

SYBASE®

Administration Guide

Sybase® Replication Agent™

12.6

[Linux, Microsoft Windows, UNIX]

DOCUMENT ID: DC38261-01-1260-01

LAST REVISED: September 2005

Copyright © 1998-2005 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, the 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, Adaptive Warehouse, Afaia, Answers Anywhere, Anywhere Studio, Application Manager, AppModeler, APT Workbench, APT-Build, APT-Edit, APT-Execute, APT-Translator, APT-Library, AvantGo Mobile Delivery, AvantGo Mobile Inspection, AvantGo Mobile Marketing Channel, AvantGo Mobile Pharma, AvantGo Mobile Sales, AvantGo Pylon, AvantGo Pylon Application Server, AvantGo Pylon Conduit, AvantGo Pylon PIM Server, AvantGo Pylon Pro, Backup Server, BizTracker, ClearConnect, Client-Library, Client Services, Convoy/DM, Copernicus, Data Pipeline, Data Workbench, DataArchitect, Database Analyzer, DataExpress, DataServer, DataWindow, DataWindow .NET, DB-Library, dbQueue, Developers Workbench, DirectConnect, DirectConnect Anywhere, Distribution Director, e-ADK, E-Anywhere, e-Biz Impact, e-Biz Integrator, E-Whatever, EC Gateway, ECMAP, ECRTP, eFulfillment Accelerator, Embedded SQL, EMS, Enterprise Application Studio, Enterprise Client/Server, Enterprise Connect, Enterprise Data Studio, Enterprise Manager, Enterprise SQL Server Manager, Enterprise Work Architecture, Enterprise Work Designer, Enterprise Work Modeler, eProcurement Accelerator, EWA, Financial Fusion, Financial Fusion Server, Gateway Manager, GlobalFIX, iAnywhere, iAnywhere Solutions, ImpactNow, Industry Warehouse Studio, InfoMaker, Information Anywhere, Information Everywhere, InformationConnect, InternetBuilder, iScript, Jaguar CTS, jConnect for JDBC, M2M Anywhere, Mach Desktop, Mail Anywhere Studio, Mainframe Connect, Maintenance Express, Manage Anywhere Studio, M-Business Channel, M-Business Network, M-Business Server, MDI Access Server, MDI Database Gateway, media.splash, MetaWorks, mFolio, Mirror Activator, MySupport, Net-Gateway, Net-Library, New Era of Networks, ObjectConnect, ObjectCycle, OmniConnect, OmniSQL Access Module, OmniSQL Toolkit, Open Biz, Open Client, Open Client/Connect, Open Client/Server, Open Client/Server Interfaces, Open Gateway, Open Server, Open ServerConnect, Open Solutions, Optima++, PB-Gen, PC APT Execute, PC DB-Net, PC Net Library, PocketBuilder, Pocket PowerBuilder, Power++, power.stop, PowerAMC, PowerBuilder, PowerBuilder Foundation Class Library, PowerDesigner, PowerDimensions, PowerDynamo, PowerScript, PowerSite, PowerSocket, Powersoft, PowerStage, PowerStudio, PowerTips, Powersoft Portfolio, Powersoft Professional, PowerWare Desktop, PowerWare Enterprise, ProcessAnalyst, QAnywhere, Rapport, RemoteWare, RepConnector, Replication Agent, Replication Driver, Replication Server, Replication Server Manager, Replication Toolkit, Report-Execute, Report Workbench, Resource Manager, RFID Anywhere, RW-DisplayLib, RW-Library, S-Designer, SDF, Search Anywhere, Secure SQL Server, Secure SQL Toolset, Security Guardian, SKILS, smart.partners, smart.parts, smart.script, SOA Anywhere, 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, STEP, SupportNow, S.W.I.F.T. Message Format Libraries, Sybase Central, Sybase Client/Server Interfaces, Sybase Financial Server, Sybase Gateways, Sybase IQ, Sybase MPP, Sybase SQL Desktop, Sybase SQL Lifecycle, Sybase SQL Workgroup, Sybase User Workbench, SybaseWare, Syber Financial, SyberAssist, SybFlex, SyBooks, System 10, System 11, System XI (logo), SystemTools, Tabular Data Stream, TradeForce, Transact-SQL, Translation Toolkit, UltraLite, UltraLite.NET, UNIBOM, Unilib, Uninull, Unisep, Unistring, URK Runtime Kit for UniCode, 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, and XP Server are trademarks of Sybase, Inc. 06/05

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	vii
CHAPTER 1	Introduction to Sybase Replication Agent..... 1
	Basic replication system concepts 1
	Transaction replication 2
	Replication system components..... 2
	Understanding Sybase Replication Agent..... 5
	Replication Agent instances 6
	Replication Agent communications 6
	Replication Agent components..... 7
CHAPTER 2	Setup and Configuration 11
	The Replication Agent instance 11
	Sybase Replication Agent instance directories 12
	Using Sybase Replication Agent utilities..... 13
	Setting the SYBASE environment 14
	Using the command line interface 15
	Using the Administrator GUI utility 28
	Starting the Sybase Replication Agent..... 32
	Start-up requirements..... 32
	Setting character sets..... 33
	Starting an instance with the ra utility..... 36
	Starting an instance with the Administrator GUI..... 37
	Starting an instance with the RUN script..... 39
	Using the Replication Agent administration port 40
	Creating an entry in the interfaces file 40
	Logging in to the Replication Agent..... 41
	Creating the Replication Agent administrator login 42
	Setting up Replication Agent connectivity 43
	Creating the primary database user login name 44
	Creating the Replication Server user login name 45
	Creating the RSSD user login name 46
	Setting up the connection configuration parameters 48

- Testing network connectivity 52
- Initializing the Replication Agent transaction log 54
- Marking objects in the primary database 56
 - Marking tables in the primary database 57
 - Marking stored procedures in the primary database 58
 - Enabling replication for LOB columns 60
 - Enabling replication for DDL 61
- Starting replication 62

CHAPTER 3

- Administering Sybase Replication Agent 65**
 - Determining current Replication Agent status 65
 - Understanding Replication Agent states 66
 - Changing the Replication Agent state 68
 - Getting Replication Agent statistics 69
 - Shutting down the Replication Agent instance 69
 - Starting replication in the Replication Agent 71
 - Stopping replication in the Replication Agent 72
 - Quiescing the Replication Agent 73
 - Suspending the Replication Agent instance 74
 - Managing the Replication Agent transaction log 74
 - Initializing the Replication Agent transaction log 76
 - Removing the Replication Agent transaction log 78
 - Forcing transaction log removal 81
 - Truncating the Replication Agent transaction log 81
 - Backing up Replication Agent objects in the primary database 83
 - Managing the Replication Agent System Database 84
 - RASD overview 84
 - Updating the RASD 86
 - Backing up the RASD 88
 - Restoring the RASD 88
 - Truncating the RASD 90
 - Identifying replicated transactions and procedures 91
 - Preparing to mark tables or stored procedures 92
 - Marking and unmarking tables 92
 - Enabling and disabling replication for marked tables 99
 - Enabling and disabling replication for LOB columns 101
 - Marking and unmarking stored procedures 105
 - Enabling and disabling replication for stored procedures 110
 - Enabling and disabling replication for DDL 112
 - Configuring and tuning the Replication Agent 114
 - Configuring Replication Agent 120
 - Customizing tuning 120

CHAPTER 4	Troubleshooting Sybase Replication Agent.....	123
	Diagnosing command errors and replication errors	123
	Troubleshooting specific command errors	124
	Connection refused	124
	Examining the Replication Agent when replication failure occurs	125
	Verify that primary database objects are marked for replication ..	125
	Checking the Replication Agent status.....	126
	Examining the Replication Agent logs.....	128
	Using the ra_statistics command to troubleshoot.....	130
	Running out of memory	131
	Checking the Replication Server.....	133
	Check replication definitions and subscriptions.....	134
	Replication Server status and operation	134
	Replication Agent login in Replication Server	135
	Replication Server stable queues.....	136
APPENDIX A	Materializing Subscriptions to Primary Data.....	137
	Understanding materialization.....	137
	Bulk materialization overview	138
	Unloading data from a primary database	139
	Loading data into replicate databases	139
	Atomic bulk materialization	139
	Preparing for atomic bulk materialization	140
	Atomic bulk materialization procedure	140
	Nonatomic bulk materialization	143
	Preparing for nonatomic bulk materialization	143
	Nonatomic bulk materialization procedure	144
	Glossary	147
	Index	155

About This Book

Sybase® Replication Agent™ version 12.6 extends the capabilities of Replication Server® to support the following non-Sybase primary data servers in a Sybase replication system:

- DB2 Universal Database (on UNIX and Microsoft Windows platforms)
- Informix Dynamic Server
- Microsoft SQL Server
- Oracle Database Server

Audience

This book is for anyone who needs to manage or administer a Sybase replication system with non-Sybase primary databases, or administer the non-Sybase primary databases in a Sybase replication system. This may include:

- Database Administrators
- Network Administrators
- System Administrators

How to use this book

Use the Sybase Replication Agent *Administration Guide* to find an overview of the Sybase Replication Agent, and detailed information about configuring and administering Replication Agent instances, and other components in a Sybase replication system.

This book is organized as follows:

Chapter 1, “Introduction to Sybase Replication Agent,” provides an introduction to replication system concepts and an overview of the Sybase Replication Agent. This chapter describes Replication Agent components and explains how they work.

Chapter 2, “Setup and Configuration,” describes how to set up and configure Sybase Replication Agent. The procedures described in this chapter must be performed after installing the software, and before replication can begin.

Chapter 3, “Administering Sybase Replication Agent,” describes administrative operations, including managing Replication Agent instances and using Replication Agent commands to perform a variety of routine tasks.

Chapter 4, “Troubleshooting Sybase Replication Agent,” describes basic Replication Agent troubleshooting and system recovery procedures.

Appendix A, “Materializing Subscriptions to Primary Data,” describes the materialization process and provides a detailed procedure for materializing subscriptions to primary tables.

Related documents

Sybase Replication Agent Refer to the following documents to learn more about the Sybase Replication Agent:

- Sybase Replication Agent *Reference Manual* – for information about all Replication Agent commands and configuration parameters, including syntax, examples, and detailed command usage notes
- Sybase Replication Agent *Primary Database Guide* – for detailed, database-specific information about each non-Sybase database that is supported by the Sybase Replication Agent
- Sybase Replication Agent *Installation Guide* – for information about installing the Sybase Replication Agent software
- The Sybase Replication Agent release bulletin – for last-minute information that was too late to be included in the books

Note A more recent version of the Sybase Replication Agent 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 Web site.

Replication Server Refer to the following documents for more information about transaction replication systems and the Replication Server software:

- Replication Server *Design Guide* – for an introduction to basic transaction replication concepts and Sybase replication technology
- Replication Server *Heterogeneous Replication Guide* – for detailed information about configuring Replication Server and implementing a Sybase replication system with non-Sybase databases

Primary data server Make sure that you have appropriate documentation for the non-Sybase primary data server that you use with the Sybase replication system.

Java environment The Sybase Replication Agent requires a Java Runtime Environment (JRE) on the Replication Agent host machine.

- The Sybase Replication Agent release bulletin contains the most up-to-date information about Java and JRE requirements.
- Java documentation available from your operating system vendor describes how to set up and manage the Java environment on your platform.

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.

❖ To find 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.

❖ **To find 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.

❖ **To create 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

❖ **To find 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.

Style conventions

The following style conventions are used in this book:

- In a sample screen display, commands that you should enter exactly as shown appear like this:

```
pdb_init
```

- In the regular text of this document, variables or user-supplied words appear like this:

Specify the *value* option to change the setting of the configuration parameter.

- In a sample screen display, variables or words that you should replace with the appropriate value for your site appear like this:

```
resume connection to pds.pdb
```

where *pds* and *pdb* are the variables you should replace.

- In the regular text of this document, names of programs, utilities, procedures, and commands appear like this:

Use the `pdb_init` command to initialize the primary database.

- In the regular text of this document, names of database objects (tables, columns, stored procedures, etc.) appear like this:

Check the price column in the widgets table.

- In the regular text of this document, names of datatypes appear like this:

Use the date or datetime datatype.

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

Log files are located in the `$SYBASE/RAX-12_6/inst_name/log` directory.

Syntax conventions

The following syntax conventions are used in this book:

Table 1: Syntax conventions

Key	Definition
{ }	Curly braces indicate that you must choose at least one of the enclosed options. Do not type the braces when you enter the command.
[]	Brackets mean that choosing one or more of the enclosed options is optional. Do not type the brackets when you enter the command.
()	Parentheses are to be typed as part of the command.
	The vertical bar means you can select only one of the options shown.
,	The comma means you can choose as many of the options shown as you like, separating your choices with commas that you type as part of the command.

In reference sections of this document, statements that show the syntax of commands appear like this:

```
ra_config [param [, value]]
```

The words *param* and *value* in the syntax are variables or user-supplied words.

The following character case conventions are used in this book:

- All command syntax and command examples are shown in lowercase. However, Sybase Replication Agent command names are *not* case sensitive. For example, RA_CONFIG, Ra_Config, and ra_config are equivalent.
- Names of configuration parameters are case sensitive. For example, Scan_Sleep_Max is not the same as scan_sleep_max, and the former would be interpreted as an invalid parameter name.
- Database object names are *not* case sensitive in Replication Agent commands. However, if you need to use a mixed-case object name in a Replication Agent command (to match a mixed-case object name in the primary database), you must delimit the object name with quote characters. For example:

```
pdb_get_tables "TableName"
```

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.

Character case conventions

Accessibility features

Sybase Replication Agent version 12.6 and the HTML documentation have 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.

For a Section 508 compliance statement for Sybase Replication Agent version 12.6, see Sybase Accessibility at http://www.sybase.com/detail_list?id=52484.

If you need help

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



Introduction to Sybase Replication Agent

Sybase Replication Agent extends the capabilities of Replication Server® by allowing non-Sybase (heterogeneous) data servers to act as primary data servers in a replication system based on Sybase replication technology.

This chapter provides an introduction to and overview of the Sybase Replication Agent.

Topic	Page
Basic replication system concepts	1
Understanding Sybase Replication Agent	5

Basic replication system concepts

Transaction replication can be used to maintain data in separate databases called **replicate databases**. Replicate databases contain accurate, current copies or subsets of data from a **primary database**.

When a table in the primary database is marked for replication, transactions that change the data in that table are captured for replication. The primary database processes the transaction, and a copy of the transaction (including all its operations) is stored in the **transaction log**.

In the case of a stored procedure marked for replication, when the stored procedure is invoked in the primary database, all parameter values provided with the procedure invocation are captured and recorded in the transaction log. When a marked stored procedure generates a transaction that affects data in marked tables in the primary database, the transaction generated by the stored procedure is ignored, so only the procedure invocation is replicated.

Transaction replication

The events captured for replication through a Sybase replication system are referred to as *transactions*, even though they might not correspond directly to an actual transaction in the primary database. For example, if a transaction affects both marked tables and unmarked tables, only the operations that affect the marked tables are captured for replication. Operations on unmarked tables are ignored.

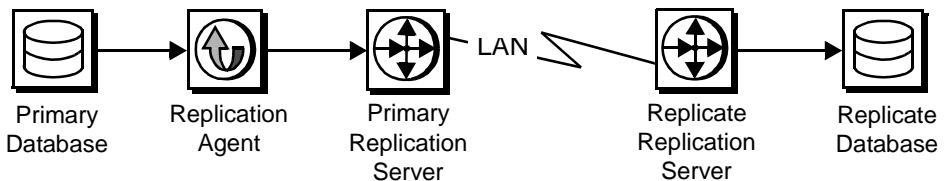
All data-change operations captured for replication exist within a *transaction context*. That is, only committed transaction operations are replicated; transactions that are rolled back are not replicated.

Even though the data-change events replicated through a Sybase replication system are really *operations*, those operations are grouped in an atomic collection, and they represent the results of a committed transaction in the primary database.

Replication system components

Figure 1-1 illustrates the basic components in a typical Sybase replication system.

Figure 1-1: Typical Sybase replication system



The following sections describe the primary-side components of a typical Sybase replication system:

- Primary databases
- Replication Agents
- Replication Servers

Primary databases

A primary database is the source of transactions that modify data in the replicate databases. Transactions are replicated by table or by procedure.

Tables marked for replication in a primary database are called primary tables. A primary table must be marked for replication so that the Replication Agent can identify and replicate the transactions that affect the data in that table.

Large-object (LOB) columns within a primary table must have replication enabled separately from the primary table. You can selectively replicate LOB columns within a primary table.

To replicate invocations of a stored procedure, the procedure must be marked for replication so that the Replication Agent can identify and replicate invocations of that procedure in the primary database.

Replication Agents

A **Replication Agent** is the Sybase replication system component that captures the replicated transactions in a primary database, and then sends those transactions to a Replication Server for distribution to replicate databases.

Sybase Replication Agent reads a transaction log in the primary database and generates Log Transfer Language (LTL) output. LTL is the language that Replication Server uses to process and distribute replicated transactions throughout a replication system.

Sybase Replication Agent can be configured to use information stored in the Replication Server System Database (RSSD) of the primary Replication Server to provide more sophisticated replication features and generate more efficient LTL.

There are two types of Sybase Replication Agents:

- Trigger-based
- Log-based

Trigger-based Replication Agents

Trigger-based Replication Agents use *triggers* on marked tables to capture the data involved in a replicated transaction. The triggers also record other information the Replication Agent needs to replicate the transaction, such as a transaction ID that identifies each operation associated with a transaction.

Stored procedure replication is accomplished by similar triggers embedded in the stored procedure code.

When fired, the triggers record the data (or procedure invocation) to be replicated in one or more transaction log tables in the primary database. The transaction log tables are user tables created and maintained by the Replication Agent.

The Replication Agent creates the data-capture triggers in the primary database when a table or stored procedure is marked for replication. The triggers are removed by the Replication Agent when the table or procedure is unmarked.

Sybase Replication Agent uses the trigger-based solution for the following primary databases:

- Informix
- Microsoft SQL Server

Log-based Replication Agents

Log-based Replication Agents retrieve the information they need for transaction replication from the native transaction log maintained by the primary data server.

Sybase Replication Agent uses the log-based solution for primary databases in following primary databases:

- DB2 Universal Database
- Oracle

Note Procedure replication is not available for DB2 Universal Database.

Replication Servers

The Replication Server that receives replicated transactions from a primary database (that is, directly from a Replication Agent) is referred to as the **primary Replication Server**. The Replication Server that sends replicated transactions to a replicate database is referred to as the **replicate Replication Server**.

Note In a simple replication system, a single Replication Server can act as both the primary Replication Server and the replicate Replication Server.

After it receives LTL from a Replication Agent, the primary Replication Server sends the replicated transaction to a replicate database, either directly or by way of a replicate Replication Server. The replicate Replication Server converts the replicated transaction from the LTL it receives to the native language of the replicate database, and then it sends the replicated transaction to the replicate data server for processing. When the replicated transaction is processed successfully by the replicate database, the replicate database is synchronized with the primary database.

Each Replication Server holds transaction operations in a stable queue and delivers them as soon as possible to other Replication Servers or replicate databases. By doing this, Replication Server guarantees that every transaction will be successfully received from a Replication Agent is guaranteed to be delivered to appropriately subscribing replicate databases.

Each Replication Server uses a database (called the RSSD) to store replication system data and metadata. Sybase Replication Agent can use some of the information stored in the RSSD to provide advanced replication features.

Understanding Sybase Replication Agent

Sybase Replication Agent supports transaction replication from a primary database through Replication Server. This section describes Sybase Replication Agent functionality in detail.

Note See the release bulletin for Sybase Replication Agent version 12.6 for information on the specific versions of DB2 Universal Database, Informix, Microsoft SQL Server, and Oracle supported by Sybase Replication Agent.

Sybase Replication Agent runs as a standalone application, independent of the primary data server, the primary Replication Server, and any other replication system components.

Trigger-based Replication Agents can reside on the same host machine as the primary database or any other replication system component, or they can reside on a machine separate from any other replication system components that has network access to the primary database. Log-based Replication Agents must execute on a machine that has access to the primary database's transaction log files.

Sybase Replication Agent is compatible with Replication Server Manager (RSM). Replication Agent instances can be configured, managed, and monitored by RSM. In addition, you can completely configure, manage, and monitor a Replication Agent instance using any Open Client™ application that is capable of communicating with the Sybase Tabular Data Stream™ (TDS) protocol (such as isql).

Replication Agent instances

An instance of the Sybase Replication Agent (a Replication Agent instance) must be created for each primary database from which you want to replicate transactions. Each Replication Agent instance is an independent application with its own configuration and log files, administration port, and connections to the primary database and the primary Replication Server.

Replication Agent instances created for a specific primary database type are referred to in this book as follows:

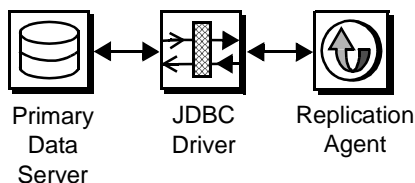
- IBM DB2 Universal Database – Replication Agent for UDB
- Informix Dynamic Server – Replication Agent for Informix
- Microsoft SQL Server (or SQL Server 2000) – Replication Agent for Microsoft SQL Server
- Oracle Database Server – Replication Agent for Oracle

Replication Agent communications

Sybase Replication Agent uses the Java Database Connectivity (JDBC) protocol for all communications. Some supported databases, however, require the Open Database Connectivity (ODBC) protocol. When connecting to a primary database, Sybase Replication Agent connects to either the JDBC driver or the JDBC/ODBC bridge provided by the database vendor.

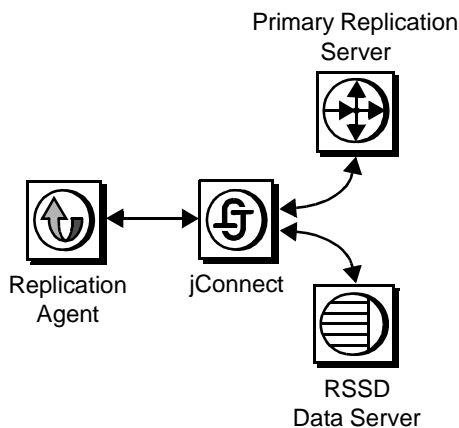
Figure 1-2 illustrates the communication between Sybase Replication Agent and a primary database using a JDBC driver.

Figure 1-2: Replication Agent primary database communication



Sybase Replication Agent uses the Sybase JDBC driver (jConnect™ for JDBC™) to communicate with all Open Client™ and Open Server™ applications, such as Adaptive Server® Enterprise and Replication Server. Each Replication Agent instance uses a single instance of jConnect for JDBC.

Figure 1-3 illustrates the communication between Sybase Replication Agent and the primary Replication Server and its RSSD.

Figure 1-3: Replication Agent communication with Replication Server

While replicating transactions, the Replication Agent maintains connections with both the primary database and the primary Replication Server, and it may occasionally connect to the RSSD of the primary Replication Server to retrieve replication definition data.

Replication Agent components

Sybase Replication Agent consists of a set of components that work together to perform all the operations required to propagate transactions from a primary database for replication.

The main Replication Agent components and their functions are:

- **Log Reader** – reads the transaction log in the primary database to retrieve transactions for replication.
- **Log Transfer Interface (LTI)** – generates Log Transfer Language (LTL) and sends it to the primary Replication Server.
- **Log Administrator** – administers the Replication Agent transaction log and manages transaction log objects.
- **Log Transfer Manager (LTM)** – manages all the other components and coordinates their operations and interactions.

Figure 1-4 illustrates the flow of data through a trigger-based Replication Agent during replication.

Figure 1-4: Replication Agent trigger-based operational data flow

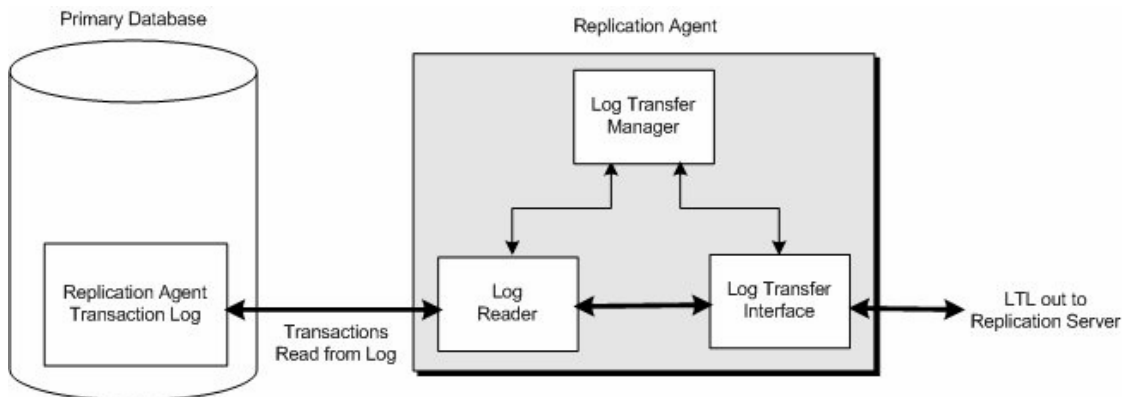
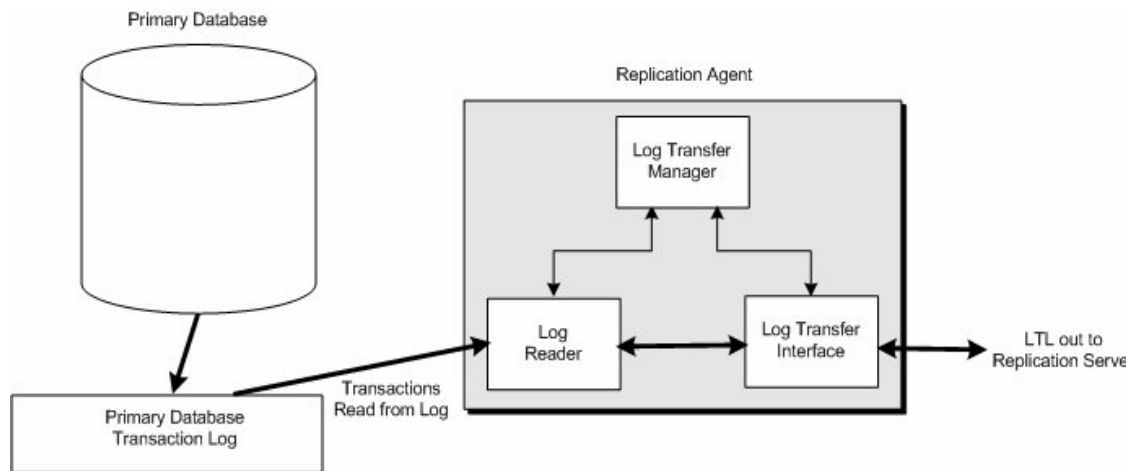


Figure 1-5 illustrates the flow of data through a log-based Replication Agent during replication.

Figure 1-5: Replication Agent log-based operational data flow



The Log Reader component retrieves transaction data from the transaction log in the primary database, generates change set data, and passes the change sets to the LTI. The LTI component processes the change set data from the Log Reader and generates the LTL to send to the primary Replication Server. The LTI component also receives messages from the primary Replication Server.

Although the LTM component is not involved in the flow of data from the primary database to the primary Replication Server, it coordinates the activities of the other Replication Agent components and processes any errors generated by those components.

Administration port

Sybase Replication Agent provides an administrative user interface through its *administration port*.

The administration port allows an Open Client application to log in to a Replication Agent instance, as if the Replication Agent were an Open Server application. Once logged in, the administrative client can issue commands to control, administer, and monitor the Replication Agent instance.

The administration port communicates with the client through the Sybase JDBC driver (jConnect for JDBC).

The administration port passes commands from the administrative client to the Replication Agent components. The administration port also processes the messages from Replication Agent components, and passes those messages out to the client.

Java requirement

Sybase Replication Agent version 12.6 requires a Java Runtime Environment (JRE) on the computer that will act as the Replication Agent host machine.

