of stable queue segments.
Valid	Valid	Complete.	None.

Bulk materialization

To use bulk materialization, execute:

- define subscription to add the subscription to the RSSD for the primary and replicate Replication Servers,
- activate subscription to add the subscription to the Subscription Resolution Engine (SRE), and
- validate subscription to verify that the data at the replicate data is consistent with the primary data.

Defining the subscription

Table 5-5 describes solutions for bulk materialization problems that occur during the definition phase based on the status returned by check subscription. The Replicate Status column shows the subscription status returned by check subscription at the replicate Replication Server. The Primary Status column shows the subscription status returned by check subscription at the primary Replication Server. The Subscription State column provides a detailed description of the subscription status. The Suggested Actions column provides the solutions.

Table 5-5: Bulk materialization problems—definition phase

Replicate Status	Primary Status	Subscription State	Suggested Actions
Defined/ Pending	Invalid	Waiting for other subscriptions for the same replication definition and replicate database.	Look for other subscriptions for the same replication definition and database. If there are no other subscriptions, wait for five minutes.
Defined/ Recovering	Invalid	Cannot connect to the primary Replication Server to define the subscription.	Check the replicate Replication Server error log for messages. Make sure the user creating the subscription has the same login name and password at the primary Replication Server and the replicate Replication Server.
Defined	Defined	Definition cycle is complete.	Activate the subscription.

Activating the subscription

After the subscription is defined, use `activate subscription` to set the subscription status to `Activating` at the replicate Replication Server and to send the activation request to the primary Replication Server.

When the primary Replication Server receives the activation request, it changes the subscription status to `Active`, sends the activation request back to the replicate Replication Server, and begins sending updates for the subscription to the replicate Replication Server. When the replicate Replication Server receives the activation request, it changes the subscription status to `Active` at the replicate Replication Server.

If the `with suspension` option was specified for the `activate subscription` command, the DSI thread is suspended when the activation request is received at the replicate Replication Server. At this stage, you load the replicate database. For more information, see the *Replication Server Reference Manual*. After loading, execute the `resume connection` command to continue applying transactions from the DSI.

Table 5-6 describes solutions for bulk materialization problems that occur during the activation phase based on the status returned by check subscription. The Replicate Status column shows the subscription status returned by check subscription at the replicate Replication Server. The Primary Status column shows the subscription status returned by check subscription at the primary Replication Server. The Subscription State column provides a detailed description of the subscription status. The Suggested Actions column provides the solutions.

Table 5-6: Bulk materialization problems—activation phase

Replicate Status	Primary Status	Subscription State	Suggested Actions
Activating/ Recovering	Defined	Cannot send the activation request to the primary Replication Server.	<p>Verify that the primary Replication Server is running and the SQM, SQT, and DIST threads for the primary database are running.</p> <p>Check the error log in the replicate Replication Server.</p> <p>Investigate whether the primary Replication Server has run out of segments.</p>
Activating	Active	<p>The primary Replication Server received the activation request and returned it to the replicate Replication Server.</p> <p>The replicate Replication Server is waiting for the activation request.</p>	<p>Verify the connection between the primary Replication Server and the replicate Replication Server.</p> <p>Verify that the DSI thread for the replicate database is running.</p>
Active	Active	The activation stage is complete.	<p>Validate the subscription.</p> <p>If you specified the with suspension option for the activate subscription command, you may now load the replicate database.</p>

Validating the subscription

After the subscription is activated, complete bulk materialization by using validate subscription at the replicate Replication Server. The subscription status is changed to Validating. The replicate Replication Server forwards the validation request to the primary Replication Server.

When the primary Replication Server receives the validation request, it changes the status at the primary to Valid and returns the validation request to the replicate Replication Server. When the replicate Replication Server receives the validation request, the subscription status becomes Valid at both sites, and the bulk materialization is complete.

Table 5-7 describes solutions for bulk materialization problems that occur during the validation phase based on the status returned by check subscription. The Replicate Status column shows the subscription status returned by check subscription at the replicate Replication Server. The Primary Status column shows the subscription status returned by check subscription at the primary Replication Server. The Subscription State column provides a detailed description of the subscription status. The Suggested Actions column provides the solutions.

Table 5-7: Bulk materialization problems—validation phase

Replicate Status	Primary Status	Subscription State	Suggested Actions
Validating/ Recovering	Active	Cannot send the validation request to the primary Replication Server.	Verify that the primary Replication Server is running.
Validating	Valid	The primary Replication Server received the validation request. The replicate Replication Server is waiting for the validation request.	Verify the route between the primary and replicate Replication Server. Verify the DSI is running. Verify that the replicate Replication Server has sufficient queue segments.
Valid	Valid	Complete.	None.

Dematerialization problems

When you execute drop subscription at the replicate Replication Server, the subscription status becomes Dematerializing in the replicate Replication Server `rs_subscriptions` table. If there are any other materializing or dematerializing subscriptions for the same replication definition and replicate database, the recovering column in the `rs_subscriptions` table is set to 2 to indicate that the dematerialization request is pending. The check subscription command indicates that the subscription has a Dematerializing or Pending status at the replicate Replication Server.

Subscription creation and drop requests are processed one at a time in the order entered. The `request_date` column in `rs_subscriptions` stores the time a request was entered.

There are two methods for dematerializing subscriptions:

- `with purge` – deletes the subscription data from the replicate database.
- `without purge` or `bulk` – does not delete the subscription data from the replicate database.

Both methods go through two stages:

- `Dematerialization` – stops sending updates for the subscription to the replicate database and, optionally, deletes the subscription data from the replicate database.
- `Removal` – deletes the subscription from the system tables of the primary and the replicate Replication Server.

with purge dematerialization

The `dematerialize with purge` method is invoked by using the `with purge` option of the `drop subscription` command. The subscription status is set to `Dematerializing` at the replicate Replication Server and a drop request is forwarded to the primary Replication Server.

When the primary Replication Server gets the drop request, it stops sending updates for the subscription to the replicate Replication Server. The subscription status is changed to `Dematerializing` at the primary Replication Server and a drop request is returned to the replicate Replication Server.

When the replicate Replication Server gets the drop request, it creates a dematerialization queue and starts a dematerialization thread to fill the queue. The dematerialization thread selects subscription data from the replicate database and puts a row delete operation into the dematerialization queue for each row that does not belong to any other active subscription.

While the dematerialization thread is writing to the dematerialization queue, the DSI thread switches to the dematerialization queue and begins applying the deletes to the replicate database.

If the dematerialization thread encounters an error, the entire dematerialization process restarts and the same error may recur. To avoid this problem, suspend the connection to the replicate database. Then resume it using the `skip transaction` option. This resumes the connection with the second transaction in the queue. The first transaction is written to the exceptions log.

When dematerialization has completed, the subscription status at the replicate Replication Server is changed to Removing. The replicate Replication Server logs in to the primary Replication Server and asks it to delete the subscription from its system tables. When that request succeeds, the replicate Replication Server removes the subscription from its own system tables and the dematerialization process is complete.

If the replicate Replication Server runs out of stable queue segments while dematerializing using the `with purge` option:

- 1 Add a new partition to the replicate Replication Server. If no partition is available, consider using bulk dematerialization.
- 2 If you have difficulty deleting subscription data, drop the subscription using the `without purge` option.
- 3 If you have difficulties because of permissions, a user with the appropriate permission should issue the `drop subscription` command.

Table 5-8 describes solutions for dematerialization problems based on the status returned by `check subscription`. The Replicate Status column shows the subscription status returned by `check subscription` at the replicate Replication Server. The Primary Status column shows the subscription status returned by `check subscription` at the primary Replication Server. The Subscription State column provides a detailed description of the subscription status. The Suggested Actions column provides the solutions.

Regardless of the problem you are having, you should first check the error logs for all of the servers involved for specific information. Also check that the user who is dropping the subscription has the same login name and password at the replicate and primary Replication Servers, and that the maintenance user for the replicate database has `select`, `delete`, and `update` permissions on the replicate table.

Table 5-8: Dematerialization problems—with purge option

Replicate Status	Primary Status	Subscription State	Suggested Actions
Dematerializing/ Pending		Waiting for other subscriptions for the same replication definition and replicate database.	Look for other subscriptions being created or dropped for the same replication definition and database. If no other subscription operations are in process, wait for five minutes.

Replicate Status	Primary Status	Subscription State	Suggested Actions
Dematerializing/ Recovering		Cannot connect to the primary Replication Server to drop the subscription.	<p>Check replicate Replication Server error log for messages.</p> <p>If the user dropping the subscription does not have the same login and password at the primary and replicate Replication Servers, then:</p> <ul style="list-style-type: none"> • Provide the user with a Replication Server account that has the same login and password at the primary and replicate, or • Have a different user with the appropriate permissions drop the subscription. <p>Also, the user must have at least primary subscribe privileges at the primary Replication Server.</p>
Dematerializing		The primary Replication Server is waiting for the drop request.	<p>Investigate whether the primary Replication Server is out of queue segments.</p> <p>Verify that the primary Replication Server is running and that the SQM, SQT and DIST threads for the primary database are running.</p>
Dematerializing	Dematerializing	<p>The primary Replication Server processed the drop request and sent it to the replicate Replication Server.</p> <p>The replicate Replication Server is waiting for the drop request.</p>	<p>Check the route between the primary Replication Server and the replicate Replication Server.</p> <p>Check the DSI thread for the replicate database.</p> <p>Investigate whether the replicate Replication Server has run out of queue segments.</p>
Dematerializing/ Recovering	Dematerializing	<p>The primary Replication Server processed the drop request and sent it to the replicate Replication Server.</p> <p>The replicate Replication Server failed to create and process the dematerialization queue.</p>	<p>Investigate whether the replicate Replication Server has run out of queue segments.</p> <p>Verify that the maintenance user has select privileges on the replicate table.</p> <p>Check the DSI thread for the replicate database.</p>

Replicate Status	Primary Status	Subscription State	Suggested Actions
Removing/ Recovering	Dematerializing	The subscription has dematerialized at the replicate Replication Server, but the replicate Replication Server could not log in to primary Replication Server to remove the subscription from the system tables.	Verify that the primary Replication Server is running. If the user dropping the subscription does not have the same login and password at the primary and replicate Replication Servers, then: <ul style="list-style-type: none"> • Provide the user with a Replication Server account that has the same login and password at the primary and replicate, or • Have a different user with the appropriate permissions drop the subscription.
Removing	Dematerializing	The primary Replication Server is deleting the subscription. The replicate Replication Server is waiting for it to finish.	Wait.
Removing	Invalid	The subscription has been removed from primary Replication Server. The replicate Replication Server will remove the subscription next.	Wait.
Invalid	Invalid	The subscription has been dropped.	None.

Bulk dematerialization

Bulk dematerialization is invoked using the `without purge` option of the `drop subscription` command. The subscription status becomes `Dematerializing` at the replicate Replication Server and a drop request is forwarded to the primary Replication Server.

When the primary Replication Server receives the drop request, it stops sending updates for the subscription to the replicate Replication Server. The subscription status becomes `Dematerializing` at the primary Replication Server and a drop request is returned to the replicate Replication Server.

When the replicate Replication Server receives the drop request, the subscription status is changed to Removing at the replicate. The replicate Replication Server logs in to the primary Replication Server and requests that it delete the subscription from its system tables. When that request has succeeded, the replicate Replication Server removes the subscription from its own system tables and the dematerialization is complete.

Table 5-9 describes solutions for bulk materialization problems based on the status returned by check subscription. The Replicate Status column shows the subscription status returned by check subscription at the replicate Replication Server. The Primary Status column shows the subscription status returned by check subscription at the primary Replication Server. The Subscription State column provides a detailed description of the subscription status. The Suggested Actions column provides the solutions.

Table 5-9: Materialization problems—without purge option

Replicate Status	Primary Status	Subscription State	Suggested Actions
Dematerializing/ Pending		Waiting for other subscription requests for the same replication definition and replicate database.	Check for other subscription being created or dropped for the same replication definition and database. If there are no other subscriptions, wait for five minutes.
Dematerializing/ Recovering		Cannot connect to the primary Replication Server to drop the subscription.	Check the replicate Replication Server error log for messages. Make sure the user who created the subscription has the same login name and password at the primary Replication Server and the replicate Replication Server. The user should also have at least primary subscribe privileges.
Dematerializing		The primary Replication Server is waiting for the drop request.	Determine whether the primary Replication Server has run out of queue segments. Verify that the primary Replication Server is up and that the SQM, SQT, and DIST threads for the primary database are running.

Replicate Status	Primary Status	Subscription State	Suggested Actions
Dematerializing	Dematerializing	The primary Replication Server processed the drop request and sent it to the replicate Replication Server. The replicate Replication Server is waiting for the drop request.	Check the route between the primary Replication Server and the replicate Replication Server. Check the DSI thread for the replicate database. Determine whether the replicate Replication Server ran out of queue segments.
Dematerializing/ Recovering	Dematerializing	The primary Replication Server processed the drop request and returned it to the replicate Replication Server. The replicate Replication Server abnormally terminated.	Wait for the subscription daemon to reset the recovering flag.
Removing/ Recovering	Dematerializing	The replicate Replication Server could not log in to the primary Replication Server to delete the subscription from the system tables.	Verify that the primary Replication Server is running.
Removing	Dematerializing	The primary Replication Server is deleting the subscription from the system tables. The replicate Replication Server is waiting for the primary Replication Server to finish.	Wait.
Removing	Invalid	The subscription is removed from the primary Replication Server. The replicate Replication Server will remove the subscription next.	Wait.
Invalid	Invalid	The subscription has been dropped.	None.

