SYBASE®

Reference Manual

Replication Agent™

15.2

[Linux, Microsoft Windows, and UNIX]

DOCUMENT ID: DC00268-01-1520-01

LAST REVISED: May 2009

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

Contents

About This Book

Sybase® Replication Agent™ 15.2 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)
- Microsoft SQL Server
- Oracle Database Server

Audience

This book is for anyone who manages or administers a Sybase replication system with primary databases, or administers the primary databases in a Sybase replication system. This may include:

- Database administrators
- Network administrators
- System administrators

How to use this book

Use the *Replication Agent Reference Manual* to look up detailed information about Replication Agent commands and configuration parameters.

This book is organized as follows:

Chapter 1, “Command Reference,” describes all Replication Agent commands, including syntax, options, examples, and detailed command usage notes.

Chapter 2, “Configuration Parameters,” describes the Replication Agent configuration file, and provides a configuration parameter reference.

Related documents

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

- *Replication Agent Administration Guide* – for an overview of the Replication Agent, information about configuring and administering Replication Agent instances, and information about configuring the other components in a Sybase replication system.

- *Replication Agent Primary Database Guide* – for detailed, database-specific information about each non-Sybase database that is supported by the Replication Agent.
- *Replication Agent Installation Guide* – for information about installing the Replication Agent software.
- *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 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 Product Manuals 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 Replication Agent requires a Java Runtime Environment (JRE) on the Replication Agent host machine.

- The 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.

❖ **Finding the latest information on product certifications**

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

❖ **Finding the latest information on component certifications**

1. Point your Web browser to Availability and Certification Reports at http://certification.sybase.com/.
2. Either select the product family and product under Search by Base Product; or select the platform and product under Search by Platform.

3 Select Search to display the availability and certification report for the selection.

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

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

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

2 Click MySybase and create a MySybase profile.

Sybase EBFs and software maintenance

❖ **Finding the latest information on EBFs and software maintenance**

1 Point your Web browser to the Sybase Support Page at http://www.sybase.com/support.

2 Select EBFs/Maintenance. If prompted, enter your MySybase user name and password.

3 Select a product.

4 Specify a time frame and click Go. A list of EBF/Maintenance releases is displayed.

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

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

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_xlog
```

- 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_xlog 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-15_1/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.

Character case conventions

The following character case conventions are used in this book:

- All command syntax and command examples are shown in lowercase. However, 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, to use a mixed-case object name in a Replication Agent command (to match a mixed-case object name in the primary database), delimit the object name with quote characters. For example:

```
pdb_get_tables "TableName"
```

Accessibility features

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

Replication Agent 15.2 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.

The online help for this product is also provided in HTML, which you can navigate using a screen reader.

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 Replication Agent 15.2, 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.

CHAPTER 1

Command Reference

This chapter describes the Replication Agent commands including targets, syntax, options, examples, and command usage.

Command reference table

This section describes the Replication Agent commands including targets, syntax, options, examples, and command usage.In Table 1-1, the targets column indicates when the command is used for “All” targets or specific targets; Oracle, Microsoft SQL Server (MSSQL), or DB2 UDB (UDB).

Table 1-1: Replication Agent commands

Command name	Target	Description	Page
log_system_name	All	Returns the path to the Replication Agent system log file.	7
pdb_asmdiskmap	Oracle	Creates a file with disk mirror paths for all ASM disks that online redo logs and archive logs are stored on.	8
pdb_capabilities	All	Returns a list of the Replication Agent capabilities.	9
pdb_date	All	Returns the current date and time from the primary data server.	9
pdb_execute_sql	All	Executes the specified SQL statement in the current database.	10
pdb_gen_id	All	Returns the current value of the database generation ID; updates the value of the database generation ID.	11
pdb_get_columns	All	Returns a list of all the columns in the specified table.	12
pdb_get_databases	All	Returns a list of all the databases in the primary data server.	14
pdb_get_primary_keys	All	Returns a list of all the columns that make up the primary keys in the specified table.	15
pdb_get_procedure_parms	Oracle MSSQL	Returns a list of the properties for the specified procedure.	16
pdb_get_procedures	Oracle MSSQL	Returns a list of all the procedures in the specified database.	18
pdb_get_sql_database	All	Returns the name of the database specified for SQL statement execution.	20

Command name	Target	Description	Page
pdb_get_tables	All	Returns a list of all the tables in the specified database.	20
pdb_ownerfilter	Oracle MSSQL	Returns a list of owners whose objects will be filtered for initialization; adds and removes owners to the list.	22
pdb_set_sql_database	All	Specifies the database to be used for SQL statement execution.	24
pdb_setrepcol	All	Returns replication marking status; enables or disables replication for all marked columns or a specified column.	25
pdb_setrepddl	Oracle MSSQL	Returns DDL replication status; enables or disables replication for DDL statements.	30
pdb_setrepproc	Oracle MSSQL	Changes and reports stored procedure marking status.	32
pdb_setrepseq	Oracle	Changes and reports sequence replication marking status.	43
pdb_setreptable	All	Changes and reports table replication marking status.	45
pdb_skip_op	Oracle MSSQL	Returns, adds, or removes record identifiers from a list of records to skip in processing.	59
pdb_truncate_xlog	All	Truncates the Replication Agent primary database transaction log.	62
pdb_version	All	Returns the type and version of the primary data server.	64

Command name	Target	Description	Page
pdb_xlog	All	Returns names of transaction log objects; creates transaction log base objects in the primary database; removes transaction log base objects from the primary database.	64
quiesce	All	Stops current Log Reader activity after all data remaining in the transaction log and Replication Agent internal queues is processed and puts Replication Agent in *Admin* state.	69
ra_config	All	Returns help information for configuration parameters; sets the value of a configuration parameter.	70
ra_date	All	Returns the current date and time from the Replication Agent server.	72
ra_helpdeviceoffset	Oracle	Changes the raw disk device read offset for a log device recorded in the RASD.	73
ra_devicepath	Oracle MSSQL	Changes the disk device path for a log device recorded in the RASD.	74
ra_dump	All	Records a dump marker in the primary database transaction log.	74
ra_help	All	Returns help information for Replication Agent commands.	75
ra_helparticle	Oracle MSSQL	Returns information about articles from the RASD.	76
ra_helpdb	Oracle MSSQL	Returns information about the primary database from the RASD.	78

Command name	Target	Description	Page
ra_helpdevice	Oracle MSSQL	Returns information about primary database log devices from the RASD.	78
ra_helpdeviceoffset	Oracle	Returns device offset information about primary database log devices from the RASD log device repository.	80
ra_helpfield	Oracle MSSQL	Returns information about fields (columns in tables, or input parameters in stored procedures) from the RASD.	82
ra_helplocator	Oracle MSSQL	Returns LTM locator field values.	84
ra_helpuser	Oracle MSSQL	Returns information about primary database users from the RASD.	85
ra_locator	All	Returns and changes the current value of the LTM Locator stored by Replication Agent.	86
ra_maintid	All	Returns the maintenance user for the Replication Agent connection.	88
ra_marker	All	Records a marker in the primary database transaction log.	89
ra_migrate	All	Performs migration tasks (as necessary) between releases of Replication Agent.	90
ra_set_login	All	Sets the Replication Agent admin user login and password.	90
ra_statistics	All	Returns statistics for either a specified Replication Agent component or all components, and resets statistics for all components.	91

Command name	Target	Description	Page
ra_status	All	Returns the current Replication Agent state.	100
ra_truncatearticles	Oracle MSSQL	Truncates older versions of primary database articles in the system data repository in the RASD.	101
ra_truncateusers	Oracle MSSQL	Truncates older versions of primary database users in the system data repository in the RASD.	101
ra_updatedevices	Oracle MSSQL	Updates the log device repository in the RASD.	102
ra_version	All	Returns the Replication Agent version.	105
ra_version_all	All	Returns Replication Agent, primary data server, Replication Server, and communications driver versions.	105
rasd_helpbackup	Oracle MSSQL	Backs up the Replication Agent System Database (RASD).	107
rasd_removebackup	Oracle MSSQL	Removes RASD backups.	107
rasd_restore	Oracle MSSQL	Restores the Replication Agent System Database (RASD).	108
rasd_trunc_schedule	Oracle MSSQL	Returns a list of the repository truncation weekly schedule; also adds or removes a specific schedule.	110
resume	All	Starts replication for the current active log and puts Replication Agent in *Replicating* state.	112
rs_create_repdef	Oracle MSSQL	Creates a replication definition at Replication Server for a marked table and procedure, or for all marked tables and procedures.	113

Command name	Target	Description	Page
rs_drop_repdef	Oracle MSSQL	A replication definition for a table is dropped at the Replication Server.	115
rs_ticket	All	Supports Replication Server rs_ticket processing by placing an rs_ticket marker in the Primary database transaction log.	117
shutdown	All	Shuts down Replication Agent.	118
suspend	All	Immediately stops all Log Reader activity, drops connections, and puts Replication Agent in *Admin* state.	119
test_connection	All	Tests Replication Agent connectivity.	120
trace	All	Returns current trace flag settings; changes a specified trace flag.	123

log_system_name

Description — Returns the full path of the Replication Agent instance log file.

Syntax — log_system_name

Usage

- When you create a Replication Agent instance, a log directory is created automatically as part of the instance directory structure. The default value of the log_directory parameter points to that directory.
- The default path of the Replication Agent log directory is:

  `%SYBASE%\RAX-15_2\`*`inst_name`*`\log\`

  where:

  - *%SYBASE%* is the Replication Agent installation directory.
  - *inst_name* is the name of the Replication Agent instance.

- If you specify a valid directory path as the value of the log_directory parameter, the Replication Agent instance places its system log file in the directory you specify.

  If you change the value of the log_directory parameter with the ra_config command, the new value is recorded in the configuration file immediately, but you must shut down and restart the Replication Agent instance to make the new value take effect.

  See the log_directory parameter in Chapter 2, "Configuration Parameters," for more information.

- The log_system_name command is valid when the Replication Agent instance is in either *Admin* or *Replicating* state.

See also ra_config, trace

pdb_asmdiskmap

Description

Note This command is available only for Oracle.

Creates a file with default disk mirror paths for all disks in the Automatic Storage Management (ASM) disk groups that are used to store online redo logs and archive redo logs. The disks required for online logs are automatically selected by querying ASM. The disks required for archive redo logs are specified by the archive log path parameter and is queried by ASM for the disks in the specified group. An entry for each disk in online redo log disk group and archive redo log disk group is added to the ASM disk map file. The default entry is the same path as the system path. The user must change the map path to the mirror device by editing the file.

When pdb_asmdiskmap is executed and the ASM disk map file already exists, new disk paths are obtained from ASM and are added to the file. New disks are only added when disks are added to the ASM disk groups used by redo logs.

Syntax pdb_asmdiskmap

Usage

- The disk map file is named *<instance_name>_ASMDisks.props* and is created in the Replication Agent instance directory.

  For example:

  ```
  <install_root>\<instance>
  \<instance_name>_ASMDisks.props
  ```

If <*install_root*> is *%SYBASE%\sybase\MA-15_2* the map file is created in:

```
%SYBASE%\sybase\MA-15_2\mra_ASMDisks.props
```

- The Replication Agent uses the disk map file, to identify the mirror log devices when log devices are created during xlog initialization and when devices are updated using the ra_updatedevices command. When Replication Agent is in the replicating state, it reads data from the mirrored disks specified in the map file. For more detailed information, see the ra_updatedevices command.

pdb_capabilities

Description — Returns a list of Replication Agent capabilities, which is used by the replication management tools.

Syntax — pdb_capabilities

Usage

- When pdb_capabilities is invoked, it returns a list of the capabilities of the Replication Agent instance.
- The purpose of the pdb_capabilities command is to support the replication management tools.
- The pdb_capabilities command is valid when the Replication Agent instance is in either *Admin* or *Replicating* state.

pdb_date

Description — Returns the current date and time from the primary data server.

Syntax — pdb_date

Usage

- When pdb_date is invoked, it returns the current date and time from the primary data server in the form of a Sybase datetime datatype, as follows:

```
Current PDB Date
 ------------------------------
     Jan 12 2009 12:09:47.310
(1 row affected)
```

- The pdb_date command is valid when the Replication Agent instance is in either *Admin* or *Replicating* state.

See also ra_date

pdb_execute_sql

Description Executes a SQL statement in the current database at the primary data server.

Syntax pdb_execute_sql *statement*

Parameters *statement*
A string in the form of a SQL statement enclosed in double quotes.

Usage

- The Replication Agent instance executes the specified SQL statement against the "current" database.

  The current database is either:

  - The *default* current database, which is the primary database specified in the Replication Agent pds_database_name configuration parameter, or
  - The database specified in the pdb_set_sql_database command (to which the Replication Agent instance is currently connected).

- To set or change the current database, use the pdb_set_sql_database command.
- To find the name of the current database, use the pdb_get_sql_database command.

  Note If the pdb_set_sql_database command has not been invoked to set or change the current database, the pdb_get_sql_database command returns the name of the default current database.

- The SQL statement specified in the pdb_execute_sql command must be a single SQL command enclosed in double quotes. For example:

  ```
  pdb_execute_sql "select * from Authors"
  ```

  The string is passed directly to the database for execution. No command to terminate is required and no syntax or other validation is performed.

- Any results returned from execution of the SQL statement are passed to the Replication Agent administrative client, by way of the Replication Agent administration port.
- The pdb_execute_sql command is valid when the Replication Agent instance is in either *Admin* or *Replicating* state.

See also pdb_get_sql_database, pdb_set_sql_database

pdb_gen_id

Description Returns the current value of the database generation ID, or updates the value of the database generation ID.

Syntax pdb_gen_id [*number*]

Parameters *number*

The value of the new database generation ID to be used when the database generation ID is updated. It must be a number between 0 and 32767.

Examples **Example 1**

```
pdb_gen_id
```

This command returns the current value of the database generation ID.

Example 2

```
pdb_gen_id 10
```

This command updates the database generation ID to the value 10.

Usage

- When pdb_gen_id is invoked with no option, it returns the current value of the database generation ID stored in the Replication Agent transaction log system table (DB2 UDB) or in the RASD (Oracle or Microsoft SQL Server).
- When pdb_gen_id is invoked with the number option, it updates the value of the database generation ID in the Replication Agent transaction log system table (DB2 UDB) or in the RASD (Oracle or Microsoft SQL Server). Changing the database generation ID takes effect immediately.
- The database generation ID is the first 2 bytes of the origin queue ID. The database generation ID is used by Replication Server to support recovery operations, which may require the Replication Agent to re-send transactions.

During recovery, if the Replication Agent must re-send operations that the Replication Server has already processed, you can change the database generation ID to prevent the Replication Server from recognizing the operations as already processed.

- For more information about the origin queue ID, see ra_helplocator on page 84, or refer to the chapter for your specific primary data server in the *Replication Agent Primary Database Guide*.
- If the Replication Agent transaction log (DB2 UDB) or the RASD (Oracle or Microsoft SQL Server) does not exist, the pdb_gen_id command returns an error.
- The pdb_gen_id command with parameters is valid when the Replication Agent instance is in the *Admin* state.

See also ra_locator, pdb_truncate_xlog

pdb_get_columns

Description Returns a list of columns in tables in the current database at the primary data server.

Syntax pdb_get_columns [*ownername*, *tablename*[, *colname*]]

Parameters

ownername
The user name of the owner of the table specified in the *tablename* option. This option can be delimited with quote characters to specify character case.

tablename
The name of the table in the current database for which information is returned. This option can be delimited with quote characters to specify character case.

colname
The name of the column for which information is returned. This option can be delimited with quote characters to specify character case.

Examples **Example 1**

```
pdb_get_columns
```

This command returns a list of all of the columns in all of the user tables in the current database.

Example 2

```
pdb_get_columns bob, authors
```

This command returns a list of all of the columns in the table authors, owned by the user “bob” in the current database.

Example 3

```
pdb_get_columns bob, authors, au_fname
```

This command returns information about the column au_fname in the table authors, owned by the user “bob” in the current database.

Usage

Note *(For Oracle and Microsoft SQL Server*) Results from these commands are taken from the Replication Agent System database (RASD).
(or DB2 UDB) Results from these commands are taken directly from the primary database.

- When pdb_get_columns is invoked with no option, it returns a result set that lists all of the columns in all of the user tables in the current database.
- When pdb_get_columns is invoked with the *ownername* and *tablename* options, it returns a result set that lists all of the columns in the specified table with the specified owner in the current database.
- When pdb_get_columns is invoked with the *ownername*, *tablename*, and *colname* options, it returns a result set with information about the specified column in the specified table with the specified owner in the current database.
- The pdb_get_columns command accepts the % wildcard character in the *ownername*, *tablename*, and *colname* options.
- The current database is either:
  - The *default* current database, which is the primary database specified in the Replication Agent pds_database_name configuration parameter, or
  - The database specified in the pdb_set_sql_database command (to which the Replication Agent instance is currently connected).

- To set or change the current database, use the pdb_set_sql_database command.

  Note If the pdb_set_sql_database command has not been invoked to set or change the current database, the pdb_get_columns command returns information from the current database.

- To find the name of the current database, use the pdb_get_sql_database command.
- The pdb_get_columns command returns 0 rows if the specified table (with the specified owner) does not exist in the current database or if the specified column does not exist in the specified table.
- The pdb_get_columns command is valid when the Replication Agent instance is in either *Admin* or *Replicating* state.

See also

pdb_get_databases, pdb_get_primary_keys, pdb_get_procedure_parms, pdb_get_procedures, pdb_get_tables

pdb_get_databases

Description

Returns a list of all user databases in the primary data server.

Note The Oracle data server does not support multiple user databases. The pdb_get_databases command returns the name of the database instance.

Syntax

pdb_get_databases

Usage

- When pdb_get_databases is invoked, it returns a result set that lists all of the user databases in the primary data server.

  Note Depending on the type of system database, the result set may or may not include the user database in the primary data server. See the chapter for your specific primary data server in the *Replication Agent Primary Database Guide*.

- The pdb_get_databases command is valid when the Replication Agent 15.2 instance is in either *Admin* or *Replicating* state.

See also pdb_get_columns, pdb_get_primary_keys, pdb_get_procedure_parms, pdb_get_procedures, pdb_get_tables

pdb_get_primary_keys

Description Returns a list of primary key columns in a specified table in the current database at the primary data server.

Syntax pdb_get_primary_keys *ownername*, *tablename*

Parameters

ownername
The user name of the owner of the table specified in *tablename*. This option can be delimited with quote characters to specify character case.

tablename
The name of the table in the current database for which primary key column information is returned. This option can be delimited with quote characters to specify character case.

Usage

Note *(For Oracle and Microsoft SQL Server*) Results from these commands are from the Replication Agent System database (RASD).
(For DB2 UDB) Results from these commands are directly from the primary database.

- When pdb_get_primary_keys is invoked, it returns a result set that lists all of the columns that are defined as primary keys in the specified table with the specified owner in the current database.
- The pdb_get_primary_keys command accepts the % wildcard character in the *ownername* option, but not in the *tablename* option.
- The current database is either:
  - The *default* current database, which is the primary database specified in the Replication Agent pds_database_name configuration parameter, or
  - The database specified in the pdb_set_sql_database command to which the Replication Agent instance is currently connected. (This is not valid for Oracle.)

- To set or change the current database, use the pdb_set_sql_database command.

  Note In Oracle, you cannot change the current database.

- To find the name of the current database, use the pdb_get_sql_database command.
- The pdb_get_primary_keys command returns 0 rows if the specified table with the specified owner does not exist in the current database.
- The pdb_get_primary_keys command is valid when the Replication Agent instance is in either *Admin* or *Replicating* state.

See also

pdb_get_columns, pdb_get_databases, pdb_get_procedure_parms, pdb_get_procedures, pdb_get_tables

pdb_get_procedure_parms

Description

Note This command is available only for Oracle and Microsoft SQL Server.

Returns a list of input parameters for procedures in the current database at the primary data server.

Syntax

pdb_get_procedure_parms [*ownername*, *procname* [, *paramname*]]

Parameters

ownername

The user name of the owner of the procedure specified in *procname*. This option can be delimited with quote characters to specify character case.

procname

The name of the procedure in the current database for which information is returned. This option can be delimited with quote characters to specify character case.

paramname

The name of the input parameter for which information is returned. This option can be delimited with quote characters to specify character case.

Examples

Example 1

```
pdb_get_procedure_parms
```

This command returns a list of all of the input parameters for all of the procedures in the current database.

Example 2

```
pdb_get_procedure_parms bob, sp_foo
```

This command returns a list of all of the input parameters for the procedure named sp_foo, owned by the user "bob" in the current database.

Example 3

```
pdb_get_procedure_parms bob, sp_foo, foo_count
```

This command returns information about the input parameter foo_count for the procedure sp_foo, owned by the user "bob" in the current database.

Usage

Note *(For Oracle and Microsoft SQL Server*) Results from these commands are from the Replication Agent System database (RASD).
(For DB2 UDB) Results from these commands are directly from the primary database.

- When pdb_get_procedure_parms is invoked with no option, it returns a result set that lists all of the input parameters for all the procedures in the current database.
- When pdb_get_procedure_parms is invoked with the *ownername* and *procname* options, it returns a result set that lists all of the input parameters for the specified procedure with the specified owner in the current database.
- When pdb_get_procedure_parms is invoked with the *ownername*, *procname*, and *paramname* options, it returns a result set with information about the specified input parameter for the specified procedure with the specified owner in the current database.
- The pdb_get_procedure_parms command accepts the % wildcard character in both the *ownername* and *procname* options.
- The current database is either:
  - The *default* current database, which is the primary database specified in the Replication Agent pds_database_name configuration parameter, or
  - The database specified in the pdb_set_sql_database command to which the Replication Agent instance is currently connected. (This is not valid for Oracle.)

- To set or change the current database, use the pdb_set_sql_database command.

  Note In Oracle, you cannot change the current database.

- To find the name of the current database, use the pdb_get_sql_database command.
- The pdb_get_procedure_parms command returns 0 rows if the specified procedure (with the specified owner) does not exist in the current database.
- The pdb_get_procedure_parms command is valid when the Replication Agent instance is in either *Admin* or *Replicating* state.

See also

pdb_get_columns, pdb_get_databases, pdb_get_primary_keys, pdb_get_procedures, pdb_get_tables

pdb_get_procedures

Description

Note This command is available only for Oracle and Microsoft SQL Server.

Returns a list of procedures in the current database at the primary data server.

Syntax

pdb_get_procedures [*ownername*, *procname*]

Parameters

ownername

The user name of the owner of the procedure specified in *procname*. This option can be delimited with quote characters to specify character case.

procname

The name of the procedure in the current database for which information is returned. This option can be delimited with quote characters to specify character case.

Examples

Example 1

```
pdb_get_procedures
```

This command returns a list of all of the procedures in the current database.

Example 2

```
pdb_get_procedures bob, sp_foo
```

This command returns information about the procedure named sp_foo, owned by the user "bob" in the current database.

Usage

Note (*For Oracle and Microsoft SQL Server*) Results from these commands are from the Replication Agent System database (RASD).
(For DB2 UDB) Results from these commands are directly from the primary database.

- When pdb_get_procedures is invoked with no option, it returns a result set that lists all of the procedures in the current database.
- When pdb_get_procedures is invoked with the *ownername* and *procname* options, it returns a result set with information about the specified procedure with the specified owner in the current database.
- The pdb_get_procedures command accepts the % wildcard character in both the *ownername* and *procname* options.
- The current database is either:
  - The *default* current database, which is the primary database specified in the Replication Agent pds_database_name configuration parameter, or
  - The database specified in the pdb_set_sql_database command to which the Replication Agent instance is currently connected. (This is not valid for Oracle.)
- To set or change the current database, use the pdb_set_sql_database command.

  Note In Oracle, you cannot change the current database.

- To find the name of the current database, use the pdb_get_sql_database command.
- The pdb_get_procedures command returns 0 rows if the specified procedure (with the specified owner) does not exist in the current database.
- The pdb_get_procedures command is valid when the Replication Agent instance is in either *Admin* or *Replicating* state.

See also

pdb_get_columns, pdb_get_databases, pdb_get_primary_keys, pdb_get_procedure_parms, pdb_get_tables

pdb_get_sql_database

Description — Returns the name of the current database, if any.

Syntax — pdb_get_sql_database

Usage

- When pdb_get_sql_database is invoked, it returns the name of the current database.
- If the pdb_set_sql_database command has not been invoked to set the current database, it returns the default current database.
- The current database is either:
  - The default current database, which is the primary database specified in the Replication Agent pds_database_name configuration parameter, or
  - The database specified in the command to which the Replication Agent instance is currently connected. (This is not valid for Oracle.)
- To set or change the current database, use the pdb_set_sql_database command.

  Note In Oracle, you cannot change the current database.

- The pdb_get_sql_database command is valid when the Replication Agent instance is in either *Admin* or *Replicating* state.

See also — pdb_execute_sql, pdb_set_sql_database

pdb_get_tables

Description — Returns a list of user tables in the current database at the primary data server.

Syntax — pdb_get_tables [*ownername*, *tablename*]

Parameters

ownername
The user name of the owner of the table specified in *tablename*. This option can be delimited with quote characters to specify character case.

tablename
The name of the table in the current database for which information is returned. This option can be delimited with quote characters to specify character case.

Examples

Example 1

```
pdb_get_tables
```

This command returns a list of all of the user tables in the current database.

Example 2

```
pdb_get_tables bob, authors
```

This command returns information about the table authors, owned by the user "bob" in the current database.

Usage

Note *(For Oracle and Microsoft SQL Server*) Results from these commands are from the Replication Agent System database (RASD).
(For DB2 UDB) Results from these commands are taken directly from the primary database.

- When pdb_get_tables is invoked with no option, it returns a result set that lists all of the user tables in the current database.

  Note System tables may or may not be returned by some primary data servers when the pdb_get_tables command is invoked.

- When pdb_get_tables is invoked with the *ownername* and *tablename* options, it returns a result set with information about the specified table with the specified owner in the current database.
- The pdb_get_tables command accepts the % wildcard character in the both the *ownername* and *tablename* options.
- The current database is either:
  - The *default* current database, which is the primary database specified in the Replication Agent pds_database_name configuration parameter, or
  - The database specified in the pdb_set_sql_database command to which the Replication Agent instance is currently connected. (This is not valid for Oracle.)

- To set or change the current database, use the pdb_set_sql_database command.

  Note In Oracle, you cannot change the current database.

- To find the name of the current database, use the pdb_get_sql_database command.
- The pdb_get_tables command returns 0 rows if the specified table (with the specified owner) does not exist in the current database.
- The pdb_get_tables command is valid when the Replication Agent instance is in either *Admin* or *Replicating* state.

See also

pdb_get_columns, pdb_get_databases, pdb_get_primary_keys, pdb_get_procedure_parms, pdb_get_procedures

pdb_ownerfilter

Description

Note This command is available only for Oracle and Microsoft SQL Server.

Returns a list of the owners whose objects will be filtered for initialization; adds, removes owners to the list.

Syntax

To return a list of the owners whose objects will be filtered for initialization:

```
pdb_ownerfilter
```

To add or remove an owner whose objects will be filtered for initialization:

```
pdb_ownerfilter {add | remove}, owner
```

Parameters

add

The add keyword filters out any objects that are owned by the owner you specify. Any objects that are owned by this owner cannot be marked for initialization.

remove

The remove keyword removes the filter for the owner you specify. Any objects that are owned by this owner can be marked for initialization.

For Oracle, you cannot remove the “SYS” owner.

owner

The name of the owner that is used for filtering.

The *owner* option can be delimited with quote characters to specify the character case.

If mixed case (uppercase and lowercase) is required, the name must be delimited. This parameter can be delimited with quotes to specify the character case. If mixed case is required, the name must be delimited. For example:

```
"Owner", "oWnEr"
```

Examples

Example 1

```
pdb_ownerfilter
```

This command returns a list of all owners whose objects will be filtered for initialization.

Example 2

```
pdb_ownerfilter add, SYSTEM
```

This command adds the "system" user to the list of owners whose objects will be filtered for replication.

Example 3

```
pdb_ownerfilter remove, SYSTEM
```

This command removes the "system" user from the list of owners whose objects will be filtered for initialization.

Usage

- pdb_ownerfilter can be used to limit the number of objects that are loaded into the Replication Agent System Database during initialization (see ra_init command). When ra_init is processed, the objects and owners in the pdb_ownerfilter list will not be loaded. You can reduce the size of the RASD and reduce the time to perform initialization by adding owners to the list whose objects are not be replicated, or for owners where the majority of objects are not to be replicated.

  Note Any object marked for replication (using commands pdb_setreptable, pdb_setrepproc), is loaded into the RASD, even if the owner is not on the list. This list affects initialization processing, but not replication (replication occurs based on marking status, not owner filtering).

- When pdb_ownerfilter is invoked, its function is determined by the keywords and options you specify.
- When multiple keywords and options are specified, each must be separated by a comma. Blank space before or after a comma is optional. For example:

  ```
  pdb_ownerfilter add, system
  ```

- When pdb_ownerfilter is invoked with no keyword, it returns a list of users whose objects will be filtered.
- The pdb_ownerfilter command is valid only when the Replication Agent instance is in *Admin* state.
- For Oracle, you cannot remove the "SYS" owner.
- For Oracle, after initialization you can replicate any object with pdb_setreptable and pdb_setrepproc, *except* for the following objects which *cannot* be replicated at any time:
  - Objects that are owned by "SYS" owner.
  - Any system table whose name begins with V$.
  - Any system procedure or package whose name begins with DBMS.

See also pdb_setrepproc, pdb_setreptable, ra_config

pdb_set_sql_database

Description Sets the current database to be used for SQL statement execution.

Syntax pdb_set_sql_database *database*

Parameters *database*

The name of the database in the primary data server against which the Replication Agent can execute SQL statements (queries). To specify character case, delimit this parameter with quote characters.

Usage

- When pdb_set_sql_database is invoked, it sets the "current" database, in which the Replication Agent can execute SQL queries.

  Note The pdb_set_sql_database command has no effect in Oracle, but it is included to provide continuity with other Replication Agents that support database servers with multiple databases.

- The Replication Agent does not validate the database name you specify with pdb_set_sql_database.

  If you specify an invalid database name, no error is returned until one of the following Replication Agent commands is invoked:

  - pdb_execute_sql
  - pdb_get_columns
  - pdb_get_primary_keys
  - pdb_get_procedure_parms
  - pdb_get_procedures
  - pdb_get_tables

- To find the name of the current database, use pdb_get_sql_database.

  Note If the pdb_set_sql_database command has not been invoked to set the current database, the pdb_get_sql_database command returns the default current database, which is the primary database specified in the Replication Agent pds_database_name configuration parameter.

- The pdb_set_sql_database command is valid when the Replication Agent instance is in either *Admin* or *Replicating* state.

See also

pdb_execute_sql, pdb_get_sql_database

pdb_setrepcol

Description

Returns LOB column replication status; enables or disables replication for LOB columns within marked tables.

Syntax

- To return replication status of all LOB columns in all tables or all LOB columns in a specific table, or to enable or disable all LOB columns in a table:

  pdb_setrepcol [*tablename* [enable|disable]]

- To return replication status of a specific LOB column in a specific table:

  pdb_setrepcol *tablename, colname*

- To enable or disable all LOB columns in all marked tables:

  pdb_setrepcol all [,{enable|disable [, force]}]

- To enable or disable replication for a specified LOB column:

  pdb_setrepcol *tablename, colname* [,{enable|disable [, force]}]

Parameters

tablename

The name of the user table in the primary database that contains the column specified in the *colname* option.

The *tablename* option can be owner-qualified (include the owner name), with each element separated by a period. For example:

```
owner.table
```

The *tablename* option can be delimited with quote characters to specify the character case.

If mixed case (uppercase and lowercase) is required, the name must be delimited. For example:

```
"Owner".table
"Owner"."Table"
```

Each mixed-case element of the *tablename* option must be delimited separately, as shown in the previous example.

Note If you must use an object name case that does not match the value of the ltl_character_case parameter, the object name must be delimited.

If an object name contains any non-alphanumeric characters, such as spaces or periods, it must be delimited with quote characters. For example:

```
"table name"
owner."table name"
```

If an object name contains a period, it must be *both* owner-qualified and delimited with quote characters. For example:

```
owner."table.name"
```

```
"table.owner"."table.name"
```

colname
The name of a LOB column in the user table specified in the *tablename* option.

The *colname* option can be delimited with quote characters to specify the character case.

If mixed character case (both uppercase and lowercase) is required, the name must be delimited. For example:

```
"Colname"
"COLname"
```

Note If you must use a column name case that does not match the value of the ltl_character_case parameter, the column name must be delimited. See "ltl_character_case" on page 153 for more information.

---

all
A keyword that refers to all LOB columns in marked tables in the primary database. By using the all keyword, you can apply an enable or disable operation to all LOB columns in marked tables.

enable
A keyword that refers to enabling replication for LOB columns.

disable
A keyword that refers to disabling replication for LOB columns.

force
A keyword that refers to forcing replication to be disabled for LOB columns.

When the force keyword follows the disable keyword, the pdb_setrepcol command immediately disables replication for the specified LOB column, without first checking for pending operations in the transaction log. When the force keyword follows the disable keyword and the all keyword, the pdb_setrepcol command immediately disables replication for all marked LOB columns in marked tables in the primary database, regardless of any pending operations in the transaction log.

Examples

Example 1

```
pdb_setrepcol
```

This command returns replication information for all enabled LOB columns in marked tables in the primary database.

Example 2

```
pdb_setrepcol authors
```

This command returns replication information for all LOB columns defined for the table authors in the primary database.

Example 3

```
pdb_setrepcol authors, picture
```

This command returns replication information for the column called “picture” in the table authors in the primary database.

Example 4

```
pdb_setrepcol authors, picture, enable
```

This command enables replication for the column “picture” in the table “authors” in the primary database.

Example 5

```
pdb_setrepcol all, disable
```

This command disables replication for all LOB columns in all marked tables in the primary database.

Usage

- If a column is renamed or dropped and a new column with the original name is created, you must explicitly enable or disable replication from this new column because the new column has no replication status related information from the original column.

  The marking information is maintained internally based on column number, not column name.

- When pdb_setrepcol is invoked, its function is determined by the keywords and options you specify.
- When multiple keywords or options are specified, each must be separated by a comma. Blank space before or after a comma is optional. For example:

  ```
  pdb_setrepcol all, disable
  ```

- When you specify a column name in the pdb_setrepcol command, you must use the name of a valid LOB column.
- You cannot specify the following items as a table name in the pdb_setrepcol command:
  - Primary database system tables

- Aliases or synonyms
- Views
- Replication Agent transaction log objects

- If a column name in the primary database is the same as a keyword, it can be identified by adding the string col= to the beginning of the column name. For example:

```
pdb_setrepcol tablename, col=enable, disable
```

- If you enable LOB column replication with the pdb_setrepcol command do not configure the Replication Agent to convert date or time datatypes in the primary database. See "pdb_convert_datetime" on page 163 for more information.
- When pdb_setrepcol is invoked with either no option or a single option, it returns information about the enabled status of LOB columns in the primary database.
  - If pdb_setrepcol is invoked with no option, it returns a list of all LOB columns for which replication is enabled in the primary database.

    Note Invoking the pdb_setrepcol command with no option produces the same result as invoking the pdb_setrepcol command with the enable keyword.

  - If pdb_setrepcol is invoked with a table name, it returns information about the enabled status of all the LOB columns in the specified primary table.
  - If pdb_setrepcol is invoked with the enable keyword, it returns a list of all LOB columns for which replication is enabled in the primary database.
  - If pdb_setrepcol is invoked with the disable keyword, it returns a list of all LOB columns for which replication is disabled in the primary database.

  For LOB columns listed as disabled, transactions are not captured for replication.
- When pdb_setrepcol is invoked with a valid primary table name and valid LOB column name, with no keywords, it returns information about the enabled status of the specified LOB column in the specified table in the primary database.

- When pdb_setrepcol is invoked with the all keyword, the operation specified by the following keyword (enable or disable) is applied to all LOB columns in marked tables in the primary database.
  - If pdb_setrepcol is invoked with the all keyword and the enable keyword, it enables replication for all LOB columns in marked tables in the primary database.
  - If pdb_setrepcol is invoked with the all keyword and the disable keyword, it disables replication for all LOB columns in marked tables in the primary database.
- When pdb_setrepcol is invoked with a valid primary table name and valid LOB column name followed by one or more keywords, the operation specified by the keyword (enable or disable) is applied to the specified LOB column in the specified primary table.
  - If pdb_setrepcol is invoked with a table name and LOB column name and the enable keyword, it enables replication for the specified LOB column in the primary database.
  - If pdb_setrepcol is invoked with a table name and LOB column name and the disable keyword, it disables replication for the specified LOB column in the primary database.

  If the table name and LOB column name combination you specify does not exist in the primary database, the pdb_setrepcol command returns an error.
- If the Replication Agent transaction log does not exist in the primary database (DB2 UDB) or the RASD is not initialized (Oracle or Microsoft SQL Server), the pdb_setrepcol command returns an error.
- 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 Replication Agent for UDB but not for Replication Agent for Oracle or Microsoft SQL Server.

See also pdb_setrepproc, pdb_setreptable, ra_config

pdb_setrepddl

Description **Note** This command is available only for Oracle and Microsoft SQL Server.

Returns DDL replication status; enables or disables replication for DDL statements.

Syntax

To return replication status of DDL:

```
pdb_setrepddl
```

To enable or disable DDL replication:

```
pdb_setrepddl {enable|disable}
```

The default setting is disable.

Examples

Example 1

```
pdb_setrepddl
```

This command returns the current DDL replication status for the primary database.

Example 2

```
pdb_setrepddl enable
```

This command enables replication of DDL commands issued into the primary database after this point in time.

Example 3

```
pdb_setrepddl disable
```

This command disables replication of DDL commands issued into the primary database after this point in time.

Usage

- This is the flag that turns DDL replication on or off. By default, it is set to off (disable).
- In addition to enabling DDL replication using pdb_setrepddl command, you must set the Replication Agent ddl_username and ddl_password parameters.
- If the Replication Agent has not been initialized, the pdb_setrepddl command returns an error.
- The pdb_setrepddl command with enable/disable option is valid only when the instance is in *Admin* state.

- The pdb_setrepddl command with no option is valid when the instance is in *Admin* or *Replicating* state.

Note Some DDL commands are filtered even when DDL replication is enabled. See the *Replication Agent Primary Database Guide* for more information.

See also

See configuration parameters: ddl_password, ddl_username

pdb_setrepproc

Description

Note This command is available only for Oracle and Microsoft SQL Server.

Returns stored procedure replication marking status; marks specified procedures for replication; unmarks all marked procedures or a specified procedure; enables or disables replication for all marked procedures or a specified procedure.

Syntax

To return stored procedure replication marking status:

pdb_setrepproc [*procname*|mark|unmark|enable|disable]

To unmark, enable, or disable all marked stored procedures:

pdb_setrepproc all, {unmark[, force]|enable|disable}

To mark a specified stored procedure:

pdb_setrepproc *procname*, [repname,] mark

To unmark, enable, or disable a specified stored procedure:

pdb_setrepproc *procname*, {unmark[, force]|enable|disable}

Note For Microsoft SQL Server, the pdb_dflt_object_repl configuration parameter has no impact on pdb_setrepproc command.

Parameters

procname

The name of a user stored procedure in the primary database.

The *procname* option can be delimited with quote characters to specify the character case.

If mixed character case (both uppercase and lowercase) is required, the name must be delimited. For example:

```
"Proc"
```

Note If you must use an object name case that does not match the setting of the ltl_character_case parameter, the object name must be delimited.

If an object name contains any non-alphanumeric characters, such as spaces, periods, and so forth, it must be delimited with quote characters. For example:

```
"proc name"
"proc.name"
```

If an object name contains a period, it must be *both* owner-qualified and delimited with quote characters. For example:

```
owner."proc.name"
"proc.owner"."proc.name"
```

repname

The name of the stored procedure specified in a function replication definition for the primary stored procedure.

The *repname* option can be delimited with quote characters to specify character case. See the previous description of the *procname* option for details.

By specifying a replicated name, stored procedure invocations can be replicated to a stored procedure invocation in the replicate database that has a different stored procedure name from the primary database.

Note The replicated name you specify with the pdb_setrepproc command must match the name specified by a “with primary function named” clause in a Replication Server function replication definition for the primary database connection. The Replication Agent cannot validate the function replication definition, but if it does not exist, function replication from the primary database will fail.

---

all
A keyword that refers to all user stored procedures in the primary database. By using the all keyword, you can mark all user stored procedures, or apply an unmark, enable, or disable operation to all *marked* stored procedures.

mark

Note You must specify an owner when using the mark keyword.

A keyword that refers to marking user stored procedures for replication.

unmark
A keyword that refers to unmarking marked stored procedures.

force
A keyword that refers to the unmark operation. When the force keyword follows the unmark keyword, the pdb_setrepproc command immediately unmarks the specified stored procedure in the primary database, without first checking the enable status of the stored procedure or checking for pending operations in the transaction log. When the force keyword follows the unmark keyword and the all keyword, the pdb_setrepproc command immediately removes replication marking from all marked stored procedures in the primary database, regardless of their enable status or pending operations in the transaction log.

The force keyword also forces complete execution of the unmarking script, even if errors occur during the unmarking process. Normally, when errors occur during script execution, the script terminates immediately without completing.

Note Sybase recommends that you read the "Usage" section that follows to better understand how scripts are used in Oracle procedure marking and unmarking.

The force keyword can be useful when a previous script execution failed and left the unmarking operation incomplete. When errors occur during a forced script execution, the pdb_setrepproc command returns the following message:

```
Errors were encountered and ignored during FORCED script
execution. See error log for details.
```

enable
A keyword that refers to enabling replication for marked stored procedures.

disable
A keyword that refers to disabling replication for marked stored procedures.

Examples

Example 1

```
pdb_setrepproc
```

This command returns replication marking information for all marked stored procedures in the primary database.

Example 2

```
pdb_setrepproc authors
```

This command returns replication marking information for the user stored procedure named “authors” in the primary database.

Example 3

```
pdb_setrepproc authors, mark
```

This command marks the user stored procedure named “authors” in the primary database.

Example 4

```
pdb_setrepproc authors, enable
```

This command enables replication for the marked stored procedure named “authors” in the primary database.

Example 5

```
pdb_setrepproc all, unmark
```

This command unmarks all marked stored procedures in the primary database.

Usage

- If a marked procedure is renamed or dropped and a new procedure with the original name is created, you must explicitly mark the new procedure because the new procedure has no marking-related information from the original procedure. The marking information is maintained internally by object id, not object name.

- How you use the pdb_setrepproc command depends on the type of replication definition that you have created at Replication Server. If you have created a database replication definition with no function replication definition, then the replicate procedure in the pdb_setrepproc command refers to the procedure in the replicate database. However, if you have created a function replication definition, then the replicate procedure in the pdb_setrepproc command refers to the name of the function replication definition, and it is the function replication definition that must map to the procedure in the replicate database.

  If *no* function replication definition exists and will not be added prior to replication, but only a database replication definition exists, use the following commands to mark a procedure for replication:

  - When the procedure in the replicate database has the same name as the procedure in the primary database:

    ```
    pdb_setrepproc pdb_proc, mark
    ```

    where:

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

  - When the procedure in the replicate database has the different name name than the procedure in the primary database:

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

    where:

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

    rep_proc is the name of the procedure in the replicate database.

  If a function replication definition exists or will be added prior to replication, regardless of whether or not a database replication definition exists, use the following commands to mark a procedure for replication:

  - When the function replication definition has the same name as the procedure in the primary database:

    ```
    pdb_setrepproc pdb_proc, mark
    ```

    where:

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

- If the procedure in the replicate database also has the *same* name as the function replication definition, then there is no need to use the "deliver as" clause in the replication definition in the primary Replication Server. For example:

  ```
  create function replication definition
  pdb_proc with primary at data_server.
  database …
  ```

- If the procedure in the replicate database has a *different* name than the name of function replication definition, then the function replication definition must map to the procedure in the replicate database. For example:

  ```
  create function replication definition
  pdb_proc with primary at data_server.database
  deliver as 'rep_proc' …
  ```

- When the name of the function replication definition is different than the procedure in the primary database:

  ```
  pdb_setrepproc pdb_proc , rdpri_proc, mark
  ```

  where:

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

  rdpri_proc is the name of the function replication definition.

  - If the procedure in the replicate database also has the *same* name as the function replication definition, then there is no need to use the "deliver as" clause in the replication definition in the primary Replication Server. For example:

    ```
    create function replication definition
    rdpri_proc with primary at
    data_server.database …
    ```

  - If the procedure in the replicate database has a *different* name from the function replication definition, then the function replication definition must map to the procedure in the replicate database. For example:

    ```
    create function replication definition
    rdpri_proc with primary at
    data_server.database deliver as 'rep_proc' …
    ```

- When multiple keywords and options are specified, each must be separated by a comma. Blank space before or after a comma is optional. For example:

  ```
  pdb_setrepproc all, unmark, force
  ```

- When you specify a stored procedure name in the pdb_setrepproc command, you must use the name of a valid user stored procedure.
- You cannot specify the following items as a stored procedure name in the pdb_setrepproc command:
  - System procedures
  - Replication Agent transaction log procedures
- If a stored procedure name in the primary database is the same as a keyword, it can be identified by adding the string `proc=` to the beginning of the stored procedure name. For example:

  ```
  pdb_setrepproc proc=unmark, mark
  ```

- When you use the unmark keyword to remove replication marking from a stored procedure, the Replication Agent verifies that replication is disabled for that stored procedure and there are no pending (unprocessed) operations for that stored procedure in the transaction log. If replication is not disabled for that procedure, or if there is a pending operation for that procedure in the transaction log, pdb_setrepproc returns an error.
- When pdb_setrepproc is invoked with either no option or a single option, it returns marking information about the stored procedures in the primary database.
  - If pdb_setrepproc is invoked with no option, it returns a list of all marked procedures in the primary database.

    Note Invoking the pdb_setrepproc command with no option produces the same result as invoking the pdb_setrepproc command with only the mark keyword.

  - If pdb_setrepproc is invoked with a procedure name, it returns complete marking information about the specified procedure.
  - If pdb_setrepproc is invoked with the mark keyword, it returns a list of all marked procedures in the primary database.
  - If pdb_setrepproc is invoked with the unmark keyword, it returns a list of all unmarked procedures in the primary database.

- If pdb_setrepproc is invoked with the enable keyword, it returns a list of all marked procedures in the primary database, for which replication is currently enabled.
- If pdb_setrepproc is invoked with the disable keyword, it returns a list of all marked procedures in the primary database, for which replication is currently disabled.

Stored procedures marked for replication are recorded in the RASD. All other user procedures are considered unmarked.

Note The Replication Agent system procedures are not included in the list of unmarked procedures. Also not included are any synonyms or aliases for these procedures.

For procedures listed as unmarked or disabled, their invocations will not be captured for replication.

- When pdb_setrepproc is invoked with the all keyword and an action keyword (unmark, enable, or disable), the action specified is applied to either all user stored procedures in the primary database, or to all marked procedures in the primary database.
  - If pdb_setrepproc is invoked with the all and unmark keywords, it removes replication marking from all marked procedures in the primary database.

    You can specify the force keyword after the unmark keyword to force immediate unmarking of all marked procedures, including procedures for which replication is still enabled or pending operations remain in the transaction log.
  - If pdb_setrepproc is invoked with the all and enable keywords, it enables replication for all marked procedures in the primary database.
  - If pdb_setrepproc is invoked with the all and disable keywords, it disables replication for all marked procedures in the primary database.

- **Note** For Microsoft SQL Server, Sybase recommends the use of [mark | unmark] instead of [enable | disable] since the results are the same.

When pdb_setrepproc is invoked with a valid user stored procedure name and followed by an action keyword (mark, unmark, enable, or disable), the action specified is applied to the specified procedure.

  - If pdb_setrepproc is invoked with a procedure name and the mark keyword, it marks the specified procedure in the primary database for replication.
  - If pdb_setrepproc is invoked with a procedure name and the unmark keyword, it removes replication marking from the specified procedure in the primary database.
  - If pdb_setrepproc is invoked with a procedure name and the enable keyword, it enables replication for the specified marked procedure in the primary database.
  - If pdb_setrepproc is invoked with a procedure name and the disable keyword, it disables replication for the specified marked procedure in the primary database.

- If you specify a stored procedure name that does not exist in the primary database, the pdb_setrepproc command returns an error.
- When pdb_setrepproc is invoked with a procedure name and a replicated name, followed by the mark keyword, the primary procedure is marked for replication with the specified replicated name.

  If the primary procedure name you specify does not exist in the primary database, the pdb_setrepproc command returns an error.

  By specifying a replicated name, procedure invocations can be replicated to a procedure in the replicate database that has a different name from the primary procedure.

  Note The replicated name you specify with the pdb_setrepproc command must match the name of a Replication Server function replication definition for the primary database connection. The Replication Agent cannot validate the function replication definition, but if it does not exist, function replication from the primary database will fail.

- If RASD is not initialized (Oracle and Microsoft SQL Server), the pdb_setrepproc command returns an error.

For Oracle:

To support stored procedure replication in Oracle, a stored procedure that is marked for replication must be modified. The modification is required to record the stored procedures execution in the Oracle transaction log. As a result of the modifications, the following behavior should be considered when marking and unmarking stored procedures in Oracle:

- You must disable DDL replication before marking or unmarking a procedure, and re-enable it after marking or unmarking to prevent modifications from replicating to standby.
- Marking and unmarking a stored procedure for replication requires that the Replication Agent drop, and then re-create the procedure. However, Replication Agent sets all the same privileges on the re-created procedure as those defined on the original procedure.

Note Do not remove or alter the Replication Agent comments in a marked stored procedure.

- When pdb_setrepproc is invoked to mark a procedure for replication, Replication Agent does the following:
  - Modifies the user procedure to add code that captures input parameter values and generates Replication Agent transaction log records.
  - Generates a SQL script that creates the procedures required for the Replication Agent transaction log in the primary database.
  - Saves the generated script in a file called *partmark.sql* in the *RAX-15_2\inst_name\scripts\procname* directory, where *inst_name* is the name of the Replication Agent instance, and *procname* is the name of the stored procedure being marked. This script can *not* be manually executed—it is for informational purposes only.

    Note If the value of the pdb_auto_run_scripts configuration parameter is false, the *partmark.sql* script will be saved but not executed automatically. You cannot manually run the script. To complete marking the procedure, you must first set pdb_auto_run_scripts to *true*, then re-run the pdb_setrepproc command.

  - Executes the script to mark the stored procedure and create the transaction log objects in the primary database (if the value of the pdb_auto_run_scripts configuration parameter is true).
  - After the script completes successfully, moves the *partmark.sql* file to the *RAX-15_2\inst_name\scripts\procname\installed* directory.
  - If the mark script fails, it is stored in a file (*partmark.sql*) in the *RAX-15_2\inst_name\scripts\procname* directory, the stored procedure is not marked, and transaction log objects are not created. You can examine the script by viewing the *mark.sql* file.

- When pdb_setrepproc is invoked to unmark a marked stored procedure, Replication Agent does the following:
  - Modifies the user procedure to remove Replication Agent code that captures input parameter values and generates transaction log records.
  - Generates a SQL script that removes the tables and procedures required for the transaction log in the primary database.
  - Saves the generated script in a file called *partunmark.sql* in the *RAX-15_2\inst_name\scripts\procname* directory, where *inst_name* is the name of the Replication Agent instance and *procname* is the name of the stored procedure being unmarked. For Oracle, this script named *partunmark.sql* because it can *not* be manually executed—it is for informational purposes only.

    Note If the value of the pdb_auto_run_scripts configuration parameter is false, the *partunmark.sql* script will be saved but not executed automatically. You cannot manually run the script. To complete unmarking the procedure, you must first set pdb_auto_run_scripts to *true*, then re-run the pdb_setrepproc command.

  - Executes the script to unmark the stored procedure and remove the transaction log objects in the primary database (if the value of the pdb_auto_run_scripts configuration parameter is true).
  - After the script completes successfully, moves the *partunmark.sql* file to the *RAX-15_2\inst_name\scripts\procname\installed* directory.
  - If the unmark script fails, it is stored in a file (*partunmark.sql*) in the RAX-15_2\inst_name\procname\scripts directory and the stored procedure is not unmarked and the transaction log objects are not removed. You can examine the script by viewing the partunmark.sql file.

    When the unmark script execution encounters a fatal error on any database object, the pdb_setrepproc command returns the following message:

    ```
    Could not unmark the following objects: ...
    See error log for details.
    ```

See also pdb_setrepcol, pdb_setreptable, ra_config

pdb_setrepseq

Description

Note This command is available only for Oracle.

Returns the sequence replication marking status; marks specified sequence for replication; unmarks all marked sequences or a specified sequence; enables or disables replication for all marked sequences or a specified sequence.

Syntax

To return sequence replication marking status:

```
pdb_setrepseq [sequence_name|mark|unmark|enable|disable]
```

To unmark, enable, or disable all marked sequences:

```
pdb_setrepseq all, {unmark[, force] |enable|disable}
```

To mark, unmark, enable, or disable a specified sequence:

```
pdb_setrepseq sequence_name, {mark|unmark[, force] |enable|disable}
```

To mark a specified sequence for replication with a replicated name:

```
pdb_setrepseq sequence_name, repname, mark
```

Parameters

sequence_name

The name of a user sequence in the primary database. The *sequence_name* option can be delimited with quote characters to specify the character case. If mixed character case (both uppercase and lowercase) is required, the name must be delimited. For example:

```
"Sequence"
```

The *sequence_name* parameter can be owner-qualified to include the primary sequence owner name, with each element separated by a period. For example:

```
owner.sequence
```

Note If you must use an object name case that does not match the setting of the ltl_character_case parameter, the object name must be delimited. If an object name contains any non-alphanumeric characters, such as spaces and periods, it must be delimited with quote characters. For example, “sequence name” or owner.”sequence name.”

repname
The replicated name of the sequence to be updated at the replicate site, if desired to be different than the sequence name at the primary site. The *repname* option can be delimited with quote characters to specify character case. See the previous description of the *sequence_name* parameter for details. By specifying a replicated name, sequence updates can be replicated to a sequence in the replicate database that has a different sequence name from the primary database.

The *repname* option can be owner-qualified to include the replicate sequence owner name, with each element separated by a period. For example:

```
repowner.repname
```

all
A keyword that refers to all user sequences in the primary database. By using the *all* keyword, you can unmark all user sequences, or apply an enable or disable operation to all marked sequences.

mark
A keyword that refers to marking user sequences for replication.

unmark
A keyword that refers to unmarking user sequences for replication.

force
A keyword that refers to the unmark operation. When the force keyword follows the unmark keyword, the pdb_setrepseq command immediately unmarks the specified sequence in the primary database, without first checking the enable status of the sequence. When the force keyword follows the unmark keyword and the all keyword, the pdb_setrepseq command immediately removes replication marking from all marked sequences in the primary database, regardless of their enable status

enable
A keyword that refers to enabling replication for marked sequences.

disable
A keyword that refers to disabling replication for marked sequences.

Usage

- When pdb_setrepseq is invoked, its function is determined by the keywords and options you specify.
- When multiple keywords and options are specified, each must be separated by a comma. Blank space before or after a comma is optional. For example:

```
pdb_setrepseq all, unmark, force
```

- When you specify a sequence in the pdb_setrepseq command, you must use the name of a valid user sequence.

pdb_setreptable

Description

Returns replication marking status; marks all user tables or a specified table for replication; unmarks all marked tables or a specified table; enables or disables replication for all marked tables or a specified table.

Syntax

To return replication marking status:

pdb_setreptable [*tablename*|mark|unmark|enable|disable]

To mark all user tables (available only for Oracle and Microsoft SQL Server):

pdb_setreptable all, mark

To unmark, enable, or disable all marked tables:

pdb_setreptable all, {unmark[, force]|enable|disable}

To mark, unmark, enable, or disable a specified table:

pdb_setreptable *tablename*, {mark[, owner][, force] |
unmark[, force] |enable|disable}

Note Marking or unmarking all tables at once in the primary database using pdb_setreptable all, mark or pdb_setreptable all, unmark is not supported in Replication Agent for UDB. You must mark or unmark each table individually.

To mark a specified table for replication with a replicated name:

pdb_setreptable *tablename*, *repname*, mark[, owner][, force]

To mark a specified table for replication immediately for any occurrence, that may or may not be marked:

pdb_setreptable *tablename*, mark[, immediate]

Parameters

tablename

The name of a user table in the primary database.

The *tablename* parameter can be owner-qualified to include the primary table owner name, with each element separated by a period. For example:

```
owner.table
```

This parameter can be delimited with quote characters to specify the character case.

If mixed character case (both uppercase and lowercase) is required, the name must be delimited. For example:

```
"Owner".table
"Owner"."Table"
```

Each mixed case element of the *tablename* option must be delimited separately, as shown in the previous example.

If an object name contains any non-alphanumeric characters, such as spaces or periods, it must be delimited with quote characters. For example:

```
"table name"
owner."table name"
```

If an object name contains a period, it must be *both* owner-qualified and delimited with quote characters. For example:

```
owner."table.name"
"table.owner"."table.name"
```

repname

The name of the table specified in the replication definition for a primary table.

Note The replicated name you specify with the pdb_setreptable command must match a table name specified by a with primary table named clause in a Replication Server replication definition for the primary database connection. The Replication Agent cannot validate the replication definition, but if it does not exist, or if the with primary table named clause does not match the replicated name specified with pdb_setreptable, replication from the primary table will fail.

The *repname* option can be owner-qualified to include the replicate table owner name, with each element separated by a period. For example:

```
repowner.reptable
```

The *repname* option can also be delimited with quote characters to specify the character case. See the previous description of the *tablename* option for details.

Note If the replicate table name contains a period (for example, table.name), without owner qualification, you must set the value of the Replication Agent use_rssd parameter to true.

---

all
: A keyword that refers to all tables in the primary database. By using the all keyword, you can mark all user tables, or apply an unmark, enable, or disable operation to all marked tables.

mark
: A keyword that refers to marking a table.

owner
: A keyword that refers to the mark operation.

  The owner keyword turns on the SEND OWNER mode. When you specify the owner of a table in a replication definition, you must always use the owner keyword if you want to enable the SEND OWNER mode.

  owner mode sets a flag in the LTL telling Replication Server that any table level Replication definition must be owner qualified to match this table.

  If the owner mode is set, the replication definition must be owner qualified. If the owner mode is not set, the replication definition must not be owner qualified.

unmark
: A keyword that refers to unmarking a marked table.

force

A keyword that refers to the unmark operation or mark (For Oracle) operation:

Note The combination of mark and force keywords with the pdb_setreptable command is only valid for Oracle.

- When the force keyword follows the unmark keyword, the pdb_setreptable command immediately removes replication marking for the specified table in the primary database, without first checking the enable status of the table, or checking for pending operations in the transaction log. When the force keyword follows the unmark keyword and the all keyword, pdb_setreptable immediately removes replication marking from all marked tables in the primary database, regardless of their enable status or any pending operations in the transaction log.

  The force keyword also forces complete execution of the unmarking script, even if errors occur during the unmarking process (Microsoft SQL Server only). Normally, when errors occur during script execution, the script terminates immediately without completing. The force keyword can be useful when a previous script execution failed and left the unmarking operation incomplete.

  When errors occur during a forced script execution, the pdb_setreptable command returns the following message:

  ```
  Errors were encountered and ignored during FORCEd
  ```

```
script execution. See error log for details.
```

- *For Oracle:*

  When the force keyword follows the mark keyword, the pdb_setreptable command allows a table that contains one or more columns with unsupported datatypes to be marked for replication. No data for the unsupported columns is sent to Replication Server. As a result, any replicate table must have a suitable default value defined for the unsupported columns, since no data is received by the replicate database to be inserted into the unsupported columns.

  The force keyword can not be used in combination with the all keyword. Tables with unsupported datatypes must be individually marked using the pdb_setreptable command and the force keyword (they will never be automatically marked, or marked by default if they have columns with unsupported datatypes). In addition, tables with unsupported datatypes are not automatically marked when configuration parameter pdb_automark_tables is true. Tables with unsupported datatypes must be individually marked using the pdb_setreptable command and the mark and force keywords.

Note If a replication definition is created using the command rs_create_repdef, for a table that was marked using the force keyword, only columns with supported datatypes are listed in the replication definition. Any column with an unsupported datatype is excluded from the replication definition.

---

enable
: A keyword that refers to enabling replication for marked tables.

disable
: A keyword that refers to disabling replication for marked tables.

immediate
: A keyword that allows a table to be immediately marked for any occurrence.

Examples

Example 1

```
pdb_setreptable authors
```

This command returns replication marking information for the table named “authors” in the primary database.

Example 2

```
pdb_setreptable mark
```

This command returns replication marking information for all marked tables in the primary database.

Example 3

```
pdb_setreptable disable
```

This command returns replication marking information for all marked tables for which replication has been disabled in the primary database.

Example 4

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

This command forces unmarking for all marked tables in the primary database.

Example 5

```
pdb_setreptable all, enable
```

This command enables replication for all marked tables in the primary database.

Example 6

```
pdb_setreptable authors, mark
```

This command marks for replication the table named "authors" in the primary database. The primary table name in the replication definition must be authors.

Example 7

```
pdb_setreptable authors, mark, owner
```

This command marks for replication the table named "authors" in the primary database so that the OWNER_MODE is enabled in the LTL. Therefore, any table replication definition created for this table must also be owner qualified.

Example 8

```
pdb_setreptable ptable, rtable, mark, owner
```

The primary table name in the replication definition must be:

```
powner.rtable
```

Example 9

```
pdb_setreptable ptable, rowner.rtable, mark,owner
```

The primary table name in the replication definition must be:

```
rowner.rtable
```

Example 10

```
pdb_setreptable ptable, rowner.rtable, mark,owner
```

The primary table name in the replication definition must be:

```
rowner.rtable
```

Example 11

```
pdb_setreptable authors, enable
```

This command enables replication for the marked table “authors” in the primary database.

Example 12

```
pdb_setreptable table=mark, enable
```

This command enables replication for the marked table named “mark” in the primary database.

Example 13

```
pdb_setreptable authors, unmark, force
```

This command forces unmarking for the marked table “authors” in the primary database.

Example 14 (Oracle only)

```
pdb_setreptable authors, mark, force
```

This command forces table “authors” to be marked, even if it contains columns with unsupported datatypes. The columns with unsupported datatypes will not be replicated.

Usage

- When a marked table is renamed or dropped and a new table with the original name is created, you must explicitly mark the new table because the new table has no marking-related information from the original table. The marking information is maintained internally by object ID, not table name.
- When a table is marked for replication and the owner mode is set to on, the replication definition must contain the owner name in the “with primary table named” clause, or the “with all tables named” clause. If the owner mode setting and the existence of the owner name in the replication definition do not match, the replication definition is not be used.

  For example:

  - Issuing the following command with the owner mode set to on:

    ```
    pdb_setreptable “mytable”, mark, owner
    ```

causes the rs_create_repdef command to generate the following replication definition for the primary and replicate database, which the Replication Server expects to receive:

```
create replication definition ra$0xda_"mytable"
with primary at ora102.dco
with primary table named "qa4user"."mytable"
with replicate table named "qa4user"."mytable"
.
.
.
```

- Issuing the following command with the owner mode set to off:

  ```
  pdb_setreptable "mytable", mark
  ```

  causes the rs_create_repdef command to generate the following replication definition for the primary and replicate database, which the Replication Server expects to receive:

  ```
  create replication definition ra$0xda_"mytable"
  with primary at ora102.dco
  with primary table named "mytable"
  with replicate table named "qa4user"."mytable"
  .
  .
  .
  ```

- How you use the pdb_setreptable command depends on the type of replication definition that you have created at the Replication Server. If you have created a database replication definition with no table replication definition, then the replicate procedure in the pdb_setreptable command refers to the table in the replicate database. However, if you have created a table replication definition, then the replicate table in the pdb_setreptable command refers to the name of the table replication definition, and it is the table replication definition that must map to the table in the replicate database.

  If *no* table replication definition exists and will not be added prior to replication, but only a database replication definition exists, use the following commands to mark a table for replication:

  - When the table in the replicate database has the same name as the table in the primary database:

    ```
    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.

- When the table in the replicate database has the different name than the table in the primary database:

  ```
  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 replicate database.

If a table replication definition exists or will be added prior to replication, regardless of whether or not a database replication definition exists, use the following commands to mark a table for replication:

- When the primary table in the table definition has the *same* name as the table in the primary database:

  ```
  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.

  - If the table in the replicate database also has the *same* name as the table replication definition, then you can use the "with all tables named" clause in the replication definition in the primary Replication Server. For example:

    ```
    create replication definition my_table_repdef
    with primary at data_server.database
    with all tables named pdb_table …
    ```

  - If the table in the replicate database has a *different* name than the primary table in the table replication definition, then the table replication definition must map to the table in the replicate database. For example:

    ```
    create replication definition my_table_repdef
    with primary at data_server.database
    with primary table named pdb_table
    with replicate table name rep_table …
    ```

- When the name of the table replication definition is *different* than the table in the primary database:

```
pdb_setreptable pdb_table, rdpri_table, mark
```

where:

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

rdpri_table is the name of the primary table in the replication definition.

- If the table in the replicate database also has the *same* name as the primary table in the table replication definition, then you can use the "with all tables named" clause in the replication definition in the primary Replication Server. For example:

```
create replication definition my_table_repdef
with primary at data_server.database
with all tables named rdpri_table …
```

- If the table in the replicate database has a *different* name from the primary table in the table replication definition, then the table replication definition must map to the table in the replicate database. For example:

```
create replication definition my_table_repdef
with primary at data_server.database
with primary table named rdpri_table
with replicate table name rep_table …
```

- When multiple keywords and options are specified, each must be separated by a comma. Blank space before or after a comma is optional. For example:

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

- When you specify a primary table in the pdb_setreptable command, you must use the name of a valid user table.
- You cannot specify the following items as a primary table in the pdb_setreptable command:
  - System tables
  - Views
  - Replication Agent transaction log tables
- If you specify an alias or synonym as a primary table in the pdb_setreptable command, the actual table that the alias or synonym refers to is acted upon. The actual table name is the table name sent to the primary Replication Server.

- If a table name in the primary database is the same as a keyword, it can be identified by adding the table=*string* to the beginning of the name. For example:

  ```
  pdb_setreptable table=unmark, mark
  ```

  This is true for both primary table names and replicated names.

- For UDB only, if the *mark* script fails, it is stored in a file (*mark.sql*) in the *RAX-15_2\inst_name\scripts\tablename* directory, the table is not marked, and transaction log objects are not created. You can examine the script by viewing the *mark.sql* file.

- For UDB only, if the *unmark* script fails, it is stored in a file (*unmark.sql*) in the *RAX-15_2\inst_name\tablename\scripts* directory, the table is not unmarked, and the transaction log objects are not removed. You can examine the script by viewing the *unmark.sql* file. When the *unmark* script execution encounters a fatal error on any database object, the pdb_setreptable command returns the following message:

  ```
  Could not unmark the following objects: ...
  See error log for details.
  ```

- If you create a new table using a table name that was previously marked you must mark the new table by executing the pdb_setreptable command with the mark option, even if you did not unmark the previous table.

- When you use the unmark keyword to remove replication marking from a primary table, the Replication Agent verifies that replication is disabled for that table and checks to make sure that there are no pending (unprocessed) operations for that table in the transaction log. If replication is not disabled, or there is a pending operation for that table in the transaction log, pdb_setreptable returns an error.

- When you use the unmark keyword to remove replication marking from primary tables, you can also specify the force keyword to immediately remove replication marking from primary tables, without regard to whether replication is disabled or pending operations exist in the transaction log.

  The force keyword also ignores script execution errors. If the *unmark* script execution encounters a fatal error on any database object, the pdb_setreptable command returns the following message:

  ```
  Could not unmark the following objects: ...
  See error log for details.
  ```

- When pdb_setreptable is invoked with either no option or a single option, it returns marking information about the user tables in the primary database:
  - If pdb_setreptable is invoked with no option, it returns a list of all marked tables in the primary database.

    Note Invoking the pdb_setreptable command with no option produces the same result as invoking the pdb_setreptable with the mark keyword.

  - If pdb_setreptable is invoked with a table name, it returns complete marking information about the specified primary table.
  - If pdb_setreptable is invoked with the mark keyword, it returns a list of all marked tables in the primary database.
  - If pdb_setreptable is invoked with the unmark keyword, it returns a list of all unmarked tables in the primary database.
  - If pdb_setreptable is invoked with the enable keyword, it returns a list of all marked tables in the primary database for which replication is enabled.
  - If pdb_setreptable is invoked with the disable keyword, it returns a list of all marked tables in the primary database for which replication is disabled.

  Tables marked for replication are listed in the marked objects table. All other user tables are considered unmarked.

  Note The Replication Agent transaction log tables and shadow tables are not included in the list of unmarked tables. Also not included are any synonyms, views, or aliases of these database objects.

  For tables listed as unmarked or disabled, transactions will not be captured for replication.
- When pdb_setreptable is invoked with the all keyword and an action keyword (mark, unmark, enable, or disable), the action specified is applied to either all tables in the primary database, or all marked tables in the primary database.

- If pdb_setreptable is invoked with the all and mark keywords, all user tables in the primary database are marked for replication.

  Note Tables owned by users contained in the owner filter list will not be marked. However, you will be able to mark any individual table.

- If pdb_setreptable is invoked with the all and unmark keywords, it removes replication marking from all marked tables in the primary database.

  You can specify the force keyword after the unmark keyword to force immediate unmarking of all marked tables, or to unmark tables for which replication is still enabled or pending operations remain in the transaction log, or to force the script execution to ignore errors and continue an unmarking operation that failed previously.

- If pdb_setreptable is invoked with the all and enable keywords, it enables replication for all marked tables in the primary database.

- If pdb_setreptable is invoked with the all and disable keywords, it disables replication for all marked tables in the primary database.

- When pdb_setreptable is invoked with a valid user table name, followed by an action keyword (mark, unmark, enable, or disable), the action specified is applied to the specified table.

  - If pdb_setreptable is invoked with a table name and the mark keyword, it marks the specified table in the primary database for replication.

    Note When an individual table is marked the owner filter list is not checked. This allows users to mark a table that has an owner in the owner filter list.

  - If pdb_setreptable is invoked with a table name and the unmark keyword, it removes replication marking from the specified table in the primary database.

    You can specify the force keyword after the unmark keyword to force immediate unmarking of the specified table, to unmark a table for which replication is still enabled or pending operations remain in the transaction log, or to force the script execution to ignore errors and continue an unmarking operation that failed previously.

  - If pdb_setreptable is invoked with a table name and the enable keyword, it enables replication for the specified marked table in the primary database.

If the enable script execution encounters a fatal error on any database object, the pdb_setreptable command returns the following message:

```
Could not enable the following objects: ...
See error log for details.
```

- If pdb_setreptable is invoked with a table name and the disable keyword, it disables replication for the specified marked table in the primary database.

  If the disable script execution encounters a fatal error on any database object, the pdb_setreptable command returns the following message:

  ```
  Could not disable the following objects: ...
  See error log for details.
  ```

- If the table name you specify does not exist in the primary database, the pdb_setreptable command returns an error.
- When pdb_setreptable is invoked with a primary table name and a replicated name, followed by the mark keyword, the primary table is marked for replication with the specified replicated name.

  If the primary table name you specify does not exist in the primary database, the pdb_setreptable command returns an error.

  By specifying a replicated name, transactions can be replicated to a table in the replicate database that has a different name from the primary table.

Note The replicated name you specify with the pdb_setreptable command must match a table name specified by a with all tables named clause in a Replication Server replication definition for the primary database connection. The Replication Agent cannot validate the replication definition, but if it does not exist, or if the with all tables named clause does not match the replicated name specified with pdb_setreptable, replication from the primary table will fail.

You can also specify the owner keyword after the mark keyword so that when operations against the primary table are replicated, the primary table owner name will be attached to the replicate table name in the form owner.tablename.

Note If you want to use an owner-qualified replicate table name with the replicate owner's name, use the owner keyword with the pdb_setreptable command. If you specify an unqualified replicate table name, the primary table owner name is sent with the replicate table name in the LTL.

- If the Replication Agent transaction log does not exist in the primary database (DB2 UDB) or the RASD is not initialized (Oracle and Microsoft SQL Server), the pdb_setreptable command returns an error.
- To replicate a table that contains column names that have spaces, you must set structured_tokens to true.
- (For Oracle) If a table contains a column with a datatype that is not supported for replication, the pdb_setreptable command using the mark keyword may fail with an error similar to:

```
Command <pdb_setreptable> failed - Table <MYTABLE>
could not be marked because:

The table contains an unsupported data type.
```

  To force the table to be marked, excluding the unsupported datatype columns from replication, add the force keyword to the pdb_setreptable command.

See also pdb_setrepcol, pdb_setrepproc, ra_config

pdb_skip_op

Description

Note This command is available only for Oracle and Microsoft SQL Server.

Returns, adds to, or removes operations from a list of operations to skip during processing. The format of the record locator is database-specific.

Syntax pdb_skip_op [add | remove], [locator | scn, thread, rba] | [remove, all]

Parameters

add

Adds a specified ID to the list of identifiers of records to skip.

remove
Removes a specified ID from the list of identifiers to skip.

locator
The locator keyword from the list of LTM locators to identify the operations to skip.

scn
(For Oracle) The system change number (SCN) keyword identifies a specified log record to skip.

thread
(For Oracle) The thread keyword of the redo log thread of the operation to skip.

rba
(For Oracle) The record byte address (RBA) keyword of the log record to skip.

all
Allows you to add or remove all IDs in the list of identifiers to skip.

Examples

Example 1

```
pdb_skip_op
```

This command with no parameters returns a list of the identifiers for the records you want to skip.

Example 2

```
pdb_skip_op add, id
```

This command adds an ID to the list of identifiers you want to skip.

Example 3

```
pdb_skip_op remove, id
```

To remove an ID to the list of identifiers you want to skip:

Example 4

```
pdb_skip_op remove, all
```

This command removes all the IDs on the list of identifiers you want to skip.

Example 5

```
pdb_skip_op add, locator
```

This command adds an operation, referred to by its location, to the list of identifiers that you want to skip.

Usage

- The pdb_skip_op command allows you to skip problem records, thereby avoiding having to reinitialize the Replication Agent.
- Skipped records are written to the system log as a warning message.
- The pdb_skip_op command is valid when the Replication Agent is in *Admin* state.
- The format of the identifier is database specific:
  - For Microsoft SQL Server, the identifier is the RBA and it has the following form:

    ```
    virtualfileseqno:pageid:operationsseqno
    ```

    where:

    - virtualfileseqno is the virtual log sequence number (LSN).
    - pageid is the page number.
    - operationsseqno is the sequence number of the operation.

    You can get the LSN using the Microsoft SQL Server DBCC command, or you can parse it from the locator. For example:

    - LSN in decimal format:

      ```
      489:112:27
      ```

    - LSN in hexadecimal format:

      ```
      0x00000489:00000112:0027
      ```

      ```
      0012.0000444.0000123
      ```

    An RBA value can be described as a hexadecimal by prefixing the identifier with an “0x” as follows:

    ```
    0x00c.00001bc.000007b
    ```

  - For Oracle, the identifier contains the system change number (SCN), redo log thread, and record byte address (RBA). It has the following form:

    ```
    wrap.base.sub, thread, lsn.blknum.blkoffset
    ```

    where:

    - wrap is the SCN wrap number.
    - base is the SCN base number.

- sub is the SCN subindex.
- thread is the redo log thread number that the operation occurred on.
- lsn is the RBA log sequence number.
- blknum is the RBA block number.
- blkoffset is the RBA offset into the block where this record resides.

The values must be specified in the pdb_skip_op command together, as shown above, enclosed in quotes, with each item separated by a period.

For example:

```
'0000.012345678.00', '1', '0012.0000444.0000123'
```

All values can be described as hexadecimal by prefixing the identifier with an "0x" as follows:

```
'0x000c.00001bc.000007b'
```

or:

```
'0x000.00BC614E'
```

See also ra_helplocator, ra_locator

pdb_truncate_xlog

Description Truncates the Replication Agent primary database transaction log on demand.

- The behavior of this command changes based on the value of the configuration parameter pdb_include_archives:
  - When pdb_include_archives is false, triggers the archive process to archive any online redo logs that have already been processed by Replication Agent.

- When pdb_include_archives is true, removes old archive redo log files from the path specified by pdb_archive_path.

Note Ttruncation of the old archive log files from the pdb_archive_path directory is performed only if the pdb_archive_remove parameter is set to true.

- For UDB, if the pdb_archive_path is not set, then archived transaction log files are truncated from the location specified by the pdb_archive_path configuration parameter, when pdb_archive_remove parameter is set to true.

  The primary transaction log files will be deleted up to, but not including, the log file that contains the lsn found.

Warning! For UDB, if pdb_archive_path is not set, then Replication Agent deletes the *primary database* log files that it no longer needs. For more information, see the *Replication Agent Primary Database Guide*.

- For Microsoft SQL Server, the primary database log is truncated. Based on the LTM locator from the Replication Server, Replication Agent queries the primary database to obtain the newest transaction ID that can be truncated, and marks the transaction log space prior to the newest transaction as reusable.

For more information on how Replication Agent affects each type of database when pdb_truncate_xlog is executed, see the *Replication Agent Primary Database Guide*.

Syntax

pdb_truncate_xlog

Usage

- When pdb_truncate_xlog is invoked, Replication Agent immediately truncates the primary database 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 update the LTM Locator from the primary Replication Server, use the ra_locator command.
- The pdb_truncate_xlog command is asynchronous and it does not return success or failure (unless an immediate error occurs). You must examine the Replication Agent system log to determine success or failure of the pdb_truncate_xlog command.

- If the Replication Agent primary database log does not exist or if a connection failure occurs, the pdb_truncate_xlog command returns an error message.
- You can use the ra_config command to specify the type of automatic truncation you want. You can use the pdb_truncate_xlog command to truncate the transaction log if automatic truncation is not sufficient to manage the size of the transaction log.
- The pdb_truncate_xlog command is valid when the Replication Agent instance is in either *Admin* or *Replicating* state.

See also ra_config, ra_locator

pdb_version

Description Returns the type and version of the primary data server.

Syntax pdb_version

Usage The actual results returned vary depending on the type of primary data server.

See also ra_version, ra_version_all

pdb_xlog

Description Returns the names of transaction log base objects; creates transaction log base objects in the primary database; or removes transaction log base objects from the primary database.

For Oracle and Microsoft SQL Server, this command verifies permissions are valid for the Replication Agent to obtain system data from the primary database. It also checks the condition of the primary database to determine if archiving is turned on or off, and then loads the RASD with system data from the primary database.

Note Because the pdb_xlog create command is deprecated, Sybase recommends that you use pdb_xlog init.

Syntax pdb_xlog [[init |create | remove] [, force]] | move_truncpt]

Parameters

init
The keyword for moving the truncation point to the end of the transaction log.

create
The keyword for creating a transaction log. Deprecated; use the init keyword instead.

remove
The keyword for removing a transaction log.

force
A keyword that refers to the remove or the init operation.

move_truncpt
A keyword that moves the truncation point; for Oracle and Microsoft SQL Server only.

Examples

```
pdb_xlog init
```

This command initializes the Replication Agent, creating any required xlog base components.