For more information on installing and setting up a JRE, see:

- Sybase Replication Agent *Installation Guide*
- The release bulletin for Sybase Replication Agent version 12.6

Setup and Configuration

This chapter describes how to set up Sybase Replication Agent after the software is installed, verify that your replication system is ready to replicate transactions, and start replication.

Topic	Page
The Replication Agent instance	11
Using Sybase Replication Agent utilities	13
Starting the Sybase Replication Agent	32
Using the Replication Agent administration port	40
Setting up Replication Agent connectivity	43
Testing network connectivity	52
Initializing the Replication Agent transaction log	54
Marking objects in the primary database	56
Starting replication	62

Note The procedures in this chapter assume you have already installed the Sybase Replication Agent software and Replication Server software, as described in the Sybase Replication Agent *Installation Guide* and the Replication Server installation and configuration guides for your platform.

The Replication Agent instance

After you install the Sybase Replication Agent software, you must create one instance of the Replication Agent for each primary database that you want to replicate transactions from.

Each Replication Agent instance is an independent process, with its own instance directories to house its configuration file, system log files, and script files. In addition, each Replication Agent instance creates some tables and stored procedures in the primary database. Replication Agent for Oracle also creates objects in the Replication Agent System Database (RASD). Each Replication Agent instance manages its own connections to the primary data server, primary Replication Server, and RSSD.

When you create a Replication Agent instance, you must specify:

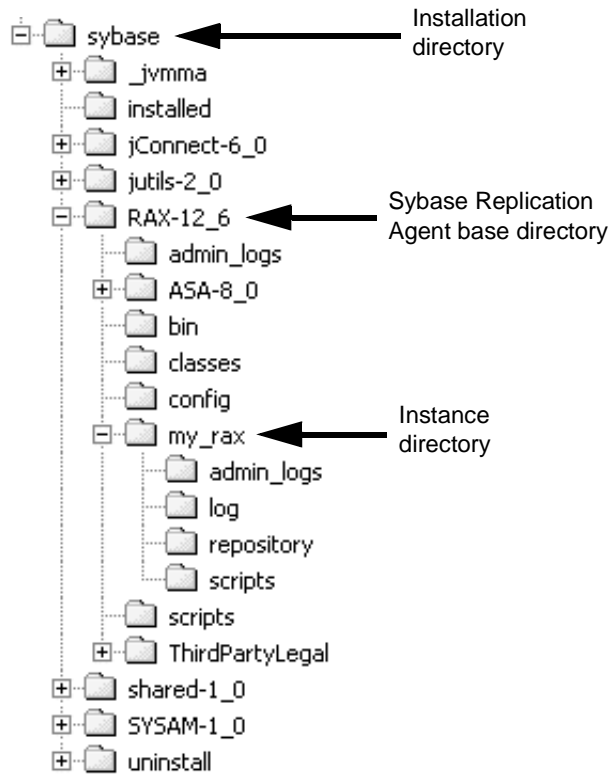
- A unique instance (server) name
- A unique client socket port number for its administration port
- The type of primary database the instance supports

You can create and run more than one Replication Agent instance on a single host machine, but each instance must have a unique name and a unique port number.

For more information, see “Creating a Replication Agent instance” on page 20.

Sybase Replication Agent instance directories

Figure 2-1 shows an example of the Sybase Replication Agent instance directories, which are created under the Sybase Replication Agent base directory when you create a Replication Agent instance.

Figure 2-1: Sybase Replication Agent directories

The Sybase Replication Agent base directory (*RAX-12_6*) and the installation directory (*sybase*) are created when you install the Sybase Replication Agent software.

Note A single installation (on a single host machine) can support multiple Replication Agent instances. Each instance directory resides under the Sybase Replication Agent base directory created when you install the software.

Using Sybase Replication Agent utilities

Three utilities are provided with the Sybase Replication Agent:

- `ra` – starts a Replication Agent instance, or returns the Sybase Replication Agent software version number.

Note On Windows platforms, when you execute a run script, you can leave off the extension: `ra -i my_rax`. However, on UNIX, you must always include the extension: `ra.sh -i my_rax`.

- `administrator` – allows you to use the Administrator GUI to create, list, copy, start, and delete Replication Agent instances.
- `ra_admin` – allows you to create, copy, verify, and delete Replication Agent instances, or list all verifiable installed Replication Agent instances on a machine.

Sybase Replication Agent utilities are supplied as batch files for Microsoft Windows platforms and shell scripts for UNIX platforms. The utility files reside in the *bin* subdirectory, under the Sybase Replication Agent base directory.

You can use the `-help` option with either the `ra_admin` or `ra` command line utility to obtain information about that utility.

For more information on using the command line utilities, see “Using the command line interface” on page 15.

For more information on using the graphical user interface utility, see “Using the Administrator GUI utility” on page 28.

Setting the SYBASE environment

Before you can invoke a Sybase Replication Agent utility, you must first:

- Log in to the operating system on the Replication Agent host machine, with a user login that has execute permission in the Sybase Replication Agent installation directory and all subdirectories (for example, the “sybase” user)
- Use the *SYBASE* environment script to make sure that the Sybase environment variables are set

The *SYBASE* environment script is supplied as a batch file for Microsoft Windows platforms (*SYBASE.bat*) and a shell script for UNIX platforms (*SYBASE.sh*).

❖ To set the SYBASE environment

- 1 Log in to the operating system on the Replication Agent host machine, with a user login that has the appropriate permissions.
- 2 Open an operating system command window.
- 3 At the operating system prompt, navigate to the Sybase Replication Agent installation directory.

- On Microsoft Windows platforms, enter:

```
cd c:\sybase
```

where *c:\sybase* is the path to the Sybase Replication Agent installation directory.

- On UNIX platforms, enter:

```
cd /opt/sybase
```

where */opt/sybase* is the path to the Sybase Replication Agent installation directory.

- 4 In the Sybase Replication Agent installation directory, invoke the *SYBASE* environment script.

- On Microsoft Windows platforms, enter:

```
SYBASE
```

- On UNIX platforms, enter:

```
source SYBASE.sh
```

Note On UNIX platforms, you can insert the `source SYBASE.sh` command in the *.login* file for the Replication Agent administrator (or “sybase” user), so that the SYBASE environment is set automatically when you log in to the Replication Agent host machine.

Using the command line interface

This section describes how to administer a Replication Agent instance using the command line interface.

Using the ra utility

The Sybase Replication Agent ra utility provides the following functions:

- Starts a specified Replication Agent instance
- Returns the Sybase Replication Agent software version string

See “Using the ra_admin utility” on page 17 for information about creating a Replication Agent instance.

To run the ra utility, invoke it as a command at the operating system prompt.

Syntax

```
ra [-help | -i inst_name [-state] | -v]
```

Parameters

-help

The option that returns command usage information.

Note You can also invoke ra with no option specified to return command usage information.

-i *inst_name*

The option that specifies a Replication Agent instance to start, where *inst_name* is the name of an existing Replication Agent instance.

-state

The keyword that specifies a start-up state for the Replication Agent instance.

Valid -state values are:

- -admin – starts the Replication Agent instance in *Admin* state. (This is the default start-up state.)
- -replicate – starts the Replication Agent instance in *Replicating* state.

-v

The option that returns the Sybase Replication Agent software version number.

Example

To start a Replication Agent instance named “my_ra” in *Replicating* state, enter the following command at the operating system prompt:

```
ra -i my_ra -replicate
```

See “Starting the Sybase Replication Agent” on page 32 for more information about starting a Replication Agent instance.

See “Understanding Replication Agent states” on page 66 for more information about *Admin* and *Replicating* states.

Start-up errors

If the Replication Agent instance encounters start-up errors:

- On Microsoft Windows platforms, start-up errors are displayed in the operating system command window.
- On UNIX platforms, start-up errors are displayed in the operating system command window and recorded in the Replication Agent system log.

See Chapter 4, “Troubleshooting Sybase Replication Agent,” for more information.

Using the ra_admin utility

The Sybase Replication Agent ra_admin utility provides the following functions:

- Creates, copies, deletes, and verifies Replication Agent instances.
- Lists all valid Replication Agent instances on the Replication Agent host machine.
- Returns the path of the Sybase Replication Agent installation directory.
- Creates an output file in the `$SYBASE/RAX-12_6/admin_logs` directory. The format of the file name is `adminmddyymm_hhmmss.log` where `mddyym` and `hhmmss` are the current date and time.

To run the ra_admin utility, invoke it as a command at the operating system prompt.

Syntax `ra_admin [option [create options]] [inst_name]`

Note You can also invoke ra_admin with no option specified to return command usage information.

Parameters

-b
The option that returns the complete path of the Sybase Replication Agent installation directory.

-c *inst_name*
The option that creates a new Replication Agent instance using the specified name (*inst_name*).
The *inst_name* string must be a valid server name, and unique on the host machine.

When you use the `-c` option, one of the following options are required:

- `-p` and `-t`, or
- `-p` and `-f`.

When you use the `-f` option to copy an existing Replication Agent configuration, you need not specify the `-t` option. The primary database type specified for the existing Replication Agent instance is copied to the configuration of the new Replication Agent instance when you specify the `-f` option.

`-h`

The option that returns command usage information.

`-p port_num`

The option that specifies a client socket port number for the administration port of the Replication Agent instance.

The *port_num* must be a valid port number, and unique on the Replication Agent host machine.

`-t database`

The option that identifies the type of data server that the primary database resides in.

The *database* string must be one of the following:

- `ibmudb` – DB2 Universal Database
- `informix` – Informix Dynamic Server
- `mssql` – Microsoft SQL Server
- `oracle` – Oracle database server

Note The *database* value is not case sensitive.

When the `-c` option is used, you also have the option of specifying that the configuration of the new Replication Agent instance should be based on the configuration file for an existing Replication Agent instance. To do this, use the `-f` option.

`-f old_inst`

The option that copies the configuration of an existing Replication Agent instance for a new Replication Agent instance.

The *old_inst* string is the name of the existing Replication Agent instance whose configuration you want to copy for the new Replication Agent instance.

When you use the `-f` option to copy an existing Replication Agent configuration, you need not specify the `-t` option; the primary database type specified for the existing Replication Agent instance is copied to the configuration of the new Replication Agent instance when you specify the `-f` option.

Note When you use the `-f` option, some configuration parameters are set to default values. See “Copying a Replication Agent configuration” on page 22 for more information.

`-d inst_name`

The option that deletes a specified Replication Agent instance.

The *inst_name* string must be the name of an existing Replication Agent instance.

When you invoke `ra_admin` with the `-d` option, the utility deletes all of the subdirectories associated with the specified instance from the Sybase Replication Agent installation directory.

Note On Microsoft Windows platforms, if any application is accessing a file or directory associated with a Replication Agent instance when you delete the instance, the open file or directory is *not* deleted. An error message informs you of the file or directory not deleted.

To finish deleting a Replication Agent instance after a file or directory access conflict on a Microsoft Windows platform, you must:

- Verify that the file or directory is not open in any application
- Manually delete the file or directory

`-l (lowercase L)`

The option that lists all verifiable Replication Agent instances.

`-v inst_name`

The option that verifies the complete directory structure for a specified Replication Agent instance.

The *inst_name* string must be the name of an existing Replication Agent instance.

Creating a Replication Agent instance

You can create a Replication Agent instance at any time after the Sybase Replication Agent software is installed by invoking `ra_admin` with the `-c` option or using the Administrator GUI utility. See, “Using the Administrator GUI utility” on page 28.

The complete syntax is:

```
ra_admin -c new_inst -p port_num {-t database|-f old_inst}
```

where:

- *new_inst* is the name of the new Replication Agent instance you are creating.
- *port_num* is the client socket port number for the administration port of the new Replication Agent instance.
- *database* is the type of data server that contains the primary database.
- *old_inst* is the name of an existing Replication Agent instance whose configuration you want to duplicate for the new Replication Agent instance.

See “Copying a Replication Agent configuration” on page 22 for information about creating a Replication Agent instance based on the configuration of an existing instance.

Use the following procedure to create a Replication Agent instance.

Note You must set the SYBASE environment before you invoke the Sybase Replication Agent `ra_admin` utility. See “Setting the SYBASE environment” on page 14 for more information.

❖ To create a Replication Agent instance using the command line

- 1 Open an operating system command window on the Replication Agent host machine.
- 2 At the operating system prompt, navigate to the Sybase Replication Agent *bin* directory.
 - On Microsoft Windows platforms, enter:

```
cd %SYBASE%\RAX-12_6\bin
```

where `%SYBASE%` is the path to the Sybase Replication Agent installation directory.

- On UNIX platforms, enter:

```
cd $SYBASE/RAX-12_6/bin
```

where *\$SYBASE* is the path to the Sybase Replication Agent installation directory.

- 3 In the Sybase Replication Agent *bin* directory, invoke the *ra_admin* utility to create a new Replication Agent instance:

```
ra_admin -c new_inst -p port_num -t database
```

where:

- *new_inst* is the name of the Replication Agent instance.
- *port_num* is the client socket port number for the administration port of the new instance.
- *database* identifies the type of data server that the primary database resides in:
 - *ibmudb* – DB2 Universal Database
 - *informix* – Informix Dynamic Server
 - *mssql* – Microsoft SQL Server (valid only on Microsoft Windows platforms)
 - *oracle* – Oracle database server

After you invoke *ra_admin*, the operating system prompt returns when the new Replication Agent instance is created.

- 4 Verify that the Replication Agent instance was created properly using one of the following methods:

- Invoke *ra_admin* with the *-v* option, and specify the name of the new Replication Agent instance:

```
ra_admin -v new_inst
```

where *new_inst* is the name of the new Replication Agent instance.

When you verify a Replication Agent instance with the *-v* option, the utility verifies the instance by checking for an instance directory with the specified instance name under the Sybase Replication Agent base directory, and checking all of the subdirectories under the Replication Agent instance directory.

- Invoke *ra_admin* with the *-l* option:

```
ra_admin -l
```

The `-l` option lists all verifiable Replication Agent instances, which should include the new one you just created.

- As an alternative to using the `ra_admin` utility, you can use operating system commands to verify that the Replication Agent instance directories were created correctly (as shown in Figure 2-1).

After you create a Replication Agent instance, you can use the `ra` utility to start the instance so that you can administer and configure it. See “Starting the Sybase Replication Agent” on page 32 for more information.

Note Sybase recommends that you create a user login name and password to replace the default `sa` login and secure access to the administration port, immediately after you create a Replication Agent instance. See “Creating the Replication Agent administrator login” on page 42 for more information.

Copying a Replication Agent configuration

When you create a new Replication Agent instance, you can copy the configuration of an existing instance by invoking `ra_admin` with the `-c` option and `-f` option.

The complete syntax is:

```
ra_admin -c new_inst -p port_num -f old_inst
```

where:

- *new_inst* is the name of the new Replication Agent instance you are creating.
- *port_num* is the client socket port number for the administration port of the new Replication Agent instance.
- *old_inst* is the name of an existing Replication Agent instance whose configuration you want to duplicate for the new Replication Agent instance.

See “Creating a Replication Agent instance” on page 20 for information about creating a Replication Agent instance with the default configuration.

Use the following procedure to create a new Replication Agent instance, based on the configuration of an existing instance.

Note You must set the SYBASE environment before you invoke the Sybase Replication Agent `ra_admin` utility. See “Setting the SYBASE environment” on page 14 for more information.

❖ **To copy an existing Replication Agent instance configuration to a new instance**

- 1 Open an operating system command window on the Replication Agent host machine.
- 2 At the operating system prompt, navigate to the Sybase Replication Agent *bin* directory.

- On Microsoft Windows platforms, enter:

```
cd %SYBASE%\RAX-12_6\bin
```

where `%SYBASE%` is the path to the Sybase Replication Agent installation directory.

- On UNIX platforms, enter:

```
cd $SYBASE/RAX-12_6/bin
```

where `$SYBASE` is the path to the Sybase Replication Agent installation directory.

- 3 In the Sybase Replication Agent *bin* directory, invoke the `ra_admin` utility to create a new Replication Agent instance whose configuration is based on the configuration of an existing instance:

```
ra_admin -c new_inst -p port_num -f old_inst
```

where:

- *new_inst* is the name of the new Replication Agent instance.
- *port_num* is the client socket port number for the administration port of the new instance.
- *old_inst* is the name of an existing Replication Agent instance whose configuration you want to copy for the new instance.

After you invoke `ra_admin`, the operating system prompt returns when the new Replication Agent instance is created.

4 Verify that the Replication Agent instance was created properly using one of the following methods:

- Invoke `ra_admin` with the `-v` option, and specify the name of the new Replication Agent instance:

```
ra_admin -v new_inst
```

where `new_inst` is the name of the new Replication Agent instance.

When you verify a Replication Agent instance with the `-v` option, the utility verifies the instance by checking for an instance directory with the specified instance name under the Sybase Replication Agent base directory, and checking all of the subdirectories under the Replication Agent instance directory.

- Invoke `ra_admin` with the `-l` (lowercase L) option:

```
ra_admin -l
```

The `-l` option lists all verifiable Replication Agent instances, which should include the new one you just created.

- As an alternative to using the `ra_admin` utility, you can use operating system commands to verify that the Replication Agent instance directories were created correctly (as shown in Figure 2-2).

Note When you create a new Replication Agent instance and copy the configuration of an existing instance, some configuration parameters are set to default values, and they are not copied from the existing configuration.

The values of the following configuration parameters are not copied from an existing configuration:

```
admin_port  
log_directory  
pds_database_name  
pds_datasource_name  
pds_host_name  
pds_password  
pds_port_number  
pds_retry_count  
pds_retry_timeout  
pds_server_name  
pds_username  
rs_source_db
```

rs_source_ds

Note The following parameters are only valid for Replication Agent for Oracle.

rasd_backup_dir
rasd_database
rasd_mirror_tran_log
rasd_trace_log_dir
rasd_tran_log
rasd_tran_log_mirror
asa_port

See the Sybase Replication Agent *Reference Manual* for more information about Replication Agent configuration parameters.

After you create a Replication Agent instance, you can use the ra utility to start the instance so that it can be administered and configured.

Note Sybase recommends that you create a user login name and password to replace the default sa login and secure access to the administration port, immediately after you create a Replication Agent instance. See “Creating the Replication Agent administrator login” on page 42 for more information.

Deleting a Replication Agent instance

You can delete a Replication Agent instance at any time by invoking ra_admin with the -d option.

Before you delete a Replication Agent instance, you should:

- Shut down the Replication Agent instance, if it is running. See “Shutting down the Replication Agent instance” on page 69 for more information.
- If the Sybase Replication Agent software is installed on a Microsoft Windows platform, make sure that none of the files in the instance subdirectories are open, and no application or window is accessing the instance subdirectories.

Note You must set the SYBASE environment before you invoke the Sybase Replication Agent ra_admin utility. See “Setting the SYBASE environment” on page 14 for more information.

❖ **To delete a Replication Agent instance**

- 1 Open an operating system command window on the Replication Agent host machine.
- 2 At the operating system prompt, navigate to the Sybase Replication Agent *bin* directory.

- On Microsoft Windows platforms, enter:

```
cd %SYBASE%\RAX-12_6\bin
```

where *%SYBASE%* is the path to the Sybase Replication Agent installation directory.

- On UNIX platforms, enter:

```
cd $SYBASE/RAX-12_6/bin
```

where *\$SYBASE* is the path to the Sybase Replication Agent installation directory.

- 3 In the Sybase Replication Agent *bin* directory, invoke the *ra_admin* utility with the *-d* option to delete a Replication Agent instance:

```
ra_admin -d inst_name
```

where *inst_name* is the name of the Replication Agent instance you want to delete.

After you invoke *ra_admin* with the *-d* option, the following message appears:

```
Are you sure you want to delete the Sybase  
Replication Agent instance inst_name? [y/n]
```

- 4 Enter *y* to delete the Replication Agent instance.

After the instance is deleted, the operating system prompt returns.

If the instance is running when you invoke *ra_admin* with the *-d* option, the utility returns an error message:

```
Cannot delete Sybase Replication Agent instance  
'inst_name' because it is currently running.
```

To shut down a Replication Agent instance, log in to its administrative port, and use the *shutdown* command. See “Shutting down the Replication Agent instance” on page 69 for more information.

- 5 Verify that the Replication Agent instance was deleted properly using one of the following methods:

- Invoke the `ra_admin` utility with the `-v` option, and specify the name of the deleted Replication Agent instance:

```
ra_admin -v inst_name
```

where *inst_name* is the name of the deleted Replication Agent instance.

When you verify a Replication Agent instance with the `-v` option, the utility looks for an instance directory with the specified instance name under the Sybase Replication Agent base directory, and looks for the correct subdirectories under the Replication Agent instance directory.

- Invoke the `ra_admin` utility with the `-l` option:

```
ra_admin -l
```

The `-l` option lists all verifiable Replication Agent instances, which should *not* include the one you just deleted.

- As an alternative to using the `ra_admin` utility, you can use operating system commands to verify that the Replication Agent instance directories were deleted correctly. (The Replication Agent instance directories are shown in Figure 2-1 on page 13.)

Note On Microsoft Windows platforms, if any application is accessing a file or directory associated with a Replication Agent instance when you delete the instance, the open file or directory is *not* deleted. An error message informs you of the file or directory not deleted.

To finish deleting a Replication Agent instance after a file or directory access conflict occurs on a Microsoft Windows platform, you must:

- Verify that the file or directory is not open in any application
- Manually delete the file or directory

Note If you delete a Replication Agent instance, Replication Agent does not unmark any primary database objects marked for replication, nor does Replication Agent delete its transaction log objects. You must unmark primary database objects and delete the transaction log before you shut down and delete a Replication Agent instance.

Using the Administrator GUI utility

This section describes how to administer Replication Agent instances using Administrator, the Sybase Replication Agent GUI utility.

Starting the Administrator GUI

To start the Administrator GUI, do one of the following:

- Enter the following command at the operating system prompt:

```
administrator
```

Note You must set the SYBASE environment before you invoke the Sybase Replication Agent administrator utility. See “Setting the SYBASE environment” on page 14 for more information.

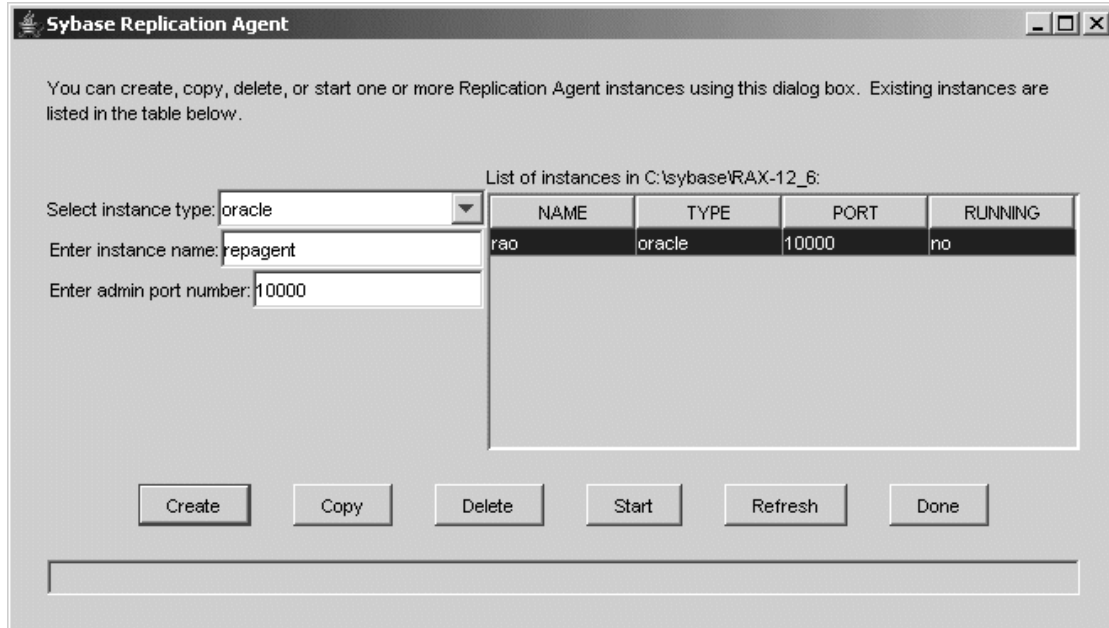
- On Windows 2000 or 2003, double-click the file name *administrator.bat* in File Manager or Explorer.

This file is located in the *RAX-12_6\bin* subdirectory, in your installation directory.

When you start Administrator, the Sybase Replication Agent Administrator GUI window opens.

When you use the Administrator GUI it creates an output file in the *\$SYBASE/RAX-12_6/admin_logs* directory. The format of the file name is *adminmddyymm_hhmmss.log* where *mddyym* and *hhmmss* are the current date and time.

Figure 2-2: Sybase Replication Agent Administrator GUI window



The Administrator GUI window provides the following information for each Replication Agent instance:

- *Name* – Name of the instance
- *Type* – Instance type
- *Port* – Port number assigned to the instance
- *Running* – Status of the instance (running or not)

Use the procedures in the following sections to administer Replication Agent instances with the Administrator GUI.

Creating an instance

Note You must set the SYBASE environment before you invoke the Sybase Replication Agent administrator utility. See “Setting the SYBASE environment” on page 14 for more information.

❖ **To create a Replication Agent instance using Administrator**

1 Select an instance type from the drop-down list:

- `oracle` – Oracle
- `informix` – Informix
- `ibmudb` – DB2 Universal Database
- `mssql` – Microsoft SQL Server

2 Enter an instance name.

The instance name must be unique; it cannot be the same as any other Replication Agent instance. Otherwise, Administrator returns an error.

The default instance name is `repagent`.

3 Enter a client socket port number that is not assigned to any other application on the machine for the administration port of the new Replication Agent instance.

Administrator returns an error if you enter a port number that is used by another application.

The default is port number 10000.

4 Click Create.

The instance name you specified appears in the List of instances. The status under Running is `no`.

5 Click Done to exit the Administrator GUI window.

Copying an instance

Note You must set the SYBASE environment before you invoke the Sybase Replication Agent administrator utility. See “Setting the SYBASE environment” on page 14 for more information.

❖ **To copy a Replication Agent instance using Administrator**

1 Select the Replication Agent instance you want to copy.

2 Enter an instance name.

This name must be unique; it cannot be the same as any other Replication Agent instance. Administrator returns an error if the instance name is not unique.

- 3 Enter a client socket port number that is not assigned to any other application on the machine for the administration port of the new Replication Agent instance.

Administrator returns an error if you enter a port number that is used by another application.

- 4 Click Copy.

The new instance appears in the List of instances.

Note Primary database server parameters and port numbers are not duplicated when you copy a Replication Agent instance.

See the Sybase Replication Agent *Administration Guide* for more information about copying Replication Agent instances.

Deleting an instance

Note You must set the SYBASE environment before you invoke the Sybase Replication Agent administrator utility. See “Setting the SYBASE environment” on page 14 for more information.

❖ To delete a Replication Agent instance using Administrator

- 1 Select the Replication Agent instance or instances you want to delete in the List of instances.
- 2 Click Delete.

Administrator provides a dialog box asking you to confirm that you want to delete the instance you selected. Click Yes to delete the instance.

Administrator deletes the instance from the instance list unless it is running. If the instance you selected is running, Administrator returns an error.

To delete an instance that is running, you must first shut the instance down by logging in to its administrative port and using the shutdown command. Then, delete the instance. See “Shutting down the Replication Agent instance” on page 69 for more information.

Note If you delete a Replication Agent instance, Replication Agent does not unmark any primary database objects marked for replication, nor does Replication Agent delete its transaction log objects. You must unmark primary database objects and delete the transaction log *before* you shut down and delete a Replication Agent instance.

Starting the Sybase Replication Agent

To start a Replication Agent instance, you must log in to the Replication Agent host machine with a user name that has `execute` permission in the Sybase Replication Agent installation directory and all subdirectories (for example, the “sybase” user).