Replication Server Interface Problems

This chapter describes how to fix Replication Server interface problems.

Name	Page
Overview	103
Connection errors	104

If you cannot find the solution in this chapter

If an error message is displayed in an error log and you cannot find it in this chapter, refer to:

- “Analyzing error logs” on page 9, or
- Chapter 3, “Common Errors”

If an error message is not displayed, refer to Chapter 1, “Identifying the Problem”

Overview

The RSI allows two Replication Servers to exchange transactions across a route. The RSI consists of:

- A stable queue and thread at the primary Replication Server. The thread reads the stable queue and writes to the network.
- A thread at the replicate Replication Server. This thread reads from the network and writes into outbound stable queues.

There are two basic types of RSI problems:

- Those occurring when Replication Server attempts to connect to another Replication Server, and
- Those that occur when the locator is invalid.

Although many errors may occur when a primary Replication Server loses its connection to the replicate Replication Server, none of these errors should cause a loss of data or cause the RSI to be suspended. The primary thread continuously attempts to connect to the replicate Replication Server until the connection is reestablished.

Losing the connection terminates the RSI thread at the replicate Replication Server. The thread at a replicate Replication Server exists only while a connection exists. Executing `admin who` in the replicate Replication Server shows that the primary Replication Server is no longer logged in as an RSI source. When the primary Replication Server reestablishes the connection to the replicate Replication Server, data is replicated across the route to the replicate Replication Server.

An RSI locator identifies the last message that the replicate Replication Server stable queue received from the primary Replication Server. If the RSI locators at the primary and replicate Replication Servers are not identical, then the primary Replication Server might delete messages from its stable queue at an incorrect location.

Connection errors

These errors can occur when two Replication Servers attempt to connect through the RSI:

- Incorrect RSI user login name or password
- Incorrect user permissions at replicate Replication Server

Incorrect RSI user login name or password

The primary Replication Server logs in to the replicate Replication Server using the RSI login name and password. The RSI user login name and password are created by `rs_init` and are also required when you create a route to the replicate Replication Server.

If there is a problem with the login name and password at the replicate Replication Server, the primary Replication Server logs one of these errors:

```
I. 2006/06/23 14:29:43. RSI: Trying to connect to
'eastRS'.
```

```

E. 2006/06/23 14:29:43. ERROR #1028 RSI(eastRS) -
seful/cm.c(3463) Message from server: Message: 14021,
State 0, Severity 12 -- 'Invalid login attempted by user
'Rep_Server_rsi'.
E. 2006/06/23 14:29:44. ERROR #1027 RSI(eastRS)-
seful/cm.c(3463) Open Client Client-Library error:
Error: 67175468, Severity 4 -- 'ct_connect(): protocol
specific layer: external error: The attempt to connect
to the server failed.'.
E. 2006/06/23 14:29:44. ERROR #13045 RSI(eastRS) -
seful/cm.c(3467) Failed to connect to server 'eastRS'
as user 'REP_Server_rsi'. See CT-Lib and/or server error
messages for more information.
E. 2006/06/23 14:29:44. ERROR #4044 RSI(eastRS) -
i/rsiint.c(329) RSI for 'eastRS': Shutting down due
to an exception.

```

To repair this problem:

- 1 Retrieve the RSI user login name and password from the RSSD of the primary Replication Server by using `isql` to execute this query on the RSSD:

```

select username, password
  from rs_maintusers, rs_sites
  where name = remote_RS_name and destid = id

```

remote_RS_name is the destination Replication Server name.

id is the destination Replication Server's site ID.

If you have password encryption, you cannot access the password by executing a query.

- 2 Perform one of these actions:
 - Use the `alter route` command to change the password for the primary Replication Server to the RSI user password, which you retrieved in the previous step.
 - Use the `alter user` command to change the password for the login name at the replicate Replication Server. Then use the `resume route` command to resume the route.

Incorrect user permissions at replicate Replication Server

The login must also have connect source or sa permission on the replicate Replication Server. If the login lacks the appropriate permission, an error message is sent to the primary Replication Server error log.

To repair this problem:

- 1 Grant connect source permission to the login at the replicate Replication Server.
- 2 Resume the route at the primary Replication Server.

Invalid RSI locator

An RSI locator points to the last message that the replicate Replication Server stable queue received from the primary Replication Server. The RSI uses the RSI locator in the `rs_locator` system table to delete messages in the stable queues. The replicate Replication Server updates the RSI locator in the `rs_locator` system table with the last messages that the replicate Replication Server added to its stable queues. Periodically, the primary Replication Server requests the RSI locator from `rs_locator` at the replicate Replication Server. The primary Replication Server deletes its stable queue messages up to and including the message identified by the RSI locator—these are the messages that the replicate Replication Server has already received.

Note The RSI locator is different from the locator that is used with database log truncation. See “Resetting the locator” on page 130 for more information about the database log truncation locator.

If the RSI locators at the primary and replicate Replication Servers are not identical, then the primary Replication Server might delete messages from its stable queue at an incorrect location and invalidate the RSI locator at the replicate Replication Server.

RSI locators must be identical. If they are not identical:

- 1 Make sure that the replicate Replication Server is not running.
- 2 Set the replicate Replication Server RSI locator to “0” by executing this command against the replicate Replication Server RSSD:

```
update rs_locator set locator=0x0
  where sender = primary_replication_server
```

RepAgent Problems

This chapter describes how to troubleshoot RepAgent problems.

Name	Page
Overview	107
Problems when starting the RepAgent	108
Errors from the Replication Server	109
Errors from the Adaptive Server	112

If you cannot find the solution in this chapter

If an error message is displayed in an error log and you cannot find it in this chapter, refer to:

- “Analyzing error logs” on page 9, or
- Chapter 3, “Common Errors”

If an error message is not displayed, refer to Chapter 1, “Identifying the Problem”

Overview

The RepAgent retrieve the data for primary objects from the Adaptive Server log and convert the log record information into Log Transfer Language (LTL) commands. These commands are sent to the primary Replication Server for distribution and replication. The RepAgent also coordinates database log truncation with the Adaptive Server and primary Replication Server.

Errors can be caused by problems with the Adaptive Server, Replication Server, or RepAgent. The RepAgent records error messages in the Adaptive Server error log. These error messages identify the server that caused the error so that you can diagnose and correct the problem.

This chapter describes these problems:

- Incorrect logins and passwords
- Incorrect permissions

- Phantom connections
- Inconsistent replication definition setup
- Conflicting text, unitext, or image column status
- Invalid truncation page
- Not enough Adaptive Server alarm resources

Note For more information about how the RepAgent processes errors, see the *Replication Server Administration Guide Volume 1*.

Problems when starting the RepAgent

These errors can occur when the RepAgent starts up:

- Invalid login to the Replication Server or Adaptive Server
- Invalid permissions for the Replication Server

Invalid login

The RepAgent requires only a login name and password for the Replication Server user. The user must have connect source permission in the Replication Server.

When the RepAgent fails to log in to the Replication Server, RepAgent retries the login and error 9216 is displayed in the Adaptive Server log. If RepAgent continues to fail logging in to the Replication Server, RepAgent error 9214 is displayed in the Adaptive Server error log.

Invalid permissions

The RepAgent user must have connect source permission for the Replication Server; otherwise, RepAgent error 9211 is displayed in the Adaptive Server error log:

```
00:00000:00024:2006/06/23 14:44:45.12 server
RepAgent(4): Received the following error message from
```

```
the Replication Server: Msg 37024. CONNECT SOURCE
permission is required to execute command..
00:00000:00024:2006/06/23 14:44:45.24 server Error:
9261, Severity: 20, State: 0
00:00000:00024:2006/06/23 14:44:45.27 server
RepAgent(4): This Rep Agent Thread is aborting due to
an unrecoverable communications or Replication Server
error.
00:00000:00024:2006/06/23 14:44:45.27 server Rep Agent
Thread for database 'westRS_RSSD' (dbid = 4) terminated
abnormally with error. (major 92, minor 61)
```

Errors from the Replication Server

The RepAgent treats most Replication Server normalization errors, which have numbers between 32000 and 32999, as recoverable. The RepAgent logs the error and continues processing.

All other Replication Server errors, except network failures, are treated as fatal by the RepAgent. For example, an incorrect Replication Server login name and password in the configuration file is a fatal error. The RepAgent disconnects from Replication Server and terminates. After you have corrected the problem, restart the RepAgent.

Normalization errors

Normalization errors result from transient inconsistencies in the setup of replication objects. For example, if a table has been marked replicated with the `sp_setreplicate` system procedure, but the replication definition has not yet been created for the table, the RepAgent retrieves log records for an object that is not yet known to the Replication Server. Replication Server error 32032 is displayed in the Replication Server error log:

```
Message: 32032, 'No table with name 'foo' is defined for
database with id MY_PDS.MY_PDB1.
```

Correct this problem by creating the replication definition for the table.

A normalization error can also occur if the replicated stored procedure is declared in the Replication Server for function replication but is marked in the Adaptive Server for table replication. Replication Server error 32047 is displayed in the Replication Server error log:

```
Message from server: Message: 32047, State 0, Severity
12 -- 'Function 't2' is associated with a function
replication definition, but an invoking stored
procedure was marked for a table replication
definition.'
```

See the *Replication Server Administration Guide Volume 1* for instructions on setting up replication definitions.

Treating normalization errors as recoverable

When you treat normalization errors as recoverable, errors may still occur even when a replication definition is created and a subscription is successfully materialized. For example, an update may be lost, causing the replicate data to be inconsistent with the primary data. This kind of error should occur only during the initial setup of a replication system. The system administrator should monitor the error logs closely and correct errors as they are identified. To reconcile subscriptions, refer to Chapter 5, “Subscription Problems” Normalization errors may also occur with replicated stored procedures.

Replication status inconsistency of *text*, *unitext*, or *image* columns

Another normalization error occurs when there are inconsistencies in the replication status of text, unitext, or image columns between the Adaptive Server database and the replication definition. Replication Server error 32046 is displayed in the Adaptive Server error log (for RepAgent):

```
Message: 32046 -- 'The status of column '%s' in repdef
is inconsistent with that of the LTL command.'
```

The replication status for text, unitext, and image columns in the Adaptive Server database is carried in the modification LTL commands that the RepAgent sends to the Replication Server. If the status of a text, unitext, or image column is not the same in the Adaptive Server database and in the replication definition, then Replication Server detects the inconsistency when the modification is being replicated, and the RepAgent shuts down.

If a text, unitext, or image column has a status of `do_not_replicate` at the Adaptive Server database and the replication definition includes that column for replication, processing continues and the Replication Server sends the modifications to the replicate database without the text, unitext, or image data. The Replication Server records a warning message in the Adaptive Server error log (for the RepAgent).

Use the following procedures to resolve replication status inconsistencies for text, unitext, or image column status and to resume operations after the RepAgent has shut down.

When `replicate_if_changed` conflicts with `always_replicate`

When the RepAgent shuts down because a text or image column has a status of `replicate_if_changed` at the Adaptive Server database and `always_replicate` in the replication definition, you must change the replication status so that they match, for example, either:

- Replicate text, unitext, or image columns only when their values change, or
- Always replicate text, unitext, or image columns.

To replicate text, unitext, or image columns only when their values change:

- 1 Execute the alter replication definition command at the primary Replication Server and change the status of the text, unitext, or image columns to `replicate_if_changed`. Wait for the modified replication definition to arrive at the replicate sites.
- 2 Restart the RepAgent.

To always replicate text, unitext, or image columns:

- 1 Stop updates at the primary table.
- 2 Execute the alter replication definition command at the primary Replication Server, and change the status of the text, unitext, or image columns to `replicate_if_changed`. Wait for the modified replication definition to arrive at the replicate sites.
- 3 Restart the RepAgent to let transactions with a `replicate_if_changed` status finish processing.
- 4 Execute the `sp_setrepcol` system procedure at the Adaptive Server and change the status to `always_replicate`.

- 5 Execute alter replication definition at the primary Replication Server and change the status of the text, unitext, or image columns to `always_replicate`. Wait for the modified replication definition to be replicated to the replicate sites.
- 6 Resume updates to the primary table.

When `do_not_replicate` columns are included in the replication definition

When the Replication Server reports that the status of a text or image column is `do_not_replicate` at the Adaptive Server database and the replication definition includes that column for replication, you must change the replication status to either:

- Replicate text, unitext, or image columns, or
- Not replicate text, unitext, or image columns.