```
pdb_xlog init, force
```

This command re-initializes the Replication Agent, creating or re-loading any required xlog base components.

```
pdb_xlog remove
```

This command removes any Replication Agent xlog base components.

```
pdb_xlog remove, force
```

This command removes any Replication Agent xlog base components and ignores any individual errors that occur during removal.

```
pdb_xlog init, move_truncpt
```

This command moves the transaction log truncation point to the end of the current transaction log (Oracle and Microsoft SQL Server).

Usage

- When pdb_xlog is invoked with no option, it returns the actual names (not synonyms or aliases) of all Replication Agent transaction log base objects in the primary database. For Oracle and Microsoft SQL Server, if you have initialized the Replication Agent, it returns the name of the component and the primary database instance name.

  Note See the chapter for your specific primary data server in the *Replication Agent Primary Database Guide* for more information on Replication Agent object names.

- If pdb_xlog is invoked with no option, and the Replication Agent transaction log base objects do not exist in the primary database, or the RASD has not been initialized (For Oracle and Microsoft SQL Server), the command returns no information.
- If pdb_xlog is invoked with the init keyword, the truncation point is established at the end of the primary database transaction log.

  Note For Microsoft SQL Server, during the pdb_xlog init process, Replication Agent may connect to the Microsoft SQL Server using pds_dac_port_number. For more information, see the Replication Agent *Primary Database Guide*.

- If pdb_xlog is invoked with the init, force keywords, the truncation point is moved to the end of the log if the Replication Agent is not already initialized. However, if the Replication Agent is already initialized, the truncation point is not moved.

  Note pdb_xlog init with the force keyword should only be used when advised by Sybase Technical support.

- If pdb_xlog is invoked with the move_truncpt keyword, the truncation point is moved to the end of the log without change or modification to any Replication Agent components. (For Oracle, this is the end of the current online redo log.) The move_truncpt option has no effect if the Replication Agent has not been initialized.

Note To prevent Replication Server from requesting a log starting point that occurs earlier in the log than the location established by the move_truncpt option, the Replication Server's LTM locator value for the primary connection must be zeroed. Execute Replication Server System Database (RSSD) command rs_zeroltm against the primary database connection to zero the LTM locator.

(For Oracle and Microsoft SQL Server) If you move the secondary truncation point to the end of the primary database transaction log using pdb_xlog init, move_truncpt, you risk skipping over any DDL commands record in the log. The DDL commands might have been used by the Replication Agent to update information stored within the Replication Agent System Database (RASD). If the RASD contents are incorrect due to skipping processing of some log records, you may force all of the schema information in the RASD to be refreshed using command pdb_xlog init, force. If only the schema for a single object stored in the RASD is of concern, you can unmark and remark just that single object, which will force the schema of the object to be reread into the RASD.

- When pdb_xlog is invoked with the init keyword, Replication Agent does the following:
  - Generates a SQL script that creates the Replication Agent tables and procedures in the primary database.
  - Saves the generated script in a file called *partinit.sql* in the *RAX-15_2\inst_name\scripts\xlog* directory, where *inst_name* is the name of the Replication Agent instance.

    Note If the value of the pdb_auto_run_scripts configuration parameter is false, the *partinit.sql* script will be saved but not executed. However, you cannot manually run the script. To complete initializing Replication Agent, you must first set pdb_auto_run_scripts to true, then re-run the pdb_xlog init command.

  - Executes the script to create the Replication Agent transaction log base objects in the primary database (if the value of the pdb_auto_run_scripts configuration parameter is true).
  - After the script completes successfully, moves the *partinit.sql* file to the *RAX-15_2\inst_name\scripts\xlog\installed* directory.

    - If the create script fails, it is stored in a file (*partinit.sql*) in the *RAX-15_2\inst_name\scripts\xlog* directory and the transaction log is not created. You can examine the script by viewing the *partinit.sql* file.

- If pdb_xlog is invoked with the init keyword and the Replication Agent objects already exist in the primary database or the RASD has been initialized (For Oracle and Microsoft SQL Server), then pdb_xlog returns an error message.
- When pdb_xlog is invoked with the remove keyword, Replication Agent does the following:
    - For UDB, pdb_xlog remove command will uninstall the jar files from the primary database (the jars are installed by the pdb_xlog init command).

      It is necessary to use pdb_xlog remove command to de-initialize Replication Agent for UDB, and remove the truncation stored procedures and jars from the database.
    - Generates a SQL script that deletes the tables and procedures required for the transaction log base objects in the primary database.
    - Saves the generated script in a file called *partdeinit.sql* in the *RAX-15_2\inst_name\scripts\xlog* directory, where *inst_name* is the name of the Replication Agent instance.

      Note If the value of the pdb_auto_run_scripts configuration parameter is false, the *partdeinit.sql* script will be saved but not executed automatically. You cannot manually run the script. To complete de-initializing Replication Agent, you must first set pdb_auto_run_scripts to true, then re-run the pdb_xlog remove command.

    - Executes the script to delete the Replication Agent objects from the primary database (if the value of the pdb_auto_run_scripts configuration parameter is true).
    - After the script completes successfully, moves the *partdeinit.sql* file to the *RAX-15_2\inst_name\scripts\xlog\installed* directory.
    - If the script fails, it is stored in a file (*partdeinit.sql*) in the *RAX-15_2\inst_name\scripts\xlog* directory and the Replication Agent objects are not deleted from the primary database. You can examine the script by viewing the *partdeinit.sql* file.

- When pdb_xlog is invoked with the remove keyword followed by the force keyword, the *partdeinit.sql* script continues executing, even if errors occur. The force keyword may be useful when a previous remove operation failed and the *partdeinit.sql* script terminated with an error.
- If pdb_xlog is invoked with the remove keyword, and Replication Agent objects do not exist in the primary database or the RASD has not been initialized (For Oracle or Microsoft SQL Server), then pdb_xlog returns an error message.
- If pdb_xlog is invoked with the remove keyword and any objects in the primary database are still marked for replication, then pdb_xlog returns an error message.

  You can use the pdb_setrepproc and pdb_setreptable commands to determine which stored procedures and tables in the primary database are still marked. You also can use the pdb_setrepddl command to determine if DDL is enabled.

  Even if objects are marked in the primary database, you can use the pdb_xlog command with the remove keyword followed by the force keyword to unmark any marked objects, and then remove the transaction log objects.
- If pdb_xlog is invoked with no option, the command is valid when the Replication Agent instance is in either *Admin* or *Replicating* state.
- If pdb_xlog is invoked with either the init or remove keyword, the command is valid only when the Replication Agent instance is in the *Admin* state.
- For more information about the Replication Agent transaction log, see the chapter for your specific primary data server in the *Replication Agent Primary Database Guide*.

See also pdb_setrepcol, pdb_setrepproc, pdb_setreptable, ra_config

quiesce

Description Stops all Replication Agent processing in *Replicating* state, and puts the Replication Agent instance in *Admin* state.

Syntax quiesce

Usage
- When the quiesce command is invoked, it stops all current replication processing in the Replication Agent instance:

  - The Log Reader component stops reading operations from the transaction log when the scan reaches the end of the log. It continues to send change-set data to the Log Transfer Interface component until it finishes processing the last operation scanned.
  - The Log Transfer Interface component stops sending LTL commands to the Replication Server as soon as it finishes processing the last change set it receives 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, and drops its connection to the primary Replication Server (and RSSD, if connected).
  - The Replication Agent instance goes from *Replicating* state to *Admin* state.
- If the Replication Agent internal queues are full when the quiesce command is invoked, 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.
- Before moving the Replication Agent to the *Admin* state, the quiesce command waits until all data in the primary log has been read and sent to the Replication server.
- If the Replication Agent instance is in *Admin* state, the quiesce command returns an error.
- The quiesce command is valid only when the Replication Agent instance is in *Replicating* state.

Note The action of the suspend command is similar to that of the quiesce command, except that the suspend command stops Replication Agent processing immediately and flushes all data in the internal queues.

See also ra_status, resume, shutdown, suspend

ra_config

Description Returns help information for Replication Agent configuration parameters, or sets the value of a specified configuration parameter.

Syntax

ra_config [*param* [, *value*]]

Parameters

param

The name of a Replication Agent configuration parameter.

value

The value to be assigned to the configuration parameter specified in the *param* option. You can use the keyword default to set the specified parameter to its default value.

Examples

Example 1

```
ra_config
```

When this command is issued with *no* parameters, it returns a list of all Replication Agent configuration parameters.

Example 2

```
ra_config use_rssd
```

This command returns the current value of the use_rssd configuration parameter.

Example 3

```
ra_config scan_sleep_max, 60
```

This command changes the value of the scan_sleep_max parameter to 60.

Usage

- If ra_config is invoked with no option, it returns a list of all Replication Agent configuration parameters.
- If ra_config is invoked with the *param* option, it returns information only for the specified configuration parameter.
- If ra_config is invoked with the *param* and *value* options, it changes the setting of the specified configuration parameter to the value specified in the *value* option.
- You can use the keyword default in place of the *value* option to reset a configuration parameter to its default value. For example:

  ```
  ra_config use_rssd, default
  ```

- The following information is returned for each configuration parameter:
  - Parameter name – the name of the parameter.
  - Parameter type – the datatype of the parameter's value (for example, string, numeric, or Boolean).

  - Current value – the value of the parameter in effect at the time ra_config is invoked.
  - Pending value – if different from the current value, the value to which the parameter was set by a previous invocation of the ra_config command, but which has not yet taken effect.
  - Default value – the value of the parameter when the Replication Agent instance configuration file is created.
  - Legal values – the values that are allowed for the parameter, for example, a range of numbers or a list of specific strings.
  - Category – refers to the Replication Agent component affected by the value of the parameter.
  - Restart – refers to parameters that require the Replication Agent instance to be shut down and restarted before a change in value takes effect.
- When ra_config is invoked with either no option, or only the param option, the command is valid when the Replication Agent instance is in either *Admin* or *Replicating* state.
- If ra_config is invoked when the Replication Agent instance is in *Replicating* state, with the param and value options for a parameter that can be changed only in *Admin* state, it returns an error.
- When ra_config is invoked with the param and value options, the command is always valid when the Replication Agent instance is in *Admin* state.
- See Chapter 2, "Configuration Parameters," for more information.

See also ra_help, ra_set_login

ra_date

Description Returns the current date and time from the Replication Agent instance.

Syntax ra_date

Usage
- When ra_date is invoked, it returns the current date and time from the Replication Agent instance in the form of a Sybase datetime datatype, as follows:

```
Current RA Date
 ----------------------------
```

```
                Jan 12 2008 12:09:47.310
        (1 row affected)
```

- The ra_date command is valid when the Replication Agent instance is in either *Admin* or *Replicating* state.

See also pdb_date, ra_config

ra_deviceoffset

Description **Note** This command is available only for Oracle.

Changes the raw disk device read offset for a log device recorded in the RASD.

Syntax ra_deviceoffset *device_id*, *device_offset*

Parameters *device*_id
The device ID is the Oracle redo log "Group number."

device_offset
The offset of the raw device from which Replication Agent starts a log scan.

Examples Example 1

```
ra_deviceoffset 1,20
```

This command specifies the raw device read offset to the log device ID "1" as:

```
20
```

Usage
- When ra_devicepath is invoked, Replication Agent records the specified raw device read offset for the specified log device in the RASD.
- To get information about log devices stored in the RASD, use the ra_helpdevice command.
- When ra_updatedevices is invoked, device offset is not reset to the default value. The default device read offset is zero.
- If ra_deviceoffset is invoked when the Replication Agent instance is in *Replicating* state, it returns an error.
- The ra_deviceoffset command is valid only when the Replication Agent instance is in *Admin* state.

See also ra_helpdevice, pdb_xlog, ra_updatedevices

ra_devicepath

Description

Note This command is available only for Oracle and Microsoft SQL Server.

Changes the disk device path for a log device recorded in the RASD. For Oracle and the ASM log devices, see the pdb_asmdiskmap command.

Syntax

ra_devicepath *device*, *dev_path*

Parameters

device
The device ID is the Oracle redo log "Group number."

dev_path
The path that points to the disk log device for the device specified in the *device* option.

Examples

Example 1

```
ra_devicepath 3,
d:\software\oracle\devices\redo001.log
```

This command specifies the disk device path to the log device ID "3" as:

```
d:\software\oracle\devices\redo001.log
```

Usage

- When ra_devicepath is invoked, Replication Agent records the specified disk device path for the specified log device in the RASD.
- To get information about log devices stored in the RASD, use the ra_helpdevice command.
- If you invoke ra_updatedevices after you set a device path using ra_devicepath, you must use ra_devicepath again to re-set the path if you need to alter the default path for a log device. The default device path is the device path returned by the primary data server.
- If you invoke ra_devicepath when the Replication Agent instance is in *Replicating* state, it returns an error.
- The ra_devicepath command is valid only when the Replication Agent instance is in *Admin* state.

See also

ra_helpdevice, pdb_xlog, ra_updatedevices

ra_dump

Description

Emulates the Replication Server rs_dumpdb and rs_dumptran system functions.

Syntax
ra_dump [database|transaction,] *dbname*, *dump_label*

Parameters
database
A keyword that causes the primary Replication Server to apply the function string associated with the rs_dumpdb system function.

transaction
A keyword that causes the primary Replication Server to apply the function string associated with the rs_dumptran system function.

dbname
The name of the database to be dumped.

dump_label
A varchar(30) value that contains information to identify the database dump.

Usage
- When ra_dump is invoked, Replication Agent places a dump marker in the Replication Agent transaction log to facilitate coordinated dumps.
- The ra_dump command returns an error message if the transaction log does not exist.
- The ra_dump command is valid when the Replication Agent instance is in either *Admin* or *Replicating* state.
- For more information about the Replication Server rs_dumpdb and rs_dumptran system functions, see the *Replication Agent Administration Guide* and *Replication Agent Primary Database Guide*.

See also
ra_config, ra_migrate

ra_help

Description
Returns help information for Replication Agent commands.

Syntax
ra_help [*command*]

Parameters
command
The name of a Replication Agent command for which you want to view help information.

Examples
Example 1

```
ra_help
```

This command returns help for all Replication Agent commands.

Example 2

```
ra_help pdb_gen_id
```

This command returns help for the pdb_gen_id command.

Usage

- If ra_help is invoked with no option, it returns help information for all Replication Agent commands.
- If ra_help is invoked with the *command* option, it returns help information only for the specified command.
- The ra_help command is valid when the Replication Agent instance is in either *Admin* or *Replicating* state.

See also

ra_config

ra_helparticle

Description

Note This command is available only for Oracle and Microsoft SQL Server.

Returns information about primary database articles from the RASD.

Syntax

ra_helparticle [*article*, [*version*]]

Parameters

article

The name or object ID of an article (table or procedure) in the primary database. Article names can be qualified with an owner name in the following form:

```
owner.article
```

Owner qualification of article names is optional.

version

A hexadecimal locator value that identifies the version of the article specified in the *article* option.

Examples

Example 1

```
ra_helparticle
```

This command returns information about all versions of all articles in the RASD.

Example 2

```
ra_helparticle table1
```

This command returns information about the current version of the article named “table1” in the RASD.

Example 3

```
ra_helparticle table1,
00000000000210a400003334000700003334000699940000d413c50000000000
```

This command returns information about version 00000000000210a400003334000700003334000699940000d413c50000000000 of the article named “table1” in the RASD.

Usage

- The ra_helparticle command returns the following information for articles (tables and procedures):
  - Article object ID
  - Primary database name
  - Article owner name or alias
  - Article name
  - Article type (table or procedure)
  - Article status (Current, Archived, or Dropped)
  - Article version number

  All information except the article type, article status, and article version number are the values returned by the primary database when the Replication Agent is initialized with the ra_init command.
- If ra_helparticle is invoked with no option, it returns information for all versions of all articles (tables and procedures) in the RASD.
- If ra_helparticle is invoked with the *article* option, it returns information only for the current version of the specified article in the RASD.
- If ra_helparticle is invoked with the *article* and *version* options, it returns information only for the specified version of the specified article in the RASD.
- The ra_helparticle command is valid when the Replication Agent instance is in either *Admin* or *Replicating* state.

See also

ra_helpdb, ra_helpfield, ra_helplocator, ra_helpuser

ra_helpdb

Description

Note This command is available only for Oracle and Microsoft SQL Server.

Returns information about the primary database from the RASD.

Syntax

ra_helpdb

Usage

- When ra_helpdb is invoked, it returns the following information about the primary database:
  - Database object ID
  - Database name

  The database ID and database name are the values returned by the primary database when the Replication Agent is initialized with the pdb_xlog init command.
- The ra_helpdb command is valid when the Replication Agent instance is in either *Admin* or *Replicating* state.
- The ra_helpdb command is valid only after the RASD has been initialized, that is, only after you have executed pdb_xlog init.

See also

ra_devicepath, ra_helparticle, ra_helpdevice, ra_helpfield, ra_helplocator, ra_helpuser, ra_updatedevices

ra_helpdevice

Description

Note This command is available only for Oracle and Microsoft SQL Server.

This command returns information about the primary database log devices from the RASD log device repository.

Syntax

ra_helpdevice [*device*]

Parameters

device
The device ID of the primary database log device.

Examples

Example 1

```
ra_helpdevice
```

This command returns information about all primary database log devices recorded in the log device repository.

Example 2

```
ra_helpdevice 1
```

This command returns information about the primary database log device ID "1" in the log device repository.

Usage

- The ra_helpdevice command returns the following information for each primary database log device recorded in the RASD:
  - Device ID – the log device ID defined by the primary data server.

    Note For Oracle, the ID is the value of the Oracle Redo Log Group that this file belongs.

  - Database name – the name of the primary database associated with the log device.
  - Device name – the logical name of the log device defined by the primary data server.
  - Server device path – the path to a multiplexed version of the log device.
  - Disk mirror path – the path to the log device (at the standby site).
  - Disk device status – the current status of the server device path (ACCESSIBLE, NOT_VALID, or OPEN).
- The log device ID, primary database name, log device name, and server log device path are values returned by the primary data server when the Replication Agent is initialized with the pdb_xlog init command, or when the log device repository is updated with the ra_updatedevices command.
- The disk mirror path is the current value recorded in the RASD. To find each log device, Replication Agent uses the disk mirror path recorded in its RASD.

  For each log device recorded in the RASD, you can set or change the disk device path with the ra_devicepath command.

  If you do not specify a disk device path using ra_devicepath, the value recorded for the disk mirror path is DEFAULT, and Replication Agent uses the value recorded for the server device path to find the log device.

  (For Oracle) The disk mirror path for ASM is specified using the file created with the pdb_asmdiskmap command.

- The disk device status is updated by the Log Reader component each time you invoke the ra_helpdevice command.
- If ra_helpdevice is invoked with no option, it returns information for all log devices recorded in the RASD log device repository.
- If ra_helpdevice is invoked with the device option, it returns information only for the specified log device.
- The ra_helpdevice command is valid when the Replication Agent instance is in either *Admin* or *Replicating* state.

See also ra_devicepath, ra_helpdb, ra_updatedevices

ra_helpdeviceoffset

Description

Note This command is available only for Oracle.

Returns device offset information about primary database log devices from the RASD log device repository.

Syntax ra_helpdeviceoffset [*device*]

Parameters

device
The device ID of the primary database log device.

Examples

Example 1

```
ra_helpdeviceoffset
```

This command returns the device offset information about all primary database log devices recorded in the log device repository.

Example 2

```
ra_helpdeviceoffset1
```

This command returns information about the primary database log device ID 1 in the log device repository.

Usage

- The ra_helpdeviceoffset command returns the following information for each primary database log device recorded in the RASD:

- Device ID – the log device ID defined by the primary data server.

  Note For Oracle, the device ID is the value of the Oracle Redo Log Group that contains this file.

- Database name – the name of the primary database associated with the log device.
- Device name – the logical name of the log device defined by the primary data server.
- Server device path – the path to a multiplexed version of the log device.
- Disk device path – the path to the log device (at the standby site).
- Disk device offset – the offset from which Replication Agent starts to scan the Oracle redo log in the log device. The value of the device offset is the value of the raw_device_offset parameter in the Replication Agent configuration file added to the offset value set by the command ra_deviceoffset.
- Disk device status – the current status of the server device path (ACCESSIBLE, NOT_VALID, or OPEN).

- The disk device path is the current value recorded in the RASD. Replication Agent uses the disk device path recorded in the RASD to locate each log device.
- For each log device recorded in the RASD, you can set or change the disk device path using ra_devicepath, and set or change the device offset using ra_deviceoffset.
- If you do not specify a disk device path using ra_devicepath, the value recorded for the disk device path is DEFAULT, and Replication Agent uses the value recorded for the server device path to find the log device.
- The disk device status is updated by the Log Reader component each time you invoke ra_helpdeviceoffset.
- If ra_helpdeviceoffset is invoked with no option, it returns information for all log devices recorded in the RASD log device repository.
- If ra_helpdeviceoffset is invoked with the device option, it returns information only for the specified log device.
- The ra_helpdevice command is valid when the Replication Agent instance is in either *Admin* or *Replicating* state.

See also ra_helpdevice, ra_helpdeviceoffset, ra_devicepath, ra_updatedevices

ra_helpfield

Description

Note This command is available only for Oracle and Microsoft SQL Server.

Returns information about primary database fields (columns in tables, or input parameters in stored procedures) from the RASD.

Syntax

ra_helpfield *article* [, *version* [, *field*]]

Parameters

article

The name or object ID of an article (table or procedure) in the primary database. Article names can be qualified with an owner name in the following form:

```
owner.article
```

Owner qualification of article names is optional.

version

A hexadecimal locator value that identifies the version of the specified article.

field

The name or object ID of a field (column or input parameter) in the specified article.

Examples

Example 1

```
ra_helpfield authors
```

This command returns information about all fields in the current version of the article named authors in the RASD.

Example 2

```
ra_helpfield authors,
00000000000210a40000333400070000333400699940000d413c50000000000
```

This command returns information about all fields in version 00000000000210a40000333400070000333400699940000d413c50000000000 of the article named “authors” in the RASD.

Example 3

```
ra_helpfield authors,
```

```
00000000000210a400003334000700003334000699940000d413c50000000000, au_fname
```

This command returns information about the field named au_fname in version 00000000000210a400003334000700003334000699940000d413c50000000000 of the article named "authors" in the RASD.

Usage

- The ra_helpfield command returns the following information for fields:
  - Field (column or input parameter) object ID
  - Field name
  - Field type ID
  - Field datatype (with precision, length, and scale)
  - Field NULL mode
  - Field IDENTITY status
  - Field primary key status

  All field information items are the values returned by the primary database when the Replication Agent is initialized with the pdb_xlog init command.
- If ra_helpfield is invoked with the *article* option, it returns information for all fields in the current version of the specified article in the RASD.
- If ra_helpfield is invoked with the *article* and *version* options, it returns information for all fields in the specified version of the specified article in the RASD.
- If ra_helpfield is invoked with the *article*, *version*, and *field* options, it returns information for the specified field in the specified version of the specified article in the RASD.
- The ra_helpfield command is valid when the Replication Agent is in either *Admin* or *Replicating* state.
- No results are returned by this command if the RASD has not yet been initialized with the pdb_xlog init command.

See also

ra_config, ra_help, ra_helparticle, ra_helpdb, ra_helpdevice, ra_helplocator, ra_helpuser

ra_helplocator

Description

Note This command is available only for Oracle and Microsoft SQL Server.

Returns information about fields in the LTM Locator value.

Syntax

ra_helplocator [*locator_value*]

Parameters

locator_value
The hexadecimal string value of an LTM Locator.

Examples

Example 1

```
ra_helplocator
```

This command returns information about fields in the current LTM Locator value.

Example 2

```
ra_helplocator locator_value
```

This command returns information about fields in the specified LTM Locator value.

Usage

- The ra_helplocator command returns the following information about the LTM Locator value:
  - Locator field names
  - Locator field hexadecimal values
  - Locator field decimal values
- If ra_helplocator is invoked with no option, it returns information about fields in the current LTM Locator value.
- If ra_helplocator is invoked with the *locator_value* option, it returns information about fields in the specified LTM Locator value.
- The ra_helplocator command is valid when the Replication Agent instance is in either *Admin* or *Replicating* state.

See also

ra_config, ra_help, ra_locator

See the *Replication Agent Primary Database Guide* for more information about locator fields and contents.

ra_helpuser

Description

Note This command is available only for Oracle and Microsoft SQL Server.

Returns information about primary database users from the RASD.

Syntax

ra_helpuser [*user* [, *version*]]

Parameters

user

The name or user ID of a user in the primary database.

version

The version number of the database user in the RASD.

Examples

Example 1

```
ra_helpuser
```

This command returns information about all versions of all users in the RASD.

Example 2

```
ra_helpuser bob
```

This command returns information about the current version of the database name “bob” in the RASD.

Example 3

```
ra_helpuser bob,
00000000000210a400003334000700003334000699940000d413c50000000000
```

This command returns information about version 00000000000210a400003334000700003334000699940000d413c50000000000 of the database user named “bob” in the RASD.

Usage

- The ra_helpuser command returns the following information about primary database users:
  - User ID
  - User name
  - User status (Current, Archived, or Dropped)
  - Primary database version (locator value)

  The user ID and user name are the values returned by the primary database when the Replication Agent is initialized with the ra_init command.
- If ra_helpuser is invoked with no option, it returns information about all users in all versions of the primary database in the RASD.

- If ra_helpuser is invoked with the *user* option, it returns information about the current version of the specified user in the primary database in the RASD.
- If ra_helpuser is invoked with the *user* and *version* options, it returns information about the specified user in the specified version of the primary database in the RASD.
- The ra_helpuser command is valid when the Replication Agent instance is in either *Admin* or *Replicating* state.
- No results are returned by this command if the RASD has not been initialized by the ra_init command.

See also

ra_config, ra_help, ra_helparticle, ra_helpdb, ra_helpdevice, ra_helpfield, ra_helplocator

ra_locator

Description

Returns the current value of the LTM Locator maintained by the Replication Agent, requests an LTM Locator value from the primary Replication Server, or sets the value of the LTM Locator maintained by the Replication Agent to zero.

Syntax

ra_locator [update|zero]

Parameters

update
The optional keyword to request a new LTM Locator value from the primary Replication Server.

zero
The optional keyword to set the value of the LTM Locator stored in the Replication Agent transaction log to zero.

Examples

Example 1