There are three ways you can start a Replication Agent instance:

- Invoke the `ra` utility and specify the instance that you want to start.
- Invoke the Administrator GUI utility and specify the instance that you want to start.
- Invoke the `RUN` script for the instance that you want to start

The `ra` utility, the Administrator GUI utility, and the `RUN` script are batch files on Microsoft Windows platforms and shell scripts on UNIX platforms.

Start-up requirements

Before you can start a Replication Agent instance and connect to the primary data server, you must set all required variables.

See the Sybase Replication Agent *Primary Database Guide* for more information about connectivity requirements specific to your primary database.

Setting the CLASSPATH environment variable

Add the location of the JDBC driver for the primary database to the CLASSPATH environment variable.

See the Sybase Replication Agent *Primary Database Guide* for more information about installing and setting up the JDBC driver for the primary database and setting up Replication Agent connectivity.

Setting the Replication Agent character set to match the primary database

If the character set on your Replication Agent is different than the one on your primary database, you need to set the RA_JAVA_DFLT_CHARSET environment variable. The Replication Agent character set must be the same as that of the primary database. For more information, see “Setting character sets” on page 33.

Setting character sets

In a heterogeneous replication system, in which the primary and replicate data servers are different types, the data servers might not support the same character sets. In that case, replication system components must perform at least one character set conversion (from the primary data server’s character set to the replicate data server’s character set).

Even in a homogeneous replication system, in which both primary and replicate data servers are the same type, character set conversions might be required if replication system components reside on more than one type of platform.

Character set problems can produce data inconsistencies between the primary database and the replicate database. To avoid character set problems, you must either:

- Use the same character set on all servers and platforms in the replication system, or
- Use compatible character sets on all servers and platforms in the replication system, and configure replication system components to perform the appropriate character set conversions.

Note Using character set conversions slows performance.

Configuring your environment's character set

By default, the Java Virtual Machine (JVM) under which a Replication Agent instance is running, finds your system's default character set. The type of character data that Replication Agent can handle is determined by the character set, also known as the encoding. Unless you want to override the default character set that the JVM finds on your system, you do *not* have to explicitly set the character set-related environment variable.

To support overriding the default character set, all of the executable scripts (or batch files) in the Replication Agent */bin* directory refer to an environment variable named `RA_JAVA_DFLT_CHARSET`. You can set this environment variable to use the character set you want. The character set you specify must be the character set configured on the primary database. For a list of valid Java character sets, see Supported Encodings on the Internationalization page under Documentation for the J2SE 1.4.2 JDK at <http://java.sun.com/j2se/corejava/intl/index.jsp>.

All Replication Agent instance RUN scripts also reference the `RA_JAVA_DFLT_CHARSET` environment variable.

Note If you are using Replication Server to replicate a number of different character sets, you must configure it for UTF8.

You can override the system default character set by either:

- Setting the value of a system variable named `RA_JAVA_DFLT_CHARSET` in your environment and using the `ra` utility to start the Replication Agent instance, or
- Setting the value of the `RA_JAVA_DFLT_CHARSET` variable in the Replication Agent instance RUN script and using the RUN script to start the Replication Agent instance.

If you start a Replication Agent instance by invoking the `ra` utility, you can override the value of the `RA_JAVA_DFLT_CHARSET` system variable in your environment to specify the character set.

If you start a Replication Agent instance by invoking the instance RUN script (or batch file), you can edit the instance RUN script to specify the default value of `RA_JAVA_DFLT_CHARSET` and specify the character set you want to use.

❖ To override the system default character set for all instances

- 1 Enter a character set value in the `ra` script:
 - For Windows, edit the `%SYBASE%\RAX-12_6\bin\ra.bat` file.

- For UNIX, edit the `$SYBASE/RAX-12_6/bin/ra.sh` file:

```
RA_JAVA_DFLT_CHARSET=charset
```

where *charset* is the Java-supported encoding.

For example: `ISO8859_1` or `Cp1252` for ISO-1 (also known as Latin-1), and `ISO8859_8` or `Cp1255` for Hebrew.

Note In UNIX, spaces are *not* allowed on either side of the equals sign.

For a list of valid Java character sets, see Supported Encodings on the Internationalization page under Documentation for the J2SE 1.4.2 JDK at <http://java.sun.com/j2se/corejava/intl/index.jsp>.

- 2 Uncomment the following lines of code:

For Windows:

```
set RA_JAVA_DFLT_CHARSET=charset
```

For UNIX:

```
RA_JAVA_DFLT_CHARSET=charset
export RA_JAVA_DFLT_CHARSET
```

❖ **To override the system default character set for a Replication Agent instance**

- Enter a character set value in the RUN script:

For Windows, edit the `%SYBASE%\RAX-12_6\<instance>\RUN_<instance>.bat` script:

```
set RA_JAVA_DFLT_CHARSET=charset
```

For UNIX, edit the `$SYBASE/RAX-12_6\<instance>/RUN_<instance>.sh` batch file:

```
RA_JAVA_DFLT_CHARSET=charset
export RA_JAVA_DFLT_CHARSET
```

where *charset* is the Java-supported encoding.

For example: `ISO8859_1` or `Cp1252` for ISO-1 (also known as Latin-1), and `ISO8859_8` or `Cp1255` for Hebrew.

Note In UNIX, spaces are *not* allowed on either side of the equals sign.

For a list of valid Java character sets, see Supported Encodings on the Internationalization page under Documentation for the J2SE 1.4.2 JDK at <http://java.sun.com/j2se/corejava/intl/index.jsp>.

Starting an instance with the ra utility

When you start the Sybase Replication Agent with the `ra` utility, you can specify the instance start-up state. If you do *not* specify a start-up state when you invoke the `ra` utility, the Replication Agent instance starts in its default *Admin* state.

Note You must set the `SYBASE` environment before you invoke the Sybase Replication Agent `ra` utility. See “Setting the `SYBASE` environment” on page 14 for more information.

❖ To start Sybase Replication Agent with the `ra` utility

- 1 Open an operating system command window on the Replication Agent host machine.
- 2 At the operating system prompt, navigate to the Sybase Replication Agent *bin* directory.
 - On Microsoft Windows platforms, enter:

```
cd %SYBASE%\RAX-12_6\bin
```

where `%SYBASE%` is the path to the Sybase Replication Agent installation directory.
 - On UNIX platforms, enter:

```
cd $SYBASE/RAX-12_6/bin
```

where `$SYBASE` is the path to the Sybase Replication Agent installation directory.
- 3 In the Sybase Replication Agent *bin* directory, invoke the `ra` utility to start the Replication Agent instance:

```
ra -i inst_name
```

or

```
ra -i inst_name -state
```

where:

- *inst_name* is the server name of the Replication Agent instance.
- *state* is the optional keyword for the start-up state:
 - *admin* – starts the Replication Agent instance in *Admin* state.
 - *replicate* – starts the Replication Agent instance in *Replicating* state.

Note If you do not specify the state option, Replication Agent starts up in *Admin* state.

For example, to start the Replication Agent instance named “my_ra” in *Replicating* state, enter:

```
ra -i my_ra -replicate
```

See “Using the ra utility” on page 16 for more information.

After you start the Replication Agent instance, you must open another operating system command window to log in to its administration port.

Starting an instance with the Administrator GUI

Note You must set the SYBASE environment before you invoke the Sybase Replication Agent administrator utility. See “Setting the SYBASE environment” on page 14 for more information.

❖ **To start a Replication Agent instance using Administrator**

The administrator utility must be running before you use it to start a Replication Agent instance. See “Starting the Administrator GUI” on page 28 for details.

Note When you start a Replication Agent instance on Windows, start-up processes are running in the background with no DOS windows open.

- 1 Select the Replication Agent instance or instances you want to start in the List of instances. Click Start.

Administrator provides a dialog box asking you to confirm that the appropriate JDBC driver is specified in the CLASSPATH environment variable. Click Yes to continue.

Note If the appropriate JDBC driver is not listed in the CLASSPATH environment variable, you can start the Replication Agent instance, but it will not be able to establish a connection to the primary database server.

- 2 Click Start.

The Sybase Replication Agent window indicates that the instance you selected is running:

Note You may need to click Refresh to see if the Replication Agent instance is running.

- On Windows 2000 or 2003, a console window opens for each Replication Agent instance you selected to start.
- On UNIX, each Replication Agent instance you selected is started in the background.

For all platforms, the Replication Agent is started using the `ra_auto` script, which is located in the `RAX-12_6/bin` directory. During start-up, standard error output is redirected to the `error.log` file, which is located in the Replication Agent instance's `log` directory. After start-up, standard error and standard output directed to the `<instance>.log` file in the `<instance>/log` directory.

Note If you try to start a Replication Agent instance with the same administration port number as an instance that is already running, Replication Agent aborts the process and logs an error message in the system log.

If the instance you try to start does not run, check the `instance log` to see if an error occurred. See “Examining the Replication Agent logs” on page 128 for more information about viewing the `instance log`.

- 3 Click Done to exit the Administrator GUI window.

Starting an instance with the RUN script

The *RUN* script is named *RUN_inst_name*, where *inst_name* is the name of the Replication Agent instance. It is created automatically when the Replication Agent instance is created.

The *RUN* script invokes the *ra* utility with the appropriate parameter values to start the Replication Agent instance. You can edit the *RUN* script to specify the start-up state.

Note You need not set the SYBASE environment variable before you invoke the *RUN* script, because the *RUN* script sets the Sybase environment variable before it starts the Replication Agent instance.

❖ To start Sybase Replication Agent with the *RUN* script

- 1 Open an operating system command window on the Replication Agent host machine.
- 2 At the operating system prompt, navigate to the Replication Agent instance directory:

- On Microsoft Windows platforms, enter:

```
cd %SYBASE%\RAX-12_6\inst_name
```

where:

- *%SYBASE%* is the path to the Sybase Replication Agent installation directory.
- *inst_name* is the name of the Replication Agent instance.
- On UNIX platforms, enter:

```
cd $SYBASE/RAX-12_6/inst_name
```

where:

- *\$SYBASE* is the path to the Sybase Replication Agent installation directory.
 - *inst_name* is the name of the Replication Agent instance.
- 3 In the Replication Agent instance directory, invoke the *RUN* script to start the Replication Agent instance:

```
RUN_inst_name
```

where *inst_name* is the server name of the Replication Agent instance.

For example, to start the Replication Agent instance named “my_ra,” enter:

```
RUN_my_ra
```

Note Because this run script is generated at the time that the instance is created, the UNIX version does not have the *.sh* extension.

After you start the Replication Agent instance, you must open another operating system command window to log in to its administration port.

Using the Replication Agent administration port

When you create a Replication Agent instance, you specify a client socket port number for its administration port. Client applications use this port to connect to the Replication Agent.

The administration port allows Open Client (or Open Client-compatible) applications to log in and execute Replication Agent commands. You can use any Sybase Open Client interface utility (such as *isql* or *SQL Advantage*) to connect to the Replication Agent administration port.

Note Client applications are *not* provided with the Sybase Replication Agent software. The *isql* utility is provided with the Replication Server software, and both *isql* and *SQL Advantage* are provided with the Adaptive Server software.

Creating an entry in the interfaces file

In general, Open Client applications (such as *isql*) require an interfaces file to identify available servers, host machines, and client ports. On Microsoft Windows platforms, the interfaces file is named *sql.ini*. On UNIX platforms, the interfaces file is named *interfaces*.

If you want Open Client applications to be able to connect to the Replication Agent administration port, as they would to any other Open Server application, you must create a server entry for the Replication Agent in the interfaces file on the Open Client application host machine.

A server entry for a Replication Agent administration port in an interfaces file appears as follows:

```
[inst_name]
query=protocol,host_name,port_num
```

where:

- *inst_name* is the name of the Replication Agent instance.
- *protocol* is the network protocol used for the connection.
- *host_name* is the name of the Replication Agent host machine.
- *port_num* is the client socket port number of the administration port.

For example, to specify an interfaces file entry for a Replication Agent instance named “my_ra,” using the Microsoft Windows socket protocol, on a host named “my_host,” with client socket port number 10002, you would add the following lines to the interfaces file:

```
[my_ra]
query=NLWNSCK,my_host,10002
```

Some systems require the interfaces file to be in the TLI form. If your system requires that, you must use a utility (such as `sybtli` or `dsedit`) that edits the interfaces file and saves the result in a form compatible with TLI.

After you create an entry for the Replication Agent instance in the interfaces file, you can connect to the administration port using any Open Client application that uses that interfaces file.

Logging in to the Replication Agent

This section describes how to use the `isql` interactive SQL utility to log in to the Replication Agent administration port.

Before you can log in to the Replication Agent administration port with an Open Client application (such as `isql`), you must first create a server entry for the Replication Agent instance in the interfaces file. See “Creating an entry in the interfaces file” on page 40 for more information.

Note The first time you log in to a newly created Replication Agent instance, use the default administrator login, “sa”, with no password.

❖ **To log in to a Replication Agent instance**

- 1 Open an operating system command window.
- 2 At the operating system prompt, enter the following command:

```
isql -Username -Ppassword -Sinst_name
```

where:

- *username* is the Replication Agent administrator login.
- *password* is the corresponding password.
- *inst_name* is the name of the Replication Agent instance.

For example, to log in to a new Replication Agent instance named “my_ra,” enter:

```
isql -Usa -P -Smy_ra
```

Note Sybase recommends that you create a new administrator login and password to replace the default “sa” login and secure access to the administration port, immediately after you create a Replication Agent instance. See “Creating the Replication Agent administrator login” on page 42 for more information.

Once you have successfully logged in to the administration port, you can use Replication Agent commands to administer the Replication Agent instance.

Creating the Replication Agent administrator login

Each Replication Agent instance has only one administrator login. The default administrator login (sa, with no password) is created when the Replication Agent instance is created.

Note Sybase recommends that you create a user login name and password to replace the default sa login and secure access to the administration port, immediately after you create a Replication Agent instance.

You can use `ra_set_login` to create (or change) the administrator login for a Replication Agent instance.

❖ **To create or change the Replication Agent administrator login**

- 1 Log in to the Replication Agent instance with the administrator login.

When you log in to the Replication Agent instance for the first time, use the default administrator login.

- 2 After you log in, enter the following command:

```
ra_set_login admin_user,admin_pw
```

where:

- *admin_user* is the new administrator login name you want to use for this Replication Agent instance.
- *admin_pw* is the password for the new administrator login.

Note Use the values from items 1e and 1f on the “Installation and Setup Worksheet” in the Sybase Replication Agent *Installation Guide* to specify the Replication Agent administrator login name and password.

The new login name replaces the current administrator login. The next time you log in to the Replication Agent instance, you must use the new administrator login name and password.

Setting up Replication Agent connectivity

You must set up connectivity between the Replication Agent instance and the following replication system components:

- Primary data server
- Replication Server
- RSSD

Primary databases require you to perform specific setup tasks *before* you can set up connectivity between the Replication Agent and a primary database. See the Sybase Replication Agent *Primary Database Guide* to verify that the required setup tasks have been performed for your primary database.

Note The term “RSSD” in this document refers to both RSSD and ERSSD, unless there is a difference.

Setting up connectivity for the Replication Agent requires:

- Creating a user login name, with the appropriate authority in the primary data server and the primary database, for the Replication Agent
- Creating a user login name, with connect source permission in the Replication Server, for the Replication Agent
- Creating a user login name, with the appropriate authority in the RSSD data server and the RSSD, for the Replication Agent
- Setting values for the Replication Agent connection configuration parameters

You can use the “Installation and Setup Worksheet” in the Sybase Replication Agent *Installation Guide*, to record the values of connection configuration parameters for each Replication Agent instance.

Creating the primary database user login name

Sybase Replication Agent requires client access to the primary database to:

- Get information about the database schema
- Create, manage, and read its transaction log objects (Replication Agents for UDB, Informix, and Microsoft SQL Server)
- Get information about database log devices (Replication Agent for Oracle)

Use the following procedure to set up a user login name in the primary data server and the primary database for the Replication Agent instance.

Note You must have a System Administrator user role in the primary data server to perform this procedure.

❖ To create a primary database user login for Sybase Replication Agent

- 1 Log in to the primary data server with a System Administrator user role.
- 2 Add the Replication Agent login name to the primary data server, and if necessary, to the primary database.
 - Refer to the Sybase Replication Agent *Primary Database Guide* for information about the permissions and authorities required in each type of primary data server and primary database.

- Refer to the documentation provided by your primary data server vendor for information about the specific commands you need to execute to create the Replication Agent login name in the primary data server (and, if necessary, in the primary database).

After you set up the Sybase Replication Agent user login in the primary data server, verify that the new user login name is valid (it can log in to the primary data server and access the primary database).

Creating the Replication Server user login name

Sybase Replication Agent requires client access to the primary Replication Server to send replicated transactions.

Use the following procedure to set up a Replication Server user login name for the Replication Agent instance.

Note You must have “sa” permission in the Replication Server to perform this procedure.

❖ To create a Replication Server user login for Sybase Replication Agent

- 1 Log in to the Replication Server with a login that has “sa” permission.
- 2 Create the Replication Agent user login name in the Replication Server:

```
create user ra_rs_user set password ra_rs_pwd
```

where:

- *ra_rs_user* is the Replication Agent user login name.
- *ra_rs_pwd* is the password for the user login name.

- 3 Grant connect source permission to the Replication Agent login name:

```
grant connect source to ra_rs_user
```

where *ra_rs_user* is the Replication Agent user login name.

After you set up the Replication Agent user login in the primary Replication Server, verify that the new user login name is valid (it can log in to the Replication Server).

Creating the RSSD user login name

Sybase Replication Agent requires client access to the RSSD or ERSSD to obtain information about replication definitions.

The following sections describe procedures for:

- Setting up the RSSD user login for Sybase Replication Agent
- Setting up the ERSSD user login for Sybase Replication Agent

Refer to the appropriate procedure for your Replication Server configuration.

Setting up the RSSD user login for Sybase Replication Agent

Use the following procedure to set up a user login name for the Replication Agent instance in an RSSD managed by Adaptive Server Enterprise.

See “Setting up the ERSSD user login for Sybase Replication Agent” on page 47 for information about setting up a user login name for the Replication Agent instance in an ERSSD managed by Adaptive Server® Anywhere.

Note You must have a System Administrator user role in the Adaptive Server that manages the RSSD to perform this procedure.

❖ To set up the RSSD user login for Sybase Replication Agent

- 1 Log in to the Adaptive Server that manages the RSSD with a System Administrator user role.
- 2 Add the Replication Agent login name to the RSSD data server:

```
use master
sp_addlogin ra_rssd_user, ra_rssd_pwd, rssid_db
```

where:

- *ra_rssd_user* is the Replication Agent user login name.
- *ra_rssd_pwd* is the password for the user login name.
- *rssd_db* is the database name of the RSSD.

- 3 Add the Replication Agent user login name to the RSSD:

```
use rssid_db
sp_adduser ra_rssd_user
```

where:

- *rssd_db* is the database name of the RSSD.
- *ra_rssd_user* is the Replication Agent user login name.

After you set up the Replication Agent user login in the RSSD, verify that the new user login name is valid (it can log in to the RSSD data server and access the RSSD).

Setting up the ERSSD user login for Sybase Replication Agent

Use the following procedure to set up a user login name for the Replication Agent instance in an ERSSD managed by Adaptive Server Anywhere.

See “Setting up the RSSD user login for Sybase Replication Agent” on page 46 for information about setting up a user login name for the Replication Agent instance in an RSSD managed by Adaptive Server Enterprise.

Note You must have the primary user role in the ERSSD (“sa” permission in the Replication Server) to perform this procedure.

❖ To set up the ERSSD user login for Sybase Replication Agent

1 Log in to the ERSSD as the primary user.

2 Add the Replication Agent login name to the ERSSD:

```
grant connect to ra_rssd_user
identified by ra_rssd_pwd
```

where:

- *ra_rssd_user* is the Replication Agent user login name.
- *ra_rssd_pwd* is the password for the user login name.

3 Give the Replication Agent user permission to read the Replication Server system tables:

```
grant membership in group rs_systabgroup
to ra_rssd_user
```

where *ra_rssd_user* is the Replication Agent user login name.

After you set up the Replication Agent user login in the ERSSD, verify that the new user login name is valid (it can log in to the ERSSD and access the Replication Server system tables).

Setting up the connection configuration parameters

When Sybase Replication Agent connects to another replication system component, it uses values stored in its configuration parameters to define the following minimal set of connection properties:

- Server host name
- Port number
- User login name
- User login password

Note The complete set of connection parameters is different for each database. See the Replication Agent *Primary Database Guide* for the complete set of connection parameters that each database requires.

For its connection to the Replication Server, Sybase Replication Agent relies on the values of two additional configuration parameters (`rs_source_db` and `rs_source_ds`) to identify the Replication Server primary database connection in the LTL connect source command.

The Replication Agent instance must be in *Admin* state to setup connection parameters. When the Replication Agent instance is *Admin* state, the instance has no connections established to other replication system components, but is available to execute administrative commands. For more information, see “Understanding Replication Agent states” on page 66.

Note The values of the `rs_source_db` and `rs_source_ds` parameters must *exactly match* the database and data server names specified in the create connection command for the Replication Server primary database connection. The values are case sensitive.

See the Sybase Replication Agent *Reference Manual* for more information about the `rs_source_db` and `rs_source_ds` parameters.

You can use the “Installation and Setup Worksheet,” in the Sybase Replication Agent *Installation Guide*, to record the values of connection configuration parameters for each Replication Agent instance.

Refer to the “Installation and Setup Worksheet” for the connection configuration parameter values you need to set in the following procedures.

Note The Replication Agent instance must be running before you can set its connection configuration parameter values. See “Starting the Sybase Replication Agent” on page 32 for more information.

❖ **To set up connection parameters for the primary database**

When the Replication Agent instance is *Admin* state, the instance has no connections established to other replication system components, but is available to execute administrative commands. The Replication Agent instance must be in *Admin* state to setup connection parameters.

- 1 Log in to the Replication Agent administration port, and verify that the Replication Agent instance is in *Admin* state.
 - a Use the following command to verify that the Replication Agent instance is in *Admin* state:

```
ra_status
```

- b If the instance is not in *Admin* state, use the following command to change it to *Admin* state:

```
suspend
```

- 2 Specify the primary data server host name:

```
ra_config pds_hostname, pds_host
```

where *pds_host* is the network name of the primary data server host machine.

- 3 Specify the primary data server port number:

```
ra_config pds_port_number, NNN
```

where *NNN* is the number of the network port where the primary data server listens for connections.

- 4 Specify the primary database name:

```
ra_config pds_database_name, pdb
```

where *pdb* is the database name of the primary database.

- 5 Specify the primary data server user login name for the Replication Agent instance:

```
ra_config pds_username, ra_pds_user
```

where *ra_pds_user* is the user login name that the Replication Agent uses to log in to the primary data server.

- 6 Specify the password for the Replication Agent user login:

```
ra_config pds_password, ra_pds_pwd
```

where *ra_pds_pwd* is the password for the user login name that the Replication Agent uses to log in to the primary data server.

After you set up connection configuration parameters for the primary database, you can use the Replication Agent `test_connection PDS` command to test connectivity between the Replication Agent and the primary database. See “Testing network connectivity” on page 52 for more information.

❖ **To set up connection parameters for the Replication Server**

- 1 Log in to the Replication Agent administration port, and verify that the Replication Agent instance is in *Admin* state.

- a Use the following command to verify that the Replication Agent instance is in *Admin* state:

```
ra_status
```

- b If the instance is not in *Admin* state, use the following command to change it to *Admin* state:

```
suspend
```

- 2 Specify the Replication Server host name:

```
ra_config rs_hostname, rs_host
```

where *rs_host* is the network name of the Replication Server host machine.

- 3 Specify the Replication Server port number:

```
ra_config rs_port_number, NNN
```

where *NNN* is the number of the network port where Replication Server listens for connections.

- 4 Specify the Replication Server character set:

```
ra_config rs_charset, <sybase_charset>
```

where *rs_charset* matches the character set used by Replication Server. If *rs_charset* is not set, Replication Agent refuses to replicate anything.

- 5 Specify the Replication Server user login name for the Replication Agent instance:

```
ra_config rs_username, ra_rs_user
```

where *ra_rs_user* is the user login name that the Replication Agent uses to log in to the primary Replication Server.

- 6 Specify the user login password for the Replication Agent instance:

```
ra_config rs_password, ra_rs_pwd
```

where *ra_rs_pwd* is the password for the user login name that the Replication Agent uses to log in to the primary Replication Server.

- 7 Specify the primary data server name for the Replication Server primary database connection:

```
ra_config rs_source_ds, pds
```

where *pds* is the primary data server name that the Replication Agent uses in the LTL connect source command.

- 8 Specify the primary database name for the Replication Server primary database connection:

```
ra_config rs_source_db, pdb
```

where *pdb* is the primary database name that the Replication Agent uses in the LTL connect source command.

❖ **To set up connection parameters for the RSSD (or ERSSD)**

- 1 Log in to the Replication Agent administration port, and verify that the Replication Agent instance is in *Admin* state:

- a Use the following command to verify that the Replication Agent instance is in *Admin* state:

```
ra_status
```

- b If the instance is not in *Admin* state, use the following command to put it in *Admin* state:

```
suspend
```

- 2 Specify the RSSD host name:

```
ra_config rssid_hostname, rssid_host
```

where *rssid_host* is the network name of the RSSD host machine.

- 3 Specify the RSSD port number:

```
ra_config rssid_port_number, NNN
```

where *NNN* is the number of the network port where the RSSD server listens for connections.

- 4 Specify the RSSD database name:

```
ra_config rssid_database_name, rssid_db
```

where *rssd_db* is the database name of the RSSD.

- 5 Specify the RSSD user login name for the Replication Agent instance:

```
ra_config rssid_username, ra_rssid_user
```

where *ra_rssid_user* is the user login name that the Replication Agent uses to log in to the RSSD.

- 6 Specify the user login password for the Replication Agent instance:

```
ra_config rssid_password, ra_rssid_pwd
```

where *ra_rssid_pwd* is the password for the user login name that the Replication Agent uses to log in to the RSSD.

After you set up connection configuration parameters for the primary Replication Server and RSSD, you can use the Replication Agent `test_connection RS` command to test connectivity between the Replication Agent and the Replication Server and RSSD.

Testing network connectivity

Sybase Replication Agent provides a simple means of testing network connections. The `test_connection` command sends a connection request and confirms the network connection to the following servers:

- Primary data server
- Primary Replication Server
- RSSD server (if so configured)

Note If the value of the `use_rssid` configuration parameter is true, the `test_connection` command tests Replication Agent connectivity to the RSSD when it tests connectivity to the Replication Server. If the value of the `use_rssid` configuration parameter is false, the `test_connection` command does *not* test Replication Agent connectivity to the RSSD.

The `test_connection` command returns a failure message if:

- The connection specifications (server name, port number, user login, and so forth) recorded in the Replication Agent configuration parameters are not correct.
- The Replication Agent cannot establish a connection to a server because of a network failure.
- The Replication Agent cannot establish a connection to a server because the server is down.

The `test_connection` command does *not* validate Replication Agent user login permissions in the primary database. It verifies only that the user login and password specified in the `pds_username` and `pds_password` parameters can log in to the primary data server.

The `test_connection` command does verify that the Replication Agent user login (specified in the `rs_username` and `rs_password` parameters) has connect source permission in the primary Replication Server.

See “Setting up Replication Agent connectivity” on page 43 for information about setting up Replication Agent connection configuration parameters.

Note You must be in *Admin* state to test network connectivity.

❖ **To verify Replication Agent connections**

- 1 Log in to the Replication Agent instance with the administrator login.
- 2 Use the following command to test Replication Agent network connections:

```
test_connection
```

This command tests all of the connections from the Replication Agent instance you logged in to.

Note You can test a specific connection (either the primary data server or the primary Replication Server) by specifying the connection you want to test.

If the `test_connection` command returns a failure, check the Replication Agent system log to determine the cause of the failure. You may also need to check the system log of the server associated with the connection to determine the cause of the failure.

See the Sybase Replication Agent *Reference Manual* for more information about the `test_connection` command.