If you want to replicate text, unitext, or image columns:

- 1 Execute the `sp_setrepcol` system procedure at the Adaptive Server database and change the status of the text, unitext, or image column to `always_replicate` or `replicate_if_changed`. It should match the status in the replication definition.
- 2 Wait for subsequent transactions that modify the text, unitext, or image column to be processed by the Replication Server.
- 3 Consider correcting any inconsistencies with the `rs_subcmp` program.

If you do not want to replicate text, unitext, or image columns:

- 1 Stop updates to the primary table.
- 2 Drop subscriptions to the replication definition.
- 3 Drop the replication definition.
- 4 Re-create the replication definition without the text, unitext, or image columns, and re-create subscriptions.
- 5 Resume updates to the primary table.

Errors from the Adaptive Server

This section describes errors from Adaptive Server that are caused by:

- An invalid truncation page, or
- Running out of Adaptive Server alarm resources.

The RepAgent does not use the Adaptive Server log scan thread; it will not encounter any errors related to the Adaptive Server log scan thread.

Dropping primary objects

Dropping primary tables or stored procedures from an Adaptive Server affects data replication and should be done with caution.

If updates are made to the primary table and the table is dropped from the Adaptive Server, Adaptive Server error 9104 is displayed in the Adaptive Server error log (for the RepAgent) :

```
Message: 9104 'Cannot identify the object on the INSERT
log record for database 'MY_PDB1', XACT ID Rid pageid =
0xa0f; row num = 0x5, RID Rid pageid = 0xa0f; row num =
0x6. Information associated with the INSERT log record
is not replicated.
```

The log record for a replicated data operation references another log page, called the oampage, that has information about the replicated object. Since the table was dropped, the oampage is deallocated and the data cannot be replicated. If the oampage is assigned to a newly created object, the Adaptive Server may associate log records for the dropped object with the new object.

In the following example, the oampage for foo may be allocated to foo_bar after foo is dropped:

```
update table foo
drop table foo
create table foo_bar
update foo_bar
```

In the preceding example, Adaptive Server sends the first update record as an update to foo_bar instead of foo. If the new table has a much larger row size, then an Adaptive Server segmentation fault may occur.

Dropping or recreating replicated stored procedures leads to unpredictable results; schema alterations have the same effect. Make these type of changes *only* after the RepAgent has processed all the records for the stored procedure.

Errors may also occur at the Replication Server, check the Replication Server for error messages.

Invalid truncation page

The RepAgent executes the dbcc settrunc command on the Adaptive Server to set the secondary truncation point in the database log. The secondary truncation point identifies the page that contains the begin transaction statement of the last transaction that was sent to the Replication Server. If the RepAgent specifies an invalid page to the Adaptive Server, an error is recorded and the RepAgent aborts. You may see Adaptive Server error 4213:

```
Message: 4213, State: 1, Severity: 16 -- 'Page 2561 in
database 'MY_DB' cannot be used as a truncation page
because the page is not allocated.'
```

The rs_locator table in the Replication Server system database tells the RepAgent where to set the secondary truncation point in the Adaptive Server log. This error is expected if the truncation state is set to ignore and the log is truncated. The RepAgent then restarts without resetting rs_locator.

To solve this problem, set rs_locator to zero and restart the RepAgent. For more information about log truncation, refer to “Resetting the locator” on page 130.

This chapter describes problems you may encounter in the DSI.

Name	Page
Overview	115
Listing databases and using admin who and admin who, dsi	116
Troubleshooting procedure	118
DSI down or suspended	119
DSI active	124

If you cannot find the solution in this chapter

If an error message is displayed in an error log and you cannot find it in this chapter, refer to:

- “Analyzing error logs” on page 9, or
- Chapter 3, “Common Errors”

If an error message is not displayed, refer to Chapter 1, “Identifying the Problem”

Overview

The DSI applies transactions from a Replication Server stable queue to a data server, which can be an Adaptive Server or another data server with an interface to the Replication Server. Before a transaction is applied, Replication Server uses function strings to convert the commands to the format specified for the data server.

If the data server is not receiving updates, the DSI may be suspended or down. A down or suspended DSI can be caused by:

- Incorrect login or permissions
- A data server error
- A Replication Server error

A suspended DSI always results in an error message being sent to the Replication Server error log. You can analyze this error message to solve the problem. After correcting the error, restart the DSI.

If the DSI is active, unique transactions might be incorrectly resolved as duplicates, or the transactions might be failing on the data server. You must manually resolve the data inconsistency between the primary and replicate databases or reapply the transactions.

To determine the exact cause of the problem, use a select command, `admin who`, and `admin who, dsi`.

Note These problems also apply to parallel DSI threads. For more information specific to parallel DSI threads, see *Replication Server Administration Guide Volume 2*.

Listing databases and using *admin who* and *admin who, dsi*

To investigate the DSI, find out which databases are controlled by the Replication Server and check the DSI status for all database connections on a Replication Server. Use this information to determine the cause of the problem.

Listing databases controlled by a Replication Server

To list the databases controlled by a Replication Server, log in to the system Adaptive Server, change to the RSSD, and enter:

```
select dsname, dbname from rs_databases, rs_sites
where prsid = id and name = replication_server
```

Checking the DSI status for a database connection

The command `admin who` displays the states of all threads in the Replication Server, including DSI scheduler and executor threads. In the output of `admin who`, the DSI scheduler threads have the name “DSI” and DSI executor threads have the name “DSI EXEC.” If the DSI is suspended when Replication Server starts up, the output shows only one DSI executor thread, even if additional DSI executor threads are configured.

The command `admin who, dsi` shows the states of all running DSI scheduler threads and provides configuration values and other information about them. If a DSI scheduler thread exists for a database but does not appear in the output of `admin who, dsi`, use the `resume connection` command to restart the data server interface for the database. For a complete description of `admin who, dsi` output, see the *Replication Server Reference Manual*.

The states of DSI scheduler threads in the output of `admin who` and `admin who, dsi` are:

- **Active** – the thread is starting, restarting after an internal error, or logging an exception to the RSSD.
- **Awaiting Command** – the thread is waiting for a transaction to become available in the stable queue for the database. The queue contains no complete transaction at this moment.
- **Awaiting Wakeup** – the thread is sleeping before it restarts. It sleeps for two minutes after an error that can be retried. During the two-minute interval, the cause of the error may disappear or the System Administrator may correct the problem. If the problem is corrected, the thread can restart without error after the sleep interval.

A failed transaction is retried when it causes a data server error that you assigned the `retry_stop` or `retry_log` option using the `assign action` command.

- **Awaiting Message** – the thread has dispatched transactions to the DSI executor threads and is waiting for them to complete.
- **Suspended** – the DSI connection has been suspended by a user command, by an error, or by a `drop subscription` or `activate subscription` command with the `with suspension` clause. This state appears only in `admin who` output.
- **Down** – the thread has not been started. This state appears only in `admin who` output.

The states of DSI executor threads in the output of `admin who` are:

- Active – the thread is starting, executing a transaction at the data server, logging an exception to the RSSD, or restarting after an internal error.
- Awaiting Message – the thread is processing a transaction and is waiting for another thread to complete processing its transaction, or the thread is waiting to receive another statement from the Stable Queue Transaction interface (SQT).
- Awaiting Command – the thread is waiting to receive another transaction from the DSI scheduler thread.
- Down – the meaning of this state depends on the state of the DSI scheduler thread as follows:
 - If the DSI scheduler thread is Down, the DSI executor thread connection was suspended when the Replication Server was started, and the connection has not been resumed.
 - If the DSI scheduler thread is Active or Awaiting Wakeup, the DSI executor thread connection is recovering from a retryable error and is starting or restarting.
- Suspended – the connection has been suspended by a user command, an error, or a drop subscription or activate subscription command using the with suspension clause.

A failed transaction does not cause the DSI connection to be suspended and is retried if the failed transaction caused a data server error that you assigned the `retry_stop` or `retry_log` option using the `assign` action command. For more information about the `assign` action command, see Chapter 3, “Replication Server Commands,” of the *Replication Server Reference Manual*.

Troubleshooting procedure

If changes made to a primary database are not arriving at a replicate database that has active subscriptions, it is possible that the DSI for the replicate database is suspended.

❖ **Troubleshooting the DSI for the replicate database**

- 1 Log in to the Replication Server that controls the replicate database and execute the `admin who, dsi` command. This returns one entry for each database with replicated data. To learn how to use `admin who, dsi`, see “Listing databases and using `admin who` and `admin who, dsi`” on page 116.
- 2 If a database does not have an entry, check the Replication Server error log to see if the DSI for the database was not started or was suspended. These are samples of error messages that are displayed in the Replication Server error log as a result of a DSI that did not start or was suspended:

```
The DSI thread for 'RDS.rdb2' is not started, because
the connection to the database has been suspended or
the connection has not been completed.
```

```
The DSI thread for database 'RDS.rdb2' is being
shutdown. DSI received data server error #1105 which
is mapped to STOP_REPLICATION. See logged data
server errors for more information. The data server
error was caused by RS output command #1 mapped from
input command #1 of the failed transaction.
```

```
The DSI thread for database 'RDS.rdb2' is shutdown.
```

If the DSI is down or suspended, see “DSI down or suspended” on page 119.

- 3 If there is an entry for the database, but the state is `Awaiting Command`, the DSI is waiting for a transaction from the primary. See Chapter 1, “Identifying the Problem” to determine why updates are not reaching this Replication Server.
- 4 If the state of the DSI thread is `Active`, then:
 - Unique transactions might be incorrectly resolved as duplicates, or
 - Transactions might be failing on the data server and written to the exceptions log.
 Refer to “DSI active” on page 124 for specific instructions.

DSI down or suspended

This section describes solving these problems:

- Incorrect login or permissions
- A data server error
- A Replication Server error

If a subscription problem causes the DSI thread for the replicate database to terminate abnormally, you can restart the thread using the `resume connection` command. If possible, fix whatever condition caused the problem before resuming the connection. For example, if the maintenance user does not have update permission on the replicate table, first grant the user update permission and then resume the connection.

If you cannot fix the problem, resuming the connection will cause the DSI thread to reexecute the command that failed, and suspend the DSI again. To prevent this cycle, assign a different action to the error returned to the DSI. You must assign error actions at the Replication Server where the error class is created. For more information about error actions and classes, see the *Replication Server Administration Guide Volume 2*.

Connection failure to the database

If the error is a connection failure to the database:

- Verify that the data server is defined in the *interfaces* file.
- Verify that the data server is running.
- Verify that the maintenance user name and password are correct.

To find the maintenance user name and password for a database, log in to the system Adaptive Server, use the RSSD, and execute the following query:

```
select username, password
       from rs_maintusers, rs_databases
       where destid = dbid
          and dsname = data_server
          and dbname = database
```

If the maintenance user name or password is not correct, you can either change them on the data server or use the `alter connection` command to change them in the Replication Server. If you have password encryption, you cannot access the password by executing a query. After correcting the user name or password, resume the connection.

Asynchronous transaction connection failures

When an asynchronous stored procedure reaches the primary Replication Server, the DSI uses the original login and password to connect to the primary data server. The maintenance login is *not* used.

If the login fails, use the `sysadmin log_first_tran` command to log the asynchronous transaction into the exceptions log. The `app_user` and `app_pwd` columns in the `rs_exceptshdr` system table contain the login and password the DSI uses to log in to the primary data server. See “Examining the exceptions log” on page 125 for more information.

Data server errors

Data server errors (for example, a permissions violation or duplicate key) are logged in the Replication Server error log.

If you cannot correct the data server error, restart the DSI by requesting that the first transaction (the one causing the error) be skipped. Do this by executing the resume connection command with the skip transaction option.

Warning! Skipping transactions may cause replication inconsistencies. See *Skipping transactions* for details.

If errors are caused by unique-key violations, you can use the `set autocorrection` command to turn automatic error correction on, before you resume the connection. These errors are likely to occur during nonatomic materialization. For more information about the `set autocorrection` command, see the *Replication Server Reference Manual*.

As a last resort, use the skip transaction option of the resume connection command. This option skips the materializing or dematerializing transaction altogether. If the problem occurred during subscription materialization, drop the subscription and create it again in a way that avoids the same problem.

For more information about subscription problems, see Chapter 5, “Subscription Problems”

Skipping transactions

A transaction intended for a database is skipped when:

- The resume connection command is executed with the skip transaction option.
- A data server error is encountered and the assigned action for the error is log, or retry_log.

Skipping a transaction that causes errors is a convenient way to force the DSI to resume applying transactions to the database. However, skipping a transaction has important implications for the database:

- Replicate data may become inconsistent with primary data.
- If an asynchronous transaction originates in a replicate database, and the transaction is skipped at the primary data server, the primary database is not updated.
- The inconsistency caused by skipping a transaction may result in even more errors if subsequent transactions depend upon the unapplied actions of the skipped transaction.
- Subscription materialization and dematerialization requests are special types of transactions. If you skip a subscription materialization request, it may invalidate the subscription and must be dropped.
- A subscription materialization or dematerialization request may cause Replication Server to create a separate stable queue, called the materialization or dematerialization queue. You cannot explicitly skip transactions in this queue using the resume connection command. They can, however, be skipped due to data server errors that are assigned the log or retry_log action.
- If a subscription materialization or dematerialization request is skipped, all transactions in the corresponding queue may be skipped. However, if a DSI is suspended in the middle of applying the transactions in a materialization or dematerialization queue, some of the transactions in the queue may already have been committed while others may not. Later, if you use the resume connection command to skip the request, the replicate database will have the effects of the previously committed transactions.