```
ra_locator
```

This command returns the current value of the LTM Locator maintained by the Replication Agent, as shown:

```
Locator
 ------------------------------------------------
 0000000052000000000000527FFFFFFFFFFFFFF0022FB3B
 (1 row affected)
```

Example 2

```
ra_locator update
```

This command requests a new LTM Locator value from the primary Replication Server.

Example 3

```
ra_locator zero
```

This command sets the value of the LTM Locator maintained by the Replication Agent to all zeros.

Usage

- When ra_locator is invoked with no option, it returns the current value of the LTM Locator maintained by the Replication Agent instance. For UDB, the Replication Agent stores the value of the LTM Locator in a table in the primary database; for Oracle and Microsoft SQL Server this value is stored in the RASD.

  Note The value of the LTM Locator that is maintained by the Replication Agent is also known as the *origin queue ID*.

- When ra_locator is invoked with the update keyword, it requests a new LTM Locator value from the primary Replication Server, and the Replication Agent saves the value.

  Note When the ra_locator command is invoked with the update keyword, the change takes effect only if the Replication Agent instance is in *Replicating* state.

- When ra_locator is invoked with the zero keyword, it sets the value of the LTM Locator maintained by the Replication Agent to zero.
- The LTM Locator contains information that the Replication Agent uses to determine where to start reading the transaction log.

  Upon start-up or recovery from a connection failure, the Replication Agent automatically requests an LTM Locator value from the primary Replication Server.

  - If the value of the LTM Locator returned from the primary Replication Server is zero, then Replication Agent uses the LTM Locator value stored in the transaction log system table.

    - If the value of the LTM Locator in the transaction log system table is zero, then Replication Agent starts reading the transaction log from either the current beginning of the log, or from the end of the log for UDB.
- For more information about the format of the origin queue ID, see the chapter for your specific primary data server in the *Replication Agent Primary Database Guide*.
- If the Replication Agent transaction log does not exist, the ra_locator command returns an error message.
- The ra_locator command with the zero keyword is valid only when the Replication Agent instance is in *Admin* state.
- Without the zero keyword, the ra_locator command is valid when the Replication Agent instance is in either *Admin* or *Replicating* state.

See also pdb_gen_id, pdb_truncate_xlog

ra_maintid

Description Returns the login name of the primary database maintenance user.

Syntax ra_maintid

Usage

- Replication Server requires a maintenance user login name for each database connection. The maintenance user login name for a database connection is specified with the Replication Server create connection or alter connection command.

  When the primary database maintenance user login name is changed in the Replication Server (using the alter connection command), Replication Server automatically sends the new maintenance user login name to the Replication Agent, if the Replication Agent is in *Replicating* state.

  Each time the Replication Agent goes into *Replicating* state, it automatically retrieves the primary database maintenance user login name from the primary Replication Server, and caches it.
- When ra_maintid is invoked, it returns the login name of the primary database maintenance user that is cached, as follows:

```
maintenance user
 ----------------
 SYS
```

```
(1 row affected)
```

- If ra_maintid is invoked when the Replication Agent is in *Replicating* state, it always returns the correct maintenance user login name.

  If ra_maintid is invoked when the Replication Agent is in *Admin* state, it may not return the correct maintenance user login name, because the maintenance user login name could have changed in the Replication Server after the last time the Replication Agent retrieved the value and stored it.

- The filter_maint_userid configuration parameter is provided to support bidirectional replication, wherein the primary database also acts as a replicate database that has transactions applied to it by a Replication Server.

  If the value of the filter_maint_userid parameter is true, database operations applied by the maintenance user are *not* replicated from the primary database. When it reads the transaction log, the Replication Agent Log Reader component filters out data-changing operations applied by the maintenance user.

- The ra_maintid command is valid when the Replication Agent instance is in either *Admin* or *Replicating* state.

See also ra_config, ra_statistics

ra_marker

Description Places a marker in the primary database transaction log.

Syntax ra_marker *command_tag*

Parameters *command_tag*
A varchar(255) value that contains information used for subscription materialization.

Examples
```
ra_marker 'activate subscription 309 0 with suspension'
```
This command places a marker object in the Primary Database transaction log that invokes the Replication Server activate subscription command.

Usage

- When ra_marker is invoked, the Replication Agent executes a transaction in the Primary Database that is captured in the Primary Database transaction log. The replicated transaction is sent as a marker object to the primary Replication Server.

- The ra_marker command returns an error message if the Replication Agent transaction log does not exist.
- The ra_marker command is valid when the Replication Agent instance is in either *Admin* or *Replicating* state.
- For more information about the Replication Server rs_marker system function, refer to the *Replication Server Administration Guide* and *Replication Agent Reference Manual*.

See also ra_dump

ra_migrate

Description Performs any migration task (as necessary) between releases of Replication Agent.

Syntax ra_migrate

Parameters *None*

There are no parameters.

Usage

- After upgrading to a new release of Replication Agent, you must first run this command to update to the latest version of the Replication Agent.
- The ra_migrate command is valid when the Replication Agent instance is in *Admin* state.

ra_set_login

Description Sets the Replication Agent administrator login and password.

Syntax ra_set_login *username*, *password*

Parameters *username*

The login name of the Replication Agent administrator.

password

The password of the Replication Agent administrator.

Examples

```
ra_set_login bob3, bug3wag
```

This command sets the Replication Agent administrator login to "bob3" and the password to "bug3wag."

Usage

- The Replication Agent administrator login has permission to log in to the Replication Agent instance through the administration port.
- Only one Replication Agent administrator login name is valid at any time.
- Any change in the Replication Agent administrator login or password takes place immediately, and you must use the new login and password the next time you log in to the Replication Agent instance.
- The password specified for the administrator login is encrypted in the Replication Agent configuration file.
- The ra_set_login command is valid when the Replication Agent instance is in either *Admin* or *Replicating* state.

See also

ra_config

ra_statistics

Description

Returns performance-related statistics for Replication Agent components and the Java Virtual Machine (Java VM), or resets the statistics counters.

Note The statistics counters may vary by primary database.

Syntax

ra_statistics [*component*|reset]

Parameters

component

The optional keyword that identifies a Replication Agent component or the Java VM. Valid *component* keywords are:

LR – Log Reader
LTI – Log Transfer Interface
LTM – Log Transfer Manager
VM – Java Virtual Machine

reset

The optional keyword that resets the statistics counters.

Examples

Example 1

```
ra_statistics
```

This command returns performance statistics for the Replication Agent instance and the Java VM.

Example 2

```
ra_statistics reset
```

This command resets the statistics counters for the Replication Agent instance.

Usage

- If you invoke ra_statistics with no option, it returns statistics for all Replication Agent components and the Java VM.
- If you invoke ra_statistics with a *component* option, it returns statistics for the Log Transfer Manager component, as well as the component (or Java VM) you specify.
- Table 1-2 lists the statistics returned for the Java VM.

Table 1-2: Java VM statistics

Component	Statistic	Description
VM	VM maximum memory	Maximum memory (in bytes) available to the Java VM
VM	VM total memory allocated	Total memory (in bytes) allocated to the Java VM at start-up
VM	VM free memory	Memory (in bytes) allocated but not used by the Java VM
VM	VM memory usage	Memory (in bytes) allocated and in use by the Java VM
VM	VM % max memory used	Percentage of the maximum memory available to the Java VM, currently in use by the Java VM

- Table 1-3 lists the statistics returned for the Log Transfer Manager component.

Table 1-3: Log Transfer Manager statistics

Component	Statistic	Description
LTM	Time statistics obtained	Day, date, and time ra_statistics was invoked and information returned
LTM	Time replication last started	Day, date, and time that *Replicating* state was entered
LTM	Time statistics last reset	Day, date, and time that statistics counters were reset
LTM	Items held in Global LRUCache	Number of object references in the internal Least Recently Used cache

- Table 1-4 lists the Log Reader statistics for UDB.

Table 1-4: Log Reader statistics for UDB

Component	Statistic	Description
LR	Number of xlogs scanned	Number of operations read from log devices
LR	Average unprocessed operations per XLog scan	Average number of unprocessed operations for each XLog scan
LR	Average XLog scan time	Average XLog scan time for operations read from log devices
LR	Number of operations replicated	Number of operations that were successfully replicated
LR	Number of transactions replicated	Number of transactions that were successfully replicated
LR	Number of XLog operations skipped (maint_user, unmarked tables)	Number of XLog operations that were skipped
LR	Average wait time on empty XLog	Average time that the XLog was not in use
LR	Average PDB Service Time/Operations	Average service and operations time for each database
LR	Operation Queue Size	The queue size used for the operations
LR	Operation Data Hash Size	The data hash size for the operations
LR	Number of transactions truncated	Number of transactions that were truncated

- Table 1-5 lists the Log Reader statistics for Microsoft SQL Server.

Table 1-5: Log Reader statistics for Microsoft SQL Server

Component	Statistic	Description
LR	Total operations scanned	Number of operations read from log devices since last reset
LR	Total operations processed	Number of operations read from log devices and passed to LTI since last reset
LR	Total operations skipped	Number of operations read from log devices and not processed for any reason since last reset
LR	Total maintenance user operations filtered	Number of maintenance user operations read from log devices and skipped since last reset

Component	Statistic	Description
LR	Avg operation processing time	Average Log Reader operation processing time (in milliseconds) since last reset
LR	Total transactions processed	Number of transactions read from log devices since last reset
LR	Total transactions skipped	Number of transactions read from log devices and not processed for any reason since last reset
LR	Total transactions opened	Number of begin transaction commands read from log devices since last reset
LR	Total transactions closed	Number of commit and rollback commands read from log devices since last reset
LR	Total transactions committed	Number of commit commands read from log devices since last reset
LR	Total transactions aborted (rolled back)	Number of rollback commands read from log devices since last reset
LR	Total system transactions skipped	Number of system transactions read from log devices and skipped since last reset
LR	Avg operations per transaction	Average number of operations in each transaction read from log devices since last reset
LR	Current scan buffer size	Current size (in bytes) of the Log Reader scan buffer
LR	Current operation queue size	Current size (in bytes) of the Log Reader input queue
LR	Current session cache size	Current size (in bytes) of the session cache
LR	Log reposition point locator	Locator value of reposition point in log device
LR	Last processed operation locator	Locator value of most recently processed operation read from log devices
LR	Average xlog operation wait time (ms)	Average time (in milliseconds) that Log Reader had to wait for each new operation to appear in the log since last reset
LR	Average sender operation processing time (ms)	Average time (in milliseconds) that Log Reader sender took to process each operation since last reset
LR	Average sender operation wait time (ms)	Average time (in milliseconds) that Log Reader sender had to wait to send each processed operation to the LTI input queue since last reset

Component	Statistic	Description
LR	Average ChangeSet send time (ms)	Average time (in milliseconds) that Log Reader sender took to send each processed operation to the LTI input queue since last reset
LR	Total sender operations processed	Number of operations that Log Reader sender processed since last reset
LR	Marked objects cache size	Current marked objects cache size

- Table 1-6 lists the statistics returned for the Log Reader component for Oracle.

Table 1-6: Log Reader statistics for Oracle

Component	Statistic	Description
LR	Average RBA search time (ms)	The average record byte address (RBA) search time during log scanner positioning
LR	Total bytes read	The total number of bytes read from the primary database transaction log
LR	Total log records read	The total number of log records read from the primary database transaction log
LR	Average number of bytes per record	The average number of bytes per log record read
LR	Average time (ms) per log read	The average time per primary database transaction log read
LR	Total log read time (ms)	The total time spent reading the primary database transaction log
LR	Average number of bytes read per second	The average number of bytes read from the primary database transaction log per second
LR	Total archive log read time (ms)	The total time spent reading primary database transaction log archives
LR	Total log records queued	The total number of log records queued for processing
LR	Total log records filtered	The total number of log records filtered
LR	Total bytes queued	The total number of bytes queued
LR	Average number of bytes per queued record	The average number of bytes per queued record

Component	Statistic	Description
LR	Average number of checkpoints per records read	The average number of checkpoint log records per records read
LR	Total operations scanned	Number of operations read from log devices since last reset
LR	Total operations processed	Number of operations read from log devices and passed to LTI since last reset
LR	Total operations skipped	Number of operations read from log devices and not processed for any reason since last reset
LR	Total maintenance user operations filtered	Number of maintenance user operations read from log devices and skipped since last reset
LR	Avg operation processing time	Average Log Reader operation processing time (in milliseconds) since last reset
LR	Total transactions processed	Number of transactions read from log devices since last reset
LR	Total transactions skipped	Number of transactions read from log devices and not processed for any reason since last reset
LR	Total transactions opened	Number of begin transaction commands read from log devices since last reset
LR	Total transactions closed	Number of commit and rollback commands read from log devices since last reset
LR	Total transactions committed	Number of commit commands read from log devices since last reset
LR	Total transactions aborted (rolled back)	Number of rollback commands read from log devices since last reset
LR	Total system transactions skipped	Number of system transactions read from log devices and skipped since last reset
LR	Avg operations per transaction	Average number of operations in each transaction read from log devices since last reset
LR	Current scan buffer size	Current size (in bytes) of the Log Reader scan buffer
LR	Current operation queue size	Current size (in bytes) of the Log Reader input queue
LR	Current session cache size	Current size (in bytes) of the session cache

Component	Statistic	Description
LR	Log reposition point locator	Locator value of reposition point in log device
LR	Last processed operation locator	Locator value of most recently processed operation read from log devices
LR	Average xlog operation wait time (ms)	Average time (in milliseconds) that Log Reader had to wait for each new operation to appear in the log since last reset
LR	Average sender operation processing time (ms)	Average time (in milliseconds) that Log Reader sender took to process each operation since last reset
LR	Average sender operation wait time (ms)	Average time (in milliseconds) that Log Reader sender had to wait to send each processed operation to the LTI input queue since last reset
LR	Average ChangeSet send time (ms)	Average time (in milliseconds) that Log Reader sender took to send each processed operation to the LTI input queue since last reset
LR	Total sender operations processed	Number of operations that Log Reader sender processed since last reset
LR	Marked objects cache size	Current marked objects cache size

- Table 1-7 lists the statistics returned when the primary database is Oracle RAC.

Table 1-7: Log Transfer statistics for Oracle RAC

Component	Statistic	Description
LR	Log scan reader current LSN	The current log sequence number of the log being read for each cluster instance
LR	Log scan reader end-of-log status	The current end of log status for each cluster log scanner
LR	Log scan reader last read time	The number of seconds since the last read for each cluster scanner
LR	Log scan record set sizes	The current scan record set size for each cluster scanner
LR	Log scan checkpoint queue sizes	The current checkpoint queue size for each cluster scanner
LR	Log scan reader last record SCN	The SCN of the last log record read by each cluster scanner

Component	Statistic	Description
LR	Log scan reader checkpoints	The checkpoint SCN of the last checkpoint log record read by each cluster scanner
LR	Log scan checkpoint SCN	The current checkpoint SCN, based on all cluster scanners
LR	Log scan active checkpoint SCN	The active checkpoint SCN, based on all cluster scanner

- Table 1-8 lists the statistics returned for the Log Transfer Interface component.

Table 1-8: Log Transfer Interface statistics

Component	Statistic	Description
LTI	Number of LTL commands sent	Total number of LTL commands sent to Replication Server since last reset
LTI	Average LTL command size	Average size (in bytes) of each LTL command sent to Replication Server since last reset
LTI	Average LTL commands/sec	Average number of LTL commands sent per second to Replication Server since last reset
LTI	Total bytes sent	Number of bytes sent to Replication Server since last reset
LTI	Average Bytes/second during transmission	Average bytes per second sent over connection to Replication Server since last reset
LTI	Average Replication Server turnaround time	Average time (in milliseconds) it takes Replication Server to acknowledge each LTL command buffer sent since last reset
LTI	Average data arrival time	Average time (in milliseconds) LTI waits between receiving change sets from Log Reader since last reset
LTI	Average time to create distributes	Average time (in milliseconds) LTI takes to convert a change-set into LTL since last reset
LTI	Average LTL buffer cache time	Average time (in milliseconds) it takes between placing the LTL commands into the LTL buffer to the time it is actually sent to the Replication Server
LTI	Average LTL buffer size	Average size (in bytes) of each LTL buffer sent to Replication Server since last reset

Component	Statistic	Description
LTI	Average LTM buffer utilization (%)	Average utilization (in percentage of LTL buffer size) of each LTL buffer sent to Replication Server since last reset
LTI	Average LTL commands/buffer	Average number of LTL commands per buffer sent to Replication Server since last reset
LTI	Input queue size	Current number of change sets in the LTI input queue
LTI	Output queue size	Current number of distributes in the LTI output queue
LTI	Last QID sent	Hex value of most recent origin queue ID sent to Replication Server
LTI	Last transaction sent	Hex value of most recent transaction ID sent to Replication Server

- Statistics counters are reset automatically each time the Replication Agent instance goes into *Replicating* state.
- If you invoke ra_statistics with the reset keyword, Replication Agent immediately resets all of the statistics, except the following:
  - Time statistics obtained (LTM)
  - Time replication last started (LTM)
  - Time statistics last reset (LTM)
  - Last QID sent (LTI)
  - Last transaction ID sent (LTI)
  - All Java VM statistics

  Note All Java VM statistics are refreshed each time you invoke ra_statistics.

- The ra_statistics command is valid when the Replication Agent instance is in either Admin or Replicating state.

See also ra_status

ra_status

Description — Returns the current state of the Replication Agent instance.

Syntax — ra_status

Usage

- When ra_status is invoked, it returns the current state of the Replication Agent instance, and a brief description of the current state, as follows:

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

Note If the first word in the description is "Transitioning," the Replication Agent instance is in transition between states. Some commands are not valid when the Replication Agent instance is in state transition.

- Replication Agent states are:
  - *Admin* – in this state, the Replication Agent instance is running, but no connections are up. You can change any configuration parameter when the Replication Agent instance is in *Admin* state.
  - *Replicating* – in this state, the Log Reader component is scanning the transaction log for operations to replicate from the primary database. If there are operations to be replicated, the Log Transfer Interface component is sending LTL commands to the Replication Server.
  - *Replicating (Waiting at end of log)* – in this state, the Log Reader component has reached the end of the transaction log, the Replication Agent has finished processing all operations in the transaction log, and the Log Transfer Interface component has successfully sent LTL commands for all replicated operations to the Replication Server.

    If the primary database is not quiesced or is otherwise inactive, transactions could arrive in the log immediately after the state is returned, so even though the state is returned as *Replicating (Waiting at end of log)*, the Replication Agent could actually be in *Replicating* state and processing log records.

  See the *Replication Agent Administration Guide* for more information about Replication Agent states.
- The ra_status command is valid when the Replication Agent instance is in either *Admin* or *Replicating* state.

See also — quiesce, ra_statistics, resume, shutdown, suspend

ra_truncatearticles

Description

Note This command is available only for Oracle and Microsoft SQL Server.

Truncates unused articles in the RASD.

Syntax

ra_truncatearticles *locator*

Parameters

locator

The log locator value (LTM Locator) that identifies the cutoff point for truncating older versions of articles from the system data repository.

Usage

- When ra_truncatearticles is invoked, it truncates all non-current versions of all primary database articles in the system data repository older than the version identified by the *locator* value.

  If the current (most recent) version of an article is older than the version identified by the *locator* value, it is not truncated.

- Most common DDL commands and stored procedures executed in the primary database (such as alter table) are recorded in the transaction log, and replicated to the standby database. When it processes those DDL transactions for replication, Replication Agent updates its RASD automatically, creating a new version of the affected primary database articles.

  Use ra_truncatearticles as part of a periodic maintenance procedure to prevent the RASD from growing indefinitely. See the *Replication Agent Administration Guide* for more information.

  Note Be sure to back up the RASD using rasd_backup before you truncate it.

- The ra_truncatearticles command is valid when the Replication Agent instance is in either *Admin* or *Replicating* state.

See also

ra_truncateusers

ra_truncateusers

Description

Note This command is available only for Oracle and Microsoft SQL Server.

Truncates older versions of primary database users in the system data repository in the RASD.

Syntax

ra_truncateusers *locator*

Parameters

locator

The log locator value (LTM Locator) that identifies the cutoff point for truncating older versions of database users from the system data repository.

Usage

- When ra_truncateusers is invoked, it truncates all non-current versions of all primary database users in the system data repository older than the version identified by the *locator* value.

  If the current (most recent) version of a user is older than the version identified by the *locator* value, it is not truncated.

- The ra_truncateusers command is valid when the Replication Agent instance is in either *Admin* or *Replicating* state.

See also

ra_truncatearticles

ra_updatedevices

Description

Note This command is available only for Oracle and Microsoft SQL Server.

Updates information about primary database log devices in the RASD.

Syntax

ra_updatedevices

Usage

- For Oracle, when ASM manages the redo log files and the disk group is changed by either adding or dropping disks, you must invoke the ra_updatedevices command to be sure the log device repository is updated with correct ASM storage information.
- When ra_updatedevices is invoked, Replication Agent does the following:
  - Refreshes the archive log information
  - Deletes all of the data in its log device repository

    Note If the device location is set, it is not overwritten.

  - Queries the primary database for information about all of its log devices

    - Re-populates the log device repository in the RASD with current information about primary database log devices returned by the primary database

- If any log device associated with the primary database is added, dropped, extended, or moved at the primary data server, you must:

    - Stop replication (using quiesce or suspend) to put the Replication Agent instance in *Admin* state
    - Invoke ra_updatedevices to update the log device repository in the RASD

    See the *Replication Agent Administration Guide* for more information.

    Note The primary database need not be quiesced when you update the log device repository.

- If the primary data server writes to a new (or altered) log device before you update the log device repository, the Replication Agent instance will stop replication processing and go to *Admin* state.

    Sybase recommends that you coordinate all log device changes at the primary database with updating the Replication Agent log device repository.

- Because Replication Agent re-creates the entire log device repository when you invoke ra_updatedevices, any log device path that you modified previously (using ra_devicepath) is overwritten with the current log device information from the primary database.

    For example:

    ```
    ID=1 serverpath=/dev1 mirror=/dev1a
    ```

    becomes the following when you change the `server path` to "`dev44`":

    ```
    ID=1 serverpath=/dev44 mirror=/dev1a
    ```

    Note If you need to alter the "default" path for a log device (that is, the log device path returned by the primary database), you must use the ra_devicepath command *after* you invoke ra_updatedevices.

- For each log device recorded in the RASD, you can set or change the disk device path with the ra_devicepath command.

If you do not specify a disk device path (using ra_devicepath), the value recorded for the disk device path is DEFAULT, and Replication Agent uses the value recorded for the server device path to find the log device.

- The ra_updatedevices command is valid only when the Replication Agent instance is in *Admin* state.
- *(For Oracle*) The Replication Agent uses the disk map file, to create mirror log devices, when log devices are created during xlog initialization and when devices are updated using the ra_updatedevices command. When Replication Agent is in the replicating state, it reads data from the mirrored disks specified in the map file.
  - The ra_updatedevices command updates the ASM disk map file. When executed the ASM disk map file is updated as follows:
    - ASM is queried to see if the disk groups required to read any redo logs have changed. If new disks have been added to any of the ASM disk groups, a default mirror entry is added in the ASM disk map file for the new disk.
    - The ASM disk group specified by the archive log path parameter is checked for new disks as well as the ASM disk group for each online redo log.
    - The ASM disk map file is updated before the log devices are updated to ensure any ASM disk path changes are included in updated log devices.
    - If new disk entries are added to the ASM disk map file, the log devices will not be updated in the repository. A message is returned to the user indicating that new entries are in the file that may need to be changed before devices are updated. The next time the ra_updatedevices command is executed, the log devices will be updated.
  - The Replication Agent command ra_helpdevice provides device information for the log device status where the device is physically located and if it is being mirrored to another device. The physical information is a simple path to a file or raw device. ASM uses disk groups with potentially many disks, the physical information for ASM devices is provided to show all disks required for the device. There will be one row output for each disk in the group where the device is stored.

See also ra_devicepath, ra_helpdevice

ra_version

Description — Returns the version of the Replication Agent instance, the host operating system version, and the JRE version.

Syntax — ra_version

Usage — When ra_version is invoked, it returns the Replication Agent version string in a row, as follows:

```
Sybase Replication Agent for Unix &
Windows/15.2.0.5602/P/generic/JDK
5.0/main/5602/VM: Sun Microsystems Inc.
1.5.0_13/OPT/Fri Nov 30 03:42:03 MST 2009
```

See also — pdb_version, ra_status, ra_version_all

ra_version_all

Description — Returns the name, type, and version of the Replication Agent instance, and version information for the primary data server, primary Replication Server, and communications drivers.

Syntax — ra_version_all

Note (For Oracle only) When the Replication Agent is configured to connect to ASM (an asm_tns_connection is configured with a non-null value that is not the default value), ra_version_all includes an additional line of output that describes the version of ASM being connected to. When asm_tns_connection is not configured, no ASM entry is listed in ra_version_all output.

Usage

- When ra_version_all is invoked, it returns the following information:

```
Component            Version
-------------------------------------------------------------------------
Instance:            mro2 - Oracle
RepAgent:            Sybase Replication Agent for Unix &
                     Windows/15.2.0.5700/P/generic/JDK
                     5.0/main/5700/VM: Sun Microsystems Inc.
                     1.5.0_13/OPT/Tue Jul 15 03:03:03 MST 2009
JRE:                 Sun Microsystems Inc. Java(TM) 2 Runtime
                     Environment, Standard Edition/1.5.0_13-b05/SunOS
```

```
                    5.8/sparc/32

RASD:               Sybase SQL Anywhere/9.0.2.3302/WindowsXP

Primary Data Server:Oracle Oracle9i Enterprise Edition Release
                     10.2.0.1.0 - Production With the Partitioning, OLAP
                    and ORACLE Data Mining options JServer
                    Release 10.2.0.1.0 - Production

PDS JDBC Driver:    Oracle JDBC driver 10.2.0.1.0

RepServer:          Replication Server/15.2
                    ESD#5/NT (IX86)/Windows 2000
                    /1/OPT/Fri Jan 01 14:01:20 2009