Initializing the Replication Agent transaction log

In trigger-based implementations (Informix and Microsoft SQL Server), Sybase Replication Agent uses its own proprietary transaction log to capture and record transactions in the primary database for replication. The Replication Agent transaction log consists of a set of database objects (tables, stored procedures, and triggers) that the Replication Agent creates in the primary database.

In log-based implementations (DB2 Universal Database and Oracle), Sybase Replication Agent uses the native transaction log maintained by the primary database to obtain transactions.

Note The Replication Agents for UDB and for Oracle also create a few user tables in the primary database to support their operation.

Before you initialize a Replication Agent that has a Replication Agent System Database (RASD), the primary database must be quiesced. Currently, only the Replication Agent for Oracle uses an RASD. The following procedure includes that quiescing.

Specifying the transaction log object name prefix

Before you create the Replication Agent transaction log base objects, you can specify the object name *prefix* string that will be used to name transaction log objects. You can set this prefix string to avoid conflicts with the names of existing database objects in your primary database.

The value of the `pdb_xlog_prefix` parameter is the prefix string used in all Replication Agent transaction log component names. Use the `ra_config` command to change the value of the `pdb_xlog_prefix` parameter.

Note Replication Agent uses the value of `pdb_xlog_prefix` to find its transaction log objects in the primary database. If you change the value of `pdb_xlog_prefix` after you create the transaction log, Replication Agent will not be able to find the transaction log objects that use the old prefix.

Initializing transaction log objects is the same process for trigger-based Replication Agents (for Informix and Microsoft SQL Server) and log-based Replication Agents (for UDB and Oracle).

Note Primary databases require you to perform specific setup tasks *before* you can initialize the Replication Agent transaction log. See the Sybase Replication Agent *Primary Database Guide* to verify that the required setup tasks have been performed for your primary database.

❖ **To initialize a Replication Agent transaction log**

- 1 Log in to the Replication Agent administration port.
- 2 To define a prefix that uniquely identifies the Replication Agent transaction log you are creating, use the following command:

```
ra_config pdb_xlog_prefix, string
```

where *string* is a character string of one to three characters that will be used as a prefix for all database object names of the Replication Agent transaction log components created in the primary database.

Note The default value of the `pdb_xlog_prefix` parameter is `ra_`. Unless this string poses a conflict with existing database object names in your primary database, you should use the default value.

- 3 To initialize the transaction log, use the following command:

```
pdb_xlog init
```

Note Releases of Replication Agent prior to 12.6 use the `pxb_xlog` command with the `create` keyword to initialize the transaction log. This keyword has been retained for backward compatibility, but the `init` keyword is the correct and preferred syntax to be used with the `pdb_xlog` command

When you invoke the `pdb_xlog` command with the `init` option, the Replication Agent does the following:

- Checks the primary database for compatible settings.
- Generates a SQL script that is run in the primary database. This script creates the Replication Agent transaction log base objects.

For Replication Agents that use a Replication Agent System Database (RASD), the RASD is initialized with information from the primary database.

Note Transaction log base objects must be created before any objects can be marked for replication in the primary database.

- 4 To verify that the Replication Agent transaction log was created, use the following command:

```
pdb_xlog
```

When you invoke the `pdb_xlog` command with no options, Replication Agent returns a list of the transaction log base objects in the primary database, if the transaction log was created successfully. If no information is returned, the transaction log does not exist in the primary database.

When the Replication Agent transaction log exists, and both primary database and Replication Server connections are defined correctly, you can put the Replication Agent instance in *Replicating* state. See “Starting replication” on page 62 for more information about putting the Replication Agent in *Replicating* state.

Marking objects in the primary database

Tables and stored procedures must be marked for replication and have replication enabled for the object (table or stored procedure). LOB columns must have replication enabled, and the table that contains the LOB column must be marked for replication and have replication enabled for that table.

There are four types of objects that can be marked for replication in a primary database:

- Tables
- Stored procedures

Note Procedure replication is not available for DB2 Universal Database.

- Large-object (LOB) columns
- DDL (Oracle only)

Marking tables in the primary database

For transactions against a table to be replicated, the primary table in the primary database must be marked for replication and replication must be enabled for that table.

❖ To mark a table in the primary database

- 1 Log in to the Replication Agent administration port.
- 2 Use the `pdb_setreptable` command to determine if the table you want to mark is already marked in the primary database:

```
pdb_setreptable pdb_table
```

where *pdb_table* is the name of the table in the primary database that you want to mark for replication.

- If the `pdb_setreptable` command returns information that the specified table is marked and replication is enabled, you need not continue this procedure.
 - If the `pdb_setreptable` command returns information that the specified table is marked, but replication is disabled, skip step 3 and continue this procedure from step 4 to enable replication for the table.
 - If the `pdb_setreptable` command returns information that the specified table is not marked, continue this procedure to mark the table for replication.
- 3 Use the `pdb_setreptable` command to mark the table for replication.

The `pdb_setreptable` command allows you to mark the primary table to be replicated and specify the name to use for replication.

- Use the following command to mark the table for replication using a replication definition with the same table name:

```
pdb_setreptable pdb_table, mark
```

where *pdb_table* is the name of the table in the primary database that you want to mark for replication.

- Use the following command to mark the table for replication using a replication definition with a different table name:

```
pdb_setreptable pdb_table, rep_table, mark
```

where:

- *pdb_table* is the name of the table in the primary database that you want to mark for replication.

- *rep_table* is the name of the table in the with all tables named *rep_table* clause in the replication definition for this table.
- When marking a table for replication, you can specify that the table owner's name is sent along with the table name in the LTL. To do this, use the *owner* keyword after the *mark* keyword, as shown in the following example:

```
pdb_setreptable pdb_table, mark, owner
```

where *pdb_table* is the name of the table in the primary database that you want to mark for replication.

If the *pdb_dflt_object_repl* parameter is set to *true* (the default), the table marked for replication with the *pdb_setreptable* command is ready for replication after you invoke the *pdb_setreptable* command successfully, and you can skip step 4 in this procedure.

If the *pdb_dflt_object_repl* parameter is set to *false*, you must enable replication for the table before replication can take place.

- 4 Use the *pdb_setreptable* command to enable replication for the marked table:

```
pdb_setreptable pdb_table, enable
```

where *pdb_table* is the name of the marked table in the primary database for which you want to enable replication.

After the table is marked and replication is enabled for the table, you can begin replicating transactions that affect data in that table.

Marking stored procedures in the primary database

To replicate invocations of a stored procedure in the primary database, the stored procedure must be marked for replication and replication must be enabled for that stored procedure.

For Oracle, DDL replication must be disabled before marking (or unmarking) a stored procedure.

Note Procedure replication is not available for DB2 Universal Database.

❖ To mark a stored procedure in the primary database

- 1 Log in to the Replication Agent administration port.

- 2 Use the `pdb_setrepproc` command to determine if the stored procedure you want to mark is already marked in the primary database:

```
pdb_setrepproc pdb_proc
```

where *pdb_proc* is the name of the stored procedure in the primary database that you want to mark for replication.

- If the `pdb_setrepproc` command returns information that the specified stored procedure is marked and replication is enabled, you need not continue this procedure.
- If the `pdb_setrepproc` command returns information that the specified stored procedure is marked, but replication is disabled, skip step 3 and continue this procedure from step 4 to enable replication for the stored procedure.
- If the `pdb_setrepproc` command returns information that the specified stored procedure is not marked, continue this procedure to mark the stored procedure for replication.

- 3 Use the `pdb_setrepproc` command to mark the stored procedure for replication.

The `pdb_setrepproc` command allows you to mark the stored procedure to be replicated and specify the name to use for replication.

- Use the following command to mark the stored procedure for replication using a function replication definition with the same procedure name:

```
pdb_setrepproc pdb_proc, mark
```

where *pdb_proc* is the name of the stored procedure in the primary database that you want to mark for replication.

- Use the following command to mark the stored procedure for replication using a function replication definition with a different procedure name:

```
pdb_setrepproc pdb_proc, rep_proc, mark
```

where:

- *pdb_proc* is the name of the stored procedure in the primary database that you want to mark for replication.
- *rep_proc* is the name of the stored procedure in the with all procedures named `rep_proc` clause in the function replication definition for this stored procedure.

If the `pdb_dflt_object_repl` parameter is set to `true` (the default), the stored procedure marked for replication with the `pdb_setrepproc` command is ready for replication after you invoke the `pdb_setrepproc` command successfully, and you can skip step 4 in this procedure.

If the `pdb_dflt_object_repl` parameter is set to `false`, you must enable replication for the stored procedure so replication can take place.

- 4 Use the `pdb_setrepproc` command to enable replication for the marked stored procedure:

```
    pdb_setrepproc pdb_proc, enable
```

where *pdb_proc* is the name of the marked stored procedure in the primary database for which you want to enable replication.

After the stored procedure is marked and replication is enabled for the stored procedure, you can begin replicating invocations of that stored procedure.

Enabling replication for LOB columns

For transactions that affect a LOB column to be replicated, the table that contains the LOB column must be marked for replication and have replication enabled, and the LOB column must have replication enabled.

If the value of the `pdb_dflt_column_repl` parameter is set to `true`, all LOB columns in a table have replication enabled automatically when you mark the table (by invoking the `pdb_setreptable` command). If the value of the `pdb_dflt_column_repl` parameter is set to `false`, you must enable replication separately for each LOB column before replication can take place.

Note The default value of the `pdb_dflt_column_repl` parameter is `false`.

❖ To enable replication for a LOB column in the primary database

- 1 Log in to the Replication Agent administration port.
- 2 Use the `pdb_setrepcol` command to determine if replication is already enabled for the LOB column you want to enable replication for in the primary database:

```
    pdb_setrepcol tablename, pdb_col
```

where:

- *tablename* is the name of the table that contains the LOB column.

- *pdb_col* is the name of the LOB column in the primary database.

Note If `pdb_setrepcol` is invoked with a table containing a “DATE” column, the primary key in the primary table must *not* include the “DATE” column. This is true for all Replication Agents except Replication Agent for Oracle.

If the `pdb_setrepcol` command returns information that replication is enabled for the specified column, you need not continue this procedure.

If the `pdb_setrepcol` command returns information that replication is not enabled for the specified column, continue this procedure to enable replication for the column.

- 3 Use the `pdb_setrepcol` command to enable replication for the LOB column:

```
pdb_setrepcol tablename, pdb_col, enable
```

where:

- *tablename* is the name of the table that contains the LOB column.
- *pdb_col* is the name of the LOB column in the primary database for which you want to enable replication.

After replication is enabled for the LOB column, you can begin replicating transactions that affect data in that column.

Enabling replication for DDL

Note DDL replication is only available for Oracle.

For database definition language (DDL) to be replicated, the `pdb_setrepddl` command must be set to enable. If `pdb_setrepddl` is set to enable, all DDL in your primary database is replicated.

Note To replicate DDL, Replication Server must have a database-level replication definition with `replicate DDL` set in the definition. See the Replication Server *Reference Manual* for details on creating a database-level replication definition.

❖ **To enable replication for DDL in the primary database**

- 1 Log in to the Replication Agent administration port.
- 2 Use the `pdb_setrepddl` command without an argument to determine if replication is already enabled for DDL in the primary database:

```
pdb_setrepddl
```

If the `pdb_setrepddl` command returns information that replication is enabled, you need not continue this procedure.

If the `pdb_setrepddl` command returns information that replication is not enabled for DDL, continue this procedure to enable replication for DDL.

- 3 Use the `pdb_setrepddl` command to enable replication for DDL:

```
pdb_setrepddl enable
```

After replication is enabled for the DDL, you can begin replicating your primary database.

See the Sybase Replication Agent *Primary Database Guide* for enabling DDL replication details specific to your primary database.

Starting replication

Note Before you attempt to replicate transactions from the primary database, you must complete all of the procedures in “Setting up Replication Agent connectivity” on page 43.

❖ **To start replication in the Replication Agent instance**

- 1 Log in to the Replication Agent administration port and use the following command to verify that the Replication Agent instance is in *Admin* state:

```
ra_status
```

- 2 Start replication by invoking the following command:

```
resume
```

- 3 Use the `ra_status` command to verify that the Replication Agent instance is in *Replicating* state.

Note The Replication Agent instance will go to the *Replicating* state only if a connection for the primary database has been created in the primary Replication Server. For more information on creating the primary database connection in Replication Server, see the Replication Agent Primary *Database Guide*.

When the Replication Agent instance is in *Replicating* state, it is scanning the transaction log for transactions to be replicated and sending LTL to the primary Replication Server.

If the Replication Agent instance is not in *Replicating* state after you invoke the resume command, see Chapter 4, “Troubleshooting Sybase Replication Agent,” for more information.

Administering Sybase Replication Agent

This chapter describes administrative tasks and procedures for the Sybase Replication Agent.

Topic	Page
Determining current Replication Agent status	65
Shutting down the Replication Agent instance	69
Starting replication in the Replication Agent	71
Stopping replication in the Replication Agent	72
Managing the Replication Agent transaction log	74
Managing the Replication Agent System Database	84
Identifying replicated transactions and procedures	91
Configuring and tuning the Replication Agent	114

See the Sybase Replication Agent *Installation Guide* for information about installing the Sybase Replication Agent software.

See Chapter 2, “Setup and Configuration,” for information about setting up the Sybase Replication Agent.

Note Example procedures in this chapter show isql as the Open Client application used to log in to the Replication Agent administration port. You can use any Open Client (or Open Client-compatible) application to log in to the Replication Agent administration port.

Determining current Replication Agent status

The Replication Agent status consists of the current state and activity of the Replication Agent instance.

❖ **To determine the status of a Replication Agent instance**

- 1 Log in to the Replication Agent instance with the administrator login.
- 2 Use the following command to get current status for the Replication Agent instance:

```
ra_status
```

This command returns the current state of the Replication Agent instance, and any current activity, as shown in the following example:

```
State  Action
-----
ADMIN  Waiting for operator command
(1 row affected)
```

Understanding Replication Agent states

When a Replication Agent instance is running, it can be in one of two discrete states:

- *Admin* – the instance has no connections established to other replication system components, but it is available to execute administrative commands, such as changing configuration parameters and maintaining the transaction log or the RASD. No replication processing occurs when the Replication Agent instance is in *Admin* state.
- *Replicating* – the instance is performing its normal replication processing—scanning the transaction log, processing log records and change-set data, and sending LTL commands to the primary Replication Server. In *Replicating* state, some administrative commands are not allowed.

Note When the state appears as *Replicating (Waiting at end of log)*, the instance is ready to perform (or continue) its normal replication processing, but the Log Reader component is idle and waiting for additional transactions to be logged.

The default start-up state is *Admin*. The Replication Agent instance goes to *Admin* state automatically when no start-up state is specified.

The state of a Replication Agent instance can be changed by either:

- An external event that occurs while the Replication Agent is processing replicated transactions (for example, a network error on the Replication Server connection), or
- Operator intervention (for example, invoking a command that changes the Replication Agent state).

From the moment a state-changing event occurs until the Replication Agent instance is actually in the new state, the instance is said to be “in transition.” During state transition, some administrative commands are ignored.

Admin state

A Replication Agent instance goes to *Admin* state when:

- The instance is started in its default state.
- The instance is started with the `ra` utility `-admin` option.
- The Replication Agent `quiesce` or `suspend` command is invoked.
- An unrecoverable error occurs when the instance is in *Replicating* state.

In *Admin* state, the Replication Agent instance is running, but it has no connection established to the primary Replication Server (or RSSD, if so configured) or the primary database.

You can perform most administrative tasks while the Replication Agent instance is in *Admin* state, including changing the value of any Replication Agent configuration parameter.

Note In *Admin* state, the instance can open a connection to the primary database, if necessary, to process a command that requests results from the primary database.

A Replication Agent instance may go to *Admin* state from *Replicating* state when a network failure or communication error causes its connection to the primary database or the primary Replication Server to be dropped.

When Replication Agent drops a connection, before it goes to *Admin* state, it first attempts to re-establish the connection using the values recorded in its configuration parameters for that connection. If it cannot reconnect, the Replication Agent instance goes to *Admin* state.

Replicating state

A Replication Agent instance goes to *Replicating* state when:

- The instance is started with the `ra` utility `-replicate` option.
- The Replication Agent `resume` command is invoked.

In *Replicating* state, the Replication Agent instance maintains a connection to the primary database and the primary Replication Server (and RSSD, if so configured), and its Log Reader component scans the transaction log for transactions to replicate.

If the Replication Agent instance has finished processing all of the records in the transaction log, its state may appear as *Replicating (Waiting at end of log)*. In that case:

- The Log Reader component's log-scanning process "sleeps" according to the values of the `scan_sleep_increment` and `scan_sleep_max` configuration parameters.
- After the Log Transfer Interface (LTI) component finishes processing all of the change sets it received from the Log Reader and sending all of the LTL to the Replication Server, no replication throughput will occur until new replicated transactions appear in the log and the Log Reader scans them.
- The Replication Agent instance remains in *Replicating* state, unless some other event causes it to go to *Admin* state.

Changing the Replication Agent state

The state of a Replication Agent instance indicates its current operational condition, and determines which administrative tasks you can perform.

Generally, there are only two reasons to change the state of a Replication Agent instance:

- To perform certain administrative or maintenance procedures (change the state from *Replicating* to *Admin*)
- To restore normal replication processing (change the state from *Admin* to *Replicating*), either after an administrative or maintenance procedure, or after recovery from an error

Changing from
Replicating state to
Admin state

To change the state of the Replication Agent instance from *Replicating* to *Admin*, you can use either the `quiesce` or `suspend` command. See "Stopping replication in the Replication Agent" on page 72 for more information.

See the Sybase Replication Agent *Reference Manual* for more detailed information about the `quiesce` and `suspend` commands.

Changing from *Admin* state to *Replicating* state

To change the state of the Replication Agent instance from *Admin* to *Replicating*, you can use the resume command. See “Starting replication in the Replication Agent” on page 71 for more information.

See the Sybase Replication Agent *Reference Manual* for more detailed information about the resume command.

Getting Replication Agent statistics

The Replication Agent records information about the performance of its internal components whenever it is in *Replicating* state. You can use this information to tune Replication Agent performance or troubleshoot problems.

To get information about Replication Agent performance, use the `ra_statistics` command. You can also use `ra_statistics` to reset the statistics counters.

Note Each time the Replication Agent instance goes to *Replicating* state, statistics counters are reset automatically.

See the Sybase Replication Agent *Reference Manual* for more information about the `ra_statistics` command and Replication Agent performance statistics.

Shutting down the Replication Agent instance

Each Replication Agent instance can be started and shut down independently of all other components in a replication system, and independently of other Replication Agent instances.

See “Starting the Sybase Replication Agent” on page 32 for information about how to start a Replication Agent instance.

Shutting down the Replication Agent instance terminates its process on the host machine.

Note You can stop all replication processing in the Replication Agent without shutting down the instance. See “Stopping replication in the Replication Agent” on page 72 for more information.

To shut down a Replication Agent instance, you must log in to the administration port and invoke the shutdown command. The shutdown command gives you two options:

- Normal shutdown – first quiesces the Replication Agent instance, and then shuts down the instance, terminating its process.
- Immediate shutdown – shuts down the Replication Agent instance and terminates its process immediately, without first quiescing. To use this method, use the immediate keyword when you invoke the shutdown command.

Note If the Replication Agent instance is in state transition, it ignores the shutdown command with no option (normal shutdown). It does *not* ignore shutdown immediate when it is in any state, including transition from one state to another.

When a Replication Agent instance is shut down normally, it does the following:

- Stops reading the transaction log
- Drops its connection to the primary database
- Finishes processing any transactions it already has in its internal queues
- Drops its connection to the Replication Server after successfully sending LTL for any transactions in its internal queues
- Terminates its process

❖ To shut down a Replication Agent instance

- 1 Log in to the Replication Agent instance with the administrator login.
- 2 Invoke the shutdown command as follows:
 - Use the following command to shut down the Replication Agent instance normally:

```
shutdown
```

This command first quiesces the Replication Agent instance before shutting it down. If the Replication Agent instance is in *Replicating* state, and the internal queues are full when you invoke shutdown, the processing may take a while to complete, and there may be a delay before the process terminates.

See “Quiescing the Replication Agent” on page 73 for more information about quiesce processing.

- Use the following command to force an immediate shutdown, regardless of the state of the Replication Agent instance:

```
shutdown immediate
```

This command shuts down and terminates the Replication Agent instance immediately, without first quiescing.

See the Sybase Replication Agent *Reference Manual* for more detailed information about the shutdown command.

Starting replication in the Replication Agent

When you start replication in the Replication Agent:

- The Replication Agent instance opens network connections to the primary database and primary Replication Server (and the RSSD, if so configured).
- The internal Log Reader and Log Transfer Interface components begin normal replication processing—scanning the transaction log for operations to replicate, and sending replicated transactions to the Replication Server.
- The Replication Agent instance goes from *Admin* state to *Replicating* state.

When the Replication Agent instance is in *Replicating* state, it maintains connections to the primary database and the primary Replication Server (and RSSD, if so configured), and its Log Reader component scans the transaction log for transactions to replicate.

The Replication Agent instance must be running before you can start replication. See “Starting the Sybase Replication Agent” on page 32 for more information.

❖ To start replication in the Replication Agent

- 1 Log in to the Replication Agent instance with the administrator login.
- 2 Use the following command to start replication:

```
resume
```

After you invoke the resume command, the Replication Agent instance should go from *Admin* state to *Replicating* state.

- 3 Use the following command to verify that the Replication Agent instance is in *Replicating* state:

```
ra_status
```

If the Replication Agent instance does not go to *Replicating* state after you invoke the resume command, see Chapter 4, “Troubleshooting Sybase Replication Agent,” for more information.

See the Sybase Replication Agent *Reference Manual* for more detailed information about the resume command and how Sybase Replication Agent starts replication processing.

Stopping replication in the Replication Agent

When you stop replication in the Sybase Replication Agent:

- The internal Log Reader and Log Transfer Interface components stop their normal replication processing.
- Any open connections to the primary database are released, and the connection to the Replication Server is dropped.
- The Replication Agent instance goes from *Replicating* state to *Admin* state.

When the Replication Agent instance is in *Admin* state, it is running and available to execute administrative commands, but it does not maintain connections to the primary database and the primary Replication Server (and RSSD, if so configured), and it does not process replicated transactions.

Some administrative tasks require the Replication Agent instance to be in *Admin* state. In a normally operating replication system, you must stop replication in the Replication Agent to perform those tasks.

There are two ways to stop replication in the Replication Agent:

- Quiesce the Replication Agent instance to stop replication gracefully. See “Quiescing the Replication Agent” on page 73 for more information.
- Suspend the Replication Agent instance to stop replication immediately. See “Suspending the Replication Agent instance” on page 74 for more information.

Quiescing the Replication Agent

Quiescing the Replication Agent instance stops its replication processing gracefully:

- The Log Reader component stops reading operations from the transaction log when the current scan is complete. It continues to send change-set data to the Log Transfer Interface component until it finishes processing the last operation it scanned from the log.
- The Log Transfer Interface component stops sending LTL commands to the Replication Server as soon as it finishes processing the last change set it received from the Log Reader.
- When the Log Transfer Interface component is finished processing its input queue and sending the resulting LTL, the Replication Agent instance releases all of its connections to the primary database (if any are open), and drops its connection to the Replication Server (and RSSD, if connected).
- The Replication Agent instance goes from *Replicating* state to *Admin* state.

❖ To quiesce a Replication Agent instance

- 1 Log in to the Replication Agent instance with the administrator login.
- 2 Use the following command to quiesce the Replication Agent:

```
quiesce
```

After you invoke the quiesce command, the Replication Agent instance should go from *Replicating* state to *Admin* state.

- 3 Use the following command to verify that the Replication Agent instance is in *Admin* state:

```
ra_status
```

Note If the internal queues are full when you invoke the quiesce command, the quiesce processing may take a while to complete, and there may be a delay before the Replication Agent instance completes its transition to *Admin* state.

See the Sybase Replication Agent *Reference Manual* for more detailed information about the quiesce command and Sybase Replication Agent quiesce processing.

Suspending the Replication Agent instance

Suspending the Replication Agent instance stops its replication processing immediately:

- The Log Reader component stops scanning the transaction log immediately, and the Log Transfer Interface component stops sending LTL commands to the Replication Server immediately.
- All data in the Replication Agent internal queues (input and output queues of the Log Reader and Log Transfer Interface components) is flushed without further processing.
- The Replication Agent instance releases all of its connections to the primary database (if any are open), and drops its connection to the Replication Server (and RSSD, if connected).
- The Replication Agent instance goes from *Replicating* state to *Admin* state.

❖ To suspend a Replication Agent instance

- 1 Log in to the Replication Agent instance with the administrator login.
- 2 Use the following command to suspend the Replication Agent:

```
suspend
```

After you invoke `suspend`, the Replication Agent instance should go from *Replicating* state to *Admin* state.

- 3 Use the following command to verify that the Replication Agent instance is in *Admin* state:

```
ra_status
```

See the Sybase Replication Agent *Reference Manual* for more detailed information about the `suspend` command.

Managing the Replication Agent transaction log

This section describes transaction log administration and maintenance procedures for the following Sybase Replication Agent variants:

- Replication Agent for Informix
- Replication Agent for Microsoft SQL Server

- Replication Agent for DB2 Universal Database (UDB)
- Replication Agent for Oracle

Replication Agents for Informix and Microsoft SQL Server

The Replication Agent for Informix and the Replication Agent for Microsoft SQL Server are trigger-based. To capture replicated transactions in the primary database, each trigger-based Replication Agent uses its own proprietary transaction log, which consists of objects (tables, stored procedures, and triggers) that it creates, owns, and maintains in the primary database.

The only routine maintenance required for the trigger-based Replication Agent transaction log is truncation. If not periodically truncated, the Replication Agent transaction log objects can grow to consume all of the resources allocated to the primary database.

Replication Agent for UDB

The Replication Agent for UDB uses the native DB2 log to capture replicated transactions. The Replication Agent for UDB does create some “transaction log” objects in the primary database to store its system data, but those database objects require no routine maintenance. See the Sybase Replication Agent *Primary Database Guide* for information on how to manually archive the transaction log.

The Replication Agent for UDB has no effect on conventional transaction log maintenance requirements in DB2 UDB.

Replication for Oracle

The Replication Agent for Oracle uses the native Oracle log to capture replicated transactions. The transaction log objects it creates in the primary database facilitate stored procedure replication. These database objects require no routine maintenance.

Oracle log archive operations do not occur until the Replication Agent has processed all the log records in a *redo* log file. Once processed, the Replication Agent will issue the archive log command to the Oracle database.

Note See the Sybase Replication Agent *Primary Database Guide* for more details.

Administration tasks

The following sections describe each Replication Agent transaction log administration and maintenance task in detail:

- Initializing the Replication Agent transaction log
- Removing the Replication Agent transaction log
- Forcing transaction log removal
- Truncating the Replication Agent transaction log

- Backing up Replication Agent objects in the primary database

Initializing the Replication Agent transaction log

The Replication Agent instance must be running and connectivity to the primary database must be established before you can initialize a transaction log. See “Starting the Sybase Replication Agent” on page 32 and “Setting up Replication Agent connectivity” on page 43 for more information.

Note Before you initialize a Replication Agent that has a Replication Agent System Database (RASD), the primary database must be quiesced. Currently, only the Replication Agent for Oracle uses an RASD. The following procedure includes quiescing.

❖ To initialize a Replication Agent transaction log in the primary database

- 1 Log in to the Replication Agent instance with the administrator login.
- 2 Use the following command to determine if a transaction log associated with this Replication Agent instance already exists in the primary database:

```
pdb_xlog
```

If a transaction log does not exist, the `pdb_xlog` command returns no information about transaction log objects. Continue this procedure to create a transaction log in the primary database.

Note The `pdb_xlog` command looks for transaction log components based on the value of the `pdb_xlog_prefix` configuration parameter. If the value of the `pdb_xlog_prefix` parameter changed *after* a transaction log was created, the `pdb_xlog` command will not find the previously created transaction log objects.