Because of these implications, you should skip transactions only after other means of correcting the error have been exhausted. After skipping a transaction, determine what you must do to bring the replicated data back to a state of consistency.

Customized handling of data server errors

You can customize how Replication Server handles data server errors. If you want the DSI to continue applying updates even when a data server error is encountered, you can change the error action from `stop_replication` to `log` or `retry_log`. To make this change, use the `assign action` command at the Replication Server where the error class was created. All databases using the error class are affected. If the primary database for a class is at a different site, then you may need to wait until the new error action is replicated to the local site.

To limit the change to one database, create a new error class with the `create error class` command. Specify the desired error actions for the class, then change the connection to the database to use the new error class.

If you want to examine the commands in the transaction that caused the connection to be suspended, use the `sysadmin log_first_tran` command. This command writes the first transaction into the exceptions log without skipping it.

Replication Server errors

If a Replication Server error is displayed in the Replication Server error log, refer to the chapter that corresponds to the following conditions.

- If the error message is about function strings or is “function string not found,” refer to “Using incorrect variables in function strings” on page 6 or “29024: no matching function strings found” on page 52.
- If the error message is about subscriptions, refer to Chapter 5, “Subscription Problems”
- If the error message is about missing objects, there is probably some inconsistency in the RSSD. Call Sybase Technical Support for assistance.

DSI active

When the DSI is active but the replicate data server is not receiving updates, unique transactions might be incorrectly resolved as duplicates, duplicate transactions might be reapplied, or the transactions might be failing on the data server. If the error's action was set to log or `retry_log` using the `assign` action command, the error and failed transaction are written to the exceptions log. You must manually resolve the data inconsistency between the primary and replicate databases or reapply the transactions.

Incorrect duplicate transaction resolution

If the `origin_qid` values stored in a data server or the `rs_exceptslast` are modified by mistake, non-duplicate transactions can be ignored or duplicate transactions could be reapplied. If you suspect that this is happening in your system, check the stored values and compare them with the transactions in the database's stable queue to determine the validity of the values. If the values are wrong, you must modify them directly.

The DSI records the last transaction committed or written into the exceptions log so that it can detect duplicates after a system restart. Each transaction is identified by a unique origin database ID and an origin queue ID that increases for each transaction. For more information about the exceptions log, see “Examining the exceptions log” on page 125.

The last transaction committed from each origin database is recorded on a data server through execution of the function strings defined for the data server's function-string class. For the default function-string class, `rs_sqlserver_function_class`, this is done in the function string of a `commit` command, that is, the `rs_commit` function. The function-string class of every data server must support the `rs_get_lastcommit` function which returns the `origin_qid` and `secondary_qid` for each origin database. The `secondary_qid` is the ID of the queue used for subscription materialization or dematerialization.

The `origin_qid` and `secondary_qid` for the last transaction written into the exception log from each origin is recorded in the `rs_exceptslast` table in the RSSD. However, transactions logged explicitly by the `sysadmin log_first_tran` command are not recorded in this table. These transactions are logged, but they are not skipped.

When a data server interface is started or restarted, it gets the `origin_qid` returned by the `rs_get_lastcommit` function and the one stored in the `rs_exceptslast` table. It assumes that any transaction in the queue with an `origin_qid` less than the larger of these two values is a duplicate and ignores it.

Refer to Appendix A, “Dumping Stable Queues” for more information on how to dump transactions in a queue.

Examining the exceptions log

If a transaction is skipped, either explicitly using the `resume connection` command with the `skip transaction` option, or as a result of the `log action` assigned to data server errors, the skipped transaction is written into the exceptions log. Orphan transactions and transactions logged by the `sysadmin log_first_tran` command are also in the exceptions log.

If a replicate database is not receiving updates, it is possible that the update transactions were skipped and written into the exceptions log. Examine the exceptions log to find out what transactions have been logged.

The exception log consists of three tables: `rs_exceptshdr`, `rs_exceptscmd`, and `rs_systext`. The `rs_exceptshdr` table has one entry for each transaction. The `rs_exceptscmd` table has one entry for each command (either source or output) of the transaction. The `rs_systext` table stores the text of the commands.

To view the header information for all logged transactions intended for a database, log in to the system Adaptive Server for the Replication Server controlling the database, and execute the following query against the RSSD:

```
select * from rs_exceptshdr
       where error_site = data_server
       and error_db = database
       order by log_time
```

The rows are returned in ascending order, by the time the transactions were logged. If you want the rows to appear in descending order, attach “`desc`” to the end of the query.

Join the three system tables to view all information about a transaction in the log. The following query gives the source commands and their corresponding output commands for each logged transaction:

```
select hdr.sys_trans_id, cmd_type, textval
       from rs_exceptshdr hdr,
            rs_exceptscmd cmd,
            rs_systext
```

```
where error_site = data_server
and error_db = database
and hdr.sys_trans_id = cmd.sys_trans_id
and cmd_id = parentid
order by log_time, src_cmd_line,
output_cmd_index, sequence
```

For more information about the system tables that make up the exceptions log, see the *Replication Server Reference Manual*.

Adaptive Server Log Problems

This chapter describes problems that occur in Adaptive Server transaction log.

Name	Page
Overview	127
Truncating an Adaptive Server log	128
Log truncation problems	131
Adaptive Server log corruption	133

If you cannot find the solution in this chapter

If an error message is displayed in an error log and you cannot find it in this chapter, see:

- “Analyzing error logs” on page 9
- Chapter 3, “Common Errors”

If an error message is not displayed, see Chapter 1, “Identifying the Problem”

Overview

The Adaptive Server primary database log is the source of the data that Replication Server distributes. A RepAgent retrieves transactions from the log and sends them to the Replication Server. Adaptive Server log problems include:

- Log files that have reached the maximum size and need to be truncated
- Missing or incomplete transactions at the primary Replication Server as a result of truncated primary database logs
- Log files corrupted by software and hardware sources

Truncating an Adaptive Server log

When the database log is full, Adaptive Server cannot continue to process updates. To solve this problem you must truncate the log. However, you should not truncate transactions from the log before the RepAgent has passed them successfully to the Replication Server.

Adaptive Server uses truncation points to ensure that only transactions processed by the RepAgent are truncated. A **secondary truncation point** marks the place in the primary database log up to which the RepAgent has processed transactions. The RepAgent periodically updates the secondary truncation point to reflect transactions successfully passed to the Replication Server. Adaptive Server will not truncate the log past the secondary truncation point. For more information about database log truncation see *Replication Server Administration Guide Volume 1*.

There are, however, times when it is necessary to truncate the log beyond the secondary truncation point. For example, if the RepAgent cannot access the Replication Server and the log fills, you may want to truncate the log at the secondary truncation point rather than extending the log or preventing clients from updating the primary database. Truncating the log can cause inconsistencies between the primary and replicate databases. To resolve these inconsistencies, see “Log truncation problems” on page 131.

To truncate the database log at the secondary truncation point:

- 1 Verify that the secondary truncation point is set for the database. See “Verifying the state of the secondary truncation point” on page 128.
- 2 Turn off the secondary truncation point in the database. See “Turning off the secondary truncation point in a database” on page 129
- 3 Dump the database log. See the *Adaptive Server Enterprise Reference Manual*.
- 4 Set the secondary truncation point. See “Setting the secondary truncation point” on page 130

Verifying the state of the secondary truncation point

To check whether the secondary truncation point is set for a database, use the `dbcc gettrunc` command. For example, to see if the secondary truncation point is set in the Parts database, log in to the primary database and execute:

```
> use Parts
```

```
> go
> dbcc gettrunc
> go
```

In the output that these commands generate, a “0” in the `ltm_trunc_state` column means the secondary truncation point is turned off for the database. A “1” in the `ltm_trunc_state` column indicates the secondary truncation point is turned on.

Turning off the secondary truncation point in a database

Truncating the log frees only log pages that the RepAgent has sent to the Replication Server with confirmation. To truncate portions of the log that have not been transferred, you must turn off the secondary truncation point for the database by executing the Transact-SQL command `dbcc settrunc(ltm, ignore)` in the database.

Note Only the Adaptive Server system administrator and the database owner have permission to execute the `dbcc settrunc` command.

First, suspend the RepAgent forwarding the log. Otherwise, executing `dbcc settrunc` results in the following error:

```
The log transfer context for the current database is
already reserved by Adaptive Server process 7. The log
transfer context for the current database is not
reserved.
```

To turn off the secondary truncation point in the database named `Parts`, for example, log in to the Adaptive Server with the database and execute:

```
> use Parts
> go
> dbcc settrunc(ltm, ignore)
> go
```

After you have turned off the secondary truncation point, you can truncate the log using the Transact-SQL `dump transaction` command. See the *Adaptive Server Enterprise Reference Manual* for more information.

Finally, you should dump the transaction log. This may result in the loss of updates to the replicate database. The primary database is not affected. You must reapply the lost updates to the replicate database. For more information about reapplying updates, see “Log truncation problems” on page 131.

Setting the secondary truncation point

Before you restart a RepAgent, you need to reestablish the secondary truncation point using `dbcc settrunc(ltm,valid)`. For example, to reestablish the secondary truncation point in the Parts database, log in to the Adaptive Server with the database and execute:

```
> use Parts
> go
> dbcc settrunc(ltm, valid)
> go
```

Use the `dbcc gettrunc` command to verify that the secondary truncation point has been enabled:

```
> use Parts
> go
> dbcc gettrunc
> go
```

The `ltm_trunc_state` column in the output from this command should contain a 1.

After resetting the locator for the database in the `rs_locator` table, resume the Replication Server to restart the RepAgent. The RepAgent begins scanning the log from the page returned in the `ltm_truncpage` column by the `dbcc gettrunc` command.

Resetting the locator

When a RepAgent starts, it uses a **locator** to determine where to begin scanning the database log. The locator for a database log is stored in the `rs_locator` table in the RSSD of the Replication Server that controls the database.

Note This locator is different from the RSI locator. See “Invalid RSI locator” on page 106 for more information about the RSI locator.

The locator points to a record on a page in the database log. If the truncation point for the database is turned off, you or Adaptive Server may truncate the page the locator points to, invalidating the locator. When the locator is invalid, the RepAgent writes error message 9215 (see “9215: log truncated (ASE 624)” on page 61) to the Adaptive Server error log.

To recover, reset the locator to 0x0. This directs the RepAgent to begin scanning the log from the truncation point.

To reset the locator, update the `rs_locator` table in the RSSD for the Replication Server controlling the database using the Adaptive Server stored procedure `rs_zeroltm`:

```
rs_zeroltm data_server, database
```

where `data_server` and `database` are the respective Adaptive Server and database whose locator value you want to reset.

For example, to reset the locator for the MYDS Adaptive Server and the Parts database, execute;

```
rs_zeroltm MYDS, Parts
```

Log truncation problems

Truncating the log of the primary database may lead to these types of inconsistencies:

- Orphaned transactions can cause the stable queues to fill.
- Inbound queue requires more disk space.
- Parts of transactions, or entire transactions, may be missing, causing inconsistent data at replicate sites. See *Replication Server Administration Guide Volume 2* for more information.
- Subscription commands may be lost, disrupting the materialization or dematerialization protocol and requiring manual cleanup.
- If a log is truncated while a route is being created, subscription commands against the RSSD may be lost. As a result, the route cannot be created. If this occurs, drop and re-create the route.

For more information about how to recover lost transactions, see *Replication Server Administration Guide Volume 2*.

Orphaned transactions

An orphaned transaction is a transaction in an inbound stable queue that is missing a terminating commit or rollback command. Because Replication Server does not free up a queue segment until every transaction in the queue segment (or in any of the preceding queue segments) has been committed or rolled back, orphaned transactions can cause the inbound queue to fill up.

❖ Checking for orphaned transactions

- 1 Log in to the Replication Server and execute `admin who, sqt` to display information about the stable queues at the Replication Server.
- 2 Use the output from `admin who, sqt` to identify the entry for the inbound queue of the database whose log was truncated.
The queue has a two-part name formed from the database ID and the queue-type identifier. For an inbound queue, the queue-type identifier is 1. For example, if the database ID is 101, the inbound queue name is 101:1.
- 3 If the output for the inbound queue shows an open transaction that does not change over long periods of time, an orphaned transaction is probably in the queue. However, because it is difficult to distinguish between an orphaned transaction and a very long transaction, you must further verify that an orphaned transaction exists.
To verify that there is an orphaned transaction:
 - Dump the stable queue and examine the information about the transaction. Refer to Appendix A, “Dumping Stable Queues” for instructions. You can dump just the `begin` record to find the user name and the time the command was executed, and then see if the user still has open transactions in the database.
 - Even if you find that the user does not have open transactions, the queue may still have an orphaned transaction. If the queue is large, the RepAgent may not be keeping up with it. To determine whether this is the case, dump the last block in the queue and look at the date for the commands. Use `admin who, sqm` to find the last block.
- 4 If the transaction is orphaned, use the `sysadmin purge_first_open` command to skip it.
- 5 Use the output from dumping the stable queue to manually apply the orphaned transaction to the replicate database.

Inbound queue requires more disk space

If the inbound queue requires more disk space, a message similar to the following is displayed in the Replication Server error log:

```
_SQM_ADD_SEGMENT: Going to wait for a segment to be
freed. Administrative action is needed.
```

To determine which stable queues hold the segments, log in to the RSSD and execute the following query:

```
select q_number, q_type, count(*)
from rs_segments
group by q_number, q_type
```

The output of this query shows the number of segments each stable queue is using. Each segment is 1MB of disk space. Look at the queues with the largest number of segments:

- A *q_type* of 0 means an outbound queue. If any of the outbound queues have more than two segments, verify that the connections they service are operating. See Chapter 8, “Data Server Interface Problems” and Chapter 6, “Replication Server Interface Problems” for more details.
- A *q_type* of 1 is an inbound queue. If one of these queues has a large number of segments, it may contain an orphaned transaction. See “Orphaned transactions” on page 132 for more information.

Adaptive Server log corruption

Adaptive Server log corruption can cause failures anywhere along the replication path. The most common symptoms are:

- RepAgent shuts down when it reads log data it cannot process.
The RepAgent shutting down is the most common symptom of a corrupt Adaptive Server log. In addition, the error log may reveal error messages that are inconsistent with the current operation; however, no specific log corruption error message is displayed.
- A component of the Replication Server (such as the DSI) shuts down.

When the RepAgent passes incorrect data to the Replication Server, Replication Server components, such as the DSI, may shut down. For instance, if the DSI encounters a mismatched data type and fails to resolve the differences, it shuts down.

- Replicated data appears out of sync.

The procedures for determining log corruption and where the corruption occurs is complex. It requires knowledge of the log format as well as Adaptive Server and Replication Server internal components.

Note If you suspect Adaptive Server log corruption, contact Sybase Technical Support to help you diagnose and resolve the issue.

Replication Monitoring Services Problems

This chapter describes tools you can use to troubleshoot the Replication Monitoring Services (RMS). It also describes common RMS problems.

Name	Page
Overview	135
RMS tracing	136
Common Problems	136

Overview

The RMS is a middle tier monitoring layer that monitors the status of the servers and components in the replication environment. It also provides information to troubleshoot problems and the commands to fix these problems.

The RMS is implemented using the Sybase Unified Agent Framework (UAF). The UAF provides flexible and extensible service-oriented approach to manage distributed resources. It provides common services and interfaces for development and deployment of agent plug-ins like RMS.

UAF and RMS Error Logs

Both the RMS and the UAF write errors, warnings, and informational messages to log files. Review the following log files when you are troubleshooting issues in the RMS:

- UAF log file: `$SYBASE/ua/log/agent.log`
- RMS log file: `$SYBASE/ua/plugins/com.sybase.rms/logs/rms.log`

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

RMS tracing

The RMS provides a tracing feature to assist the user when troubleshooting problems. Information about the internal workings of the RMS will be written to the RMS log file.

Table 10-1 lists trace flags that are useful when troubleshooting problems in the RMS.

Table 10-1: Trace flags for troubleshooting RMS

Trace flags	Description
trace RMS_Command on	Writes to the error log every command RMS receives. Commands can be sent from the RM plug-in, an isql session, or some other client application. This flag is useful in finding the command that caused a specific problem.
trace Server_Command on	Writes to the error log every command RMS sends to a monitored server. Use this flag to determine how RMS retrieves information about the replication environment.
trace Startup on	Adds trace messages to the RMS log file at each step of the start-up process. Use this flag to diagnose UAF start-up problems.
trace Network_Connection on	Adds trace messages to the RMS log file whenever a connection to a server is created. The RMS will include all connection information except the password in the trace message. Use this flag to trace network connection errors.

The user controls the type of information that is logged by setting trace flags in the RMS. For information on the RMS trace commands and a complete list of the trace flags, see the *Replication Server Reference Manual*.

Common Problems

RMS problems may occur when:

- Using the RMS with UAF version 2.0
- Using multiple UAF servers on the same computer
- Using RMS heartbeat with Replication Server 12.6

- Monitoring the ASA Replication Agent

Using the RMS with UAF version 2.0

Symptom	UAF starts but plug-ins are not loaded.
Explanation	Sybase has released a newer version of the UAF and the RMS configuration file <code>agent-plugin.xml</code> must be modified to work with UAF version 2.0.
Action	Execute the following command to determine the version of the UAF: <pre>cd \$SYBASE/ua/bin cd ./agent -v</pre> <p>If the version of the UAF is 2.0 or greater, the <code>mbean-descriptor</code> attribute must be added to the <code>agent-plugin.xml</code> file. The file is located at <code>\$SYBASE/ua/plugins/com.sybase.rms</code></p> <p>Shutdown the UAF and add the following attribute to the second line of the <code>agent-plugin.xml</code> file:</p> <pre>mbean-descriptor="mbean-descriptor.xml"</pre>

Using multiple UAF servers on the same computer

Symptom	Sybase does not support running multiple UAF servers on the same computer. Executing multiple UAF servers on the same computer causes the RMS to report inconsistent and false server status. At UAF server start up, the RMS generates the following message if a UAF server is already running: <pre>2006-03-10 12:57:16,520 INFO [main] RMIService(136) - Starting RMI Service... 2006-03-10 12:57:16,629 ERROR [RMIRegistryThread] RMIRegistryServiceStarter(34)- Failed to start rmi registry. java.rmi.server.ExportException: Port already in use: 9999; nested exception is: java.net.BindException: Address already in use at sun.rmi.transport.tcp.TCPTransport.listen (UnknownSource) at sun.rmi.transport.tcp.TCPTransport.exportObject (Unknown Source) at sun.rmi.transport.tcp.TCPEndpoint.exportObject (Unknown Source) at sun.rmi.transport.LiveRef.exportObject</pre>
---------	---

```
(Unknown Source) at
sun.rmi.server.UnicastServerRef.exportObject
(Unknown Source) at
sun.rmi.registry.RegistryImpl.setup(Unknown Source) at
sun.rmi.registry.RegistryImpl.<init>
(Unknown Source) at
java.rmi.registry.LocateRegistry.createRegistry
(Unknown Source) at
com.sybase.ua.services.rmi.RMIRegistryServiceStarter.
run(RMIRegistryServiceStarter.java:28)
Caused by: java.net.BindException: Address already in
use at java.net.PlainSocketImpl.socketBind
(Native Method) at
java.net.PlainSocketImpl.bind(Unknown Source) at
java.net.ServerSocket.bind(Unknown Source) at
java.net.ServerSocket.<init>(Unknown Source) at
java.net.ServerSocket.<init>(Unknown Source) at
sun.rmi.transport.proxy.RMIDirectSocketFactory.
createServerSocket(Unknown Source) at
sun.rmi.transport.proxy.RMIMasterSocketFactory.
createServerSocket(Unknown Source) at
sun.rmi.transport.tcp.TCPEndpoint.newServerSocket
(Unknown Source) ... 9 more
2006-03-10 12:57:17,799 ERROR [main] RMIService(142) -
Failed to start RMI
Connector server. java.io.IOException: Cannot bind to
URL [rmi://abonner-sun:9999/agent]:
javax.naming.NameAlready BoundException: agent
[Root exception is java.rmi.AlreadyBoundException:
agent]
```

Explanation Only one UAF server can run on a computer.

Action Reinstall the RMS into the existing UAF installation.

Using RMS heartbeat with Replication Server 12.6

Symptom Connection is suspended when the RMS heartbeat is used with Replication Server 12.6.

Explanation The RMS heartbeat process uses the Replication Server `rs_ticket` feature. This feature is only supported by Replication Server version 12.6 and later. However, Replication Server 12.6, unlike Replication Server 15.0, sends `rs_ticket` transactions to versions of the Replication Server earlier than 12.6, and this causes the connection suspensions.

Action Do not set up a heartbeat process if there are Replication Servers between the primary and replicate databases that are less than version 12.6.

To restart the connection, remove all `rs_ticket` command from the connection's queue.

Monitoring the ASA Replication Agent

Symptom The ASA Replication Agent (*dbltn.exe*) fails when the RMS attempts to establish a connection.

Explanation The RMS uses jConnect and therefore causes the *dbltn* to fail when RMS connects to determine its status.

Action Apply the following EBF to the *dbltn.exe*: ASA v8.0.3 EBF(5369), ASA v9.0.2 EBF(3272) and ASA v10.0.0.

Dumping Stable Queues

This appendix describes how to dump stable queues.

Name	Page
Introduction	141
Using sysadmin dump_queue	143
Using sysadmin sqt_dump_queue	154
Using sysadmin dump_file	156
sysadmin purge commands	157

Introduction

Stable queues are composed of segments stored on disk partitions. Each segment has a megabyte of message space, divided into 64 blocks of 16KB. Segments store messages passed by the system. Each block in a segment corresponds to a physical data transfer between disk and memory. A segment is allocated to just one queue—different queues cannot share segments. Each block within a segment may store one or more messages. A map of the system segments is stored in the `rs_segments` table of the RSSD.

Replication Server deletes messages from queues on a segment basis. Internally, it deletes blocks from the segments. However, the freed space does not appear in the system tables until all of the blocks contained in the 1MB segment are deleted.

When a stable queue is dumped, a segment may contain deleted blocks along with the undeleted blocks. You can identify the undeleted blocks of a segment by checking the `First Seg.Block` column in the `admin who, sqm` output for the queue.

Sometimes queues are dumped while Replication Server is actively adding and deleting segments from the queues. At such times, the output of the `admin who, sqm` command may not accurately represent the contents of the stable queues. Segments may be deleted before you can enter the `sysadmin dump_queue` command. To prevent Replication Server from changing a stable queue while you examine its contents, you can suspend the database connection or route that uses the stable queue. You can also start the Replication Server in standalone mode to prevent changes to a stable queue. For more information, see the *Replication Server Administration Guide Volume 2*.

This list briefly describes each command:

- `sysadmin dump_queue` – dumps the contents of the transaction cache of an inbound or outbound stable queue.
- `sysadmin sqt_dump_queue` – dumps the contents of the transaction cache of an inbound queue.
- `sysadmin dump_file` – specifies an alternate log file to be used by `sysadmin dump_queue` or `sysadmin dump_queue`.
- `sysadmin purge_all_open` – purges all open transactions from an inbound queue.
- `sysadmin purge_first_open` – purges only the first open transaction from an inbound queue.

Using trace

The outbound queue for the replicate database contains default commands. If you have created function strings in addition to the defaults, you may want to use a special DSI trace flag to print the actual commands that are being sent to replicate databases.

To set traces, you need to run the debug replication server, `repserver.diag`, which should be located in the *bin* directory.

To set traces, add the following lines into the `.cfg` file of the Replication Server you want to trace.

- 1 This line forwards the output to the standard output or to the *errorlog*:

```
trace=GEN,TRC_STDERR or trace=GEN,TRC_ERRLOG
```

- 2 This line gives the output of commands that are sent to the replicate site.

```
trace=DSI,DSI_CMD_DUMP
```

It logs the commands in the form:

```
Command sent to 'RDS.RDB' : command
```

Using *sysadmin dump_queue*

The `sysadmin dump_queue` command dumps the contents of a Replication Server stable queue to the Replication Server log, an alternate log file, the client, or to tables in the RSSD. You can examine the dump to see the messages in the stable queue. This may help you to identify and reapply transactions that have not been applied or applied incorrectly to the replicate database.

Here is the syntax for the `sysadmin dump_queue` command:

```
sysadmin dump_queue, q_number, q_type, seg, blk, cnt [,RSSD | client]
```

q_number, *q_type* – identifies the stable queue to dump. Examine the output of the `admin who` command or the `admin who, sqm` command to find these values.

seg – identifies the starting segment. If set to -1, the starting segment is the first active segment in the queue. If set to -2, the starting segment is the first segment in the queue and includes any inactive segments retained by setting a save interval.

blk – identifies the 16K block in the segment where the dump is to begin. Block numbering starts at 1 and ends at 64. The following two additional options are used with the *seg* setting:

- If *seg* is -1 and *blk* is -1, then start with the first undeleted block of the queue.
- If *seg* is -1 and *blk* is -2, then start with the first unread block of the queue. For example, if the RSI and DSI are down, then executing `sysadmin dump_queue` with these parameters dumps the unread segments in the queue.

cnt – specifies the number of blocks to dump. This number can span multiple segments. If set to -1, the end of the current segment is the last block dumped. If set to -2, the end of the queue is the last block dumped.

RSSD – optional flag that forces the output to system tables in the RSSD. If you use this option, insert a comma before it. Sending output to the RSSD is the slowest option.