If a Replication Agent transaction log exists in the primary database, the `pdb_xlog` command returns a list of the transaction log objects. For Replication Agent for Oracle, if replication has been initialized, it returns the Oracle database instance name.

If a transaction log exists for the Replication Agent instance, you need not complete this procedure.

- 3 If you want to use a particular string for the database object name prefix of the transaction log components, use the `ra_config` command to set the value of the `pdb_xlog_prefix` parameter:

```
ra_config pdb_xlog_prefix, XXX
```

where `XXX` is a one- to three-character string that will be the new value of the `pdb_xlog_prefix` parameter, and the prefix string used in the database object names of transaction log components when the transaction log is created. The default value of the `pdb_xlog_prefix` parameter is `ra_`.

Note The value of the `pdb_xlog_prefix_chars` parameter specifies the non-alphabetic characters that are allowed in the prefix string (the value of the `pdb_xlog_prefix` parameter). The primary data server may restrict the characters that can be used in database object names.

You can also use `ra_config` to determine the current value of the `pdb_xlog_prefix` parameter:

```
ra_config pdb_xlog_prefix
```

When you invoke `ra_config` and specify a configuration parameter with no value, it returns the current value of that parameter.

- 4 If your Replication Agent has an RASD, you must quiesce the primary database, or otherwise prevent any DDL operations that can change the database objects or schema.

Currently, only the Replication Agent for Oracle uses an RASD.

Log in to the primary data server with a user login that has appropriate permissions or authority, and then quiesce the primary database (or execute the commands necessary to prevent any DDL operations that change the database objects or schema).

- 5 Use the `pdb_xlog` command to initialize the Replication Agent transaction log:

```
pdb_xlog init
```

Note When you invoke `pdb_xlog` with the `init` keyword, the command returns an error message if a transaction log (using the prefix string currently specified in the `pdb_xlog_prefix` parameter) already exists in the primary database.

When you invoke the `pdb_xlog` command with the `init` option, the Replication Agent does the following:

- Checks the primary database for compatible settings.
- Generates a SQL script that is run in the primary database. This script creates the Replication Agent transaction log base objects.

For Replication Agents that use a RASD, the RASD is initialized with information from the primary database.

Note You can configure the Replication Agent to generate the script—but not execute—by setting the value of the `pdb_auto_run_scripts` parameter to *false* before you invoke the `pdb_xlog` command. To complete the transaction log creation, you must manually execute that script in the primary database.

If the log-creation script executes successfully, the script is stored in a file named *create.sql* in the *scripts/xlog/installed* directory.

If the log-creation script does *not* execute successfully, the primary database is not changed, and the script is stored in a file named *create.sql* in the *scripts/xlog* directory.

Check the primary database error log to determine why the log-creation script did not execute successfully. To get the log-creation script to execute successfully, you may need to edit the script file. See Chapter 4, “Troubleshooting Sybase Replication Agent,” for more information.

Removing the Replication Agent transaction log

The Replication Agent instance must be running in *Admin* state to remove the transaction log. See “Starting the Sybase Replication Agent” on page 32 for more information.

❖ **To remove the Replication Agent transaction log from the primary database**

- 1 Log in to the Replication Agent instance with the administrator login.
- 2 Use the `pdb_xlog` command to verify that a transaction log associated with this Replication Agent instance exists in the primary database:

```
pdb_xlog
```

If a transaction log does not exist for this Replication Agent instance, the `pdb_xlog` command returns no information about transaction log objects. If no transaction log exists, you need not complete this procedure.

Note The `pdb_xlog` command looks for transaction log components based on the current value of the `pdb_xlog_prefix` configuration parameter. If the value of the `pdb_xlog_prefix` parameter changed after a transaction log was created for a Replication Agent instance, the `pdb_xlog` command will not find the previously created transaction log.

If a transaction log exists for this Replication Agent instance, the `pdb_xlog` command returns a list of the names of the transaction log database objects. Continue this procedure to remove the transaction log from the primary database.

- 3 Use the `pdb_setreptable` command to disable replication for all marked tables in the primary database:

```
pdb_setreptable all, disable
```

When you invoke the `pdb_setreptable` command with the `all` and `disable` keywords, Replication Agent disables replication for all marked tables in the primary database.

- 4 Use the `pdb_setrepproc` command to disable replication for all marked procedures in the primary database:

```
pdb_setrepproc all, disable
```

- 5 Use the `pdb_setreptable` command to unmark all marked tables in the primary database:

```
pdb_setreptable all, unmark
```

When you invoke the `pdb_setreptable` command with the `all` and `unmark` keywords, Replication Agent removes replication marking from all marked tables in the primary database.

If there are pending transactions for any trigger-based table that you attempt to unmark, `pdb_setreptable` returns an error message. You can force an unmark, regardless of pending transactions or the table's enable status, by following the `unmark` keyword with the `force` keyword:

```
pdb_setreptable all, unmark, force
```

- 6 Use the `pdb_setrepproc` command to unmark all marked procedures in the primary database:

```
pdb_setrepproc all, unmark
```

When you invoke the `pdb_setreproc` command with the `all` and `unmark` keywords, Replication Agent removes replication marking from all marked procedures in the primary database.

If there are pending transactions for any trigger-based procedure that you attempt to unmark, `pdb_setreproc` returns an error message. You can force an unmark, regardless of pending transactions or the procedure's enable status, by following the `unmark` keyword with the `force` keyword:

```
pdb_setreproc all, unmark, force
```

Note Normally, if any objects in the primary database are marked for replication, you cannot remove the Replication Agent transaction log.

- 7 Use the `pdb_xlog` command to remove the Replication Agent transaction log:

```
pdb_xlog remove
```

Note When you invoke the `pdb_xlog` command with the `remove` keyword, the command returns an error message if no transaction log exists in the primary database.

After you invoke the `pdb_xlog` command with the `remove` keyword, Replication Agent generates a script that, when executed, removes the transaction log base objects from the primary database.

Note You can configure Replication Agent to simply build the script, but not execute it, by setting the value of the `pdb_auto_run_scripts` parameter to `false` prior to invoking the `pdb_xlog` command. To complete the transaction log removal process when there is no RASD involved, you have to manually execute the script.

If the log removal script executes successfully, the script is stored in a file named `remove.sql` in the `RAX-12_6\inst_name\scripts\xlog\installed` directory.

If the log removal script does not execute successfully, the script is stored in a file named `remove.sql` in the `RAX-12_6\inst_name\scripts\xlog` directory.

Forcing transaction log removal

When you invoke the `pdb_xlog` command with the `remove` keyword, Replication Agent creates a script (`remove.sql`) that, when executed successfully, removes all transaction log objects from the primary database.

In the event that the `remove.sql` script fails for some reason, some transaction log components may be removed from the primary database and some components may remain.

Note If errors cause a script execution failure, you should refer to your primary database error logs and the Replication Agent system log to evaluate the any errors and determine if any corrective action is necessary.

Even after you correct the cause of the failure, if you simply re-execute the `remove.sql` script after a failure in which some (but not all) transaction log components were removed, the script will immediately fail again because of the missing (successfully removed) transaction log components.

To avoid this problem and finish removing transaction log components after a script execution failure, you can invoke the `pdb_xlog` command with the `remove` keyword followed by the `force` keyword:

```
pdb_xlog remove, force
```

When you use the `force` keyword, the execution of the `remove.sql` script continues, even when errors are encountered, until the script is finished.

Truncating the Replication Agent transaction log

The Replication Agents for Informix, Microsoft SQL Server, and Oracle support both automatic and manual transaction log truncation.

You can enable or disable automatic log truncation at any time, and you can truncate the Replication Agent transaction log manually at any time, with automatic log truncation either enabled or disabled.

Note The Replication Agent for UDB does *not* support transaction log truncation.

The Replication Agent for Oracle supports transaction log truncation by controlling archive operations against the Oracle *redo* logs. See the Replication Agent *Primary Database Guide* for details.

When the Replication Agent truncates its transaction log, either automatically or on command (manually), the truncation point is determined by the most recent LTM Locator received from the primary Replication Server.

Automatic truncation

You have two options for automatic transaction log truncation:

- Automatic truncation each time the Replication Agent receives a new LTM Locator value from the primary Replication Server
- Periodic truncation on a time interval you specify

Replication Agent truncates the transaction log based on the most recent truncation point received from the primary Replication Server. The truncation point is part of the information contained in the LTM Locator.

❖ **To enable automatic log truncation**

- 1 Log in to the Replication Agent instance with the administrator login.
- 2 Use the `ra_config` command to enable automatic log truncation and specify the type of automatic truncation:

- Use the following command to enable automatic log truncation at a specified time interval:

```
ra_config truncation_type, interval
```

- Use the following command to enable automatic log truncation whenever the Replication Agent receives a new LTM Locator value from the primary Replication Server:

```
ra_config truncation_type, locator_update
```

- 3 If you enabled automatic log truncation based on a time interval, use `ra_config` to specify the time interval for automatic truncation:

```
ra_config truncation_interval, N
```

where *N* is the number of seconds between automatic truncations.

Note The maximum `truncation_interval` value is 720.

See the Sybase Replication Agent *Reference Manual* for more information about the `truncation_interval` and `truncation_type` configuration parameters.

❖ **To disable automatic log truncation**

- 1 Log in to the Replication Agent instance with the administrator login.
- 2 Use the `ra_config` command to disable automatic log truncation:

```
ra_config truncation_type, command
```

Note If the value of the `truncation_type` parameter is `interval`, and the value of the `truncation_interval` parameter is 0 (zero), automatic log truncation is effectively disabled.

Manual truncation

If automatic log truncation is disabled, you must periodically truncate the Replication Agent transaction log manually.

❖ **To truncate the Replication Agent transaction log manually**

- 1 Log in to the Replication Agent instance with the administrator login.
- 2 Use the following command to truncate the Replication Agent transaction log:

```
pdb_truncate_xlog
```

The `pdb_truncate_xlog` command is asynchronous; it does not return success or failure, unless an immediate error occurs.

Note To determine the results of `pdb_truncate_xlog`, you can use the `ra_statistics` command, both *before* and *after* you invoke `pdb_truncate_xlog`, to compare the Number of transactions truncated value. Replication Agent for Oracle does *not* provide this statistic.

See the Sybase Replication Agent *Reference Manual* for more information about the `pdb_truncate_xlog` command.

Note As an alternative to the Replication Agent automatic log truncation feature, you can use a scheduler utility to execute the `pdb_truncate_xlog` command in a script.

Backing up Replication Agent objects in the primary database

The Sybase Replication Agent does not support automatically backing up and restoring Replication Agent objects in the primary database.

Sybase recommends that you use the database backup utilities provided by your primary database vendor to periodically back up the Replication Agent transaction log objects in the primary database.

Managing the Replication Agent System Database

A new feature introduced in Sybase Replication Agent version 12.6, the Replication Agent for Oracle uses an embedded database, managed by Adaptive Server Anywhere, for its Replication Agent System Database (RASD).

There are four tasks you can perform to maintain the RASD:

- Updating the RASD
- Backing up the RASD
- Restoring the RASD
- Truncating the RASD

RASD overview

Each instance of the Replication Agent for Oracle depends on the information in its Replication Agent System Database (RASD) to recognize database structure or schema objects in the transaction log.

When you create a Replication Agent instance, the RASD is created automatically, but it contains no information until you *initialize* the Replication Agent instance.

When you initialize a Replication Agent instance, it does the following:

- Queries the primary database to get information about the database structure or schema
- Stores information about the database schema in its RASD

Initializing the Replication Agent is one of the tasks required to set up the replication system, and it has several prerequisites. See “The Replication Agent instance” on page 11 for more information about these tasks, and initializing the Replication Agent.

After the RASD is initially populated, its contents will be synchronized with the primary database automatically during normal replication (without intervention).

If replication does not occur, the contents of the RASD become stale (not synchronized with the primary database), and you should rebuild them before use.

Most of the common data definition language (DDL) commands and system procedures executed in the primary database are recorded in the transaction log, and they are replicated to the standby database. When it processes those DDL transactions for replication, the Replication Agent updates the RASD automatically.

If a DDL command or system procedure produces a change in the primary database schema and the Replication Agent cannot recognize that command or procedure and update its RASD automatically, a replication failure will occur if a subsequent transaction changes data in an object that is not recorded in the RASD. In that event, you must re-initialize and quiesce the primary database, and force the Replication Agent to update its RASD. See “Updating the RASD” on page 86 for more information.

Each time it processes a DDL transaction that affects an existing database object, the Replication Agent creates a new *version* of the object metadata in its RASD. The version of each object is identified by the LTM Locator value of the DDL transaction that changed it.

Previous versions of objects must be kept in the RASD long enough to allow system recovery. For example, replaying a transaction that involved an object before it was changed by DDL could produce an error (or data inconsistency) with the current version of the object.

Note The Sybase Replication Agent does not support replaying transactions from a restored transaction log.

The Replication Agent determines which version of each object to use by comparing the object’s current version string with the current LTM Locator value. If the current LTM Locator value is greater than or equal to the value of the object version, the current object metadata is used. If the current LTM Locator value is less than the value of the object version, a previous version of the object metadata must be used.

Without periodic truncation, the size of the RASD can grow indefinitely, as more and more versions of object metadata are added. See “Truncating the RASD” on page 90 for more information.

Updating the RASD

Note The RASD is normally updated automatically during normal replication activity. The following procedure to force an update of the RASD should only be used with the recommendation of Sybase Technical Support when the RASD is suspected of being corrupt.

Once the data in the RASD is created by initializing the Replication Agent instance, the only way to update the RASD is to *re-initialize* the instance using the `pbd_xlog` command with the `init` and `force` keywords.

Note Before you re-initialize the Replication Agent, the primary database must be quiesced. The following procedure includes quiescing.

❖ To update the RASD

- 1 Log in to the Replication Agent instance with the administrator login.
- 2 Use the following command to determine the state of the Replication Agent instance:

```
ra_status
```

- 3 If the Replication Agent is in *Admin* state, skip this step and continue with step 4.

If the Replication Agent is in *Replicating* state:

- a Use the following command to quiesce the Replication Agent instance:

```
quiesce
```

- b Use the following command to verify that the Replication Agent is in *Admin* state:

```
ra_status
```

- 4 Before you re-initialize the RASD, you must quiesce the primary database, or otherwise prevent any DDL operations that can change the database objects or schema.

Log in to the primary data server, using a user login that has appropriate permissions or authority, and then quiesce the primary database (or execute the commands necessary to prevent any DDL operations that change the database objects or schema).

- 5 Use the following command to re-initialize the Replication Agent and force it to update its RASD:

```
    pbd_xlog init, force
```

Note The `pbd_xlog init, force` command does *not* overwrite any marking information or configurations. It also does *not* overwrite any existing path information to the log device (or redo log in Oracle) in the RASD, if all of the following log information in the RASD matches that returned by the primary data server.

The following log information pertains only to Oracle:

- Redo group number
- Redo name
- Redo path

For each log device or redo log identified in the RASD, if any item in the preceding list does *not* match the information returned by the primary data server, `pbd_xlog init, force` overwrites the RASD record for that log device or redo log with the information returned by the primary data server.

- 6 After the Replication Agent is initialized, you must release the quiesce hold on the primary database to restore normal client access to the database. Use *one* of the following methods to do this:

Log in to the primary data server, using a user login that has appropriate permissions or authority, and then release the quiesce hold on the primary database (or execute the commands necessary to restore normal client access to the database).

- 7 Use the following command to resume replication in the Replication Agent:

```
    resume
```

- 8 Use the following command to verify that the Replication Agent is in *Replicating* state:

```
    ra_status
```

If the Replication Agent does not return to *Replicating* state, see Chapter 4, “Troubleshooting Sybase Replication Agent,” for more information.

Backing up the RASD

Like any database, you should periodically back up the RASD to prevent data loss in the event of a device failure.

Note Sybase recommends that you always back up the RASD *after* you truncate the primary database log.

The Replication Agent places RASD backup files in the directory identified by the `rasd_backup_dir` configuration parameter. See the Sybase Replication Agent *Reference Manual* for more information about the `rasd_backup_dir` parameter.

You need not interrupt replication to back up the RASD. You can back up the RASD at any time, when the Replication Agent instance is in any state.

❖ To back up the RASD

- 1 Log in to the Replication Agent instance with the administrator login.
- 2 Use the following command to back up the RASD:

```
rasd_backup
```

After the backup completes successfully, the Replication Agent returns a message to confirm that the RASD backup was successful.

If the Replication Agent could not find the directory identified in the `rasd_backup_dir` parameter, or if it could not write the RASD backup files in that directory (for example, because of a permission problem), it returns an error. You must correct the cause of the error to successfully back up the RASD.

Restoring the RASD

If the RASD becomes corrupt (for example, because of a device failure), you can restore the database from the most recent backup files.

The Replication Agent retrieves the RASD backup files from the directory identified by the `rasd_backup_dir` configuration parameter. See the Sybase Replication Agent *Reference Manual* for more information about the `rasd_backup_dir` parameter.

Note To restore the RASD, the Replication Agent instance must be in *Admin* state.

❖ **To restore the RASD**

- 1 Log in to the Replication Agent instance with the administrator login.
- 2 Use the following command to determine the state of the Replication Agent instance:

```
ra_status
```

- 3 If the Replication Agent is in *Admin* state, skip this step and continue with step 4.

If the Replication Agent is in *Replicating* state:

- a Use the following command to quiesce the Replication Agent instance:

```
quiesce
```

- b Use the following command to verify that the Replication Agent is in *Admin* state:

```
ra_status
```

- 4 After you verify that the Replication Agent is in *Admin* state, use the following command to restore the RASD:

```
rasd_restore
```

After the restore completes successfully, the Replication Agent returns a message to confirm that the RASD restore was successful.

If the Replication Agent cannot find the directory identified in the `rasd_backup_dir` parameter, or if it cannot read the RASD backup files in that directory (for example, because of a permission problem), it returns an error. You must correct the cause of the error to restore the RASD.

- 5 After the RASD is successfully restored from the most recent backup, use the following command to resume replication in the Replication Agent instance:

```
resume
```

If the Replication Agent does not return to *Replicating* state, see Chapter 4, “Troubleshooting Sybase Replication Agent,” for more information.

Truncating the RASD

To keep the RASD from growing indefinitely, you can periodically truncate older versions of its primary database object metadata.

Note Back up the RASD using `rasd_backup` *before* you truncate it. See “Backing up the RASD” on page 88 for more information.

The RASD stores definitions for two types of database objects:

- Articles – tables and stored procedures that are marked for replication.
- Users – database users who apply transactions in the primary database.

Use the `ra_truncatearticles` and `ra_truncateusers` commands to manage the size of the RASD.

You need not interrupt replication to truncate the RASD. You can truncate the RASD at any time, when the Replication Agent instance is in any state.

❖ To truncate older versions of articles in the RASD

- 1 Log in to the Replication Agent instance with the administrator login.
- 2 Use the following command to truncate articles in the RASD:

```
ra_truncatearticles NNN
```

where *NNN* is an LTM Locator value that identifies the oldest non-current version of any article to be kept.

All non-current versions of all articles that are *less than* the LTM Locator value you specify are truncated from the RASD. If the current (most recent) version of an article is older than the version identified by the LTM Locator value, it is *not* truncated.

❖ To truncate older versions of users in the RASD

- 1 Log in to the Replication Agent instance with the administrator login.
- 2 Use the following command to truncate users in the RASD:

```
ra_truncateusers NNN
```

where *NNN* is an LTM Locator value that identifies the oldest non-current version of any user to be kept.

All non-current versions of all users that are *less than* the LTM Locator value you specify are truncated from the RASD. If the current (most recent) version of a user is older than the version identified by the LTM Locator value, it is *not* truncated.

Identifying replicated transactions and procedures

In a Sybase transaction replication system, the Sybase Replication Agent and Replication Server components both provide features that allow you to identify (or select) the transactions that you want to replicate. You need not replicate all transactions, or all data-changing operations, in the primary database.

The ability to select transactions for replication is particularly useful when you need to implement a replication system to support an application that uses some of the tables in a database, but not all of them.

By marking tables, you identify the specific tables in the primary database for which transactions are replicated. Transactions that affect the data in marked tables are referred to as *replicated transactions*.

Note If a transaction affects data in both marked and unmarked tables, only the operations that affect data in marked tables are replicated.

By marking stored procedures, you identify (or select) the specific procedures in the primary database that will be replicated as *applied functions*. When a marked procedure is invoked in the primary database, its invocation is replicated, along with its input parameter values, to the replicate database.

The ability to select procedures for replication is particularly useful when you need to implement a replication system to support an application that uses stored procedures, or when replicating a single procedure invocation is more efficient than replicating numerous, individual data-changing operations that are produced by a single procedure invocation.

Sybase Replication Agent provides the following features to allow you to select replicated transactions and procedures:

- Marking and unmarking tables

- Enabling and disabling replication for marked tables
- Enabling and disabling replication for LOB columns
- Marking and unmarking stored procedures
- Enabling and disabling replication for stored procedures
- Enabling and disabling replication for DDL (Replication Agent for Oracle)

Preparing to mark tables or stored procedures

Before you can mark tables or stored procedures for replication, you must create the Replication Agent transaction log objects.

See the following for more information:

- “Initializing the Replication Agent transaction log” on page 76
- “Managing the Replication Agent System Database” on page 84

Marking and unmarking tables

To replicate transactions that affect the data in a table in the primary database, that table must be marked for replication, and replication must be enabled for the marked table.

Marking a table can be separate from enabling replication for that table. If the value of the `pdb_dflt_object_repl` parameter is true, replication is enabled automatically at the time a table is marked. See “Enabling and disabling replication for marked tables” on page 99 for more information.

The table-marking process of trigger-based Replication Agents (for Informix and Microsoft SQL Server) is different from the table-marking processes of the Replication Agent for UDB and the Replication Agent for Oracle.

Table marking with
trigger-based
Replication Agents

When a table is marked for replication, a trigger-based Replication Agent creates several transaction log objects in the primary database to support replication for that table:

- Shadow table – prefixSH_xxx
- Blob shadow table – prefixBSH_xxx
- Shadow-row procedure – prefixSRP_xxx

- Blob shadow-row procedure – prefixBSRP_xxx
- Insert trigger – prefixINSTRG_xxx
- Update trigger – prefixUPDTRG_xxx
- Delete trigger – prefixDELTRG_xxx

where:

- *prefix* is the 1 to 3-character transaction log prefix string specified by the `pdb_xlog_prefix` parameter.
- *xxx* is an automatically generated alphanumeric counter, which is incremented each time an object is marked in the primary database.

Note The prefix string recorded in the `pdb_xlog_prefix` parameter is used for naming all Replication Agent transaction log database objects.

For example, if the value of the `pdb_xlog_prefix` parameter is `ra_` (the default), and two tables have already been marked, the Replication Agent generates the following transaction log object names for the third marked table:

- Shadow table – `ra_SH_c`
- Blob shadow table – `ra_BSH_c`
- Shadow-row procedure – `ra_SRP_c`
- Blob shadow-row procedure – `ra_BSRP_c`
- Insert trigger – `ra_INSTRG_c`
- Update trigger – `ra_UPDTRG_c`
- Delete trigger – `ra_DELTRG_c`

Note If triggers exist on the table, the Replication Agent attempts to insert its trigger code, leaving the existing trigger code intact. You can configure the Replication Agent instance to simply build a script to mark the table—but not execute it—by setting the value of the `pdb_auto_run_scripts` parameter to `false`. To complete the table marking, you have to manually execute the script.

If you need to change the schema of a marked table in the primary database, you must unmark the table to remove the shadow table, stored procedure, and triggers that Replication Agent creates for the primary table, then change the primary table's schema and re-mark the table.

Note If you change the schema of a primary table, you may need to rematerialize the replicate table.

Table unmarking with trigger-based Replication Agents

When you unmark a table marked for replication, a trigger-based Replication Agent removes the transaction log objects for that table from the primary database.

Sybase Replication Agent provides two ways to unmark an object:

- Normal unmarking – The Replication Agent first checks to see if replication is disabled for the table, then it checks for any unprocessed transactions in the transaction log. If replication is not disabled, or if there are any unprocessed transactions in the transaction log, the unmarking operation fails.
- Force unmarking – The Replication Agent unmarks the table immediately, without checking the replication enabled status or looking for unprocessed transactions in the transaction log.

Note Forced unmarking can cause transactional inconsistency between the primary database and replicate database. In general, it should be used only when you are removing the entire transaction log and discontinuing replication from the primary database, or in troubleshooting procedures.

Table marking with the Replication Agent for UDB

When a table is marked for replication with the log-based Replication Agent for UDB (DB2), the Replication Agent does the following:

- Logs in to the primary database and sets the value of the table's DATA CAPTURE option to DATA CAPTURE CHANGES.
- Adds a row to the Replication Agent marked objects table in the primary database. Each row in the marked objects table lists attributes of a table marked for replication in the primary database.

If you need to change the schema of a marked table, you must:

- 1 Lock the table so that new operations cannot change any data in the table.
- 2 Wait for the Replication Agent to complete its processing of any logged transactions in the table.

- 3 Quiesce the Replication Agent instance.
- 4 Change the table's schema, without changing the DATA CAPTURE option.
- 5 Unlock the table to allow normal user or client access.
- 6 Use the Replication Agent resume command to restart replication.

Note If you change the schema of a primary table, you may need to rematerialize the replicate table.

Table unmarking with the Replication Agent for UDB

When you unmark a table marked for replication with the log-based Replication Agent for UDB, the Replication Agent does the following:

- Logs in to the primary database and restores the value of the table's DATA CAPTURE option to the value it had before the table was marked.
- Deletes the table's row in the Replication Agent marked objects table.

When a table is unmarked, any subsequent operations that affect the data in that table are ignored (not replicated).

Note In the event that the Replication Agent for UDB must re-scan the transaction log, such as when recovering from a replication error, transactions recorded prior to unmarking a table will not be replicated.

Table marking with the Replication Agent for Oracle

When a table is marked for replication with the log-based Replication Agent for Oracle, the Replication Agent does the following:

- Connects to the RASD
- Records the mark status for the table in the RASD Article for that table.

When a table is marked, any subsequent operations that affect the data in that table are replicated.

Table unmarking with the Replication Agent for Oracle

When you unmark a table marked for replication with the log-based Replication Agent for Oracle, the Replication Agent does the following:

- Connects to the RASD
- Records the unmark status for the table in the RASD Article for that table

When a table is unmarked, any subsequent operations that affect the data in that table are ignored (not replicated).

Marking a table for replication

Use the following procedure to mark tables for replication with any Sybase Replication Agent (either log-based or trigger-based).

❖ To mark a table in the primary database for replication

- 1 Log in to the Replication Agent instance with the administrator login.
- 2 Use the `pdb_setreptable` command to determine if the table is already marked:

```
    pdb_setreptable pdb_table
```

where *pdb_table* is the name of the table that you want to mark for replication.

If the `pdb_setreptable` command returns information that the specified table is marked for replication, you need not continue this procedure.

If the `pdb_setreptable` command returns information that the specified table is not marked, continue this procedure to mark the table for replication.

- 3 Use the `pdb_setreptable` command to mark the table for replication.

The `pdb_setreptable` command allows you to mark the primary table to be replicated and specify a different table name to use in the replicate database (as specified in a replication definition).

- Use the following command to mark the table for replication when the replicate table (in the replicate database) has the same table name:

```
    pdb_setreptable pdb_table, mark
```

where *pdb_table* is the name of the table that you want to mark for replication.

- Use the following command to mark the table for replication when the replicate table (in the replicate database) has a different table name:

```
    pdb_setreptable pdb_table, rep_table, mark
```

where:

- *pdb_table* is the name of the table that you want to mark for replication.

- *rep_table* is the name of the replicate table in the replicate database.

Note When the primary table and replicate table have different names, you must use the `with all tables named rep_table` clause when you create the replication definition in the primary Replication Server.

- When marking a table for replication, you have the option to specify that the table owner's name is sent with the table name in the LTL. To do this, use the `owner` keyword after the `mark` keyword, as shown in the following example:

```
pdb_setreptable pdb_table, mark, owner
```

where *pdb_table* is the name of the table that you want to mark for replication.