- If the RSSD option is used, then:

- System tables are first cleared of the segments with the same *q_number*, *q_type*, *seg*, and *blk* as the blocks are being dumped, and
- Dump is written into two tables in the Replication Server system database: *rs_queuemsg* and *rs_queuemsgtxt*.
- The *rs_queuemsg* table has the following columns:
 - *q_number*
 - *q_type*
 - *q_seg*
 - *q_blk*
 - *q_row*
 - *len*
 - *origin_site_id*
 - *origin_q_id*
 - *origin_time* (used only at the beginning of a transaction)
 - *origin_user*
 - *tran_name*
 - *local_q_id*
 - *status*
 - *reserved*
 - *tran_len*
 - *txt_len*
 - *tran_id*
 - *lorigin_site_id*
 - *version*

Only the begin transaction command uses the *origin_time*, *origin_user*, and *tran_name* columns.
- The *rs_queuemsgtxt* table holds the text of commands dumped from the stable queue. If the text of a command exceeds 255 characters, it is stored in multiple rows numbered with the *q_seg* column.

client – optional flag that forces the output to the client that is issuing this command.

- If the client option is used, the dump is written into the client that is issuing this command, such as isql or Replication Server Manager.
- The client output has the following columns:
 - Q Number
 - Q Type
 - Segment
 - Block
 - Row
 - Message Len
 - Orgn Siteid
 - Orgn Time
 - Orgn Qid
 - Orgn User
 - Tran Name
 - Local Qid
 - Status
 - Tranid
 - Logical Orgn Siteid
 - Version
 - Command Len
 - Seq No
 - Command

If you do not specify the RSSD or client option, the output will go into the Replication Server log. If you use the `sysadmin dump_file` command to specify an alternative log file for dumping queues, the output will go into the alternate dump file.

Confirming suspected problems

This section describes how to use `sysadmin dump_queue` to confirm that a suspected problem exists.

When the Replication Server runs out of stable queue segments, an uncommitted transaction may remain in the inbound stable queue. To find out if this has happened, follow these steps:

- 1 Log in to the Replication Server.
- 2 Execute `admin who, sqt`.
- 3 Use the output from `admin who, sqt` to determine which inbound queue has an open transaction. It will appear as an open transaction with the state marked “st:O”. The Info column of this row holds the queue number and type. The transaction state is followed by the number of commands and then a local queue ID for the begin transaction command in the form `segment:block:row`.

In the following example of `admin who, sqt` output, the transaction has a queue number of 103 and a queue type of 1, is in an open state and consists of three commands. The begin transaction command is located in segment 21, block 28, row 0:

```
103:1 st:O,cmds:3,qid:21:28:0
```

- 4 Execute the following command to dump block 28 of segment 21:

```
sysadmin dump_queue, 103, 1, 21, 28, 1, RSSD
```
- 5 Replication Server writes the text of the commands in block 28 of segment 1 into the `rs_queuemsg` and `rs_queuemsgtxt` system tables or into the Replication Server log.
- 6 If you use the RSSD option, you can view information about the command in the `rs_queuemsg` system table by executing this command on the RSSD:

```
select * from rs_queuemsg
  where q_number = 103 and q_type = 1 and
  q_seg = 21 and q_blk=28 and q_row = 0
```

Execute a similar query against `rs_queuemsgtxt` to obtain the text of the command.

sysadmin dump_queue examples

The following examples demonstrate the use of the `sysadmin dump_queue` command.

- This command acts on queue 103:1. It dumps blocks 15 through 64 of segment 0 and blocks 1 through 15 of segment 1 into the Replication Server log:

```
sysadmin dump_queue, 103, 1, 0, 15, 65
```

- The following command dumps all of queue 103:1 into the Replication Server log:

```
sysadmin dump_queue, 103, 1, -1, 1, -2
```

- This command dumps all of queue 103:1 into the RSSD:

```
sysadmin dump_queue, 103, 1, -1, 1, -2, RSSD
```

- This command dumps all of queue 103:1 into the client:

```
sysadmin dump_queue, 103, 1, -1, 1, -2, client
```

- The following series of commands dump all of queue 103:1 into a file named *qdump*. The last `sysadmin dump_file` command closes *qdump*.

```
sysadmin dump_file, qdump
sysadmin dump_queue, 103, 1, -1, 1, -2
sysadmin dump_file
```

Interpreting the output of a dump queue

Stable queues consist of partitions. Partitions are created with the `add partition` command. Whenever a partition is added to the replication server, it is divided into 1 megabyte segments. Segments are divided into 64 blocks of 16 KB. Messages have fixed size headers, variable size messages, and are written into blocks.

The following examples shows the output of an `sysadmin dump_queue` command on various types of queues and how to read the output of each.

Example 1: outbound queue after *create subscription*

In this example, the `create subscription` command at the replicate Replication Server starts the materialization process by selecting from the primary database:

```
create subscription emp_queue_sub
for emp_queue
with replicate at NYDS.nydb
```

To view the entire queue:

```
sysadmin dump_queue, 103, 0,-1,1,-2
```

Output

```
I. 2006/07/05 08:20:20. QUEUE DUMP FOR 103:0
I. 2006/07/05 08:20:20. BLOCK BEGIN
q_number=103 q_type=0 blk=0:1 cnt=1
I. 2006/07/05 08:20:20. Begin Transaction
Origin User= Tran Name=
I. 2006/07/05 08:20:20. ENTRY ver=1100 len=188 orig=102
lorig=0 oqid=00000000000016c7000004ff0004000004ff0000
000097f50087a0720000000000000000 lqid=0:1:0 st=21
tr='00000000000016c70000' NYDSnydb comlen=69
activate subscription 0x0100006580000065 0
I. 2006/07/05 08:20:20. END QUEUE DUMP FOR 103:0
```

Explanation

- **BLOCK BEGIN**
The beginning of a 16K block.
- **q_number=103**
The queue ID.
- **q_type=0**
The queue type (0 = outbound, 1 = inbound).
- **blk=0:1**
Block identifier (segment 0: block 1).
- **cnt=1**
The number of entries (commands) in this block.
- **Begin Transaction Origin User= Tran Name=**
Beginning of a transaction with originating user and the transaction name. *Tran Name* is empty when the original transaction at the primary database was explicitly started with a begin transaction without supplying a transaction name.

- ENTRY
The actual message.
- len=188 orig=102
The message length and the originating site ID.
- oqid=00000000000016c7000004ff0004000004ff0000000097f50087a0720000000000000000
The origin queue ID: the queue ID assigned to the command string at the place the command originated from. This ID is used for duplicate detection.
- Lqid=0:1:0
Local queue ID: Commands logged by an SQM are assigned a local queue ID. This queue ID is sent to the next queue as the sender queue ID and is used by the destination to acknowledge the command. *Lqid* consists of segment number, block number and row number. Row numbers start at 0.
- St=21
Status and command type. The field is a bitmask which has status and command type set. This bit's settings are 0x01 and 0x20, a commit and materialization command.
0x01 = status of a commit
0x02 = status of a rollback
0x04 = status of a begin
0x08 = status of a orphan statement
0x10 = status of a materialization statement
0x20 = status of a materialization statement
0x40 = status of a tran going to primary
0x80 = ignore duplicates for DDL commands
0x0100 = status of a materialization command
0x0200 = dump/load message
0x0400 = routing message
0x0800 = flag for DDL generation
0x1000 = DDL generation at DSI flag

0x2000 = special SQM message

0x4000 = SQT mini-abort

0x8000 = message has been zapped

- tr= '0000000000016c70000' NYDSnydb comlen=69
activate subscription 0x0100006580000065 0

Transaction ID and command, *comlen* indicates the length of the command.

Example 2: inbound queue

In this example, these commands were executed at a primary database, NYDS.nydb:

```
sp_setreplicate emp_queue, true
insert emp_queue
    values("123456789", "Davis", "Gen",
          "Process Engineer","1/1/95", 1111)
insert emp_queue
    values("987654321", "Irvine", "Ben",
          "Microwave Engineer","3/5/94", 2222)
grant all on emp_queue to public
```

To view the inbound queue of the primary Replication Server:

```
sysadmin dump_queue, 102,1,-1,1,-2
```

Output

```
I. 2006/07/05 08:27:05. BLOCK BEGIN
q_number=102 q_type=1 blk=0:4 cnt=10
I. 2006/07/05 08:27:05. Begin Transaction
Origin User=sa Tran Name=rs_logexec
I. 2006/07/05 08:27:05. ENTRY ver=1100 len=216 orig=102
lorig=0 oqid=0000000000016c7000004ff0005000004ff0005
000097f5008af20b00000000000000000 lqid=0:4:0 st=4
tr= '0000000000016c70005' NYDSnydb comlen=97
begin transaction
I. 2006/07/05 08:27:05. ENTRY ver=1100 len=204 orig=102
lorig=0 oqid=0000000000016c7000004ff0007000004ff0005
000097f5008af20b00000000000000000 lqid=0:4:1 st=1
tr= '0000000000016c70005' NYDSnydb comlen=86
commit transaction
I. 2006/07/05 08:27:05. Begin Transaction
Origin User=sa Tran Name=_ins
```

```
I. 2006/07/05 08:27:05. ENTRY ver=1100 len=208 orig=102
lorig=0 oqid=00000000000016c7000004ff0008000004ff0008
000097f5008af20b00000000000000000 lqid=0:4:2 st=4
tr= '00000000000016c70008' NYDSnydb comlen=91
begin transaction
I. 2006/07/05 08:27:05. ENTRY ver=1100 len=524 orig=102
lorig=0 oqid=00000000000016c7000004ff0009000004ff0008
000097f5008af20b00000000000000000 lqid=0:4:3 st=2097152
tr= '00000000000016c70008' NYDSnydb comlen=406
insert into dbo.emp_queue
(emp_id, emp_first, emp_last, emp_title, emp_date, id)
values ('123456789', 'Gen', 'Davis',
        'Process Engineer', '1/1/95', 1111)
I. 2006/07/05 08:27:05. ENTRY ver=1100 len=204 orig=102
lorig=0 oqid=00000000000016c7000004ff000a000004ff0008
000097f5008af20b00000000000000000 lqid=0:4:4 st=1
tr= '00000000000016c70008' NYDSnydb comlen=85
commit transaction
I. 2006/07/05 08:27:05. Begin Transaction
Origin User=sa Tran Name=_ins
I. 2006/07/05 08:27:05. ENTRY ver=1100 len=208 orig=102
lorig=0 oqid=00000000000016c7000004ff000b000004ff000b
000097f5008af20b00000000000000000 lqid=0:4:5 st=4
tr= '00000000000016c7000b' NYDSnydb comlen=91
begin transaction
I. 2006/07/05 08:27:05. ENTRY ver=1100 len=528 orig=102
lorig=0 oqid=00000000000016c7000004ff000c000004ff000b
000097f5008af20b00000000000000000 lqid=0:4:6 st=2097152
tr= '00000000000016c7000b' NYDSnydb comlen=409
insert into dbo.emp_queue
(emp_id, emp_first, emp_last, emp_title, emp_date, id)
values ('987654321', 'Ben', 'Irvine',
        'Microwave Engineer', '3/5/94', 2222)
I. 2006/07/05 08:27:05. ENTRY ver=1100 len=204 orig=102
lorig=0 oqid=00000000000016c7000004ff000d000004ff000b
000097f5008af20b00000000000000000 lqid=0:4:7 st=1
tr= '00000000000016c7000b' NYDSnydb comlen=85
commit transaction
I. 2006/07/05 08:27:05. Begin Transaction
Origin User=sa Tran Name=_grrev
I. 2006/07/05 08:27:05. ENTRY ver=1100 len=212 orig=102
lorig=0 oqid=00000000000016c7000004ff000e000004ff000e
000097f5008af20b00000000000000000 lqid=0:4:8 st=4
tr= '00000000000016c7000e' NYDSnydb comlen=93
begin transaction
I. 2006/07/05 08:27:05. ENTRY ver=1100 len=204 orig=102
```

```
lorig=0 oqid=00000000000016d2000005010016000004ff000e
000097f5008af20b00000000000000000 lqid=0:4:9 st=1
tr= '00000000000016c7000e' NYDSnydb comlen=85
commit transaction
```

Explanation

- Q_type=1
An inbound queue.
- Tran Name=rs_logexec
Name of the transaction sp_setreplicate.
- Tran Name=_ins
Adaptive Server uses transaction name _ins for implicit insert transactions.
- Tran Name=_grrev
The grant and revoke permission commands get a special transaction name from the Adaptive Server.

Note You will see all begin, commit, and rollback commands regardless of whether or not the transactions contain changes that need to be replicated. When the RepAgent reads the log, it does not know whether or not the transaction includes changes to objects that are marked for replication. Because future versions of Replication Server may change this scheme to optimize for network traffic, you should check your current version of Replication Server for this behavior.

Example 3: inbound queue

In this example, this update command was executed at the primary database:

```
update emp_queue
  set emp_first = "General"
  where emp_id = "123456789"
```

Execute the sysadmin dump_queue command to view the inbound queue of the primary Replication Server:

```
sysadmin dump_queue, 102,1,-1,1,-2
```

Output

```

I. 2006/07/05 08:29:52. BLOCK BEGIN
q_number=102 q_type=1 blk=0:5 cnt=3
I. 2006/07/05 08:29:52. Begin Transaction
Origin User=sa Tran Name=_upd
I. 2006/07/05 08:29:52. ENTRY ver=1100 len=208 orig=102
lorig=0 oqid=00000000000016e3000005020002000005020002
000097f5008bedfe0000000000000000 lqid=0:5:0 st=4
tr= '00000000000016e30002' NYDSnydb comlen=91
begin transaction
I. 2006/07/05 08:29:52. ENTRY ver=1100 len=568 orig=102
lorig=0 oqid=00000000000016e3000005020004000005020002
000097f5008bedfe0000000000000000 lqid=0:5:1 st=2097152
tr= '00000000000016e30002' NYDSnydb comlen=451
update dbo.emp_queue
    set emp_id='123456789', emp_first='General',
        emp_last='Davis', emp_title='Process Engineer',
        emp_date='1/1/95', id=1111 where id=1111
I. 2006/07/05 08:29:52. ENTRY ver=1100 len=204 orig=102
lorig=0 oqid=00000000000016e3000005020005000005020002
000097f5008bedfe0000000000000000 lqid=0:5:2 st=1
tr= '00000000000016e30002' NYDSnydb comlen=85
commit transaction

```

Explanation

- Tran Name=_upd

Adaptive Server uses transaction name upd for implicit update transactions. RepAgent reads this name from the log.

Example 4: outbound queue after update

In this example, an update has been performed at the primary database. The minimal column feature has not been set for the replication definition:

```

update emp_queue
    set emp_first = "General"
    where emp_id = "123456789"

```

To view the outbound queue or the replicate:

```

sysadmin dump_queue, 103, 0, -1,1,-2

```

Output

```

I. 2006/07/05 08:31:32. BLOCK BEGIN

```

```
q_number=103 q_type=0 blk=0:3 cnt=3
I. 2006/07/05 08:31:32. Begin Transaction
Origin User=sa Tran Name=_upd
I. 2006/07/05 08:31:32. ENTRY ver=1100 len=204 orig=102
lorig=0 oqid=00000000000016e3000005020005000005020002
000097f5008bedfe000000000000001 lqid=0:3:0 st=4
tr='00000000000016e30002' NYDSnydb comlen=88
begin transaction
I. 2006/07/05 08:31:32. ENTRY ver=1100 len=268 orig=102
lorig=0 oqid=00000000000016e3000005020005000005020002
000097f5008bedfe000000000000002 lqid=0:3:1 st=0
tr= '00000000000016e30002' NYDSnydb comlen=149
update emp_queue
    set emp_id='123456789', emp_first='General',
        emp_last='Davis', emp_title='Process Engineer',
        emp_date='1/1/95', id=1111
    where id=1111
I. 2006/07/05 08:31:32. ENTRY ver=1100 len=180 orig=102
lorig=0 oqid=00000000000016e3000005020005000005020002
000097f5008bedfe0000000000000003 lqid=0:3:2 st=1
tr= '00000000000016e30002' NYDSnydb comlen=63
commit transaction
```

Explanation

- cnt=3
There are 3 rows in this block.
- Tran Name=_upd
If you perform an update and did not explicitly start a transaction, Adaptive Server uses _upd as the transaction name. Likewise, Adaptive Server uses _del for a delete command and _ins for an insert command.
- st=4, st=0, st=1
Status for begin, other, and commit.

Using sysadmin sqt_dump_queue

Use this command to dump contents of the transaction cache for each inbound queue. sysadmin sqt_dump_queue dumps all the statements of transactions that have not yet been deleted.

Commands are dumped into the Replication Server log or an alternate log file created using the `sysadmin dump_file` command. In the current version of Replication Server, you cannot dump the transaction cache into the RSSD.

Before using this command, make sure the transaction cache for the database exists using `admin who, sqt`.

Output from `sysadmin sqt_dump_queue` shows the state of transactions in the transaction cache: Open, Closed, or Read. Open transactions do not yet have a commit. Closed transactions have a commit but have not yet been completely read. Read transactions have been completely read but have not yet been deleted.

Following is the syntax for the `sysadmin dump_queue` command:

```
sysadmin sqt_dump_queue, q_number, q_type, reader [, open]
```

q_number, q_type – identifies the stable queue to dump. These values can be found by using the `admin who` command, the `admin who, sqm` command, and the `admin who, sqt` command.

reader – identifies the reader you want to dump the stable queue. This parameter applies to features that require multiple readers, such as warm standby applications. You can get the reader number by running `admin sqm_readers` or `admin who, sqt`. If you are not using multiple readers, enter “0.”

open – allows you to dump only open transactions. If you use this option, insert a comma before it.

Examples

The following examples demonstrate the use of `sysadmin sqt_dump_queue`.

- This command acts on inbound queue 103:1. All undeleted transactions in the queue are dumped from the transaction cache:

```
sysadmin sqt_dump_queue, 103, 1, 0
```

- The following series of commands dumps open transactions into the file *qdump* from the transaction cache for queue 103:1. The last command closes *qdump*:

```
sysadmin dump_file, qdump
sysadmin sqt_dump_queue, 103, 1, 0, open
sysadmin dump_file
```

- In an `isql` session against the NYDS.nydb database, run a `begin tran` test to start an open transaction. `admin who, sqt`

```
Spid      State                Info
-----  -
      20      Awaiting Wakeup     102:1 NYDS.nydb
```

```

Closed      Read      Open      Trunc      Removed
-----
          0          0          1          1          0

Full      SQM Blocked      First Trans
-----
          0          1      st:O,cmds:1,qid:0:42:0

Parsed
-----
          0

```

The output of the command shows one queue with an open transaction:

- Running `sysadmin sqt_dump_queue,102,1,0` returns the dump output:


```

I. 96/06/28 17:18:50. TRANSACTION CACHE for 102:1
I. 96/06/28 17:18:51. TRAN_OPEN TID='0000000100000a'
g '0013' NYDSnydb Origin=102. 1 commands. in memory
I. 96/06/28 17:18:51. Tran Name 'test' User:'sa'
I. 96/06/28 17:18:51. LQID 0:42:0 begin transaction
test
I. 96/06/28 17:18:51. END TRANSACTION CACHE for
102:1

```

Using sysadmin dump_file

Use this command to specify an alternate log file to be used by the `sysadmin dump_queue` and `sysadmin sqt_dump_queue` commands for output.

Here is the syntax for the `sysadmin dump_file` command:

```
sysadmin dump_file [, file_name]
```

where *file_name* is the local name of the file where dumps will be directed.

If you omit *file_name*, the current dump file is closed. If you specify *file_name* and a dump file currently exists, then the current dump file is closed and a new file is opened. The new file uses the name specified with the optional *file_name* entry.

Examples

These examples demonstrate how to use `sysadmin dump_file`.

- This command dumps the contents of queue 100:1 into the Replication Server log:

```
sysadmin dump_queue, 100, 1, -1, 1, -2
```

- The following series of commands use `sysadmin dump_file` to specify an alternate log file.

This command specifies *file1* as the alternate log file:

```
sysadmin dump_file, file1
```

This command outputs the contents of queue 100:1 into *file1*:

```
sysadmin dump_queue, 100, 1, -1, 1, -2
```

This command closes *file1* and opens *file2*. *file2* receives the output from subsequent dumps:

```
sysadmin dump_file, file2
```

This command dumps transaction statements from queue 100:1 into *file2*:

```
sysadmin sqt_dump_queue, 100, 1, 0
```

This command closes *file2*:

```
sysadmin dump_file
```

This command dumps the output from queue 103:1 into the Replication Server log:

```
sysadmin dump_queue, 103, 1, 0, 11, 63
```

sysadmin purge commands

Use the `sysadmin purge_all_open` command to purge all open transactions from a Replication Server inbound queue. Use `sysadmin purge_first_open` to purge only the first open transaction from a Replication Server inbound queue.

Replication Server needs sufficient storage to execute the `sysadmin purge` commands. If you do not have enough storage, the following error message may appear:

```
This RS is out of Disk Space. Use another session to add
disk space for this command to proceed.
```

If this occurs, start another `isql` session and add stable storage to the Replication Server. `sysadmin purge` commands cannot proceed until sufficient storage is available.

If you want to review the contents of the transactions being dropped, then execute the `sysadmin sqt_dump_queue` command prior to using the `sysadmin purge_all_open` or `sysadmin purge_first_open` command.

The syntax for the `sysadmin purge_all_open` command is:

```
sysadmin purge_all_open, q_number, q_type
```

where *q_number*, *q_type* identifies the stable queue to purge. These values can be found by using the `admin who` command, the `admin who, sqm` command, and the `admin who, sqt` command. Only inbound queues can be purged.

The syntax for the `sysadmin purge_first_open` command is:

```
sysadmin purge_first_open, q_number, q_type
```

where *q_number*, *q_type* identifies the stable queue to purge. These values can be found by using the `admin who` command, the `admin who, sqm` command, and the `admin who, sqt` command. Only inbound queues can be purged.

Examples

The next set of examples describes the use of `sysadmin purge_first_open` and `sysadmin purge_all_open`.

- The following command purges the first open transaction from the specified queue:

```
sysadmin purge_first_open, 103, 1
```

- This command purges all open transactions from the specified queue:

```
sysadmin purge_all_open, 103, 1
```

Language, Sort Order, and Character Set Issues

This chapter describes localization of messages into different languages, sort order, character sets, and the problems that can occur with each feature.

Name	Page
Overview	159
Message language	160
Sort order	161
Character sets	162

Overview

Replication Server provides support for international environments with:

- Localization of messages into English, French, German, and Japanese languages
- Support for all Sybase-supported character sets, with character set conversion between Replication Server sites
- Support for non-binary sort orders

Because Replication Server works in a distributed environment with many independent processes, you must set up its languages, sort orders, and character sets carefully to avoid problems. This appendix describes some of the problems that can arise when you do not set up these items according to the guidelines recommended in the *Replication Server Design Guide*.

Note RepAgent uses the same language, sort order, and character set settings as the Adaptive Server.

Message language

You can configure Replication Servers to print messages to the error log and to clients in English, and other foreign languages. This allows server programs in your replication system, including Replication Servers, and Adaptive Servers, to write messages to their error logs in the configured language. However, depending on the server, they may or may not send messages to a client in the client's language.

For example, Adaptive Server checks for the language setting of its client (Replication Server) and returns messages in that language. Replication Server, however, does not check for a client's language; instead, they return messages to a client in their own language. This can result in error logs with messages in different languages if the servers are configured in different languages.

To avoid the confusion that can result from a mixed-language error log, Sybase recommends that you configure the Replication Server so that all languages are the same.

Stored procedure messages

Stored procedures use the language of their Adaptive Server connection. Therefore, if you have a mixed-language environment, Adaptive Server stored procedure messages may be in a different language from the Replication Server.

For example, if you log in to Adaptive Server using French and execute a stored procedure, the stored procedure generates messages in French, even if the language setting of Replication Server is English or Japanese. If you log in to Adaptive Server with a language that is not installed at the Adaptive Server, Adaptive Server returns stored procedure messages in English.

Other situations that produce mixed languages

- In order to quickly and efficiently release EBFs and maintenance releases, there may be times when Sybase does not translate new or changed messages into French, German, or Japanese. If this is the case, the Replication Server, or rs_subcmp messages use the English version of the message.
- Message localization files may be misplaced. Verify that the files are in the correct directories.

Sort order

The sort order, or collation sequence, used by a server determines how character data and identifiers are compared and ordered. A sort order is determined by the character set.

Replication Server supports all Sybase-supported sort orders, including non-binary sort orders. Non-binary sort orders are necessary for the correct ordering of character data and identifiers in European languages.

Use a non-binary sort order if you have *both* of the following:

- Differing character sets between the primary and replicate data servers, and
- Data with 8-bit characters,

and you intend to do *either* of the following:

- Include columns with this data in the where clause of a subscription, or
- Query the database with an order by clause involving the columns with this data.

Sort order problems

Most sort order problems in Replication Server are a result of different sort orders being used in different servers throughout your system. If you are using different sort orders, you may experience the following problems:

- You receive a DSI error when trying to insert or update a record into a table with a unique index on a character column. Under the primary data server's sort order, the value is unique; but under the replicate data server's sort order, it is not unique. For example, "Ivis" and "LVIS" are considered distinct under a binary sort order but are considered equal under the nocase sort order.
- A drop subscription with purge command does not completely remove the rows at the replicate database. A query to select records to purge misses some records under the replicate data server's sort order.
- Replicated data is not what the replicate site expects.

When this occurs, a subscription probably has one or more character clauses. Updates during the lifetime of a subscription are processed with the primary Replication Server's sort order, leading to unexpected semantics at the replicate data server if the sort orders differ.

These problems can be corrected by editing the Replication Server configuration file to use the same sort order and, if necessary, by running `sybinit` to make the primary data server and replicate data server use this sort order as well.

Character sets

The following problems can arise when servers have different or incompatible character sets. These problems can be corrected by configuring all servers in your replication system to use compatible character sets.

- 8-bit or multibyte data is corrupted when replication is taking place between a multibyte and a single-byte data server.

Sybase does not support any character-set conversion between single and multibyte systems; it simply passes the data and object names as is. Since 8-bit and multibyte characters are character-set-specific, their semantics change after replication.