Note The table owner's name returned from the primary database must be the same as the owner name specified in the replication definition for the table.

If the value of the `pdb_dflt_object_repl` parameter is `true`, the table you marked for replication is ready for replication immediately after the `pdb_setreptable` command returns successfully.

The default value of the `pdb_dflt_object_repl` parameter is `true`.

If the value of the `pdb_dflt_object_repl` parameter is `true`, you can skip step 4 in this procedure.

If the value of the `pdb_dflt_object_repl` parameter is `false`, you must enable replication for the table, as described in step 4.

- 4 Use the `pdb_setreptable` command to enable replication for a marked table:

```
pdb_setreptable pdb_table, enable
```

where *pdb_table* is the name of the marked table.

After replication is enabled for the table, the Replication Agent can begin replicating transactions that affect data in that table.

Unmarking a table

Use the same procedure to unmark tables for replication with either a log-based Replication Agent or a trigger-based Replication Agent. For non-RASD instances, Replication Agent must be in *Admin* state when unmarking.

❖ To unmark a table in the primary database

- 1 Log in to the Replication Agent instance with the administrator login.
- 2 Use the `pdb_setreptable` command to confirm that the table is marked in the primary database:

```
    pdb_setreptable pdb_table
```

where *pdb_table* is the name of the table in the primary database that you want to unmark.

If the `pdb_setreptable` command returns information that the specified table is marked, continue this procedure to unmark the table.

If the `pdb_setreptable` command does not return information that the specified table is marked, you need not continue this procedure.

- 3 Use the `pdb_setreptable` command to disable replication from the table:

```
    pdb_setreptable pdb_table, disable
```

where *pdb_table* is the name of the table in the primary database that you want to disable.

- 4 Use the `pdb_setreptable` command to remove the replication marking from the table:

```
    pdb_setreptable pdb_table, unmark
```

where *pdb_table* is the name of the table in the primary database that you want to unmark.

If you need to force the unmark, you can use the following command:

```
    pdb_setreptable pdb_table, unmark, force
```

- 5 Use the `pdb_setreptable` command to confirm that the table is no longer marked for replication:

```
    pdb_setreptable pdb_table
```

where *pdb_table* is the name of the table in the primary database that you unmarked.

Note You can unmark all marked objects in the primary database by invoking the `pdb_setreptable` command with the `all` keyword.

Enabling and disabling replication for marked tables

If you need to temporarily stop replication for a marked table (for example, when maintenance operations are performed in the primary database), you can disable replication for a marked table without affecting replication for other tables in the primary database. Then, when you are ready to resume replication from that table, you can enable replication for that table without affecting other tables in the database.

To replicate transactions that affect the data in a table, that table must be marked for replication, and replication must be enabled for the marked table. See “Marking and unmarking tables” on page 92 for more information.

Replication Agents for Informix, Microsoft SQL Server, and UDB have a marked objects table which contains an entry for each marked table in the primary database. Each marked table row contains a flag indicating whether replication is enabled or disabled for the marked table. Replication Agent for Oracle has *articles* in the RASD. An article is an object that has a one-to-one relationship to the tables and has a marked indicator.

When replication is disabled for a marked object, the marking infrastructure remains in place, but no transactions for that object are captured in the transaction log.

Note For Replication Agents for Informix, Microsoft SQL Server, or DB2 UDB, if you need to change the schema of a marked table in the primary database, you must first unmark the table to remove the transaction log objects that Replication Agent creates for the primary table.

This is not required for Replication Agent for Oracle since DDL commands are captured in the Oracle database and the RASD will be updated automatically.

See “Marking and unmarking tables” on page 92 for more information.

Enabling replication for marked tables

Use the same procedure to enable replication for marked tables with either a log-based Replication Agent or a trigger-based Replication Agent.

❖ To enable replication for a marked table

- 1 Log in to the Replication Agent instance with the administrator login.
- 2 Use the `pdb_setreptable` command to verify that replication is disabled for the table:

```
pdb_setreptable pdb_table
```

where *pdb_table* is the name of the marked table you want to enable replication for.

If the `pdb_setreptable` command returns information that the table is marked and has replication disabled, continue this procedure to enable replication for the table.

Note A table must be marked for replication before replication can be enabled or disabled for the table.

- 3 Use the `pdb_setreptable` command to enable replication for the table:

```
pdb_setreptable pdb_table, enable
```

where *pdb_table* is the name of the marked table in the primary database for which you want to enable replication.

After replication is enabled for the table, any transaction that affects the data in that table will be captured for replication.

- 4 You can use the `pdb_setreptable` command again to verify that replication is now enabled for the table:

```
pdb_setreptable pdb_table
```

where *pdb_table* is the name of the marked table for which you want to verify that replication is enabled.

Disabling replication for marked tables

Use the same procedure to disable replication for marked tables with either a log-based Replication Agent or a trigger-based Replication Agent.

❖ To disable replication for a marked table

- 1 Log in to the Replication Agent instance with the administrator login.

- 2 Use the `pdb_setreptable` command to verify that replication is enabled for the table:

```
    pdb_setreptable pdb_table
```

where *pdb_table* is the name of the marked table for which you want to disable replication.

If the `pdb_setreptable` command returns information that the table is marked and has replication enabled, continue this procedure to disable replication for the table.

Note A table must be marked for replication before replication can be enabled or disabled for the table.

- 3 Use the `pdb_setreptable` command to disable replication for the table:

```
    pdb_setreptable pdb_table, disable
```

where *pdb_table* is the name of the marked table in the primary database for which you want to disable replication.

After replication is disabled for the table, transactions that affect the data in that table will not be captured for replication until replication is enabled again.

- 4 You can use the `pdb_setreptable` command again to verify that replication is now disabled for the table:

```
    pdb_setreptable pdb_table
```

where *pdb_table* is the name of the marked table for which you want to verify that replication is disabled.

Enabling and disabling replication for LOB columns

In this document, all columns that contain large object (LOB) datatypes are referred to as LOB columns, regardless of the actual datatype name used by the primary database vendor. To replicate transactions that affect a LOB column, replication must be enabled for that column.

You must enable replication for each LOB column you want to replicate, in addition to marking and enabling replication for the table that contains the LOB column.

If the value of the `pdb_dflt_column_repl` parameter is true, replication is enabled automatically for all LOB columns in a table at the time the table is marked. If the value of the `pdb_dflt_column_repl` parameter is false, replication is not enabled automatically for any LOB columns in a table at the time the table is marked. See “Marking and unmarking tables” on page 92 for more information on marking a table for replication.

When a table is marked for replication and replication is enabled for that table but not for a LOB column in that table, any part of a transaction that affects the LOB column is not replicated. The portion of a transaction that affects all other non-LOB columns is replicated if the table is marked for replication and replication is enabled for the table.

When replication is enabled for a LOB column, Replication Agent makes an entry in the `prefixBLOB_COLUMNS_` table to support replication for that column.

When Replication Agent triggers process a transaction that affects a LOB column, the LOB data is not stored in the transaction log because of its possible size. Instead, the Replication Agent Log Reader component reads the LOB data directly from the primary database at the time it processes the transaction.

Because of the way Replication Agent processes the LOB column data when replicating transactions, it is possible to compromise transaction integrity. For example, if two transactions change the data in a LOB column and the Log Reader does not process the first transaction until after the second transaction has been committed, when the LOB data is read from the primary database, the value of that data is the result of the second transaction. In this event, the value of the LOB data in the first transaction is never sent to the replicate database. After the second transaction is processed by the Log Reader, the primary and replicate databases will be synchronized again, but for a period of time between processing the first and second transactions, the replicate database contains data that does not match the originating transaction.

This problem occurs only when a LOB column is changed more than once by a sequence of transactions. The period of time over which the problem exists could be significant if the replication system throughput is slow or if a replication system component fails. As soon as the last transaction that changes the LOB column is processed at the replicate site, the problem will be corrected.

In contrast, Oracle logs all LOB data (except for *BFILE* datatypes) in the Oracle redo log. This allows the Replication Agent to apply each individual LOB change. However, for *BFILE* data, the same technique is used as for the trigger-based solution and the same limitation exists—*BFILE* data is not logged, but read from the database at the time the rest of the transaction is processed. If two consecutive transactions modify the same *bfile*, the same inconsistency described above can occur.

See the Replication Agent *Primary Database Guide* for more information on LOB handling for Oracle.

Enabling replication for LOB columns

❖ To enable replication for a LOB column in a marked table

- 1 Log in to the Replication Agent instance with the administrator login.
- 2 Use the `pdb_setrepcol` command to verify that replication is disabled for the LOB column:

```
pdb_setrepcol pdb_table, pdb_col
```

where:

- *pdb_table* is the name of the marked table that contains the LOB column.
- *pdb_col* is the name of the LOB column.

If the `pdb_setrepcol` command returns information that the LOB column has replication disabled, continue this procedure to enable replication for the column.

Note The table that contains the LOB column must be marked for replication before replication can be enabled or disabled for a LOB column.

- 3 Use the `pdb_setrepcol` command to enable replication for the LOB column:

```
pdb_setrepcol pdb_table, pdb_col, enable
```

where:

- *pdb_table* is the name of the marked table that contains the LOB column.

- *pdb_col* is the name of the LOB column for which you want to enable replication.

After replication is enabled for the LOB column (and if replication is enabled for the table that contains the column), any transaction that affects the data in that column will be replicated.

- 4 You can use the `pdb_setrepcol` command again to verify that replication is now enabled for the LOB column:

```
pdb_setrepcol pdb_table, pdb_col
```

where *pdb_table* is the name of the marked table that contains the LOB column and *pdb_col* is the name of the LOB column for which you want to verify that replication is enabled.

Disabling replication for LOB columns

❖ To disable replication for a LOB column in a marked table

- 1 Log in to the Replication Agent instance with the administrator login.
- 2 Use the `pdb_setrepcol` command to verify that replication is enabled for the LOB column:

```
pdb_setrepcol pdb_table, pdb_col
```

where:

- *pdb_table* is the name of the marked table that contains the LOB column.
- *pdb_col* is the name of the LOB column you want to disable replication for.

If the `pdb_setrepcol` command returns information that the LOB column has replication enabled, continue this procedure to disable replication for the column.

Note The table that contains the LOB column must be marked for replication before replication can be enabled or disabled for a LOB column.

- 3 Use the `pdb_setrepcol` command to disable replication for the LOB column:

```
pdb_setrepcol pdb_table, pdb_col, disable
```

where:

- *pdb_table* is the name of the marked table that contains the LOB column.
- *pdb_col* is the name of the LOB column for which you want to disable replication.

After replication is disabled for the LOB column, transactions that affect the data in that column will not be replicated unless replication is enabled for that column again.

- 4 You can use the `pdb_setrepcol` command again to verify that replication is now disabled for the LOB column:

```
    pdb_setrepcol pdb_table, pdb_col
```

where:

- *pdb_table* is the name of the marked table that contains the LOB column.
- *pdb_col* is the name of the LOB column for which you want to verify that replication is disabled.

Marking and unmarking stored procedures

Sybase Replication Agent supports Replication Server function replication by replicating the invocation of stored procedures in the primary database.

Note In this document, the terms *function* and *stored procedure* are synonyms.

Sybase Replication Agent can replicate both *applied functions* and *request functions*:

- Applied functions are stored procedures that are executed in the primary database and generate transactions that affect data in the primary database.
- Request functions are stored procedures that are invoked in one database (for example, a replicate database), then executed in another database (for example, a primary database).

Replication Agent does not distinguish between these two function types, except to supply a specific user and password for use with request functions. If you are using request functions, the configuration parameters `function_username` and `function_password` must be supplied. For more information about applied and request functions, see the Managing Replicated Functions chapter of the Replication Server Administration Guide. For more information about the `function_username` and `function_password` configuration parameters, see the Replication Agent *Reference Manual*.

In order to replicate a stored procedure invoked in a primary database, the stored procedure must be marked for replication and replication must be enabled for that stored procedure. (This is analogous to marking and enabling replication for tables.)

Note Marking a stored procedure for replication is separate from enabling replication for the stored procedure. If the value of the `pdb_dflt_object_repl` parameter is `true`, replication is enabled automatically at the time a stored procedure is marked. See “Enabling and disabling replication for stored procedures” on page 110 for more information on enabling and disabling replication for a marked stored procedure.

If a marked stored procedure performs operations that affect a marked table, the operations that affect the marked table are not captured for replication, only the invocation of the marked stored procedure is replicated.

When you mark a stored procedure for replication, Replication Agent creates a shadow-row procedure for that stored procedure.

Replication Agent also modifies the marked stored procedure as follows:

- Inserts a new first step to execute the associated shadow-row procedure
- Inserts a new last step to again execute the shadow-row procedure with different parameters.

If you need to temporarily suspend replication of a marked stored procedure (for example, when database maintenance operations are performed in the primary database) you can disable replication for the stored procedure. See “Enabling and disabling replication for stored procedures” on page 110 for more information.

When you unmark an object that has been marked for replication, the transaction log objects that were created to facilitate the replication for that object are removed from the primary database.

Note Procedure replication is not supported for DB2 UDB.

For more information on the Replication Server function replication feature, see the Replication Server *Administration Guide*.

Marking a stored procedure for replication

When you mark a stored procedure for replication, Replication Agent creates the transaction log objects that capture the stored procedure invocation in the transaction log.

Note For Oracle, DDL replication must be disabled during the marking of stored procedures. See “Disabling replication for DDL” on page 113.

❖ To mark a stored procedure for replication

- 1 Log in to the Replication Agent instance with the administrator login.
- 2 Use the `pdb_setrepproc` command to determine if the stored procedure is already marked in the primary database:

```
    pdb_setrepproc pdb_proc
```

where *pdb_proc* is the name of the stored procedure in the primary database that you want to mark for replication.

If the `pdb_setrepproc` command returns information that the specified stored procedure is marked, you need not continue this procedure.

If the `pdb_setrepproc` command returns information that the specified stored procedure is not marked, continue this procedure to mark the stored procedure for replication.

- 3 Use the `pdb_setrepproc` command to mark the stored procedure for replication.

The `pdb_setrepproc` command allows you to mark the primary stored procedure to be replicated and specify a different stored procedure name to use in the replicate database (as specified in a function replication definition).

- Use the following command to mark the stored procedure for replication using a function replication definition with the same stored procedure name:

```
pdb_setrepproc pdb_proc, mark
```

where *pdb_proc* is the name of the stored procedure in the primary database that you want to mark for replication.

- Use the following command to mark the stored procedure for replication using a function replication definition with a different stored procedure name:

```
pdb_setrepproc pdb_proc, rep_proc, mark
```

where:

- *pdb_proc* is the name of the stored procedure in the primary database that you want to mark for replication.
- *rep_proc* is the name of the stored procedure in the function replication definition for this stored procedure.

If the value of the `pdb_dflt_object_repl` parameter is true, the stored procedure marked for replication with the `pdb_setrepproc` command is ready for replication after you invoke the `pdb_setrepproc` command successfully.

If the value of the `pdb_dflt_object_repl` parameter is true (the default value), you can skip step 4 in this procedure.

If the value of the `pdb_dflt_object_repl` parameter is false, you must enable replication for the stored procedure before replication can take place.

- 4 Use the `pdb_setrepproc` command to enable replication for the marked stored procedure:

```
pdb_setrepproc pdb_proc, enable
```

where *pdb_proc* is the name of the marked stored procedure for which you want to enable replication.

After replication is enabled for the stored procedure, you can begin replicating invocations of that stored procedure in the primary database.

Note If your stored procedure is in Oracle, and you disabled DDL replication during stored procedure marking, remember to re-enable DDL replication. See “Enabling replication for DDL” on page 113.

Unmarking a stored procedure

When you unmark a stored procedure, Replication Agent removes the transaction log objects that were created when the stored procedure was marked. For non-RASD instances, Replication Agent must be in *Admin* state when you are unmarking a stored procedure.

Note For Oracle, DDL replication must be disabled during the unmarking of stored procedures. See “Disabling replication for DDL” on page 113.

❖ To unmark a stored procedure

- 1 Log in to the Replication Agent instance with the administrator login.
- 2 Use the `pdb_setrepproc` command to confirm that the stored procedure is marked in the primary database:

```
pdb_setrepproc pdb_proc
```

where *pdb_proc* is the name of the stored procedure that you want to unmark.

If the `pdb_setrepproc` command returns information that the specified stored procedure is marked, continue this procedure to unmark the stored procedure.

If the `pdb_setrepproc` command does not return information that the specified stored procedure is marked, you need not continue this procedure.

- 3 Use the `pdb_setrepproc` command to disable replication of the stored procedure:

```
pdb_setrepproc pdb_proc, disable
```

where *pdb_proc* is the name of the stored procedure that you want to unmark.

- 4 Use the `pdb_setrepproc` command to remove the replication marking from the stored procedure:

```
pdb_setrepproc pdb_proc, unmark
```

where *pdb_proc* is the name of the stored procedure that you want to unmark.

If you need to force the unmark, you can use the following command:

```
pdb_setrepproc pdb_proc, unmark, force
```

- 5 Use the `pdb_setrepproc` command to confirm that the stored procedure is no longer marked for replication:

```
    pdb_setrepproc pdb_proc
```

where *pdb_proc* is the name of the stored procedure in the primary database that you unmarked.

You can unmark all marked stored procedures in the primary database by invoking the `pdb_setrepproc` command with the `all` keyword.

Note If your stored procedure is in Oracle, and you disabled DDL replication during stored procedure unmarking, remember to re-enable DDL replication. See “Enabling replication for DDL” on page 113.

Enabling and disabling replication for stored procedures

If you need to temporarily suspend replication of a stored procedure, you can use the `pdb_setrepproc` command to disable replication for the marked stored procedure. When you are ready to resume replication of the marked stored procedure, you can use the `pdb_setrepproc` command to enable replication.

Note Procedure replication is not supported for DB2 UDB.

To replicate invocations of a stored procedure in the primary database, the stored procedure must be marked for replication and replication must be enabled for that stored procedure.

Marking a stored procedure for replication is separate from enabling replication for the stored procedure. See “Marking and unmarking stored procedures” on page 105 for more information on marking a stored procedure for replication.

Enabling replication for stored procedures

❖ To enable replication for a marked stored procedure

- 1 Log in to the Replication Agent instance with the administrator login.
- 2 Use the `pdb_setrepproc` command to verify that replication is disabled for the stored procedure:

```
    pdb_setrepproc pdb_proc
```

where *pdb_proc* is the name of the marked stored procedure you want to enable replication for.

If the `pdb_setrepproc` command returns information that the stored procedure is marked and has replication disabled, continue this procedure to enable replication for the stored procedure.

Note A stored procedure must be marked for replication before replication can be enabled or disabled for the stored procedure.

- 3 Use the `pdb_setrepproc` command to enable replication for the stored procedure:

```
pdb_setrepproc pdb_proc, enable
```

where *pdb_proc* is the name of the marked stored procedure for which you want to enable replication.

After replication is enabled for the stored procedure, any invocation of that stored procedure will be replicated.

- 4 You can use the `pdb_setrepproc` command again to verify that replication is now enabled for the stored procedure:

```
pdb_setrepproc pdb_proc
```

where *pdb_proc* is the name of the marked stored procedure for which you want to verify that replication is enabled.

Disabling replication for stored procedures

❖ To disable replication for a marked stored procedure

- 1 Log in to the Replication Agent instance with the administrator login.
- 2 Use the `pdb_setrepproc` command to verify that replication is enabled for the stored procedure:

```
pdb_setrepproc pdb_proc
```

where *pdb_proc* is the name of the marked stored procedure you want to disable replication for.

If the `pdb_setrepproc` command returns information that the stored procedure is marked and has replication enabled, continue this procedure to disable replication for the stored procedure.

Note A stored procedure must be marked for replication before replication can be enabled or disabled for that stored procedure.

- 3 Use the `pdb_setrepproc` command to disable replication for the stored procedure:

```
    pdb_setrepproc pdb_proc, disable
```

where *pdb_proc* is the name of the marked stored procedure for which you want to disable replication.

After replication is disabled for the stored procedure, any invocation of that stored procedure will not be captured for replication until replication is enabled again.

- 4 You can use the `pdb_setrepproc` command again to verify that replication is now disabled for the stored procedure:

```
    pdb_setrepproc pdb_proc
```

where *pdb_proc* is the name of the marked stored procedure for which you want to verify that replication is disabled.

Enabling and disabling replication for DDL

Note DDL replication is available only for Oracle. See the Replication Agent *Primary Database Guide* for more information on the DDL commands that are not replicated.

If you need to temporarily suspend replication of DDL, you can use the `pdb_setrepddl` command to disable replication of DDL. When you are ready to resume replication of DDL, you can use the `pdb_setrepddl` command to enable replication.

When you set the value of `pdb_setrepddl` to enable, all DDL in your primary database is replicated.

Note To replicate DDL, Replication Server must have a database-level replication definition with `replicate DDL` set in the definition. See the Replication Server *Reference Manual* for details on creating a database-level replication definition.

Enabling replication for DDL

❖ To enable replication for DDL in the primary database

- 1 Log in to the Replication Agent administration port.
- 2 Use the `pdb_setrepddl` command without an argument to determine if replication is already enabled for DDL in the primary database:

```
pdb_setrepddl
```

If the `pdb_setrepddl` command returns information that replication is enabled, you need not continue this procedure.

If the `pdb_setrepddl` command returns information that replication is not enabled for DDL, continue this procedure to enable replication for DDL.

- 3 Use the `pdb_setrepddl` command to enable replication for DDL:

```
pdb_setrepddl enable
```

After replication is enabled for the DDL, you can resume replicating your primary database.

See the Sybase Replication Agent *Primary Database Guide* for enabling DDL replication details specific to your primary database.

Disabling replication for DDL

❖ To disable replication for DDL in the primary database

- 1 Log in to the Replication Agent administration port.
- 2 Use the `pdb_setrepddl` command without an argument to determine if replication is already disabled for DDL in the primary database:

```
pdb_setrepddl
```

If the `pdb_setrepddl` command returns information that replication is disabled, you need not continue this procedure.

If the `pdb_setrepddl` command returns information that replication is enabled for DDL, continue this procedure to disable replication for DDL.

- 3 Use the `pdb_setrepddl` command to disable replication for DDL:

```
    pdb_setrepddl disable
```

After replication is disabled for the DDL, you can resume replicating your primary database.

See the Sybase Replication Agent *Primary Database Guide* for enabling DDL replication details specific to your primary database.

Configuring and tuning the Replication Agent

The performance of Sybase Replication Agent can be tuned or optimized by adjusting some of the Replication Agent configuration parameters.

You can set or change a Replication Agent configuration parameter with the `ra_config` command.

Because the Replication Agent overwrites its entire configuration file whenever `ra_config` or `ra_set_login` is invoked, Sybase recommends that you do *not* edit the configuration file. Furthermore, each Replication Agent instance reads its configuration file only at start-up. You must use the `ra_config` command if you want a new configuration parameter value to take effect before the instance is shut down and restarted.

Note Some configuration parameter changes are recorded in the configuration file when you invoke `ra_config`, but they do not take effect until the Replication Agent instance is shut down and restarted.

All Replication Agent configuration parameters can be changed when the Replication Agent instance is in *Admin* state. Some configuration parameters *cannot* be changed when the instance is in *Replicating* state.

Table 3-1 lists all Replication Agent configuration parameters.

Table 3-1: Replication Agent configuration parameters

Parameter name	Description	Default
admin_port	Port number on which Replication Agent listens for incoming user connections.	10000
column_compression	Use minimal column information.	true
compress_ltl_syntax	Use abbreviated LTL syntax.	true
connect_to_rs	Enable/disable connection from LTI to Replication Server.	true
ddl_password	Password for user ID passed in LTL with replicated DDL invocations.	<not_configured>
ddl_username	User ID passed in LTL with replicated DDL invocations.	<not_configured>
dump_batch_timeout	Time to send incomplete buffer to Replication Server.	5
filter_maint_userid	Log Reader filters operations with maintenance user ID.	true
function_password	Password for user ID passed in LTL with replicated stored procedure invocations.	"" (empty string)
function_username	User ID passed in LTL with replicated stored procedure invocations.	sa
log_backup_files	Determines the number of log backup files kept in the log directory.	3
log_directory	Directory where Replication Agent system log file is located.	Current directory on the Replication Agent host machine from which the Replication Agent instance was started.
log_trace_verbose	Switch on/off verbose mode in trace log file.	false
log_wrap	Number of 1KB byte blocks written to log file before wrapping.	1000

Parameter name	Description	Default
lr_nxtext_byte_order	Specifies which byte order to use when replicating NTEXT data: "big (BigEndian), little (LittleEndian)." For Microsoft SQL Server only.	big
lti_batch_mode	Specifies whether to batch the LTL commands sent to Replication Server.	true
lti_max_buffer_size	Maximum number of change sets stored in the LTI input buffer.	5000
lti_update_trunc_point	Number of LTL commands sent before LTI requests new LTM Locator.	1000
ltl_batch_size	Size of the LTL batch buffer.	40000
ltl_character_case	Case of database object names sent to Replication Server.	asis
ltl_origin_time_required	Specifies whether to send origin_time command tag in LTL.	false
ltm_admin_pw	Password for Replication Agent administrative port.	"" (empty string)
ltm_admin_user	User ID for Replication Agent administrative port.	sa
max_ops_per_scan	Maximum number of operations Log Reader will read in a single log scan.	1000
pdb_auto_run_scripts	Automatic execution of SQL scripts used to create/remove transaction log objects and mark/unmark primary database objects.	true
pdb_convert_datetime	Converts native date/time formats to Sybase datetime format.	false
pdb_dflt_column_repl	Enables replication for LOB columns by default when table is marked.	false
pdb_dflt_object_repl	Enables replication by default when object is marked.	true
pdb_exception_handling	Allows primary database to continue operation if Replication Agent triggers fail.	false

Parameter name	Description	Default
pdb_update_stats_count	Indicates the total number of rows that must be truncated before update statistics is run at the primary database. A value of zero indicates that update statistics should not be run at all. The range is 0 to 2,147,483,647. For Informix only.	10,000
pdb_update_stats_type	Indicates the type of update statistics to run at the primary database: low, medium, high. For Informix only.	medium
pdb_xlog_device	Name of the primary database device.	NULL
pdb_xlog_prefix	Character string prefix used to identify transaction log objects.	ra_
pdb_xlog_prefix_chars	Non-alphabetic characters allowed in <code>pdb_xlog_prefix</code> .	For DB2 Universal Database: <code>_#@</code> For Informix: <code>_</code> For Microsoft SQL Server: <code>_\$#@</code> For Oracle: <code>_#\$</code>
pds_connection_type	Type of connection to primary data server.	For DB2 Universal Database: UDBODBC For Informix: IFXJDBC For Microsoft SQL Server: MSMERJDBC For Oracle: ORAJDBC
pds_database_name	Name of database replicated from primary data server.	<not_configured>
pds_datasource_name	Data source name of database replicated from primary data server.	<not_configured>
pds_host_name	Name of primary data server host machine.	<not_configured>

Parameter name	Description	Default
pds_password	Password for user ID Replication Agent uses to access primary data server.	"" (empty string)
pds_port_number	Port number for primary data server.	1111
pds_retry_count	Number of times to retry connection to primary data server.	5
pds_retry_timeout	Number of seconds to wait between connection retry attempts.	10
pds_server_name	Server name of primary data server.	<not_configured>
pds_username	User ID that Replication Agent uses to access primary data server.	<not_configured>
ra_retry_count	Number of times LTM attempts to get back to <i>Replicating</i> state after a failure.	2
ra_retry_timeout	Number of seconds to wait between LTM attempts to get back to <i>Replicating</i> state.	10
rs_charset	Character set used to communicate with Replication Server.	"" (empty string)
rs_host_name	Name of primary Replication Server host machine.	<not_configured>
rs_packet_size	Network I/O packet size sent to Replication Server.	2048
rs_password	Password for user ID Replication Agent uses to access Replication Server.	"" (empty string)
rs_port_number	Port number for primary Replication Server.	1111
rs_retry_count	Number of times to retry connection to primary Replication Server.	5
rs_retry_timeout	Number of seconds to wait between connection retry attempts.	10
rs_source_db	Name of primary database identified to Replication Server.	<not_configured>

Parameter name	Description	Default
rs_source_ds	Name of primary data server identified to Replication Server.	<not_configured>
rs_username	User ID that Replication Agent uses to access primary Replication Server.	<not_configured>
rssd_charset	Character set used to communicate with RSSD.	"" (empty string)
rssd_database_name	Name of RSSD database.	<not_configured>
rssd_host_name	Name of RSSD host machine.	<not_configured>
rssd_password	Password for user ID Replication Agent uses to access RSSD.	"" (empty string)
rssd_port_number	Port number for RSSD.	1111
rssd_username	User ID that Replication Agent uses to access RSSD.	<not_configured>
scan_sleep_increment	Number of seconds to increase Log Reader wait before next scan after finding no operations to replicate.	5
scan_sleep_max	Maximum number of seconds for Log Reader to wait before next scan after finding no operations to replicate.	60
skip_ltl_errors	LTI ignores error messages returned by Replication Server.	false
structured_tokens	LTI uses structured tokens when generating LTL output.	true
truncation_interval	Number of minutes to wait between automatic log truncations.	0
truncation_type	Method(s) of log truncation allowed.	For Oracle: locator_update For all others: interval
use_rssd	Switches on/off access to RSSD for replication definitions.	true

Note Log truncation is not available with log-based Replication Agent for UDB. Therefore, the `truncation_interval` and `truncation_type` parameters in Table 3-1 do not apply to Replication Agent for UDB.

See the Sybase Replication Agent *Reference Manual* for more information about the `ra_config` command and Replication Agent configuration parameters.

Configuring Replication Agent

To set or change a Replication Agent configuration parameter, use the `ra_config` command.

Because the Replication Agent overwrites its entire configuration file whenever `ra_config` or `ra_set_login` is invoked, Sybase recommends that you do *not* edit the configuration file. Furthermore, Replication Agent reads the configuration file only at start up. You must use the `ra_config` command if you want a new configuration parameter value to take effect before the Replication Agent is shut down and restarted.

Note Some configuration parameter changes are recorded in the configuration file when you invoke `ra_config`, but do not take effect until the Replication Agent is shut down and restarted.

Customizing tuning

Adjusting the size and volume of the Replication Agent system logs

Generally, Replication Agent's default configuration values provide optimal performance. However, there may be certain situations where the configuration should be changed to suit or optimize your particular environment.

By default, the system logs produced by the Replication Agent are a pre-set size. They roll over occasionally to prevent continual disk consumption.

You can adjust the size of a log and adjust the number of backup files. By increasing these sizes, you can save log data for a longer period of time. By decreasing them, you can increase the unused space in your environment.

❖ To adjust the size and volume of log files

- 1 Start the Replication Agent instance using the following command at the operating system prompt:

```
ra -i inst_name
```

where *inst_name* is the name of the Replication Agent instance you want to start.

- 2 Log in to the Replication Agent instance using the administrator login.

- 3 Use the `ra_status` command to verify that the Replication Agent instance is in *Admin* state:

```
ra_status
```

- 4 Increase the following values if you want to increase the size and number of backup files. Decrease the following values if you want to make more space available in your environment. Use the `ra_config` command to set the values of the following Replication Agent configuration parameters for the primary database:

```
ra_config log_backup_files, n
```

```
ra_config log_wrap, m
```

Preventing continual spinning at the end of a log scan

Replication Agent uses the configuration parameters `scan_sleep_increment` and `scan_sleep_max` to “pause” scanning when the end of the log is reached. This prevents Replication Agent from continually “spinning” on the end of the log. The downside is that Replication Agent may pause up to 60 seconds (by default) before a new transaction appears, because it was sleeping. When you need the maximum possible latency for a transaction to be less than the 60-second default, the scan parameters can be reduced. This results in additional CPU usage when the end of the log is reached.

Conversely, if CPU maximization is a greater concern than latency, you can increase these parameters to allow Replication Agent to use less CPU on an inactive log, at the cost of having the latency of the ‘next’ transaction increased.

Note These parameters have effect *only* when the end of the log has been reached and there is no additional activity to be replicated. By default, Replication Agent immediately re-scans (without pause) when the end of the log has not been reached.

Troubleshooting Sybase Replication Agent

This chapter describes basic troubleshooting procedures for Sybase Replication Agent and the replication system.

Topic	Page
Diagnosing command errors and replication errors	123
Troubleshooting specific command errors	124
Examining the Replication Agent when replication failure occurs	125
Checking the Replication Server	133

Diagnosing command errors and replication errors

Two types of failures can occur in your replication system: command and replication. Command failures occur when you are in setting up your replication system. They return specific error messages that help you troubleshoot the problem. Replication failures occur after the replication system has been set up and replicated transactions do not appear in the replicate database.

Often, problems that prevent replication from occurring do not result in an error message from any replication system component. For example, a component may not recognize a problem in its own configuration that prevents replication from starting.

In a functioning Sybase Replication Agent system—one that has previously replicated transactions successfully—most system problems result in an error message from one or more of the system's components. However, some problems that interrupt replication might not be interpreted as errors by the system's components. In that case, replication fails, but no error message is returned.

Use the diagnostic and troubleshooting tips in the following sections to identify and correct the cause of a replication system problem:

- Troubleshooting specific command errors
- Examining the Replication Agent when replication failure occurs
- Checking the Replication Server

Troubleshooting specific command errors

This section describes troubleshooting for specific errors you may encounter in a Replication Agent. The following error message can be returned from a command or appear in the log file.

Connection refused

Error message	Could not connect to <jdbc:sybase:Tds:localhost:5001/emb>: JZ006: Caught IOException: java.net.ConnectException: Connection refused: connectJZ006:
Explanation	The Replication Agent attempted to connect to a Sybase server on a host called <i>localhost</i> and port 5001. The error indicates no server was found.
Action	This is usually a configuration error. Either the primary database server that Replication Agent is attempting to connect to is not running, or the host and port information configured in Replication Agent is incorrect. Verify that your primary database server is running. Verify that your Replication Agent is configured with the correct host and port information. See “Setting up the connection configuration parameters” on page 48 for more information. Test your connection after you have verified them. See, “Testing network connectivity” on page 52.

Examining the Replication Agent when replication failure occurs

When no transactions appear to be replicated to the replicate database, and you receive specific error messages, see “Troubleshooting specific command errors” on page 124. When no errors are returned by any replication system components, the first things to check are:

- Verify that primary database objects are marked for replication
- Examining the Replication Agent logs
- Checking the Replication Agent status
- Using the `ra_statistics` command to troubleshoot

Verify that primary database objects are marked for replication

In a Sybase transaction replication system, the Sybase Replication Agent and Replication Server components both provide features that allow you to select the objects that you want to replicate. You need not replicate all objects, or all data-changing operations, in the primary database.

If a primary database object, such as a table or stored procedure, is not replicating, verify the object you intended to replicate is marked.

❖ To verify that a primary database object is marked for replication

- 1 Log in to the Replication Agent instance with the administrator login.
- 2 Use the appropriate command to determine if the object is already marked:

- For a table:

```
pdb_setreptable pdb_table
```

where *pdb_table* is the name of the table that you want to verify is marked for replication.

- For a LOB column:

```
pdb_setrepcol pdb_table, pdb_col
```

where:

- *pdb_table* is the name of the marked table that contains the LOB column.
- *pdb_col* is the name of the LOB column.

- For a stored procedure:

```
pdb_setrepproc pdb_proc
```

where *pdb_proc* is the name of the stored procedure in the primary database that you want to verify is marked for replication.

- For DDL:

```
pdb_setrepddl pdb_ddl
```

where *pdb_ddl* is the name of the DDL in the primary database that you want to verify is marked for replication.

What's next

After you verify the primary database objects are marked, see the following table:

If...	Then...
The primary database object is not marked,	Mark the object: <ul style="list-style-type: none"> • Table – see “Marking a table for replication” on page 96 • LOB column – “Enabling replication for LOB columns” on page 103 • Stored procedure – see “Marking and unmarking stored procedures” on page 105 • DDL – “Enabling and disabling replication for DDL” on page 112
The primary database object is marked,	Check the Replication Agent status. See “Checking the Replication Agent status” on page 126.

Checking the Replication Agent status

The status of the Replication Agent instance indicates whether it is:

- In *Replicating* state, or
- In *Admin* state.

No replication takes place when the Replication Agent instance is in *Admin* state. See “Understanding Replication Agent states” on page 66 for more information.

❖ To check the current Replication Agent status

- 1 Log in to the Replication Agent instance with the administrator login.

- 2 Use the following command to check the current status of the Replication Agent:

```
ra_status
```

This command returns the current state of the Replication Agent instance, as shown in the following example:

```
State Action
-----
ADMIN Waiting for operator command
(1 row affected)
```

See the Sybase Replication Agent *Reference Manual* for more information about the `ra_status` command.

What's next

When the Replication Agent instance is in one of the following states, take the suggested actions.

If...	Then...
The Replication Agent instance is in <i>Replicating (Waiting at end of log)</i> state	Examine the statistics output to check the progress of the replication process. See “Using the <code>ra_statistics</code> command to troubleshoot” on page 130 for more information.
The Replication Agent instance is <i>Replicating</i> <hr/> Note When the Replication Agent instance is in the <i>Replicating</i> state, all data may <i>not</i> have yet been replicated. You can only be sure that the Replication Agent instance is finished replicating when the state is <i>Replicating (Waiting at end of log)</i> .	The instance is operating normally, but it has not reached the end of the transaction log. Wait until Replication Agent instance is in <i>Replicating (Waiting at end of log)</i> state. Then repeat the procedure on page 126.

If...	Then...
If the Replication Agent instance is in <i>Admin</i> state	<p>Start replication and put the Replication Agent instance in <i>Replicating</i> state by executing the Replication Agent resume command. See “Starting replication in the Replication Agent” on page 71 for more information.</p> <p>If the Replication Agent instance returns to <i>Admin</i> state after you invoke the resume command, then there is at least one unresolved problem that prevents the instance from going to <i>Replicating</i> state. See “Examining the Replication Agent logs” on page 128 for more information.</p>

Examining the Replication Agent logs

The Replication Agent system log files contain warning and error messages, as well as information about the Replication Agent connections to the primary database and the primary Replication Server. Look for the most recent command you executed at the bottom of the log file to find the most recent message. The logs are located in the `$$SYBASE/RAX-12_6/inst_name/log` directory.

The following is sample output from an instance's log file:

```

W.      2005/04/26 11:33:23.075 OracleLogScanne
com.sybase.ds.oracle.log.OracleLogScanner  scanForward 23      The change
vector list for log record <00001610.000002d2.0170> is empty.

W.      2005/04/26 11:33:43.313 OracleLogScanne
com.sybase.ds.oracle.log.OracleLogScanner  scanForward 23      The change
vector list for log record <00001610.00000483.011c> is empty.

W.      2005/04/26 11:33:47.879 OracleLogScanne
com.sybase.ds.oracle.log.OracleLogScanner  scanForward 23      The change
vector list for log record <00001610.000004f7.012c> is empty.

T.      2005/04/26 11:35:28.867 OracleLogScanne
com.sybase.ds.oracle.log.OracleLogScanner  scanForward 23      Moving to log
<5649>.

E.      2005/04/26 11:35:30.359 ERROR
com.sybase.ds.oracle.log.record.RecordFactoryparseLogRecord 23
com.sybase.ds.oracle.record.UnknownRecordException: Unkown CVxE_4 inner op
type: <63>.
```

```
E.      2005/04/26 11:35:30.359  ERROR
com.sybase.ds.oracle.log.record.RecordFactoryparseLogRecord 23
java.lang.RuntimeException:
com.sybase.ds.oracle.record.UnknownRecordException: Unkown CVxE_4 inner op
type: <63>.

E.      2005/04/26 11:35:30.359  ERROR
com.sybase.ds.oracle.log.record.RecordFactoryparseLogRecord 23
com.sybase.ds.oracle.log.record.RecordFactory.createChangeVector (RecordFactor
y.java:430)
```

where:

- The first column displays a single character indicating the type of message:
 - I = information
 - W = warning
 - E = error
 - T = trace
 - S = severe
- The second column is a time stamp indicating when the message was written.
- The third column is a description.
- The fourth column identifies the Java class that produced the error.

Note The following two columns appear only when configuration property `log_trace_verbose` is set to `TRUE`.

- The fifth column includes the method.
- The sixth column includes the line number.
- The final column is a text description of the message.

Note In some cases, the information in a specific column will not be consistent with these descriptions. In these cases, other information is generated that is used by technical support to determine where the message was generated from.

Using the *ra_statistics* command to troubleshoot

The *ra_statistics* command returns activity-related statistics that you can use to evaluate Replication Agent operations and performance. By comparing the statistics returned when you first run the command, to the statistics returned after you have successfully replicated something you know will work, you can analyze the differences in the statistics and troubleshoot where the problem lies. The statistics help you determine if the instance is:

- Scanning the transaction log
- Processing replicated transactions
- Sending LTL to the Replication Server

❖ To check Replication Agent operations

- 1 Log in to the Replication Agent instance with the administrator login.
- 2 Verify that you are in *Replicating* state. If you are not, change the state to *Admin*. See “Checking the Replication Agent status” on page 126.
- 3 Use the following command to get current Replication Agent activity statistics:

```
ra_statistics
```

This command returns statistics for all of the Replication Agent components and the Java VM.

- 4 Save the statistics returned to use as a baseline for comparison.
- 5 Perform activity against the object that is not being replicated.
For example, update a table that is not being replicated.
- 6 Repeat step 2.

Note Be sure to allow enough time for the Replication Agent to process the transaction.

- 7 Compare the new returned statistics activity with the baseline.
Check for differences and see the table below.

If...	Then...
The value returned for Total Maintenance User operations filtered increases	<p>You are executing the transaction as the Repserver maintenance user for this Replication Server connection. By default, these transactions are not sent to Replication Server. You must either connect to the primary database as a different user, or you can set the configuration value of filter_maint_userid to false.</p> <p>See “Setting up the connection configuration parameters” on page 48.</p>

See the Sybase Replication Agent *Reference Manual* for more information about the ra_statistics command.

Running out of memory

If you are running out of memory you will get the following error message:

```
java.lang.OutOfMemoryError
```

When you are running out of memory, the Replication Agent drops out of *Replicating* state, or the entire Replication Agent server may stop executing.

To support adjusting the amount of memory available to the JRE, all of the executable scripts (or batch files) in the Replication Agent */bin* directory refer to an environment variable named RA_JAVA_MAX_MEM.

All Replication Agent instance RUN scripts also reference the RA_JAVA_MAX_MEM environment variable.

You can adjust the amount of memory available to the JRE by either:

- Setting the value of a system variable named RA_JAVA_MAX_MEM in your environment and using the ra utility to start the Replication Agent instance, or
- Setting the value of the RA_JAVA_MAX_MEM variable in the Replication Agent instance RUN script and using the RUN script to start the Replication Agent instance.

If you start a Replication Agent instance by invoking the `ra` utility, you can set the value of the `RA_JAVA_MAX_MEM` system variable in your environment to specify the amount of memory available to the JRE. Before it sets the `RA_JAVA_MAX_MEM` variable to a default value, the `ra` and `ra_admin` utilities first check to see if it is already set.

If you start a Replication Agent instance by invoking the instance `RUN` script (or batch file), you can edit the instance `RUN` script to change the default value of `RA_JAVA_MAX_MEM` and specify the amount of memory available to the JRE.

Note When a Replication Agent instance is started with the instance `RUN` script, the value of the `RA_JAVA_MAX_MEM` variable specified in the `RUN` script overrides the value set elsewhere. Therefore, you can edit the `RUN` script to adjust the memory available to the JRE for each instance individually.

Debugging LTL

LTL (Log Transfer Language) is the syntax used to communicate or distribute replication data to Replication Server. It is the principle output from a Replication Agent. For more details about LTL syntax, see the Replication Server *Design Guide*.

❖ To debug LTL

- 1 Start the Replication Agent instance using the following command at the operating system prompt:

```
ra -i inst_name
```

where *inst_name* is the name of the Replication Agent instance you want to start.

- 2 Log in to the Replication Agent instance using the administrator login.
- 3 Use the `ra_status` command to verify that the Replication Agent instance is in *Admin* state:

```
ra_status
```

- 4 Use the `ra_config` command to set the values of the following Replication Agent configuration parameters for the primary database:

```
ra_config LITTRACELTL, true
```

- 5 Use the `resume` command to change the Replication Agent state to *Replicating*:

```
resume
```

- 6 When new replication activity is generated, check the *LTITRACELTL.log* file in the log directory to debug your problem.

Once new activity is generated, check the *LTITRACELTL.log* file in the log directory to debug your problem.

Uncompressing LTL for debugging a problem

By default, the LTL generated by the Replication Agent is compressed to reduce the amount of data sent to Replication Server, reducing network bandwidth. In cases where you require more verbose output to help debug a problem, change the following configuration parameters to produce more verbose LTL.

❖ **To produce more verbose LTL**

- 1 Start the Replication Agent instance using the following command at the operating system prompt:

```
ra -i inst_name
```

where *inst_name* is the name of the Replication Agent instance you want to start.

- 2 Log in to the Replication Agent instance using the administrator login.
- 3 Use the `ra_status` command to verify that the Replication Agent instance is in *Admin* state:

```
ra_status
```

- 4 Use the `ra_config` command to set the values of the following Replication Agent configuration parameters for the primary database:

```
ra_config column_compression = false
```

```
ra_config compress_ltl_syntax = false
```

```
ra_config structured_tokens = false
```

- 5 When new replication activity is generated, check the *LTITRACELTL.log* file in the log directory to debug your problem.

Checking the Replication Server

This section describes how to use Replication Server commands to check for the most common replication problems. See the *Replication Server Troubleshooting Guide* for more detailed information about diagnosing and solving Replication Server problems.

Check replication definitions and subscriptions

Verify that you created replication definitions with the appropriate information.

Verify that you defined and activated subscriptions for all of the replication definitions.

Replication Server status and operation

Replication Server provides several admin commands that you can use to check on its status and operation.

❖ To check the status and operation of the Replication Server

- 1 Log in to the Replication Server with a user login that has “sa” permission.
- 2 Use the following command to check the current status of the Replication Server:

```
admin health
```

This command returns the current status of the Replication Server, as shown in the following example:

Mode	Quiesce	Status
-----	-----	-----
NORMAL	FALSE	HEALTHY

If the Replication Server status is `SUSPECT`, use the `admin who_is_down` command to check for Replication Server threads that may be down or attempting to connect to other servers.

- 3 Use the following command to check the current status of the Replication Server primary database connection (the connection from the Replication Agent to the primary Replication Server):

```
admin show_connections
```

You can also use the `admin who, dsi` command to get more information about the Replication Agent connection in the primary Replication Server.

Note You can use the `admin show_connections` or `admin who, dsi` command output to verify that the primary data server and primary database names are correct for the Replication Agent connection in the primary Replication Server.

See the Replication Server *Reference Manual* for more information about the `admin show_connections` and `admin who` commands.

Replication Agent login in Replication Server

The Replication Server `connect source lti` command accomplishes the following:

- Verifies that the Replication Server database connection used by the Replication Agent exists in the primary Replication Server
- Verifies that the login name specified in the Replication Agent `rs_username` parameter has permission to connect to the primary Replication Server as a data source
- Returns a version string that shows the highest numbered version of LTL that the primary Replication Server supports

❖ To verify that the `rs_username` login has appropriate permissions

- 1 Log in to the primary Replication Server with the Replication Agent user login name specified in the `rs_username` configuration parameter.

Refer to the “Installation and Setup Worksheet” in the Sybase Replication Agent *Installation Guide* for this login name.

- 2 Execute the `connect source lti` command using the following example syntax:

```
connect source lti pds.pdb version
```

where:

- *pds* is the value specified for the Replication Agent `rs_source_ds` configuration parameter.
- *pdb* is the value specified for the Replication Agent `rs_source_db` configuration parameter.
- *version* is the proposed LTL version number.

Refer to the “Installation and Setup Worksheet” in the Sybase Replication Agent *Installation Guide* for the values of the `rs_source_ds` and `rs_source_db` parameters.

Enter 999 for the value of the LTL version number. Replication Server returns the highest numbered version of LTL that it supports.

- 3 Disconnect from the primary Replication Server as `rs_username`, and then log in to the Replication Agent instance with the administrator login and invoke the `resume` command.

See the Replication Server *Design Guide* and *Reference Manual* for more information about the `connect source lti` command.

Replication Server stable queues

Check the Replication Server stable queues to determine which transactions are being processed or ignored, and to determine whether transactions are open (not committed).

❖ To display information about SQM and SQT threads

- 1 Log in to the primary Replication Server and execute the `admin who, sqm` command.
- 2 View the results to determine the number of duplicate messages being detected and ignored, and the number of blocks being written in the Replication Server stable queues.
- 3 In the primary Replication Server, execute the `admin who, sqt` command.
- 4 View the results to find open transactions.

See the Replication Server *Reference Manual* for more information about the `admin who` command.

Materializing Subscriptions to Primary Data

This appendix introduces the concept of bulk materialization and how to use it to set up replication from primary tables in a primary database. It also describes the process of materializing subscriptions to primary tables in a non-Sybase database.

Topic	Page
Understanding materialization	137
Unloading data from a primary database	139
Loading data into replicate databases	139
Atomic bulk materialization	139
Nonatomic bulk materialization	143

Understanding materialization

Materialization is a process of creating and activating subscriptions and copying data from the primary database to the replicate database, thereby initializing the replicate database. Before you can replicate data from a primary database, you must set up and populate each replicate database so that it is in a state consistent with that of the primary database.

There are two types of subscription materialization supported by the Sybase Replication Server:

- *Bulk materialization* – the process of manually creating and activating a subscription and populating a replicate database using data unload and load utilities outside the control of the replication system.

- *Atomic materialization* – the process of creating a subscription and populating a replicate database using Replication Server commands.

Note Sybase Replication Agent does not support atomic materialization.

See the Replication Server *Administration Guide* for more information on subscription materialization methods.

Bulk materialization overview

Sybase recommends that you use bulk materialization to materialize subscriptions to primary data in a non-Sybase database. When you use bulk materialization, you must coordinate and manually perform the following materialization activities:

- Define, activate, and validate the subscription (or create the subscription without materialization)
- Unload the subscription data at the primary site
- Move the unloaded data to the replicate database
- Load data into the replicate tables
- Resume the database connection from the replicate Replication Server to the replicate database so that the replicate database can receive replicated transactions
- Resume replication at the Replication Agent instance

There are two bulk materialization options for subscriptions to primary data in a non-Sybase database:

- Atomic bulk materialization
- Nonatomic bulk materialization

Unloading data from a primary database

Part of the subscription materialization process is unloading the subscription data from the primary table so it can be loaded into the replicate table. Subscription data is the data in the primary table that is requested by the subscription.

Data unloading utilities are usually provided with the primary data server software. You can use one of the OEM-supplied unloading utilities, or a database unload utility of your choice.

Loading data into replicate databases

Part of the subscription materialization process is loading the subscription data from the primary table into the replicate table.

If you are using Sybase Adaptive Server Enterprise as the data server for your replicate database, you can use the Sybase bcp utility to load subscription data into the replicate database.

If you are using a non-Sybase data server as the data server for your replicate database, you can use the load utility of your choice to load subscription data into the replicate database.

See Also

- *Sybase Adaptive Server Enterprise Utility Programs* for more information about using the bcp utility to load subscription data into a replicate database in Sybase Adaptive Server version 11.5 or later.
- *Sybase SQL Server Utility Programs* for more information about using the bcp utility to load subscription data into a replicate database in Sybase SQL Server version 11.0.x.

Atomic bulk materialization

Atomic bulk materialization assumes that all applications updating the primary table can be suspended while a copy of the table is made. This copy will be loaded at the replicate site.

You can use this method to retrieve data from the primary database if you can suspend updates to the primary data.

Preparing for atomic bulk materialization

Before you start an atomic bulk materialization procedure, verify the following:

- The primary table exists and contains data.
- You have a user ID with ownership or select privilege on the primary table (or a column to be replicated in the primary table).
- The replicate table exists and contains the appropriate columns.
- You successfully configured every Replication Server in your replication system.
- You created the replication definition correctly at the primary Replication Server.
- You successfully created the Replication Agent transaction log in the primary database.
- You marked and enabled replication for the primary table in the primary database.
- You started the Replication Agent instance and put it in the *Replicating* state.

Atomic bulk materialization procedure

❖ To perform atomic bulk materialization

- 1 Log in to the replicate Replication Server as the System Administrator (*sa*) using *isql*:

```
isql -Usa -Psa_password -SRRS_servername
```

where:

- *sa* is the System Administrator user ID.
 - *sa_password* is the password for the System Administrator user ID.
 - *RRS_servername* is the name of the replicate Replication Server.
- 2 Define the subscription at the replicate Replication Server using the following syntax:

```
1> define subscription subscription_name  
2> for replication_definition  
3> with replicate at dataserver.database
```

```
4> [where search_conditions]  
5> go
```

The `dataserver.database` name must match the name you used for your replicate database.

- 3 Check the subscription at both the primary and replicate Replication Servers. Use the following command to verify that the subscription status is DEFINED:

```
1> check subscription subscription_name  
2> for replication_definition  
3> with replicate at dataserver.database  
4> go
```

- 4 Lock the primary table to prevent primary transaction activity. This prevents updates to the primary table during materialization.
- 5 Unload the subscription data at the primary site using your site's preferred database unload method to select or dump the data from the primary table.

Note When unloading subscription data from the primary table, make sure you select only the columns specified in the replication definition and the rows specified in the subscription.

- 6 Activate the subscription using the `with suspension` option at the replicate Replication Server by using the following syntax:

```
1> activate subscription subscription_name  
2> for replication_definition  
3> with replicate at dataserver.database  
4> with suspension  
5> go
```

- 7 Wait for the subscription to become active at both the primary and replicate Replication Servers. Execute the `check subscription` command at both the primary and replicate Replication Servers to verify that the subscription status is ACTIVE.

When the subscription status is ACTIVE at the replicate Replication Server, the database connection for the replicate database is suspended.

- 8 Restore the primary table to read-write access (unlock).

- 9 Load the subscription data into the replicate database using the bcp utility or your site's preferred database load utility.

Note Before loading the subscription data into the replicate table, make sure that any data manipulation that will be performed by Sybase Replication Agent (such as datetime conversion) or by Replication Server function strings is applied to the unload file.

- 10 From the replicate Replication Server, resume the database connection for the replicate database:

```
1> resume connection
2> to dataserver.database
3> go
```

- 11 Validate the subscription at the replicate Replication Server:

```
1> validate subscription subscription_name
2> for replication_definition
3> with replicate at dataserver.database
4> go
```

- 12 Wait for the subscription to become valid at both the primary and replicate Replication Servers, then execute the check subscription command at both the primary and replicate Replication Servers to verify that the status is VALID.

When you complete this procedure, the subscription is created, the replicate data is consistent with the primary data, and replication is in progress.

If replication is not in progress when you complete this procedure, see Chapter 4, "Troubleshooting Sybase Replication Agent."

Nonatomic bulk materialization

Nonatomic bulk materialization assumes applications updating the primary table cannot be suspended while a copy of the table is made. Therefore, a nonatomic materialization requires the use of the Replication Server autocorrection feature to get the replicate database synchronized with the primary database.

Note You cannot use nonatomic materialization if the replicate minimal columns feature is set for the replication definition for the primary table.

Preparing for nonatomic bulk materialization

Before you start a nonatomic bulk materialization procedure, verify the following:

- The primary table exists and contains data.
- You have a user ID with ownership or select privilege on the primary table (or a column to be replicated in the primary table).
- The replicate table exists and contains the appropriate columns.
- You successfully configured every Replication Server in your replication system.
- You created the replication definition correctly at the primary Replication Server.
- You successfully created the Replication Agent transaction log in the primary database.
- You marked and enabled replication for the primary table in the primary database.
- You started the Replication Agent instance and put it in the *Replicating* state.