- Character data in the replicate data server has some question marks (?) substituted for non-ASCII characters, even though the primary and replicate data servers use compatible character sets. Also, object names originating at a remote site have some question mark (?) characters.

This happens because different character sets, even compatible ones, have some unique characters. When trying to convert a character that does not exist in the replicate data server's character set, a question mark (?) is substituted for the unrecognized character.

If you are replicating between compatible character sets (for example, `iso-1` and `cp850`), make sure that your object names and character data do not include any of the characters that are not common to both character sets.

Index

Symbols

@ xvi
_ xvi

Numerics

8-bit data 162

A

Adaptive Server
 adding data servers 3
 error logs 9
 errors 112
 errors causing Replication Server errors 26
 errors, correcting 39
alter route command 69
analyzing error logs 9
analyzing Replication Server error messages 28
asynchronous commands, errors from 26, 121
asynchronous transaction failures 121
atomic materialization 82, 83
 determining if subscription needs recovery 82
 monitoring 82
automatic error correction 121

B

binaries, changing 3
binary sort orders 161
bulk dematerialization 96, 99, 100
bulk materialization 92

C

cache for RSSD too small 54
case in RCL commands xv
character sets
 conversion of 162
 guidelines 162
 problems with 162
client applications, adding 4
command log 2
configuration problems 5
 troubleshooting 12
connect source command, not executed 51
connections
 DSI failures 120
 none 51
 resuming 120
 RSI errors 103
conventions
 examples xiii
 syntax statements xiv
conversion, character set 162
create route command 65
create subscription errors 79

D

data loss
 DSI 57
 RSI 57
data servers
 customizing error handling 123
 error logs 9
 error logs, non-Adaptive Server 26
 errors 121
database log, locator 130
databases
 finding maintenance user name and password 120
 large 4

Index

- listing 116
 - no connection to 51
 - skipping transactions for 121
 - truncation point 128
 - dbcc gettrunc command 128
 - definition
 - identifiers xvi
 - dematerialization
 - bulk 99
 - bulk, using without purge option 100
 - deleting subscription data and 96
 - errors 96
 - options for 95
 - overview 95
 - purge, list of states 97
 - purging subscriptions and 96, 97
 - subscription problem, troubleshooting 15
 - without deleting subscription data 96, 100
 - detecting loss message 22
 - diagnostic tools, description of 11
 - DIST thread down 50
 - drop route with nowait command 72
 - drop subscription command 95
 - with purge option 96, 97
 - without purge option 96
 - dropping replicated objects 112
 - DSI
 - connection failures 120
 - data loss 57
 - detecting duplicates 124
 - rs_update_lastcommit not replicated 57
 - states 117
 - verifying for replicate 119
 - dumping
 - stable queues 141
 - transaction logs 43, 48
- ## E
- Embedded Replication Server System Database. *See* ERSSD
 - errmsg.rs 10
 - errmsg.sub 10
 - error block, definition 28
 - error correction, automatic 121
 - error logs
 - Adaptive Server 9
 - analyzing 9
 - data server 9
 - gateway 9, 26
 - mixed languages 160
 - networks 26
 - overview of analyzing 25
 - replication agents 9
 - Replication Server 9, 11, 29
 - RMS 135
 - rs_init 10
 - UAF 135
 - error messages
 - analyzing Replication Server 28
 - error block 28
 - fatal 26
 - format for Replication Server 27
 - informational 26
 - internal 26
 - multiple, for same problem 28
 - Open Client/Server 29
 - procedure for finding 9
 - RepAgent, analyzing 32
 - RepAgent, format 31
 - RepAgent, overview 31
 - RepAgent, state 31
 - Replication Server, description 37
 - rs_init, description 36
 - thread-terminated 26
 - warnings 26
 - error numbers 28
 - errors
 - 1028 38
 - 1033 59
 - 11061 46
 - 13045 48
 - 15020 50
 - 15040 51
 - 15052 51
 - 21 38
 - 28028 51
 - 29024 52
 - 37022 53
 - 37023 53
 - 7035 45
 - 9202 59

- 9210 60
- 9215 61
- asynchronous transactions 121
- connection failures 121
- creating routes 66
- data server 121
- data server, customizing handling 123
- error numbers 28
- from Adaptive Server 112
- normalization 109
- recoverable 110
- Replication Server out of memory 45
- RSI connection 103
- RSSD deadlocks 46
- `srv_spawn` failed 38
- ERSSD 2
- events causing problems 2
- examples
 - RepAgent error message 32
 - Replication Server error log 29
 - style conventions xiii
- exceptions log 125

F

- fatal errors 26
- function strings
 - incorrect variables 6
 - no match found 52

G

- gateway error logs 9, 26
- generation number, increasing 23

H

- hardware problems, identifying 2

I

- icons

- Adaptive Server xvii
 - client application xvii
 - Replication Agent xvii
 - Replication Manager xvii
 - Replication Server xvii
- identifiers
 - definition of xvi
 - format xvi
 - function parameters xvi
 - length xvi
 - types of xvi
- image columns
 - replication problem 22
 - replication status inconsistency 110
- inbound queue requires more disk space 133
- informational messages 26
- interfaces file
 - no destination Replication Server entry 67
 - no source Replication Server entry 66
- internal errors 26
- international environments, problems with 159, 162

K

- kernel, rebuilding 4
- key violation errors 121

L

- languages
 - configuring 160
 - mixed 160
 - stored procedure 160
- locator
 - database log 130
 - inconsistencies between primary and replicate 106
 - resetting 130
- log files
 - Replication Server 11, 26
 - `rs_init` 10
- log, RSSD 47
- log, transaction
 - dumping 43, 48
 - full

Index

truncating 43, 48
login, incorrect when creating route 67

M

maintenance user name, finding for database 120
matching function string, cannot find 52
materialization
 atomic 82, 83
 bulk 92
 description of 79
 methods 79
 nonatomic 87, 92
 schema inconsistency 81
 subscription problem, troubleshooting 14
memory_limit parameter, increasing 45
message language for stored procedures 160
multibyte data 162

N

network error logs 26
network problems
 identifying 2
non-ASCII characters replaced by ? 162
nonatomic materialization 87
non-binary sort orders 161
normalization errors 109
num_thread parameter, increasing 38

O

objects, replicated, dropping 112
Open Client/Server error messages 29
operating system problems
 identifying 2
 on Windows NT 2
 upgrading 4
origin_qid, incorrect 23
orphaned transactions 132

P

password, finding for database 120
performance problems, critical 3
permissions
 missing or incorrect 53, 80
 primary subscribe required for subscriptions 53
problems
 configuration 5
 hardware 2
 network 2
 operating system 2
 recovery, manual 9
 replication 7
 RMS 136
 subscriptions 6
 types of 5

Q

question marks replacing non-ASCII characters 162
queue data, viewing 2

R

recoverable normalization errors 110
recovery, manual
 generation number, increasing 23
 problems 9
 troubleshooting procedure for 23
rejecting messages after data loss 22
REP AGENT USER threads down 61
RepAgent
 adding 3
 error messages, analyzing 32
 error messages, format 31
 error messages, overview 31
 error messages, severity of 31
 error messages, state 31
 nested stored procedures 59
 problems starting 108
replicated objects, dropping 112
replication
 definitions not found 50
 determining where it stopped 22

- failures, troubleshooting 16
- image column not replicating 22
- problems 7
- text column not replicating 22
- unitext column not replicating 22
- replication agents
 - adding 3
 - error logs, non-Adaptive Server 26
 - error logs, role in troubleshooting 9
- Replication Manager. *See* RM
- Replication Monitoring Services. *See* RMS
- Replication Server
 - adding 3
 - common errors 54
 - common informational and warning messages 59
 - diagnostic version, running 22
 - error logs, overview of analyzing 25
 - error logs, reading 29
 - error logs, role in troubleshooting 9, 11
 - error message descriptions 37
 - error message format 27
 - error message information file 10
 - errors caused by Adaptive Server errors 26
 - errors, types of 26
 - incorrect login when creating route 67
 - interfaces file, no entry 66, 67
 - intermediate for indirect route 66
- Replication Server error message
 - connect source** not executed 51
 - create object** required 53
 - data server errors 38
 - function strings not matched 52
 - no database connection 51
 - out of memory 45
 - primary subscribe** required 53
 - replication definition not found 50
 - RSSD deadlocks 46
 - RSSD log device full 47
 - RSSD restarted 48
 - same primary and replicate database 51
 - srv_spawn** failed 38
- Replication Server System Database. *See* RSSD
- replication status inconsistency
 - image columns 110
 - text columns 110
 - unitext columns 110
- repsrvr.diag, using 22
- resource contention 3
- resume connection command
 - skip transaction option and 96
 - using 120
- RM
 - command log 2
 - queue data, viewing 2
- RMS
 - ASA Replication Agent, and 139
 - dbltm, and 139
 - error logs 135
 - heartbeat with Replication Server 12.6 138
 - overview 135
 - problems, common 136
 - tracing feature 136
 - UAF version 2.0, and 137
- routes
 - altering 69
 - cleaning up after drop route with nowait 72
 - create route protocol message waiting 68
 - creating system table subscriptions 69
 - creating, after dropped 66
 - creating, problems with 65
 - drop route protocol in queue 71
 - drop route protocol not sent 71
 - drop route protocol waiting for delivery 71
 - dropping 70, 73
 - error messages while creating 66
 - incorrect login at destination 67
 - indirect, creating 66
 - procedure for checking problems 23
 - protocol message not sent 68
 - rs_helproute diagnostic messages 67, 69
 - system table subscriptions, creating 69
 - system table subscriptions, dropping 71
 - troubleshooting overview 63
- rows in system table, swapped out 54
- rs_helproute stored procedure
 - diagnostic messages 67, 69
 - protocol message not sent 68
- rs_init
 - common errors 37
 - error logs 10
 - error message descriptions 36
 - problems 5

Index

- rs_init error
 - Adaptive Server entry not found 36
 - unknown host name 37
- rs_lastcommit system table 23
- rs_marker stored procedure 68
- rs_oqid system table 22
- rs_recovery system table
 - recoverable actions 22
- rs_subcmp
 - error message information file 10
 - mixed languages 160
- rs_update_lastcommit 57
- RSI
 - connection errors 103
 - data loss 57
 - locator inconsistencies 106
- RSSD
 - deadlocks 46
 - log device full 46, 47
 - restarted 48
 - routes, creating 69
 - routes, dropping 71
 - system table rows swapped out 54
 - system tables, checking 22
- S**
 - schema inconsistency 81
 - set autocorrection command, key violations 121
 - severity of RepAgent error messages 31
 - single-byte data 162
 - skipping transactions 40
 - sort orders
 - binary vs. non-binary 161
 - described 161
 - problems with 161
 - SQT thread, down 50
 - stable queues
 - dumping 141
 - dumps, using 22
 - inbound, requires more disk space 133
 - stack traces on Windows NT 2
 - standard error output. *See* stderr
 - state of RepAgent error messages 31
 - stderr
 - description 26
 - role in troubleshooting 9
 - stored procedures
 - connection failures 121
 - messages in mixed languages 160
 - nested 59
 - subscriptions
 - connect source command, not executed 51
 - dematerialization 95, 100
 - dematerialization of 95
 - failures, troubleshooting 13
 - maintenance user permission 81
 - missing rs_select function string 81
 - monitoring 76
 - no database connection 51
 - overview 75
 - permissions, missing or incorrect 53, 80
 - primary subscribe required 53
 - problems 6
 - replication definition not found 50
 - RSSD log device full 47
 - status 100
 - update permission and maintenance user 81
 - verifying login account requirements for 80
- Sybase Unified Agent Framework. *See* UAF
- syntax conventions
 - identifiers xvi
 - syntax statements, conventions xiv
- sysadmin dump_file command 156
- sysadmin dump_queue command 143
- sysadmin purge_all_open command 157
- sysadmin purge_first_open command 157
- sysadmin sqt_dump_queue command 154
- system tables
 - checking 22
 - rows swapped out of 54
 - rs_oqid system table, checking 22
 - rs_recovery system table, checking 22
- T**
 - tasks causing problems 2
 - text column
 - replication problem 22
 - text columns

- replication status inconsistency 110
- thread-terminated messages 26
- trace flags
 - Replication Server 22
 - RMS 136
- transaction log. *See* log, transaction
- transactions
 - asynchronous 121
 - large 4
 - nested 59
 - open 4
 - orphaned 132
 - skipping 40
 - viewing header information of 125
- truncating
 - databases 128
 - problems caused by 131
 - transaction log 43, 48
- truncation point
 - checking 128
 - setting 131
 - turning off 129
- types of problems 5

U

- UAF
 - error logs 135
 - multiple servers 137
 - overview 135
 - version 2.0 and RMS 137
- Unified Agent Framework. *See* UAF
- unique-key violations 121
- unitext columns
 - replication problem 22
 - replication status inconsistency 110
- upgrading
 - Adaptive Server 3
 - Replication Server components 3

V

- variables, incorrect for function string 6

W

- warnings 26
- Windows NT operating system problems 2