Nonatomic bulk materialization procedure

❖ To perform nonatomic bulk materialization

- 1 Log in to the replicate Replication Server as the System Administrator (sa) using isql:

```
isql -Usa -Psa_password -SRRS_servername
```

where:

- *sa* is the System Administrator user ID.
- *sa_password* is the password for the System Administrator user ID.
- *RRS_servername* is the name of the replicate Replication Server.

- 2 Turn on the autocorrection feature at the replicate Replication Server:

```
1> set autocorrection on
2> for replication_definition
3> with replicate at dataserver.database
4> go
```

- 3 Define the subscription at the replicate Replication Server:

```
1> define subscription subscription_name
2> for replication_definition
3> with replicate at dataserver.database
4> with suspension
5> go
```

The *dataserver.database* name must match the name you used for your replicate database.

- 4 In the primary database, invoke the *rs_marker* stored procedure to activate the subscription.
- 5 Check the subscription at both the primary and replicate Replication Servers. Use the following command to verify that the subscription status is ACTIVE:

```
1> check subscription subscription_name
2> for replication_definition
3> with replicate at dataserver.database
4> go
```

When the subscription status is ACTIVE at the replicate Replication Server, the database connection for the replicate database is suspended.

- 6 Unload the subscription data at the primary site using your site's preferred database unload method to select or dump the data from the primary tables.

Note When unloading subscription data from the primary table, make sure you select only the columns specified in the replication definition and the rows specified in the subscription.

- 7 In the primary database, invoke the `rs_marker` stored procedure to validate the subscription.
- 8 Wait for the subscription to become valid at both the primary and replicate Replication Servers, then execute the `check subscription` command at both the primary and replicate Replication Servers to verify that the status is `VALID`.
- 9 Load the subscription data into the replicate database using the `bcp` utility or your site's preferred database load utility.

Note Before loading the subscription data into the replicate table, make sure that any data manipulation that will be performed by Sybase Replication Agent (such as `datetime` conversion) or by Replication Server function strings is applied to the unload file.

- 10 From the replicate Replication Server, resume the database connection for the replicate database:

```
1> resume connection
2> to dataserver.database
3> go
```

- 11 Wait for the subscription to become valid at both the primary and replicate Replication Servers, then execute the `check subscription` command at both the primary and replicate Replication Servers to verify that the status is `VALID`.

When the subscription's status is `VALID` at the replicate Replication Server, the replicate database is synchronized with the primary database and you can turn off autocorrection.

- 12 Turn off the autocorrection feature at the replicate Replication Server using the following syntax:

```
1> set autocorrection off
2> for replication_definition
3> with replicate at dataserver.database
```

4 > go

When you complete this procedure, the subscription is created, the replicate data is consistent with the primary data, and replication is in progress.

If replication is not in progress when you complete this procedure, see Chapter 4, “Troubleshooting Sybase Replication Agent.”

See also

- *Replication Server Commands Reference* for information on Replication Command Language (RCL) commands.
- *Replication Server Administration Guide* for information on configuring Replication Servers and materialization methods.

Glossary

This glossary describes Replication Server and Sybase Replication Agent terms used in this book.

Adaptive Server

The brand name for Sybase relational database management system (RDBMS) software products.

- *Adaptive Server Enterprise* manages multiple, large relational databases for high-volume online transaction processing (OLTP) systems and client applications.
- *Adaptive Server IQ* manages multiple, large relational databases with special indexing algorithms to support high-speed, high-volume business intelligence, decision support, and reporting client applications.
- *Adaptive Server Anywhere* manages relational databases with a small RDBMS footprint, which is ideal for embedded applications and mobile device applications.

See also **database** and **RDBMS**.

atomic materialization

A materialization method that copies subscription data from a primary database to a replicate database in a single, atomic operation. No changes to primary data are allowed until the subscription data is captured at the primary database. See also **bulk materialization** and **nonatomic materialization**.

BCP utility

A bulk copy transfer utility that provides the ability to load multiple rows of data into a table in a target database. See also **bulk copy**.

bulk copy

An Open Client interface for the high-speed transfer of data between a database table and program variables. It provides an alternative to using SQL insert and select commands to transfer data. See also **BCP utility** and **materialization**.

bulk materialization	A materialization method whereby subscription data in a replicate database is initialized outside of the replication system. You can use bulk materialization for subscriptions to table replication definitions or function replication definitions. See also atomic materialization , materialization , and nonatomic materialization .
client	In client/server systems, the part of the system that sends requests to servers and processes the results of those requests. See also client application .
client application	Software that is responsible for the user interface, including menus, data entry screens, and report formats. See also client .
commit	An instruction to the DBMS to make permanent the changes requested in a transaction. Contrast with rollback . See also DBMS and transaction .
data client	A client application that provides access to data by connecting to a data server. See also client , client application , and data server .
data distribution	A method of locating (or placing) discrete parts of a single set of data in multiple systems or at multiple sites. Data distribution is distinct from data replication, although a data replication system can be used to implement or support data distribution. Contrast with data replication .
data replication	The process of copying data to remote locations, and then keeping the replicated data synchronized with the primary data. Data replication is distinct from data distribution. Replicated data is stored copies of data at one or more remote sites throughout a system, and it is not necessarily distributed data. Contrast with data distribution . See also transaction replication .
data server	A server that provides the functionality necessary to maintain the physical representation of a table in a database. Data servers are usually database servers, but they can be any data repository with the interface and functionality a data client requires. See also client , client application , and data client .
database	A collection of data with a specific structure (or schema) for accepting, storing, and providing data for users. See also data server and relational database .
database connection	A connection that allows Replication Server to manage the database and distribute transactions to the database. Each database in a replication system can have only one database connection defined in Replication Server. See also Replication Server and route .
datatype	A keyword that identifies the characteristics of stored information on a computer. Some common datatypes are: char, int, smallint, date, time, numeric, and float. Different data servers support different datatypes.

DBMS	An abbreviation for <i>database management system</i> . A DBMS is a computer-based system for defining, creating, manipulating, controlling, managing, and using databases. The DBMS can include the user interface for using the database, or it can be a stand-alone data server system. Compare with RDBMS . See also database .
ERSSD	An abbreviation for embedded <i>Replication Server System Database</i> . The ERSSD manages replication system information for a Replication Server. See also Replication Server .
function	A Replication Server object that represents a data server operation, such as insert, delete, or begin transaction. Replication Server distributes operations to replicate databases as functions. See also function string .
function string	A string that Replication Server uses to map a function and its parameters to a data server API. Function strings allow Replication Server to support replication between (homogeneous) non-Sybase data servers, and heterogeneous replication, in which the primary and replicate databases are different types, with different SQL extensions and different command features. See also function .
gateway	Connectivity software that allows two or more computer systems with different network architectures to communicate.
inbound queue	A stable queue managed by Replication Server to spool messages received from a Replication Agent. See also outbound queue and stable queue .
interfaces file	A file containing information that Sybase Open Client and Open Server applications need to establish connections to other Open Client and Open Server applications. See also Open Client and Open Server .
isql	An interactive SQL client application that can connect and communicate with any Sybase Open Server application, including Adaptive Server, Sybase Replication Agent, and Replication Server. See also Open Client and Open Server .
Java	An object-oriented, platform-independent, “write once, run anywhere” programming language developed by Sun Microsystems. The Sybase Replication Agent is a Java application.
Java VM	The Java Virtual Machine. The Java VM (or JVM) is the part of the Java Runtime Environment (JRE) that interprets Java byte codes. See also Java and JRE .

JDBC	An abbreviation for <i>Java Database Connectivity</i> . JDBC is the standard communication protocol for connectivity between Java clients and data servers. See also client , data server , and Java .
jConnect	The Sybase JDBC driver that Replication Agent uses to connect to Replication Server and the RSSD.
JRE	An abbreviation for <i>Java Runtime Environment</i> . The JRE consists of the Java Virtual Machine (Java VM or JVM), the Java Core Classes, and supporting files. To run a Java application, such as the Sybase Replication Agent, a JRE must be installed on the machine. See also Java and Java VM .
LAN	An abbreviation for “local area network.” A local area network is a computer network located on the user’s premises and covering a limited geographical area (usually a single site). Communication within a local area network is not subject to external regulations; however, communication across the LAN boundary can be subject to some form of regulation. Contrast with WAN .
latency	In transaction replication, the time it takes to replicate a transaction from a primary database to a replicate database. Specifically, latency is the time elapsed between committing an original transaction in the primary database and committing the replicated transaction in the replicate database. See also transaction replication .
LOB	An abbreviation for <i>large object</i> . A LOB is a type of data element (or datatype) associated with a column that contains extremely large quantities of data.
Log Reader	An internal component of the Replication Agent that interacts with the primary database to capture transactions for replication. See also Log Transfer Interface and Log Transfer Manager .
Log Transfer Interface	An internal component of the Replication Agent that interacts with Replication Server to forward transactions for distribution to a replicate database. See also Log Reader and Log Transfer Manager .
Log Transfer Interface	An internal component of the Replication Agent that interacts with Replication Server to forward transactions for distribution to a replicate database. See also Log Reader and Log Transfer Manager .
Log Transfer Language	The proprietary protocol used between Replication Agent and Replication Server to replicate data from the primary database to Replication Server. See also Log Reader and Log Transfer Interface .
Maintenance User	A special user login name in the replicate database that Replication Server uses to apply replicated transactions to the database. See also replicate database and Replication Server .

materialization	The process of copying the data from a primary database to a replicate database, initializing the replicate database so that the replication system can begin replicating transactions. See also atomic materialization , bulk materialization , and non-atomic materialization .
nonatomic materialization	A materialization method that copies subscription data without a lock on the primary database. Changes to primary data are allowed during data transfer, which may cause temporary inconsistencies between the primary and replicate databases. Contrast with atomic materialization . See also bulk materialization .
ODBC	An abbreviation for <i>Open Database Connectivity</i> . ODBC is an industry standard communication protocol for clients connecting to data servers. See also client , data server , and JDBC .
Open Client	A Sybase product that provides customer applications, third-party products, and other Sybase products with the interfaces needed to communicate with Open Server applications. See also Open Server .
Open Client application	An application that uses Sybase Open Client libraries to implement Open Client communication protocols. See also Open Client and Open Server .
Open Server	A Sybase product that provides the tools and interfaces required to create a custom server. See also Open Client .
Open Server application	A server application that uses Sybase Open Server libraries to implement Open Server communication protocols. See also Open Client and Open Server .
outbound queue	A stable queue managed by Replication Server to spool messages to a replicate database. See also inbound queue , replicate database , and stable queue .
primary data	The version of a set of data that is the source used for replication. Primary data is stored and managed by the primary database. See also primary database .
primary database	The database that contains the data to be replicated to another database (the replicate database) through a replication system. The primary database is the source of replicated transactions and data in a replication system. Sometimes called the <i>active database</i> . Contrast with replicate database . See also primary data and replicated transaction .
primary key	The column or columns whose data uniquely identify each row in a table.
primary table	A table used as a source for replication. Primary tables are defined in the primary database schema. See also primary data and primary database .

primary transaction	A transaction that is committed in the primary database and recorded in the primary database transaction log. See also primary database and transaction log .
quiesce	To cause a system to go into a state in which further data changes are not allowed. See also quiescent .
quiescent	<p>In a replication system, a state in which all data-changing operations have been propagated to their destinations. Some Replication Server commands require that you quiesce the replication system.</p> <p>In a database, a state in which all data-changing operations are suspended so that transactions cannot change any data.</p> <p>This term is interchangeable with <i>quiesced</i> and <i>in quiesce</i>. See also quiesce.</p>
RASD	An abbreviation for <i>Replication Agent System Database</i> . Information in the RASD is used by the primary database to recognize database structure or schema objects in the transaction log.
RCL	An abbreviation for <i>Replication Command Language</i> . RCL is the command language used to manage Replication Server. See also Replication Server .
RDBMS	An abbreviation for <i>relational database management system</i> . An RDBMS is an application that manages and controls relational databases. Compare with DBMS . See also relational database .
relational database	A collection of data in which data is viewed as being stored in tables, which consist of columns (data items) and rows (units of information). Relational databases can be accessed by SQL requests. Compare with database . See also SQL .
replicate data	The data managed by a replicate database, which is the destination (or target) of a replication system. Contrast with primary data . See also replicate database and replication system .
replicate database	A database that contains data replicated from another database (the primary database) through a replication system. The replicate database is the database that receives replicated transactions and/or data in a replication system. Sometimes called the <i>standby database</i> . Contrast with primary database . See also replicate data , replicated transaction , and replication system .
replicated data	A set of data that is replicated from a primary database to a replicate database by a replication system. See also primary database , replication system , and replicate database .

replicated transaction	A primary transaction that is replicated from a primary database to a replicate database by a transaction replication system. See also primary database , primary transaction , replicate database , and transaction replication .
Replication Agent	An application that reads a primary database transaction log to acquire information about data-changing transactions in the primary database, processes the log information, and then sends it to a Replication Server for distribution to a replicate database. See also primary database , replicate database , and Replication Server .
replication definition	A description of a table or stored procedure in a primary database, for which subscriptions can be created. The replication definition, maintained by Replication Server, includes information about the columns to be replicated and the location of the primary table or stored procedure. See also Replication Server and subscription .
Replication Server	The Sybase software product that provides the infrastructure for a robust transaction replication system. See also Replication Agent .
RSSD	An abbreviation for <i>Replication Server System Database</i> . The RSSD manages replication system information for a Replication Server. See also Replication Server .
replication system	A data processing system that replicates data from one location to another. Data can be replicated between separate systems at a single site, or from one or more local systems to one or more remote systems. See also data replication and transaction replication .
rollback	An instruction to a database to reverse the data changes requested in a unit of work (a transaction). Contrast with commit . See also transaction .
route	A one-way message stream from a primary Replication Server to a replicate Replication Server. Routes carry data-changing commands (including those for RSSDs) and replicated functions (database procedures) between separate Replication Servers. See also Replication Server .
SQL	An abbreviation for <i>Structured Query Language</i> . SQL is a non-procedural programming language used to process data in a relational database. ANSI SQL is an industry standard. See also transaction .

stable queue	A disk device-based, store-and-forward queue managed by Replication Server. Messages written into the stable queue remain there until they can be delivered to the appropriate process or replicate database. Replication Server provides a stable queue for both incoming messages (the inbound queue) and outgoing messages (the outbound queue). See also database connection , Replication Server , and route .
subscription	A request for Replication Server to maintain a replicated copy of a table, or a set of rows from a table, in a replicate database at a specified location. See also replicate database , replication definition , and Replication Server .
table	In a relational database, a two-dimensional array of data, or a named data object that contains a specific number of unordered rows composed of a group of columns that are specific to the table. See also database and relational database .
transaction	A unit of work in a database that can include zero, one, or many operations (including insert, update, and delete operations), and that is either applied or rejected as a whole. Each SQL statement that modifies data can be treated as a separate transaction, if the database is so configured. See also replicated transaction and SQL .
transaction log	Generally, the log of transactions that affect the data managed by a database or a data server. Replication Agent reads the transaction log to identify and acquire the transactions to be replicated from the primary database. See also primary database , Replication Agent , and transaction .
transaction replication	A data replication method that copies data-changing operations from a primary database to a replicate database. See also data replication , primary database , and replicate database .
transactional consistency	A condition in which all transactions in the primary database are applied in the replicate database, and in the same order that they were applied in the primary database. See also primary database , replicate database , and transaction .
WAN	An abbreviation for “wide area network.” A wide area network is a system of local-area networks (LANs) connected together with data communication lines. Contrast with LAN .

Index

A

Adaptive Server
 user logins 46–47
Admin state 66–67
administration port 9, 18, 40–43
 connecting to 40–42
Administrator GUI
 starting 28–29
administrator login 42–43
Administrator utility 28–31
atomic bulk materialization 138, 139–142

B

backing up
 RASD 88
base directory, Replication Agent 12–13
base directory, Sybase Replication Agent 17
bcp (bulk copy) utility 142, 145
bulk materialization 138
 atomic 138, 139–142
 nonatomic 138, 143–146

C

changing
 Replication Agent state 68–69
 schema of primary tables 94
client ports
 interfaces file 40–41
 primary database 49–50
 Replication Agent 18, 40–43
 Replication Server 52
 RSSD 46–47
commands
 pdb_setrepcol 60–61, 103, 104
 pdb_setrepDDL 114

pdb_setreppedl 62
 pdb_setrepproc 59–60, 79, 107–110, 112
 pdb_setreptable 57, 58, 79, 96, 97, 98, 99, 100, 101
 pdb_xlog 55, 56, 76, 78, 80
 quiesce 73
 ra_config 77, 114, 120
 ra_set_login 43
 ra_statistics 69, 130–131
 ra_status 66, 125–127
 resume 62, 71–72
 shutdown 69–71
 suspend 74
 test_connection 52–54
communications
 administration port 40–42
 JDBC driver 6–7
 Replication Agent protocols 6–7
 RSSD parameters 46–47
 setting up connectivity 43–52
 testing connections 52–54
components
 of Replication Agent 7–9
 of replication system 1
configuration files 18–19, 22–25
configuration parameters
 copied from existing instance 24
 pdb_auto_run_scripts 78, 80, 93
 pdb_dflt_object_repl 92, 97, 106, 108
 pdb_xlog_prefix 54, 55, 76, 79, 93
 tuning recommendations 120–121
configuring
 Replication Agent 48–52
connect source permission 45
connections
 configuring 48–52
 rssd_port_number parameter 46–47
copying a Replication Agent instance 30–31
creating
 transaction log 54–56, 76–78

creating a Replication Agent instance 20–25, 30

D

data definition language

See DDL commands

database connections

Replication Agent 49–50

database objects

DDL 61, 113

LOB columns 60

transaction log 92–94

transaction log object names 54–56

transaction log prefix 54

databases

loading data into 139

unloading data from 139

DB2 Universal Database

See also Replication Agent for UDB

connection from Replication Agent 49–50

user logins 44–45

DDL

disabling replication 114

enabling replication for 61–62, 113

DDL commands

effect on RASD 85

replicated 85

deleting

transaction log 78–80

deleting a Replication Agent instance 25–27, 31–32

disabling column replication 104–105

disabling DDL replication 114

disabling stored procedure replication 106, 111–112

disabling table replication 99, 100–101

E

enabling column replication 60–61, 103–104

enabling DDL replication 61–62, 113

enabling stored procedure replication 110–111

enabling table replication 100

environment, *SYBASE* variable 14–15

errors

replication failure 123

starting up Replication agent 17

F

files

configuration 18–19, 22–25

interfaces 40–41

Replication Agent base directory 12–13

Replication Agent scripts directory 78

Sybase Replication Agent base directory 17

system log file 128

G

granting permissions

connect source in Replication Server 45

H

host machines

Replication Agent 12, 17, 40–42

RSSD 46–47

I

IBM DB2 Universal Database

See DB2 Universal Database

immediate shutdown 70

Informix

See also Replication Agent for Informix

versions supported 5

Informix Dynamic Server

connection from Replication Agent 49–50

user logins 44–45

initializing

Replication Agent 84

initializing Replication Agent 85

installing

Replication Agent 11–13

instance, Replication Agent 11–13

changing state 68–69

creating 20–25

- deleting 25–27
- initializing 84, 85
- name 12, 16, 17
- setting up connectivity 43–52
- shutting down 69–71
- starting 32–40
- status 65–68

interfaces file 40–41

J

- Java Runtime Environment (JRE) 9
- JDBC driver 6–7

L

- loading data into databases 139
- LOB columns
 - disabling replication for 104–105
 - enabling replication for 60–61, 103–104
- Log Administrator component 7
- log files
 - Replication Agent system log 128
- Log Reader component 7
- Log Transfer Interface component 7, 8
- Log Transfer Language (LTL) 3
- Log Transfer Manager component 7, 8
- log-based Replication Agent 4
- logs, transaction
 - See* transaction logs
- LTI
 - See* Log Transfer Interface component
- LTM
 - See* Log Transfer Manager component

M

- marking
 - tables 92–97
- marking a primary table 56–58, 92–97
- marking a stored procedure 58–60, 105–108
- materializing subscriptions
 - bulk materialization 138

- Microsoft SQL Server
 - See also* Replication Agent for Microsoft SQL Server
 - connection from Replication Agent 49–50
 - user logins 44–45
 - versions supported 5

N

- names
 - Adaptive Server user logins 46–47
 - host machine 46–47
 - primary database user logins 44–45
 - Replication Agent instance 12, 16, 17
 - Replication Server user logins 45
 - RSSD database name 46–47
 - RSSD user logins 46–47
 - transaction log objects 54–56, 92–93
- nonatomic bulk materialization 138, 143–146

O

- Open Client interfaces file 40–41
- Oracle
 - See also* Replication Agent for Oracle
 - versions supported 5
- Oracle database server
 - connection from Replication Agent 49–50
 - user logins 44–45

P

- passwords
 - RSSD user login 46–47
- pdb_auto_run_scripts** configuration parameter 78, 80, 93
- pdb_dflt_object_repl** configuration parameter 92, 97, 106, 108
- pdb_setrepcol** command 60–61, 103, 104
- pdb_setrepDDL** command 114
- pdb_setrepddl** command 62
- pdb_setrepproc** command 59–60, 79, 107–110, 112

Index

pdb_setreptable command 57, 58, 79, 96, 97, 98, 99, 100, 101
pdb_xlog command 55, 56, 76, 78, 80
pdb_xlog_prefix configuration parameter 54, 55, 76, 79, 93
performance statistics 69
performance tuning 114, 120–121
permissions
 connect source in Replication Server 45
port numbers
 RSSD 46–47
prefix, transaction log 54, 55, 76, 79
primary databases
 connection from Replication Agent 49–50
 testing connections 52–54
 transaction log 74, 80
 user logins 44–45
primary tables 2
 disabling replication 79, 99, 100–101
 enabling replication 100
 marking 56–58, 92–97
 materializing subscription to 137
 schema change 94, 94–95
 subscriptions to 5, 134, 137
 transaction log objects 92–95, 102, 106
 unloading data from 139
 unmarking 79, 94, 98–99

Q

queues
 Replication Server 136
quiesce command 73
quiescing Replication Agent 73

R

ra utility 16
 start-up state 16
 syntax 16
ra_admin utility 17–27
 syntax 17–19
ra_config command 77, 114, 120
ra_set_login command 43

ra_statistics command 69, 130–131
ra_status command 66, 125–127
RASD 84–91
 backing up 88
 DDL commands 85
 forcing update 86–87
 initializing 84
 operations 84–85
 restoring 88–90
 truncating 90–91
 updating 84–87
replicate databases
 loading data into 139
replicate tables
 loading data into 139
replicating DDL commands 85
Replicating state 66–68
Replication Agent
 Admin state 66–67
 administration port 9, 18, 40–43
 administrator login 42–43
 base directory 12–13, 17
 communications 6–7, 40–42, 43–52
 configuration file 18–19, 22–25
 configuring connections 48–52
 copying an instance 30–31
 creating an instance 20–25, 30
 creating transaction log 54–56, 76–78
 DB2 Universal Database, *See* Replication Agent for UDB
 deleting an instance 25–27, 31–32
 host machine 12, 17, 40–42
 Informix, *See* Replication Agent for Informix
 initializing an instance 84, 85
 installing 11–13
 instance name 16, 17
 Log Administrator component 7
 Log Reader component 7
 Log Transfer Interface component 7, 8
 Log Transfer Manager component 7, 8
 log-based design 4
 Microsoft SQL Server, *See* Replication Agent for Microsoft SQL Server
 Oracle, *See* Replication Agent for Oracle
 performance statistics 69
 performance tuning 114–115, 120–121

- primary database user login 44–45
 - quiescing 73
 - RASD 84–91
 - removing transaction log 78–80
 - Replicating* state 66–68
 - Replication Server user login 45
 - RSSD user login 46–47
 - RUN** script 39–40
 - scripts directory 78
 - setting up 54–56
 - shutting down an instance 69–71
 - starting an instance 32–40
 - starting replication 62, 71–72
 - start-up state 16
 - statistics, performance 69
 - stopping replication 72–74
 - suspending 74
 - system data repository 84–87
 - system log file 128
 - testing connections 52–54
 - transaction log 3, 74–83
 - transaction log prefix 54, 55, 76, 79
 - trigger-based design 3
 - troubleshooting 123–136
 - utilities 14–27
 - version 16
 - replication definitions 5
 - troubleshooting 134
 - Replication Server
 - connect source lti** command 135–136
 - connect source** permission 45
 - connection from Replication Agent 50–52
 - materializing subscriptions 137–146
 - primary Replication Server 4, 5
 - replicate Replication Server 4, 5
 - replication definitions 5, 134
 - stable queues 136
 - subscriptions 5, 134
 - testing connections 52–54
 - troubleshooting 133–136
 - user logins 45
 - Replication Server Manager (RSM) 5
 - Replication Server System Database
 - See* RSSD
 - repository
 - system data 84–91
 - restoring
 - RASD 88–90
 - resume** command 62, 71–72
 - RSSD 3, 5
 - connection from Replication Agent 46–47
 - database name 46–47
 - host machine name 46–47
 - port number 46–47
 - Replication Agent user login 46–47
 - user logins 46–47
 - RUN** script
 - Replication Agent 39–40
- ## S
- scripts
 - directory 78
 - transaction log creation 78
 - setting up
 - Replication Agent connections 48–52
 - Replication Agent connectivity 43–52
 - shutdown** command 69–71
 - immediate** option 70
 - shutting down Replication Agent 69–71
 - socket port number
 - RSSD 46–47
 - stable queues 136
 - See also* queues
 - starting
 - replication 62, 71–72
 - Replication Agent 32–40
 - starting the Administrator GUI 28–29
 - states
 - Replication Agent start-up 16
 - states of Replication Agent 65–68
 - Admin* state 66–67
 - changing 68–69
 - Replicating* state 66–68
 - statistics, performance 69
 - stopping
 - replication 72–74
 - stored procedures
 - disabling replication 106, 111–112
 - enabling replication 110–111
 - marking 58–60, 105–108

Index

- unmarking 79, 109–110
- subscription to primary table 134
- subscriptions to primary tables 5
 - atomic materialization 139–142
 - materializing 137–146
 - nonatomic materialization 143–146
- suspend** command 74
- suspending Replication Agent 74
- SYBASE* environment variable 14–15
- syntax
 - ra** utility 16
 - ra_admin** utility 17–19
- system data repository 84–91
 - forcing update 86–87
 - initializing 84–85
 - updating 84–87

T

- tables, primary
 - See* primary tables
- test_connection** command 52–54
- transaction logs 3, 74
 - creating 54–56, 76–78
 - creation script 78
 - object names 54–56
 - prefix 54, 76, 79
 - prefix characters 55
 - primary table objects 92–94, 102, 106
 - removing 78–80
 - Replication Agent 74–83
 - truncating 81–83
- trigger-based Replication Agent 3
- trigger-based Replication Agents
 - transaction log objects 92–95
- troubleshooting 123–136
 - replication definitions 134
 - Replication Server 133–136
 - Replication Server connections 135–136
 - start-up errors 17
 - subscriptions 134
- truncating
 - RASD 90–91
- tuning Replication Agent performance 114–121

U

- unloading data from databases 139
- unmarking
 - tables 94
- unmarking a primary table 79, 98–99
- unmarking a stored procedure 79, 109–110
- updating
 - RASD 84–87
 - system data repository 84–87
- user IDs
 - Adaptive Server user logins 46–47
 - primary database logins 44–45
 - Replication Agent administrator login 42–43
 - Replication Server 135–136
 - Replication Server logins 45
 - RSSD logins 46–47
- utilites
 - bcp** (bulk copy) utility 142, 145
- utilities
 - Administrator GUI 28–31
 - isql** 40–42
 - ra** utility 16
 - ra_admin** utility 17–27
 - Replication Agent 14–27

V

- variable, *SYBASE* environment 14–15
- version
 - Replication Agent 16