RSSD:               Sybase SQL Anywhere/8.0.2.4322/WindowsXP

Sybase JDBC Driver:jConnect (TM) for JDBC(TM)/6.05(Build
                    26169)/P/EBF15181/JDK14/Wed Oct 24 1:06:46 2007
```

See also pdb_version, ra_status, ra_version

rasd_backup

Description

Note This command is available only for Oracle and Microsoft SQL Server.

Backs up the Replication Agent System Database (RASD).

Syntax

rasd_backup

Usage

- When rasd_backup is invoked, it starts the database backup process for the RASD.

  Note Sybase recommends that you always back up the RASD before you truncate using ra_truncatearticles or ra_truncateusers.

- Replication Agent places RASD backup files in the directory identified by the rasd_backup_dir configuration parameter.

  When you create a Replication Agent instance, a RASD backup directory is created automatically as part of the instance directory structure. The default value of the rasd_backup_dir parameter points to that directory.

- The rasd_backup command is valid when the Replication Agent instance is in *Admin* or *Replicating* state.

See also rasd_restore, ra_truncatearticles, ra_truncateusers

rasd_helpbackup

Description **Note** This command is available only for Oracle and Microsoft SQL Server.

Displays a list of RASD backups.

Syntax rasd_helpbackup

Usage
- When rasd_helpbackup is invoked, it displays a list of RASD backups stored in the directory, identified by the rasd_backup_dir configuration parameter.
- The backups are named using the date and time the backup was created.

See also rasd_restore, rasd_backup, rasd_removebackup

rasd_removebackup

Description **Note** This command is available only for Oracle and Microsoft SQL Server.

Removes RASD backups.

Syntax rasd_removebackup [*backup_name* | *all*]

Parameters
backup_name
The name of the backup that you are removing.

all
A keyword that allows you to remove all RASD backups.

Examples **Example 1**

```
rasd_removebackup 2008-07-24_15.41.10

This command causes the backup named
2008-07-24_15.41.10 to be removed

rasd_removebackup 2008-07-24_15.41.10
go
```

```
RASD Backups removed
--------------------
2008-07-24_15.41.10
(1 row affected)
```

Example 2

```
rasd_removebackup all
This command causes all of the backups to be removed.

rasd_removebackup all
2> go

RASD Backups removed
--------------------
2008-07-28_10.08.27
2008-07-28_10.09.29
2008-07-28_10.11.31
2008-07-28_10.20.55
(4 rows affected)
```

Usage

- When rasd_removebackup is invoked, it removes a RASD backup that is stored in the directory identified by the rasd_backup_dir configuration parameter.
- When rasd_removebackup is invoked with the all keyword, all RASD backups that are stored in the directory identified by the rasd_backup_dir configuration parameter are removed.
- When rasd_backup is invoked, the names of all backups removed appear.
- The backups are named using the date and time the backup was created.

See also

rasd_restore, rasd_backup, rasd_helpbackup

rasd_restore

Description

Note This command is available only for Oracle and Microsoft SQL Server.

Allows you to restore the RASD.

Syntax

rasd_restore [*backup_name]*

Note After executing rasd_restore, Replication Agent automatically shuts down if rasd_restore successful.

Parameters

backup_name

The name of the backup that you are restoring from. If you omit the backup name, the most recent backup is restored.

Examples

Example 1

```
rasd_restore
```

This command with no parameters restores the RASD from the most recent backup.

Example 2

```
rasd_restore 2008-07-24_15.41.10
```

This command restores the RASD from the 2008-07-24_15.41.10 backup.

Usage

- When rasd_restore is invoked, it starts the restore process for the RASD.
- When no parameters are used. Replication Agent looks for the most recent RASD backup files in the directory identified by the rasd_backup_dir configuration parameter.
- If a backup name is provided as a parameter, Replication Agent restores from the specified backup in the directory identified by the rasd_backup_dir configuration parameter.
- When you create a Replication Agent instance, an RASD backup directory is automatically created as part of the instance directory structure. The default value of the rasd_backup_dir parameter points to that directory.
- If you invoke rasd_restore when the Replication Agent instance is in *Replicating* state, it returns an error.
- The rasd_restore command is valid only when the Replication Agent instance is in *Admin* state.

See also

rasd_backup, rasd_helpbackup, rasd_removebackup

rasd_trunc_schedule

Description

Note This command is available only for Oracle and Microsoft SQL Server.

Manages a truncation schedule. rasd_trunc_schedule returns a list of the repository truncation schedule, and can also add and remove a specific schedule.

Syntax

rasd_trunc_schedule [add | remove | clear | force] schedule

Parameters

schedule

The day and time string in the form of restricted UNIX cron style that indicates the time automatic repository truncation is to be performed.

The following is a valid schedule string in UNIX cron style format:

```
[mm][HH][DOM][MON][DOW]
```

where:

- mm – is the minutes past the hour.
- HH – is the hour in 24-hour notation.
- DOM – represents the days of the month, 2-digit number between 1 and 31, which represents the day of the month.
- MON – represents the month of the year, abbreviated in 3-character format, such as “Jan”, “Feb”, and so on, or a 2-digit number between 1 and 12, which represents the month in a year from January to December.
- DOW – represents the day of the week, abbreviated in 3-character format, such as “Sun”, “Sat”, and so on, or a 2-digit number between 1 and 7, which represents the day in a week from Sunday to Saturday.
- Use an asterisk to match any valid value in a specific schedule field, ([mm],[HH],[DOM],[MON],[DOW]):
  - For example, “1720*** represents a daily schedule at 8:17 p.m.
  - When both the DOW and DOM are specified, the schedule represents two days that match either DOW or DOM.

    For example, “*1216*Mon” represents 12:00 a.m. every Monday or 12:00 a.m. the 16th of every month.
  - Multiple entries can be provided using a semicolon.

    For example, “*1216*Mon” or “1720***;*1216*Mon”.

- Do not leave spaces between fields; otherwise, the schedule is rejected as an invalid schedule format.

  For example,“* 12 16 *Mon” is as an invalid schedule.

clear

To remove all repository truncation schedules, enter:

```
rasd_trunc_schedule clear
```

When the repository truncation schedule list is not set or empty, repository truncation by schedule is disabled.

force

To perform an immediate repository truncation manually, regardless of the automatic truncation time schedule, enter:

```
rasd_trunc_schedule force
```

Examples

Example 1

```
rasd_trunc_schedule
```

This command returns a list of all repository truncation schedule times when repository truncation occurs.

Example 2

```
rasd_trunc_schedule add, 1720***
```

This command adds daily repository truncation schedule at 8:17 PM to the schedule list.

Example 3

```
rasd_trunc_schedule remove, 1720***
```

This command removes the daily repository truncation schedule at 8:17 PM from the schedule list.

Example 4

```
rasd_trunc_schedule clear
```

This command clears all repository truncation schedules that have been set.

Example 5

```
rasd_trunc_schedule force
```

This command truncates the repository immediately, regardless of the existence of any truncation schedule.

Usage

- When rasd_trunc_schedule is invoked, its function is determined by the keywords and options you specify.
- When you specify multiple keywords and options, separate each must using a comma. A blank space before or after a comma is optional.

  For example:

  ```
  rasd_trunc_schedule add, 1720***
  ```

  .

See also ra_truncatearticles, ra_truncateusers

resume

Description Starts replication processing in the Replication Agent instance.

Syntax resume

Usage

- When resume is invoked, the Replication Agent instance attempts to go to *Replicating* state and start replication operations, as follows:
  - Replication Agent attempts to open network connections to the primary database, primary Replication Server, and RSSD.

    If it fails to establish a connection, the Replication Agent logs a warning message in its system log, and it attempts to retry the connection, based on its configuration parameters for the connection.
  - If the Replication Agent cannot establish a connection to the primary database after exhausting its configured retry attempts, it aborts all subsequent resume processing, returns to *Admin* state, and logs the error.
  - Replication Agent requests the current LTM Locator value from the primary Replication Server, and it stores the value in the Replication Agent transaction log.
  - The Log Reader component begins scanning the transaction log, looking for operations to be replicated. Log Reader begins scanning the log at the point identified by the LTM Locator value.
  - When it finds transactions to replicate, Log Reader passes them (as change-set data) to the input queue of the Log Transfer Interface component.

    - The Log Transfer Interface component reads the change-set data from its input queue, generates LTL commands, and places the LTL commands in its output queue for transmission to the Replication Server.
- If any start-up operation fails, the Replication Agent instance returns to *Admin* state, and it logs the error.
- If the resume command is successful, the Replication Agent instance goes to *Replicating* state. To determine the current state of the Replication Agent, use the ra_status command.
- The resume command returns an error under any of the following conditions:
    - The Replication Agent instance is already in *Replicating* state.
    - The system data repository in the RASD does not exist or is not initialized (Oracle and Microsoft SQL Server).
    - The Replication Agent connection configuration parameters are not set correctly, or it fails otherwise to connect with the primary database or the primary Replication Server.
    - The database connection for the primary database is not defined correctly in the primary Replication Server.
- If the resume command is successful, the Replication Agent instance goes into *Replicating* state.
- The resume command is valid only when the Replication Agent instance is in *Admin* state.

See also quiesce, ra_status, shutdown, suspend

rs_create_repdef

Description **Note** This command is available only for Oracle and Microsoft SQL Server.

Creates a replication definition at Replication Server for a marked table and procedure, or for all marked tables and procedures.

Note Replication Agent 15.2 is pre-configured to match replication definition datatypes available in Replication Server 15.0 and later. If replication definitions are to be generated against an earlier version of Replication Server, this configuration needs to be changed. Contact Sybase Technical Support for assistance in making this adjustment.

Syntax

rs_create_repdef {all | *name*}

Parameters

all

A replication definition is created for all tables and procedures that are marked for replication.

name

A replication definition is created for the table or procedure specified by *name*.

Note rs_create_repdef always assumes that a database replication definition exists for the primary database.

Usage

- When a table is marked for replication and the owner mode is set to on, the replication definition created by rs_create_repdef includes the owner name as part of the table name for a table replication definition in the “with primary table named” clause.
- This command always assumes that a database replication definition exists for the primary database. All replication definitions created by this command include the send standby clause, which means the replication definition will only be used by Replication Server if there is already a database level replication definition. The replication definition created by rs_create_repdef can not be individually subscribed to. If you do not wish to have a database level replication definition, you must use a different tool, or create replication definitions manually, and not use rs_create_repdef.
- Replication definitions created by rs_create_repdef always define the datatypes using available user defined datatypes that are installed in Replication Server. This means that customers using rs_create_repdef should not set Replication Agent configuration parameter pdb_convert_datetime to true, as doing so converts date and timestamp datatypes to Sybase format, instead of UDD format.

- Using the Replication Agent configuration parameter pdb_auto_create_repdefs has the same result as executing rs_create_repdef.
- When rs_create_repdef is invoked and the parameter "all" or "ALL" is entered, a replication definition is created for all tables or procedures that are marked for replication.
- When rs_create_repdef is invoked and the name of a table or procedure that is marked for replication is entered, a replication definition is created only for that table or procedure.
- For each table or procedure for which a replication definition create is attempted, a result set is returned. The result set contains the replication definition name and status of the create. If the replication definition was created, the status will be "created." If an error occurred, an error message from Replication Server will be returned.
- The character case of the object names in the replication definition will be set according to the ltl_character_case setting.
- The following applies to replication definition table and procedure names:
  - All non-alphanumeric characters and spaces are removed and are not part of the table or procedure name.
  - Underscores are kept as part of the name even though they are non-alphanumeric characters.
  - Periods are replaced with underscores.
- Replication definition names for tables always begin with the prefix "*ra$*," followed by a unique alphanumeric identifier (maximum of 8 characters), and ending with a table or object name. For example, for a replicate name of "My Table," the resulting repdef name is "ra$0x7952_mytable." For an especially long replicate name of "mytable8901234567890123456789 0" (30 characters), the resulting repdef name is "ra$0x7952_mytable8901234567890" (30 characters maximum).

See also rs_drop_repdef

rs_drop_repdef

Description **Note** This command is available only for Oracle and Microsoft SQL Server.

A replication definition at the configured Replication Server for a table and procedure is dropped.

Syntax

rs_drop_repdef *name*

Parameters

name

A replication definition is dropped for that table or procedure.

Usage

- When rs_drop_repdef is invoked, a replication definition for that table is dropped at the Replication Server.
- When rs_drop_repdef is invoked and the name of a table or procedure that is marked for replication is entered, a replication definition is created for that table or procedure.
- For each table or procedure for which a replication definition is dropped, a result set is returned. The result set contains the table name and status of the create. If the replication definition was created, the status will be "dropped." If an error occurred, an error message from Replication Server will be returned.
- The character case of the object names in the replication definition will be set according to the ltl_character_case setting.
- The following applies to replication definition table and procedure names:
  - All non-alphanumeric characters and spaces are removed and are not part of the table or procedure name.
  - Underscores are kept as part of the name even though they are non-alphanumeric characters.
  - Periods are replaced with underscores.
- Replication definition names for tables always begin with the prefix "*ra$*," followed by a unique alphanumeric identifier (maximum of 8 characters), and ending with a table or object name. For example, for a replicate name of "My Table," the resulting replication definition name is "ra$0x7952_mytable." For an especially long replicate name of "mytable8901234567890123456789O" (30 characters), the resulting replication definition name is "ra$0x7952_mytable8901234567890" (30 characters maximum).

See also

rs_create_repdef

rs_ticket

Description

Supports Replication Server rs_ticket processing by placing an rs_ticket marker in the primary database transaction log. This command was created in support of the Replication Server rs_ticket feature.

Syntax

rs_ticket *H1* [, *H2*[, *H3* [, *H4*]]]

Parameters

H1, *H2*, *H3*

Each parameter contains from 1-10 characters. It is free form and is to be used as an identifier.

H4

It contains from 1-50 characters. It is free form and is also to be used as an identifier.

Examples

Example 1

The following executes rs_ticket and monitors the processing time for the record identified by the four parameters (only one parameter is required):

```
rs_ticket test1, 1221, appxyz.monitoring_system
```

Note The parameters are optional, and can be used to identify or differentiate executions of rs_ticket.

It can be used independently or grouped with additional executions to allow processing times to be compared.

In this example, the following information will be sent to Replication Server.

```
rs_ticket 'V=1;H1=test1;H2=1221;H3=appxyz;
H4=monitoring_system;PDB(name)=hh:mm:ss.ddd'
```

where “name” is the name of the primary database.

When this command reaches the replicate database, Replication Server will add additional time values for the EXEC, DIST and DSI components of Replication Server. The final result seen by the replicate database will look similar to:

```
rs_ticket 'V=1;H1=test1;H2=1221;H3=appxyz;
H4=monitoring_system;PDB(name)=hh:mm:ss.ddd;
EXEC=hh:mm:ss.ddd;DIST=hh:mm:ss.ddd;
DSI(name)=hh:mm:ss.ddd;RDB(name)=hh:mm:ss.ddd'
```

You can use the information provided to monitor replication latency and performance. By using different or descriptive H1-H4 parameters, users can more easily identify which rs_ticket data matches the activity or timing of the command when entered at the primary database.

Example 2

To measure performance of a batch of work, you can surround the work with rs_ticket executions, similar to the following sequence:

(Execute in Replication Agent)

```
rs_ticket start
```

(Execute in primary data server)

```
execute replication benchmarks
```

(Execute in Replication Agent)

```
rs_ticket stop
```

Usage

- The Replication Server EXEC, DIST, and DSI modules parse and process rs_ticket subcommands.
- There are no subscriptions for rs_ticket. DIST does not send rs_ticket to DSI unless there is at least one subscription from the replicate site.

For more information regarding the rs_ticket usage in Replication Server, refer to the Replication Server documentation.

shutdown

Description — Shuts down the Replication Agent instance, terminating its process.

Syntax — shutdown [immediate]

Parameters

immediate
The optional keyword that shuts down the Replication Agent instance immediately.

Usage

- When shutdown is invoked with no option, the Replication Agent starts a normal (graceful) shutdown.

  In a normal shutdown, the Replication Agent first quiesces, and then the process terminates. See quiesce on page 69 for more information about quiescing the Replication Agent.

- When shutdown is invoked with the immediate keyword, the Replication Agent starts an immediate shutdown.

  In an immediate shutdown, the Replication Agent:

  - Stops all of its replication processing, without regard to transactions in process or in transit
  - Drops all of its connections
  - Terminates the application process
- The shutdown command with the immediate keyword is valid at any time, when the Replication Agent instance is in any state, including transition between states.
- The shutdown command with no keyword (normal shutdown) is valid when the Replication Agent instance is in either *Admin* or *Replicating* state, but not in state transition.

See also quiesce, ra_status, resume, suspend

suspend

Description Stops all current replication processing and puts the Replication Agent instance into *Admin* state.

Syntax suspend

Usage

- When suspend is invoked, it stops all current replication processing in the Replication Agent instance.
  - The Log Reader component stops scanning the transaction log immediately, and the Log Transfer Interface component stops sending LTL to the Replication Server immediately.
  - Any data in the Replication Agent internal queues (input and output queues of the Log Reader and Log Transfer Interface components) is removed without further processing.
  - The Replication Agent instance immediately releases all of its connections to the primary database, and drops its connection to the primary Replication Server (and RSSD, if connected).

- The Replication Agent instance goes from *Replicating* state to *Admin* state.

Note The action of the quiesce command is similar to that of the suspend command, except that quiesce allows pending transactions in the Replication Agent internal queues to be processed first, before putting the Replication Agent instance in *Admin* state.

- If the Replication Agent instance is in *Admin* state, the suspend command returns an error.
- The suspend command is valid only when the Replication Agent instance is in *Replicating* state.

See also quiesce, ra_status, resume, shutdown

test_connection

Description Tests Replication Agent connection configurations and network connectivity.

Syntax test_connection [*conn_name*]

Parameters *conn_name*

The keyword for a Replication Agent connection to be tested. Valid keywords are:

- PDS – primary data server
- RS – primary Replication Server (and RSSD, if so configured)

Note If the value of the use_rssd 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_rssd configuration parameter is false, the test_connection command does *not* test Replication Agent connectivity to the RSSD.

Examples **Example 1**

```
test_connection
```

This command tests all Replication Agent connections, including the primary data server connection, the primary Replication Server connection, and the RSSD connection (if so configured).

Example 2

```
test_connection PDS
```

This command tests only the Replication Agent connection for the primary data server.

Usage

- When test_connection is invoked with no option, Replication Agent tests all of its connections by attempting to log in to the corresponding server for each connection, using the connection parameters stored in its configuration file.
- When test_connection is invoked with either the RS or PDS keyword, Replication Agent tests the specified connection.
- The test_connection command verifies both network connectivity and the following Replication Agent connection configuration parameters for the primary database:
  - connection type (connectivity driver and protocol) – pds_connection_type
  - database name – pds_database_name
  - data server name – pds_server_name
  - Data source name (ODBC drivers only) – pds_datasource_name
  - host machine name – pds_host_name
  - port number – pds_port_number
  - user login access – pds_password and pds_username

  Note 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 verifies both network connectivity and the following Replication Agent connection configuration parameters for the primary Replication Server (and RSSD, if so configured):
  - Database name – rssd_database_name (RSSD only)
  - Replication Server data source (as specified in the Replication Server primary database connection) – rs_source_db and rs_source_ds (Replication Server only)
  - Host machine name – rs_host_name (and rssd_host_name)

- Network packet size – rs_packet_size (Replication Server only)
- Port number – rs_port_number (and rssd_port_number)
- User login access – rs_password, rs_username (and rssd_password and rssd_username)

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

- The test_connection command returns the connection type and its status, as follows:

```
Type Connection
---- ----------
PDS  succeeded
RS   succeeded

(2 rows affected)
```

  If the connection status is failed, it indicates one of the following:

  - The Replication Agent connection configuration parameters are not set correctly.
  - A network failure or communication error prevents the connection.
  - The server associated with the connection is down.
- If the connection status is failed, check the Replication Agent system log to determine the cause of the failure.

Note 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 *Replication Agent Administration Guide* for information about setting up Replication Agent connection configuration parameters.
- See Chapter 2, “Configuration Parameters,” for information about specific connection configuration parameters.
- The test_connection command is valid when the Replication Agent instance is in either *Admin* or *Replicating* state.

See also ra_config, ra_statistics, ra_status

trace

Description

Returns current trace flag settings, or changes trace flag settings for the Replication Agent instance.

Syntax

trace [*flag*|all, *switch*]

Parameters

flag

The name of the trace flag to change the setting for.

all

A keyword that allows you to apply a switch value to all of the trace flags at once.

switch

A Boolean (true or false) value that enables or disables tracing for the trace point identified in the *flag* option.

Usage

- The trace command is intended for use by Sybase Technical Support engineers when troubleshooting Replication Agent.
- When trace is invoked with no option, it returns the current settings for all Replication Agent trace flags.
- When trace is invoked with the *flag* and *switch* options, it changes the setting of the trace flag identified, and it returns the current (new) setting for the trace flag.
- When trace is invoked with the all keyword and a *switch* option, it sets all Replication Agent trace flags to the value specified in the *switch* option, and it returns the current (new) setting for all of the trace flags.
- Changes made with the trace command take effect immediately.
- When a trace flag is set to true, tracing is enabled for the trace points identified by the flag. When set to false, tracing is disabled for the trace points.
- Output from all trace points (except *LTITRACELTL*) is sent to the Replication Agent system log file. Use the log_system_name command to find the name and path of the Replication Agent system log file.

- Output from the *LTITRACELTL* trace point is sent to a separate trace output file named *LTITRACELTL.log*. To view the contents of the *LTITRACELTL.log* file, your file viewer must be capable of handling very long lines.

  Note The *LTITRACELTL.log* file contains a human-readable representation of the LTL, not the actual LTL commands sent to the primary Replication Server.

- Table 1-9 lists Replication Agent trace flags:

Table 1-9: Replication Agent trace flags

Trace flag	Description
BMGRTRACE	When set to "true," this flag enables Bean Management event tracing.
CACHETRC	When set to "true," this flag enables tracing of internal cache events.
DBCONTEXT	When set to "true," this flag turns on tracing of database context events.
LATRC	When set to "true," this flag traces general Log Administrator operations.
LATRCSQL	When set to "true," this flag traces SQL conversations between Log Administrator and the primary database.
LICTRACE	When set to "true," this flag traces feature license check-in/checkout events.
LOGREADTRC	When set to "true", turns on trace of database log reading.
LRTRACE	When set to "true," this flag traces general execution of the Log Reader component.
LTITRACE	When set to "true," this trace flag enables tracing operations of the Log Transfer Interface component.
LTITRACELTL	When set to "true," this trace flag enables LTL statement tracing in the *LTITRACELTL.log* file.
LTMCI	When set to "true," causes tracing of LTM component interface invocations and LTM invocations of other components' interfaces.
LTLFMTTRC	When set to "true," this trace flag enables tracing of the LTL formatter.
LTMHL	When set to "true," causes highlights in the LTM execution path to be noted.
LTMSC	When set to "true," causes tracing of all Replication Agent state changes.

Trace flag	Description
RACONTRC	When set to “true,” causes tracing of connection and query execution.
RACONTRCSQL	When set to “true,” causes tracing of SQL statements to be executed.
RASDTRC	(For Oracle and Microsoft SQL Server) When set to “true,” turns on tracing of Replication Agent System Data Repository events.
RATRACE	When set to “true,” causes tracing of Replication Agent events.
STMTRACE	When set to “true,” causes tracing of LTM state monitor events.
THREADTRC	When set to “true,” logs ThreadPool trace events.

- You *cannot* change the settings of SYSTEM trace flags.

  Table 1-10 lists Replication Agent SYSTEM trace flags:

Table 1-10: Replication Agent SYSTEM trace flags

Trace flag	Description
CONFIG	Configuration change event logged.
ERROR	Serious error; manual intervention may be needed to recover.
FATAL	Critical error; application shut down; manual intervention required to recover.
INFORMATION	Information only; no action required.
WARNING	Minor error; operation not affected, or problem is recoverable.

- The trace command is valid when the Replication Agent instance is in either *Admin* or *Replicating* state.

See also log_system_name

CHAPTER 2

Configuration Parameters

This chapter describes the Replication Agent configuration file and configuration parameters.

Configuration parameter overview

Configuration parameters record the user-configurable settings that control how a Replication Agent instance operates. The current values of all configuration parameters are stored in the *configuration* file of each Replication Agent instance.

Replication Agent configuration file

The configuration file is created automatically when you create a Replication Agent instance. It resides in the instance subdirectory, under the Replication Agent base directory.

The configuration file is named after the Replication Agent instance, with the extension *.cfg* (for example, if the instance is named "my_ra," the configuration file is *my_ra.cfg*).

Each time a Replication Agent instance starts up, it reads the configuration file to get the configuration information it needs to run. After start-up, the only time the Replication Agent accesses the configuration file is when the ra_config or ra_set_login command is invoked to change the value of a configuration parameter.

When the value of a configuration parameter is changed, Replication Agent saves the new value, overwriting the entire configuration file.

Configuration file format

The configuration file is a flat ASCII file that contains configuration information for a single Replication Agent instance.

The first two lines in the configuration file identify the file as a Replication Agent configuration file and record the time that the file was last modified. For example:

```
#RAX Property File
#Fri Jan 12 07:33:18 MST 2008
```

Each configuration parameter name appears on a separate line, followed by the equal symbol (=) and the current value of the parameter. For example:

```
compress_ltl_syntax=true
```

If the Replication Agent instance is not running, you can view the configuration file to examine the current Replication Agent configuration.

Note Sybase recommends that you do *not* edit the configuration file, because Replication Agent overwrites the entire configuration file every time the ra_config or ra_set_login command is invoked to change a parameter value.

If the Replication Agent instance is running, use the ra_config command to view the current Replication Agent configuration.

Changing configuration parameters

To view, set, or change the current value of a Replication Agent configuration parameter, use the ra_config command.

To change the current Replication Agent administrator login (ltm_admin_user) or administrator password (ltm_admin_pw), you must use the ra_set_login command.

Note You cannot use the ltm_admin_user and ltm_admin_pw parameters; additionally, they do not appear in the parameter list returned by ra_config.

See Chapter 1, “Command Reference,” for more information about using the ra_config and ra_set_login commands.

Copying a Replication Agent configuration

When you create a new Replication Agent instance with the ra_admin utility, you can specify the new instance to use the same configuration parameter values as an existing Replication Agent instance.

Note When you copy an existing configuration instance when creating a new Replication Agent instance, certain configuration parameter values are not copied to the new configuration. See the *Replication Agent Administration Guide* for more information.

If you do not copy an existing configuration when you create a new Replication Agent instance, the ra_admin utility creates a default configuration file, with default values for all configuration parameters.

Configuration parameter reference

Table 2-1 lists all of the Replication Agent configuration parameters and a brief description of each parameter. The target column indicates when the parameter is used for "All" targets or specific targets; Oracle, Microsoft SQL Server (MSSQL), or DB2 UDB (UDB).

Table 2-1: Replication Agent configuration parameters

Parameter name	Target	Description
admin_port	All	Port number that the Replication Agent will use to listen for administrative connections.
archive_read_block_count	Oracle	Determines the number of blocks or pages that are read at a time from the primary database archive log.
asm_password	Oracle	Password for Oracle Automatic Storage Management (ASM) access for the user specified in the asm_username.
asm_tns_connection	Oracle	Identifies the Oracle ASM connection name found in the *tnsnames.ora* file.
asm_tns_filename	Oracle	Identifies the Oracle *tnsnames.ora* filename where the ASM connection information is located.
asm_username	Oracle	Identifies the Oracle user name to be used when connecting to the ASM server.
auto_adjust_block_count	Oracle MSSQL	Determines whether or not the Replication Agent should automatically adjust the log_read_block_count parameter if any part of a read is discarded
column_compression	All	Use minimal column information in LTL.
compress_ltl_syntax	All	Use abbreviated LTL syntax.
connect_to_rs	All	Enable/disable connection from LTI to Replication Server.
ddl_password	Oracle MSSQL	Password for ddl_username.
ddl_username	Oracle MSSQL	The database user name included in LTL for replicating DDL commands to the replicate database.
dump_batch_timeout	All	Number of seconds to wait before sending an incomplete LTL buffer to Replication Server.
filter_maint_userid	All	Log Reader filters operations with maintenance user ID.
function_password	Oracle MSSQL	Password for user ID passed in LTL with replicated stored procedure invocations.

Parameter name	Target	Description
function_username	Oracle MSSQL	User ID passed in LTL with replicated stored procedure invocations.
log_backup_files	All	Determines the number of log backup files kept in the log directory.
log_directory	All	Directory where Replication Agent system log file is located.
log_read_block_count	Oracle MSSQL	Determines the number of blocks or pages that are read at a time from the primary database transaction log.
log_trace_verbose	All	Switch on/off verbose mode in trace log file.
log_wrap	All	Number of 1KB blocks written to log file before wrapping.
lr_ntext_byte_order	Oracle MSSQL	Specifies which byte order to use when replicating NCLOB for Oracle, and NTEXT for Microsoft SQL Server.
lr_send_trunc_partition_ddl	Oracle	Determines whether truncate partition commands are sent as DDL or DML to the replicate database.
lti_batch_mode	All	Switches on/off LTI batch mode.
lti_max_buffer_size	All	Maximum number of change sets stored in the LTI input buffer.
lti_update_trunc_point	All	Number of LTL commands sent before LTI requests new LTM Locator.
ltl_batch_size	All	Size of the LTL batch buffer.
ltl_big_endian_unitext	All	Specifies whether unicode LOB data should be converted from little endian to big endian before sending LTL to Replication Server.
ltl_character_case	All	Case of database object names sent to Replication Server.
ltl_origin_time_required	All	Specifies whether to send origin_time command tag in LTL.
ltl_send_only_primary_keys	All	Controls whether the Replication Agent sends only primary key columns data for the *before* image for update and delete operations.
ltm_admin_pw	All	Password for Replication Agent administrative port.
ltm_admin_user	All	User ID for Replication Agent administrative port.

Parameter name	Target	Description
max_ops_per_scan	All	Maximum number of operations Log Reader will read in a single log scan.
pdb_archive_path	Oracle UDB	Identifies the directory path where the Replication Agent expects to find archived Oracle redo log files or the archived UDB transaction log files.
pdb_archive_remove	Oracle UDB	Enables or disables the removal of archived transaction log files from the path specified by pdb_archive_path.
pdb_auto_create_repdefs	Oracle MSSQL	If set to true, when tables and procedures are marked for replication, a replication definition is automatically created at Replication Server for that table or procedure.
pdb_automark_tables	Oracle MSSQL	Determines if the Replication Agent automatically marks tables for replication during initialization or DDL replication.
pdb_auto_run_scripts	All	Automatic execution of SQL scripts used to create/remove transaction log objects and mark/unmark primary database objects.
pdb_convert_datetime	All	Converts native date/time formats to Sybase datetime format.
pdb_dflt_column_repl	All	Enables replication for LOB columns by default when table is marked.
pdb_dflt_object_repl	Oracle UDB	Enables replication by default when object is marked.
pdb_include_archives	Oracle	Enables or disables the use of Oracle archive log files.
pdb_support_large_identifier	All	To support replication of large identifiers up to 255 characters in length with Replication Server 12.6 or later.
pdb_timezone_file	Oracle	Specifies the file to read at Replication Agent initialization to obtain Oracle time zone information.
pdb_xlog_device	All	Name of the primary database device.
pdb_xlog_prefix	All	Character string prefix used to identify transaction log objects.
pdb_xlog_prefix_chars	All	Non-alphabetic characters allowed in pdb_xlog_prefix.

Parameter name	Target	Description
pds_connection_type	All	Type of connection to primary data server.
pds_dac_port_number	MSSQL	The primary database dedicated administrative connection port number used by Replication Agent during the server level initialization of the primary database.
pds_database_name	All	Name of database replicated from the primary data server.
pds_datasource_name	UDB	Data source name or database alias of the database replicated from the primary data server.
pds_host_name	All	Name of primary data server host machine.
pds_integrated_security	MSSQL	Determines if the Replication Agent should use Windows authentication when connecting to the primary SQL Server.
pds_password	All	Password for user ID that the Replication Agent uses to access the primary data server.
pds_port_number	All	Port number for the primary data server.
pds_retry_count	All	Number of times to retry connection to primary data server.
pds_retry_timeout	All	Number of seconds to wait between connection retry attempts.
pds_server_name	All	Server name of primary data server.
pds_tns_connection	Oracle	Oracle connection name found in the *tnsmame.ora* file which identifies the connection information for the primary database.
pds_tns_filename	Oracle	Oracle file name identifying the Oracle *tnsname.ora* file to be used to identify the connection information for the primary database.
pds_username	All	User ID that Replication Agent uses to access primary data server.
ra_retry_count	All	Number of times LTM attempts to get back to *Replicating* state after a failure.

Parameter name	Target	Description
ra_retry_timeout	All	Number of seconds to wait between LTM attempts to get back to *Replicating* state.
rasd_backup_dir	Oracle MSSQL	The directory path for Replication Agent System Database (RASD) backup files.
rasd_database	Oracle MSSQL	The directory path for the Replication Agent System Database (RASD) database file.
rasd_mirror_tran_log	Oracle MSSQL	Enables or disables Replication Agent System Database (RASD) transaction log mirroring.
rasd_trace_log_dir	Oracle MSSQL	The directory path for the Replication Agent System Database (RASD) trace log file.
rasd_tran_log	Oracle MSSQL	The directory path for the Replication Agent System Database (RASD) transaction log file.
rasd_tran_log_mirror	Oracle MSSQL	The directory path for the Replication Agent System Database (RASD) transaction log file mirror.
rs_charset	All	Character set used to communicate with Replication Server.
rs_host_name	All	Name of primary Replication Server host machine.
rs_packet_size	All	Network I/O packet size sent to Replication Server.
rs_password	All	Password for user ID Replication Agent uses to access Replication Server.
rs_port_number	All	Port number for primary Replication Server.
rs_replicate_owner_required	All	Indicates if the owner is always included with the replicate table clause when generating replication definitions.
rs_retry_count	All	Number of times to retry connection to primary Replication Server.
rs_retry_timeout	All	Number of seconds to wait between connection retry attempts.
rs_source_db	All	Name of primary database identified to Replication Server.

Parameter name	Target	Description
rs_source_ds	All	Name of primary data server identified to Replication Server.
rs_ticket_version	All	Determines whether Replication Agent records the primary database time or the primary database date and time into the rs_ticket marker.
rs_username	All	User ID that Replication Agent uses to access primary Replication Server.
rssd_charset	All	Character set used to communicate with RSSD.
rssd_database_name	All	Name of RSSD database.
rssd_host_name	All	Name of RSSD host machine.
rssd_password	All	Password for user ID Replication Agent uses to access RSSD.
rssd_port_number	All	Port number for RSSD.
rssd_username	All	User ID that Replication Agent uses to access RSSD.
scan_sleep_increment	All	Number of seconds to increase Log Reader wait before next scan after finding no operations to replicate.
scan_sleep_max	All	Maximum number of seconds for Log Reader to wait before next scan after finding no operations to replicate.
skip_ltl_errors	All	LTI ignores error messages returned by Replication Server.
structured_tokens	All	LTI uses structured tokens when generating LTL output.
truncation_interval	All	Number of minutes to wait between automatic log truncations.
truncation_type	All	Methods of log truncation allowed.
use_rssd	All	Switches on/off access to RSSD for replication definitions.

admin_port

The client socket port number of the Replication Agent.

Default 10000

Value A valid port number on the Replication Agent host machine.

Comments

- When you create a Replication Agent instance, you must specify a client socket port number for the instance administration port. Client applications use this port number to connect to the Replication Agent instance.
- You must specify a port number that does not conflict with any port numbers already in use on the Replication Agent host machine.
- If you change the value of the admin_port parameter with the ra_config command, the new value is recorded in the configuration file immediately, but you must shut down and restart the Replication Agent instance to make the new port number take effect.
- After you change the value of the admin_port parameter with the ra_config command, the next time you log in to the Replication Agent administration port, you must use the new port number.

archive _read_block_count

Note This parameter is available only for Oracle.

Determines the number of blocks or pages that are read at a time from the primary database archive log.

Default

1 (one)

Value

An integer between 1 - 8192

Comments

- Redo block sizes are 512 bytes for all platforms except HP, for which the block size is 1024 bytes.
- The block size is not configurable.
- The archive_read_block_count affects only the read of Oracle archive redo logs.
- The log_read_block_count configuration determines the block count for Oracle online redo logs.

asm_password

Note This parameter is available only for Oracle.

The password associated with the configuration parameter asm_username user to access the Oracle Automatic Storage Management (ASM) server instance.

Default: "" (empty string)

Value: A valid password.

Comments:

- Setting of the configuration parameter is only required when the Oracle ASM product is being used to manage Oracle online redo or archive redo storage.
- The value of the asm_password parameter is encrypted in the Replication Agent instance configuration file.

See also: asm_tns_connection, asm_tns_filename, and asm_username configuration parameters

asm_tns_connection

Note This parameter is available only for Oracle.

The Oracle connection name that identifies the connection parameters for the Oracle Automatic Storage Management (ASM) connection in the Oracle *tnsnames.ora* file. If configuration parameter asm_tns_filename is not configured, the *tnsnames.ora* file identified by pds_tns_filename will be used.

Default: <not configured>

Value: A valid ASM connection name from the Oracle *tnsnames.ora* file specified by either the asm_tns_filename or pds_tns_filename configuration parameter.

Comments: Setting of the configuration parameter is only required when the Oracle ASM product is being used to manage Oracle online redo or archive redo storage

See also: pds_tns_filename, asm_tns_filename, asm_username, and asm_tns_password configuration parameters

asm_tns_filename

Note This parameter is available only for Oracle.

The fully-qualified file name identifying the Oracle *tnsnames.ora* file that contains the Oracle ASM connection parameters. This configuration parameter is required only when the connection parameter information required for ASM does not exist in the *tnsnames.ora* file pointed to by the pds_tns_filename configuration parameter.

Default: <not configured>

Value: A valid Oracle *tnsnames.ora* file that contains the connection parameters to the Oracle ASM server. This file normally resides in the ASM instance's *ORACLE_HOME\network\admin* directory.

Comments:

- Setting of the configuration parameter is only required when the Oracle ASM product is being used to manage Oracle online redo or archive redo storage.
- This parameter is required only when the pds_tns_filename configuration parameter is not populated, or the *tnsnames.ora* file pointed to by configuration parameter pds_tns_filename does not contain the connection definition for the ASM connection to be used. This may occur if the primary Oracle instance and ASM instance was installed in different ORACLE_HOME locations, resulting in different *tnsnames.ora* files for each server.

  Warning! The Replication Agent process must have *read* permission to this file. Access failures prevent the Replication Agent from connecting to the Oracle ASM server.

See also: pds_tns_filename and asm_tns_filename configuration parameters

asm_username

Note This parameter is available only for Oracle.

The login name that the Replication Agent uses to access the Oracle ASM server.

Default	<not configured>
Value	A valid Oracle ASM user name.
Comments	• Setting of the configuration parameter is only required when the Oracle ASM product is being used to manage Oracle online redo or archive redo storage • The value of the asm_username parameter is the login name that the Replication Agent uses to log in to the Oracle ASM server that is managing storage for Oracle online or archived redo logs. • This login name must be defined in the ASM server, with appropriate privileges to query ASM system views. • The Replication Agent uses this login to query ASM in order to determine Oracle online or archived redo log locations. • To remove archive log files from ASM that are no longer required for Replication, the user must have update privileges in ASM.
See also	asm_tns_connection, asm_tns_filename, asm_password, and pds_archive_remove configuration parameters.

auto_adjust_block_count

Note This parameter is available only for Oracle and Microsoft SQL Server.

Determines whether or not the Replication Agent should automatically adjust the log_read_block_count parameter if any part of a read is discarded.

Default	true
Value	true – enables automatic adjusting of the log read block count. false – disables automatic adjusting of the log read block count.
Comments	• The auto_adjust_block_count is relevant only if the log_read_block_count is configured to a value greater than 1. • The Replication Agent never automatically adjusts to a block count greater than the block count specified by the log_read_block_count configuration parameter. • For Oracle, the auto_adjust_block_count is relevant only for online redo log reading, since archive logs have a defined end.

See also log_read_block_count configuration parameter

column_compression

Determines whether the Log Transfer Interface component sends all columns in row *after* images, or only the columns that changed in an update operation.

Default true

Values true – enables minimal column information (only changed columns in row after images) in Log Transfer Language (LTL) for update operations.

false – disables minimal column information in LTL for update operations.

Comments
- When the column_compression parameter is set to false, the LTI component sends complete row *after* images in LTL, including columns in which no data changed as a result of an update operation.
- When the column_compression parameter is set to true, the LTI component sends minimal column information in the row after images in LTL, with only the columns that changed as a result of an update operation. Columns in which no data changed as a result of the update are not sent in LTL.
- In general, setting the value of the column_compression parameter to true provides better Replication Agent throughput.

compress_ltl_syntax

Determines whether the Log Transfer Interface component compresses Log Transfer Language (LTL) commands using abbreviated syntax.

Default true

Values true – enables LTL compression, using abbreviated LTL syntax.

false – disables LTL compression.

Comments
- Setting the value of the compress_ltl_syntax parameter to true will provide better Replication Agent throughput.
- See the *Replication Agent Administration Guide* for more information about LTL commands and abbreviated LTL syntax.

connect_to_rs

Enables or disables the network connection to the primary Replication Server.

Default

true

Values

true – enables the network connection to the Replication Server.

false – disables the network connection to the Replication Server.

Comments

- When the value of the connect_to_rs parameter is false, the network connection from the Replication Agent to the Replication Server is disabled, and no replication can occur.
- When the network connection to the Replication Server is disabled by the connect_to_rs parameter, the Replication Agent instance can still go to *Replicating* state, with the following limitations:
  - A “dummy” connection in the Replication Agent emulates a real connection to the Replication Server.
  - The value of the LTM Locator stored in the Replication Agent transaction log is set to zero.
  - The maintenance user name is set to an invalid user ID.

  Note maintenance user operations cannot be filtered when the value of the connect_to_rs parameter is false.
- You can use the connect_to_rs parameter to temporarily disable the network connection to the Replication Server for testing.
- When the value of the connect_to_rs parameter is false, you can put the Replication Agent instance in *Replicating* state, set the value of the LTITRACELTL trace flag to true, and view a readable representation of the LTL that would have been sent to the Replication Server if the connection had not been disabled.
- During normal Replication Agent operation, the value of the connect_to_rs parameter must be true.

ddl_password

Note This parameter is available only for Oracle and Microsoft SQL Server.

Updates the log device repository in the RASD. Identifies the password for ddl_username.

Default: "" (empty string)

Value: A valid password.

Comments:

- The value of the ddl_password parameter can be up to 30 characters.
- The value of the ddl_password parameter is the password for the database user name specified in the ddl_username parameter.
- The value of the ddl_password parameter is encrypted in the Replication Agent configuration file.

ddl_username

Note This parameter is available only for Oracle and Microsoft SQL Server.

The database user name included in LTL for replicating DDL commands to the standby database.

This user must have permission to execute all replicated DDL commands at the standby database.

Default: <not_configured>

Value: A valid user name in the standby database.

Comments:

- The value for the ddl_username must not be the same as the value of the maintenance user defined in Replication Server for the standby connection. Failure to provide different names results in a Replication Server error.
- The value of the ddl_username parameter is sent in the LTL for all replicated DDL statements.
- The value of the ddl_password parameter is the password for the database user name specified in the ddl_username parameter.
- When DDL is replicated, Replication Server will connect to the replicate database using the ddl_username and ddl_password.
- For Oracle, Replication Server issues the following message:

  ```
  ALTER SESSION SET CURRENT_SCHEMA=user
  ```

where *user* is the user ID that generated the DDL operation at the primary database. The actual DDL command is then executed against the replicate database. If the ddl_username does not have permission to issue ALTER SESSION SET CURRENT_SCHEMA or to execute the DDL command against the *user* schema, the command fails.

- For Microsoft SQL Server, Replication Agent will send:

  ```
  execute as login = user
  ```

  where *user* is the user ID that generated the DDL operation at the primary database. The actual DDL command is then executed against the replicate database. If the ddl_username does not have permission to issue execute as login or to execute the DDL command against the *user* schema, the command fails.

dump_batch_timeout

Specifies the number of seconds to wait before sending the contents of the Log Transfer Interface (LTI) buffer to the Replication Server, even though the buffer is not full.

Default: 5

Value: An integer from 1 to 60.

Comments:

- The value of the dump_batch_timeout parameter is the number of seconds from the time the previous LTI buffer was sent to the Replication Server until the next buffer will be sent.
- The dump_batch_timeout parameter has no effect if the value of the lti_batch_mode parameter is false.

filter_maint_userid

Determines whether operations applied by the maintenance user are ignored.

Default: true

Values: true – enables the Log Reader to ignore maintenance user operations.

false – disables the Log Reader filter to allow replicating maintenance user operations.

Comments

- The filter_maint_userid configuration parameter is provided to support bidirectional replication, in which the primary database also serves as a replicate database that has transactions applied to it by a Replication Server maintenance user.
- If the value of the filter_maint_userid parameter is true, database operations applied by the maintenance user are *not* replicated. The Log Reader component filters out (ignores) operations applied by the maintenance user when it reads the transaction log.
- If the value of the filter_maint_userid parameter is false, database operations applied by the maintenance user are replicated. The Log Reader component replicates all operations on marked objects, regardless of the user that applied the operation.
- The maintenance user login is specified when the database connection for the primary database is created in the Replication Server.

function_password

Note This parameter is available only for Oracle and Microsoft SQL Server.

The password included in Log Transfer Language for replication of “request” stored procedures.

Default

"" (empty string)

Values

A valid password.

Comments

- The value of the function_password parameter can be up to 30 characters.
- The value of the function_password parameter is the password for the database user name specified in the function_username parameter.
- The value of the function_password parameter is encrypted in the Replication Agent configuration file.
- For more information about “request” stored procedures, see Replication Server documentation.

function_username

Note This parameter is available only for Oracle and Microsoft SQL Server.

The database user name included in Log Transfer Language (LTL) for replication of "request" stored procedures.

Default: sa

Values: A valid user name in the primary database.

Comments:

- The value of the function_username parameter is sent in the LTL for all replicated stored procedures in the primary database.
- The value of the function_password parameter is the password for the database user name specified in the function_username parameter.
- For more information about "request" stored procedures, see Replication Server documentation.

log_backup_files

The number of backup log files kept in the Replication Agent instance *log* directory.

Default: 3

Values: An integer greater than or equal to 1.

Comments:

- When the system log wraps, Replication Agent copies the current log file to a backup file, with a generated number appended to the file's name.

  For example, if the system log file is named *my_ra.log*, the first backup file created when the system log wraps would be named *my_ra1.log*. The second backup file created would be named *my_ra2.log*, and so on.
- When the number of backup files exceeds the value of the log_backup_files parameter, the oldest backup file (that is, the one with the lowest generated number) is deleted from the *log* directory before the next backup file is created.

log_directory

The directory for Replication Agent system log files.

Default

The path to the *log* directory created when the Replication Agent instance was created. For example:

- On Microsoft Windows platforms:

  %SYBASE%\RA-15_2*inst_name*\log

  where:

  - *%SYBASE%* is the path to the Replication Agent installation directory.
  - *inst_name* is the name of the Replication Agent instance.

- On UNIX platforms:

  $SYBASE/RA-15_2/*inst_name*/log

  where:

  - *$SYBASE* is the path to the Replication Agent installation directory.
  - *inst_name* is the name of the Replication Agent instance.

Value

A valid path on the Replication Agent host machine.

Comments

- When a Replication Agent instance is created, the *log* directory is created as part of the instance directories. The default value of the log_directory parameter points to that directory.
- If you specify any valid path as the value of the log_directory parameter, the Replication Agent instance places its system log files in the directory you specify the next time it is started.
- If you specify the default value of the log_directory parameter by using the default keyword in the ra_config command, then the next time it is started, Replication Agent will place its system log files in the *log* directory that was created when the Replication Agent instance was created.
- If you change the value of the log_directory parameter with the ra_config command, the new value is recorded in the configuration file immediately, however, you must shut down and restart the Replication Agent instance to make the new value take effect.

log_read_block_count

Note This parameter is available for Oracle and Microsoft SQL Server.

	Determines the number of blocks or pages that are read at a time from the primary database transaction log.
Default	1 (one)
Values	An integer between 1 - 8192
Comment	• In Microsoft SQL Server, transaction log page sizes can be 2K, 4K, 8K, or 16K. • In Oracle, redo block sizes are 512 bytes for all platforms except HP, for which the block size is 1024 bytes. The block size is not configurable. • In Oracle, the log_read_block_count affects only the read for online redo logs. The archive_log_read_block_count controls the read for archive redo logs.
See also	auto_adjust_block_count configuration parameter.

log_trace_verbose

	Enables or disables additional diagnostic information in Replication Agent system log files.
Default	false
Values	true – enables detailed diagnostic information in log files. false – disables detailed diagnostic information in log files.
Comment	Detailed diagnostic information is intended for troubleshooting only, with assistance from Sybase Technical Support.

log_wrap

	The maximum size of the Replication Agent system log file before wrapping.
Default	1000
Value	An integer greater than or equal to 1000.
Comments	• The value of the log_wrap parameter is the number of 1KB blocks written by Replication Agent, before it wraps the system log file. • Larger values for the log_wrap parameter allow more log history in each file. Smaller values produce smaller log files.

- When the log file wraps, Replication Agent copies the current log file to a backup file, with a generated number appended to the file's name.

  For example, if the system log file is named *my_ra.log*, the first backup file created when the system log wraps would be named *my_ra1.log*. The second backup file created would be named *my_ra2.log*, and so on.

- When the number of backup files exceeds the value of the log_backup_files parameter, the oldest backup file (that is, the one with the lowest generated number) is deleted from the *log* directory before the next backup file is created.

lr_ntext_byte_order

Note This parameter is available only for Oracle and Microsoft SQL Server.

Ensures the byte order of NCLOB data (for Oracle) or NTEXT data (for Microsoft SQL Server) is sent correctly to the replicate database. The byte order being correct is necessary when you are replicating with different primary and replicate database types (Oracle to Microsoft SQL Server), or primary and replicate databases that are on different platforms (Windows to big endian UNIX).

Default

big

Values

big – big endian

little – little endian

Examples

- For Microsoft SQL Server:

  When replicating Microsoft SQL Server to Microsoft SQL Server, both the primary and replicate database must be little endian because Microsoft SQL Server runs only on Windows. The parameter should be set to little endian. If the replicate database is not Microsoft SQL Server, you need to know whether it is little endian or big endian, and set the parameter to match.

- For Oracle:

  When replicating Oracle to Oracle, you need to know the endian for the replicate database and set the parameter so the correct endian is sent to the replicate. The same applies when replicating to databases other than Oracle.

Comments

- If lr_ntext_byte_order is set to little, ltl_big_endian_unitext must be set to false.
- Big endian indicates a left-to-right byte order architecture; little endian indicates a right-to-left byte order architecture.
- Microsoft SQL Server stores double byte ntext datatype values in little endian byte order. By default, the byte order of ntext data will be converted during replication to big endian, to allow the data to be transmitted over networks using the common “network” byte order, or big endian.

  If the target database is also Microsoft SQL Server, the SQL Server will *not* automatically convert the replicated data from the sent big endian order to the SQL Server desired little endian order.To support replicating ntext data to a Microsoft SQL Server (or other replicate server that fails to provide the necessary conversion), you may 'force' the byte order to be sent using the lr_ntext_byte_order parameter by specifying a value of big or little, as desired to meet the expectations of your replicate database.

  Note The default behavior of the Replication Agent is to force any unicode data to big endian order, as defined by configuration parameter ltl_big_endian_unitext. In order to allow configuration parameter lr_ntext_byte_order to successfully override the SQL Server byte order, you must also set ltl_big_endian_unitext configuration parameter to false whenever the lr_next_byte_order parameter is used.

- The following describes the relationship between ltl_big_endian_unitext and lr_ntext_byte_order configuration parameters:
  - When ltl_big_endian_unitext is set true by default, the Replication Agent ensures that all unicode data is sent in big endian order.
  - When set to false, ltl_big_endian_unitext allows unicode data to be sent in the byte order that is used when the data is stored in the transaction log file
  - In contrast, lr_ntext_byte_order, forces the result of Unicode data read from the transaction log to be in the requested byte order, regardless of how it normally exists in the transaction log file.

See also

ltl_big_endian_unitext configuration parameter.

lr_send_trunc_partition_ddl

Note This parameter is available only for Oracle.

Determines whether truncate partition commands are sent as DDL or DML to the replicate database.

Default: true

Values: true – the truncate partition command is sent as a DDL command (alter table).

false – the truncate partition is sent as a DML operation.

Comments:

- If set to true, the truncate partition command is sent as a DDL command. Normally, it is set to replicate to Oracle.
- Set to false, when replicating to databases that treat truncate partition commands as DML.

lti_batch_mode

Enables or disables the Log Transfer Interface component LTL batch mode.

Default: true

Values: true – enables LTL batch mode.

false – disables LTL batch mode.

Comments:

- If the value of the lti_batch_mode parameter is true, the LTI component sends LTL commands to the Replication Server in batches, instead of one command at a time:
  - The LTI component fits as many LTL commands as it can into its LTL batch mode buffer, before it sends any commands to the Replication Server.
  - When the time interval specified in the dump_batch_timeout parameter expires, the LTI component sends the current LTL batch mode buffer contents to the Replication Server, even if the buffer is not full.
- If the value of the lti_batch_mode parameter is false, the LTI component sends individual LTL commands to the Replication Server for each change set in its input queue.

- When Replication Agent connects to the Replication Server, it determines the version of the Replication Server:
  - If the Replication Server version is earlier than 12.5, the size of the LTL batch mode buffer is set to 16KB automatically.
  - If the Replication Server version is 12.5 or later, Replication Agent sets the size of the LTL batch mode buffer to the size specified by the ltl_batch_size parameter.
- If the Replication Server version is 12.5 or later, you can use the Replication Agent ltl_batch_size parameter to set the size of the LTI component's LTL batch mode buffer.

Note Adjusting the size of the LTL batch mode buffer can help you optimize the performance of the replication system.

- If the Replication Server version is earlier than 12.5 and the value of the lti_batch_mode parameter is true, if any single LTL distribute command exceeds the 16K size of the LTL batch mode buffer, Replication Server returns an error and Replication Agent goes into *Admin* state.
- In general, setting the value of the lti_batch_mode parameter to true provides better Replication Agent throughput.

lti_max_buffer_size

The maximum size of the Log Transfer Interface (LTI) component's queues.

Default: 5000

Value: An integer in the range of 1000 to 100000.

Comments:

- The value of the lti_max_buffer_size parameter is the maximum number of operations that can be stored in the LTI component's inbound and outbound queues:
  - Operations in the inbound queue represent change sets received from the Log Reader component.
  - Operations in the outbound queue are the Log Transfer Language commands to be sent to the Replication Server.
- The LTI component's inbound queue is a bounded buffer that blocks the processing of the Log Reader component when it gets full.

lti_update_trunc_point

The number of Log Transfer Language (LTL) commands sent before requesting a new LTM Locator from the Replication Server.

Default: 1000

Value: An integer from 1 to 100000.

Comments:

- The value of the lti_update_trunc_point parameter is the number of LTL commands that Replication Agent sends to the Replication Server, before it requests a new LTM Locator (secondary truncation point) from the Replication Server.
- Lower numbers cause Replication Agent to request a new LTM Locator from the Replication Server more often.
- If the value of the truncation_type parameter is locator_update, setting the value of the lti_update_trunc_point parameter to a lower number causes automatic log truncation to occur more frequently.
- The value of the lti_update_trunc_point parameter is a trade-off between better system performance and longer recovery time:
  - Lower values reduce the time it takes to recover from a replication failure, but they may have an adverse affect on overall system throughput.
  - Higher values improve overall system throughput, but they may increase the time it takes to recover from a replication failure.
- If the Replication Agent is operating in an unreliable network environment, it may be prudent to set the lti_update_trunc_point parameter to a lower value to ensure faster recovery.

ltl_batch_size

The size of the Log Transfer Interface component's Log Transfer Language (LTL) batch mode buffer.

Default: 40000

Value: An integer from 16384 to 10485760.

Comments:

- The value of the ltl_batch_size parameter is the size (in bytes) of the LTI component's LTL batch mode buffer.

- When Replication Agent connects to the Replication Server, it determines the version of the Replication Server:
  - If the Replication Server version is earlier than 12.5, the size of the LTL batch mode buffer is set to 16K automatically, and the value of the ltl_batch_size parameter is ignored.
  - If the Replication Server version is 12.5 or later, Replication Agent sets the size of the LTL batch mode buffer to the size specified by the ltl_batch_size parameter.
- The Log Transfer Interface component uses the LTL batch mode buffer only if the value of the lti_batch_mode parameter is true. If the value of the lti_batch_mode parameter is false, the LTL batch mode buffer is not used.

ltl_big_endian_unitext

Specifies whether "unitext" data is converted from little endian to big endian before sending LTL to Replication Server.

Default true

Values true – Unitext data that is in little endian byte order will be changed to big endian byte order.

false – Unitext data byte order is *not* changed.

Comments When setting this parameter, you must know how the lr_ntext_byte_order is set. If parameter lr_ntext_byte_order is set to send the correct byte order for the replicate database, then ltl_big_endian_unitext must be set to false so the byte order will not be changed.

See also lr_next_byte_order configuration parameter.

ltl_character_case

The character case used for database object names in Log Transfer Language (LTL) sent to the Replication Server.

Default asis

Values asis – database object names are sent in the same character case as they are returned from the primary database, or (if the value of the use_rssd parameter is true) in the same character case as they are specified in replication definitions.

lower – database object names are sent in *all lowercase*, regardless of how they are returned from the primary database, or specified in replication definitions.

upper – database object names in LTL are sent in *all uppercase*, regardless of how they are returned from the primary database, or are specified in replication definitions.

Comments

- The ltl_character_case configuration parameter allows you to customize the handling of database object names in LTL to work with replication definitions that specify the object names differently than the way the primary database returns them.
- If the value of the ltl_character_case parameter is asis, and the value of the use_rssd parameter is true, database object names are sent in the same character case as they are specified in replication definitions.
- If the value of the ltl_character_case parameter is asis, and the value of the use_rssd parameter is false, database object names are sent in the same character case as they are returned from the primary database.
- If replication definitions specify database object names in all lowercase, set the value of the ltl_character_case parameter to lower.
- If replication definitions specify database object names in all uppercase, set the value of the ltl_character_case parameter to upper.
- If you want to send database object names with "mixed" character case (for example, MyTable), set the value of the ltl_character_case parameter to asis.

ltl_origin_time_required

Enables or disables the Log Transfer Language (LTL) origin_time command tag.

Default

false

Values

true – enables the origin_time command tag in LTL.

false – disables the origin_time command tag in LTL.

Comments

- If the value of the ltl_origin_time_required parameter is true, the Log Transfer Interface component includes the origin_time command tag in the LTL it generates.
- If a Replication Server function string checks for the origin_time command tag, set the value of the ltl_origin_time_required parameter to true.

- The datetime value placed in the LTL origin_time command tag is the time that the original primary database operation was recorded in the transaction log, not the time it was scanned and processed by the Log Reader component.
- Setting the value of the ltl_origin_time_required parameter to false provides better Replication Agent throughput.
- If you use Replication Manager to report latency, you must set the value of the ltl_origin_time_required parameter to true.

ltl_send_only_primary_keys

Determines whether Replication Agent sends only primary key columns, or sends all columns to Replication Server for update and delete operations to the replicate database.

Default: true

Value: true – sends only the primary key columns to Replication Server.

false – sends all columns to Replication Server.

Comments:

- When set to true and a replication definition exists that identifies the primary key column(s) for a table, only the primary key column value(s) are sent for the *before* image in update and delete operations. Sending only primary key column data reduces the amount of data sent to Replication Server, since only primary keys are used to construct the "where" clauses for update and delete operations.
- When set to false, *before* image values are sent for all columns available, regardless of primary key definition.
- Setting of ltl_send_only_primary_keys to false, is only recommended when additional *before* image values provide benefit, such as for supporting customer function strings at the Replicate database, or for resolving other issues where additional column data provides benefit.
- Primary keys are defined within a table level Replication Definition. If configuration use_rssd is set to false, the setting of ltl_send_only_primary_keys has no impact, since Replication Definition information will not be gathered from the Replication Server System Database (RSSD).
- If you set the value to false, the performance will be slower.

ltm_admin_pw

The Replication Agent administrator login password.

Default: "" (empty string)

Value: A valid password.

Comments:

- The value of the ltm_admin_pw parameter is the password for the user name authorized to log in to the Replication Agent.
- The value of the ltm_admin_pw parameter is encrypted in the Replication Agent configuration file.
- To change the value of the ltm_admin_pw parameter, use ra_set_login.
- When you change the value of the ltm_admin_pw parameter with ra_set_login, the new value is recorded in the configuration file immediately. However, you must shut down and restart the Replication Agent instance to make the new password take effect.

  After you change the value of the ltm_admin_pw parameter with ra_set_login, you must use the new password the next time you log in to the Replication Agent.

ltm_admin_user

The Replication Agent administrator login name.

Default: sa

Value: A valid user name on the Replication Agent host machine.

Comments:

- The value of the ltm_admin_user parameter is the user name authorized to log in to the Replication Agent.
- To change the value of the ltm_admin_user parameter, use the ra_set_login command.
- If you change the value of the ltm_admin_user parameter with the ra_set_login command, the new value is recorded in the configuration file immediately. However, you must shut down and restart the Replication Agent instance to make the new administrator name take effect.
- After you change the value of the ltm_admin_user parameter with ra_set_login, you must use the new administrator name the next time you log in to the Replication Agent.

max_ops_per_scan

The maximum number of operations the Log Reader component reads during each log scan operation.

Default: 1000

Values: An integer from 25 to 2147483647.

Comments:

- The value of the max_ops_per_scan parameter is the maximum number of database operations that can be read from the Replication Agent transaction log during each Log Reader scan operation (the size of the Log Reader operation queue).
- The Log Reader component always reads at least one transaction in each scan, regardless of how many operations are in the transaction.

  For example, if the value of the max_ops_per_scan parameter is 1000, and a transaction contains 1200 operations, the Log Reader component reads all 1200 operations in one scan when it reads that transaction.
- See the *Replication Agent Primary Database Guide* for more information about how the max_ops_per_scan parameter affects Replication Agent performance.

pdb_archive_path

Note This parameter is available only for Oracle and UDB.

Identifies the directory path where the Replication Agent expects to find archived redo log files (For Oracle) or archived transaction log files (for UDB). When archived redo log files or archived transaction log files are stored in the file system, the configuration parameter is set to a file system path.

Default: <not configured>

Values: For Oracle, a valid directory path on the machine hosting the Replication Agent that points to a location where Oracle places the archived redo log files. For example,

```
ORACLE_HOME\oradata\orcl\archive
```

Archive logs stored in and managed by ASM are owned by the corresponding unique Oracle database name. If the Oracle database name differs from the global unique database name, you must set pdb_archive_path to both the name of the ASM disk group and the globally unique name of the database in which the archive logs are stored:

```
pdb_archive_path=+DISK_GROUP1/database_name
```

You can also set pdb_truncate_xlog to manually remove archive logs. Set the pdb_archive_path to the ASM disk group name, and precede the archive logs to be manually removed with a plus "+" sign.

For UDB, a valid directory path on disk, as defined by the UDB LOGARCHMETH1 or LOGARCHMETH2 database configuration parameter, when it is configured to DISK. For example,

```
<path>
```

Comments

- You must set pdb_archive_path when configuration parameter pdb_include_archives is set to true, and you must set it to a valid location before the Replication Agent can be placed in a *Replicating* state.
- If the Replication Agent cannot find an expected log record in the Oracle online redo logs, the Replication Agent will search this directory for the archived log file containing the required record.
- For UDB, pdb_archive_path must be configured before pdb_archive_remove is enabled (set to true).

  When pdb_archive_remove parameter is set to true, Replication Agent for UDB truncates the archived log files from the location specified by pdb_archive_path.

See also

pdb_archive_remove, and pdb_include_archives configuration parameters

pdb_archive_remove

Note This parameter is available only for Oracle and UDB.

Enables or disables the removal of Oracle archived redo log files or UDB archived transaction logs from the path specified by pdb_archive_path.

Default

false

Values	true – Allows the removal of archived Oracle redo log and UDB archived transaction log files from the path specified by pdb_archive_path. Removal occurs based on the execution of command pdb_truncate_xlog, or the timing of automatic truncation based on parameters truncation_type and truncation_interval. false – Disables the removal of archived Oracle redo log files or UDB archived transaction log files.
Comments	• For Oracle, set this configuration to true when the path specified by pdb_archive_path is established solely for Replication support, and automatic removal of unneeded archived log files is desired. • For UDB, set this configuration to true when UDB is configured to archive logs by setting LOGARCHMETH1 configuration parameter to DISK:<path>, and the automatic removal of archived log files is desired. • If the path specified by pdb_archive_path is shared by other processes, or the removal of archived log files is expected to be performed by processes other than the Replication Agent, this parameter should be false. • Configuration parameters truncation_type and truncation_interval, and command pdb_truncate_xlog have no impact when this configuration parameter is set to false.
See also	pdb_archive_path, truncation_type, truncation_interval configuration parameters, and the pdb_truncate_xlog command

pdb_auto_create_repdefs

Note This parameter is available only for Oracle and Microsoft SQL Server.

Configures Replication Agent to automatically create replication definitions at Replication Server when a table or procedure is marked for replication, after initialization. To improve performance pdb_auto_create_repdefs is ignored during initialization.

Note The pdb_xlog create command no longer checks the setting of pdb_auto_create_repdefs during initialization. To create replication definitions for all marked tables after executing pdb_xlog create, execute command rs_create_repdef all.

Default	false
Values	true – Replication Agent automatically creates replication definitions at the Replication Server when tables or procedures are marked after initialization. false – no replication definitions are created when tables or procedures are marked.
Comments	**Note** Replication Agent always assumes that a database replication definition exists for the primary database. • The table and procedure replication definitions that Replication Agent creates assume that a database level replication definition for the primary database already exists at Replication Server. All replication definitions created by this command include the send standby clause, which means the replication definition will only be used by Replication Server if there is a database level replication definition or the primary Replication Server connection is for a warm standby configuration. The replication definition created by rs_create_repdef can not be individually subscribed to. If you do not wish to have a database level replication definition, or warm standby configuration, you must use a different tool, or create replication definitions manually, and not use rs_create_repdef. • Replication definitions created by rs_create_repdef will always define the datatypes using available user defined datatypes that are installed in Replication Server. This means that customers using rs_create_repdef should not set Replication Agent configuration parameter pdb_convert_datetime to true, as doing so converts date and timestamp datatypes to Sybase format, instead of UDD format. • If this parameter is set to true and when pdb_setrepproc is invoked to mark a procedure or procedures, a replication definition is created at Replication Server for each procedure that gets marked for replication. • If this parameter is set to true and when pdb_setreptable is invoked to mark a table or tables, a replication definition is created at Replication Server for each table that gets marked for replication. • If this parameter is set to true and when pdb_setreptable is invoked to unmark a table or tables, the replication definition is dropped at Replication Server for each table that gets unmarked for replication.

- If this parameter is set to true and when pdb_setrepproc is invoked to unmark a procedure or procedures, a replication definition is dropped at Replication Server for each procedure that gets unmarked for replication if this parameter is set to true.
- The following applies to replication definition table and procedure names:
  - All non-alphanumeric characters and spaces are removed and are not part of the table or procedure name.
  - Underscores are kept as part of the name even though they are non-alphanumeric characters.
  - Periods are replaced with underscores.
- Replication definition names for tables always begin with the prefix "*ra$*," followed by a unique alphanumeric identifier (maximum of 8 characters), and ending with a table or object name. For example, for a replicate name of "My Table," the resulting repdef name is "ra$0x7952_mytable." For an especially long replicate name of "mytable8901234567890123456789" (30 characters), the resulting repdef name is "ra$0x7952_mytable8901234567890" (30 characters maximum).
- Replication definition names for procedures are the same name as the procedure.

pdb_automark_tables

Determines if the Replication Agent automatically marks tables for replication during DDL replication.

Default: true

Values: true – user tables are automatically marked during DDL replication.

false – user tables are not automatically marked during DDL replication. They must always be marked using the pdb_setreptable command (default).

Comments

- The default value for pdb_automark_tables is set to true when a Replication Agent instance is created. In this default setting, all user tables (those whose owners are not contained in the owner_filter_list) are marked for replication when the pdb_xlog command is executed with the init keyword. In addition, when replication of DDL commands is enabled (pdb_setrepddl setting is enabled by default), any create table command for a user table (those whose owners are not contained in the owner_filter_list) is automatically marked for replication. If automatic marking of tables is not desired, this configuration parameter value should be changed to false. When set to true, tables are never automatically marked for replication.
- Automatic marking of new tables (those created in the primary database with the create table command) will only occur when replication of DDL commands is enabled (pdb_setrepddl is set to enable) and the table is a user table (those whose owners are not contained in the owner_filter_list) and pdb_automark_tables is set to true. Modifying the owner_filter_list after the initialization may cause inconsistencies.
- Tables are automatically unmarked for replication when a drop table command issued at the primary and is recorded in the transaction log, regardless of the settings of pdb_setrepddl or pdb_automark_tables. This is due to the fact that a dropped table cannot be replicated from.
- Automatic marking of user tables is independent of manual marking of tables using the pdb_setreptable command. In other words, you can always mark or unmark individual or all tables for replication using the pdb_setreptable command, regardless of the setting of pdb_automark_tables.

pdb_auto_run_scripts

Determines whether Replication Agent automatically runs scripts (for transaction log creation and removal, and object marking and unmarking) at the primary database.

Default

true

Values

true – Replication Agent automatically runs scripts.

false – Replication Agent generates and saves the scripts, but it does not automatically run them at the primary database.

Comments

- When the pdb_xlog command is invoked to create or remove the transaction log, Replication Agent generates a script to create or remove the transaction log base objects.

- When the pdb_setrepproc command is invoked to mark or unmark an object in the primary database, Replication Agent generates a script to create or remove the transaction log objects necessary for object marking.
- Replication Agent always saves the scripts in a file. Log creation and removal scripts are saved in files named *partinit.sql* and *partdeinit.sql*. Object marking and unmarking scripts are saved in files named *partmark.sql* and *unmark.sql*.
- When the pdb_auto_run_scripts parameter is set to false, the scripts are created but no action is taken. This allows you to review the scripts to see what action will be taken before execution. You cannot execute the scripts. You must set pdb_auto_run_scripts parameter back to true and re-execute the command to have the desired action take place.
- As described above for the pdb_xlog and pdb_setrepproc commands, Oracle and Microsoft SQL Server creates the *partinit, partdeinit, partmark,* and *partunmark* scripts. For Oracle and Microsoft SQL Server, these scripts can *not* be executed (since they do not update the RASD) and are for informational purposes only.
- This parameter must be set to true for initialization to occur.

pdb_convert_datetime

Determines whether Replication Agent converts non-Sybase temporal datatypes to the Sybase datetime format.

Default true

Values true – Replication Agent converts all data in the primary database native date/time datatypes to the Sybase datetime format.

false – Replication Agent replicates data in the primary database native datetime datatypes as character strings.

Comments

- Replication definitions created by the rs_create_repdef command always define the datatypes using available user defined datatypes (UDDs) that are installed in Replication Server. If you use the rs_create_repdef command, do not set the Replication Agent configuration parameter pdb_convert_datetime to true, as doing so converts date and timestamp datatypes to Sybase format, instead of UDD format.

- The pdb_convert_datetime parameter is provided for backward compatibility with previous versions of Replication Agents and Replication Server. If you use Replication Server version 12.0 or later, Sybase recommends that you use the Replication Server heterogeneous datatype support (HDS) feature for all datatype conversion and translation.
- Replication Agent checks the value of the pdb_convert_datetime parameter at the time an object is marked for replication. Transaction log objects that support replication of the marked object are constructed to provide the desired date format.

  If you change the value of the pdb_convert_datetime parameter after an object is marked, it has no effect on the marked object. To change the datetime datatype conversion for a marked object, you must unmark the object, change the value of the pdb_convert_datetime parameter, then re-mark the object.
- For log-based Replication Agents, the conversion takes place after the log records have been read and before LTL is generated to send to the Replication Server.
- Any missing component in the non-Sybase date/time datatype format is treated as an implied 0 (zero) when it is converted to the Sybase datetime format.
- When the value of the pdb_convert_datetime parameter is true, the replication definition for each table should specify that the declared datatype for all date/time columns is datetime.
- If the value of the pdb_convert_datetime parameter is false, the Replication Agent sends date/time data to the primary Replication Server as default-sized character strings. The default character string size varies by database and datatype:
  - DB2 Universal Database: DATE = char(10), TIME = char(8), TIMESTAMP = char(26)
  - Microsoft SQL Server: datetime or smalldatetime = char(23), timestamp = binary(8)
  - Oracle: DATE = char(19)
- Set the value of the pdb_convert_datetime parameter to true if *all* date/time values replicated from the primary database will be replicated as the Sybase datetime datatype.
- pdb_convert_datetime must be false if a table containing replicated LOB columns has datetime datatype in the primary key.

- Replication Agent date/time datatype conversion does not work with LOB column replication, unless either of the following conditions exist (these conditions are *not* required for Oracle and Microsoft SQL Server):
  - There are no date/time columns in the tables that have LOB column replication enabled, or
  - The primary keys in tables that have LOB column replication enabled do not contain date/time datatypes.

  Otherwise, if you use the pdb_setrepcol command to enable LOB column replication, you must set the value of the pdb_convert_datetime parameter to false.

  The Replication Agent for UDB is the only Replication Agent that must query LOB data directly from the primary database. To successfully query a primary database table for a LOB column value, any date column value must retain the primary database format and structure for the date value to appear correctly in the query. The format and structure for the date value cannot be converted to the Sybase datetime format.
- Sybase recommends that you set the value of the pdb_convert_datetime parameter to false for better Replication Agent throughput performance and optimal datatype handling.

pdb_dflt_column_repl

Determines whether LOB column replication is enabled by default when tables are marked.

Default

false

Values

true – LOB column replication is enabled by default (automatically) when tables are marked.

false – LOB column replication is disabled by default when tables are marked.

Comments

- If the value of the pdb_dflt_column_repl parameter is false when a table is marked for replication, no transactions that affect LOB columns in the table can be replicated until replication is explicitly enabled with the pdb_setrepcol command.
- You can use the pdb_setrepcol command to enable or disable replication for all LOB columns in all marked tables at once.

- When replication is disabled for a LOB column, any part of an operation that affects that column will not be recorded in the transaction log, even if the operation also affects other columns for which replication is enabled.

pdb_dflt_object_repl

Note This parameter is available only for Oracle and UDB.

Determines whether replication is enabled by default when objects (tables or stored procedures) are marked.

Default

true

Values

true – enables replication by default (automatically) when objects are marked.

false – disables replication by default when objects are marked.

Comments

- If the value of the pdb_dflt_object_repl parameter is false when a table is marked for replication, no transactions can be replicated from that table until replication is explicitly enabled with the pdb_setreptable command.
- If the value of the pdb_dflt_object_repl parameter is false when a stored procedure is marked for replication, no invocations of that stored procedure can be replicated until replication is explicitly enabled with the pdb_setrepproc command.
- You can use the pdb_setrepproc or pdb_setreptable command to enable or disable replication for all marked stored procedures or tables at once.
- When replication is disabled for a table, no operations that affect that table will be recorded in the transaction log.
- When replication is disabled for a stored procedure, no invocations of that stored procedure are recorded in the transaction log.
- For Microsoft SQL Server, the pdb_setrepproc command ignores the pdb_dflt_object_repl parameter. As a result, this command will always enable replication by default when objects are marked.

pdb_include_archives

Note This parameter is available only for Oracle.

Enables or disables the use of Oracle archive log files.

Default: true

Values: true – allows reading of the archived Oracle redo log files from the path specified by pdb_archive_path. The configuration of Oracle automatic archiving is supported under this mode. Use pdb_archive_remove to remove old archives logs that are no longer needed to support replication.

false – only online redo logs files are read. Oracle automatic archiving must be disabled. The Replication Agent executes Oracle archive commands to archive the redo logs once they are no longer needed for replication.

Comments:

- Set the configuration to true when use of archive logs is preferred or when Oracle must be configured to perform automatic archiving. Set this value to false if accessing only the on-line redo logs is preferred.
- Set this value to false if using only the online redo logs is preferred.

See also: pdb_archive_path, truncation_type, truncation_interval configuration parameters.

pdb_support_large_identifier

To support replication of large identifiers up to 255 characters in length with Replication Server 12.6 and later.

Default: false

Value: true – objects containing large identifiers may be marked for replication.

false – objects containing large identifiers may *not* be marked for replication.

Comments:

- If pdb_support_large_identifier value is false, when an object (Table/Procedure/Function) is being marked for replication, the object is checked for any identifiers that are longer than 30 characters. An error is returned and the object is not marked for replication if the object has identifiers longer than 30 characters.
- This parameter may be set to true if the Replication Server being used is at version 12.6 or later and the replicate database can support large identifiers.

- When pdb_support_large_identifier is set to true, objects being marked for replication are not checked for identifiers longer than 30 characters.

pdb_timezone_file

Note This parameter is available only for Oracle.

Specifies the file to read at Replication Agent initialization to obtain Oracle time zone information.

Default: <not configured>

Value: A valid path to the Oracle time zone file including the *time zone* file name.

Comments:

- If the value is not specified, it will default to the Oracle installation's *oracore/zoneinfo/timezone.dat* file. For example,

  ```
  $ORACLE_HOME/oracore/zoneinfo/timezone.dat
  ```

- The *timezone* file specified must be for the same release and platform as the primary Oracle database. For example, an Oracle 9i timezone file is not compatible with an Oracle 10g primary database, and a Windows *timezone* file is not compatible with UNIX.

pdb_xlog_device

The primary database device on which Replication Agent transaction log objects are created.

Default: NULL

Value: A valid primary database device name or NULL.

Comments:

- The value of the pdb_xlog_device parameter is the device specification of the primary database device to be used in SQL scripts generated by the Replication Agent to create transaction log objects.
- The pdb_xlog_device parameter allows you to specify a single device on which all Replication Agent transaction log objects will be created, even if the database uses multiple devices.

- If the value of the pdb_xlog_device parameter is NULL, no device is specified in the SQL create statements, and Replication Agent transaction log objects are placed on the primary data server's system-defined default device.

pdb_xlog_prefix

The prefix string used in database object names to identify Replication Agent transaction log objects.

Default

ra_

Value

A character string of 1 to 3 characters.

Comments

- When Replication Agent generates database object names for transaction log components in the primary database, it uses the value of the pdb_xlog_prefix parameter as an object name prefix.
- Replication Agent uses the value of the pdb_xlog_prefix parameter to recognize its transaction log objects in the primary database. Therefore, if you change the value of the pdb_xlog_prefix parameter after the transaction log objects are created, Replication Agent will not be able to find its transaction log objects.
- The value of the pdb_xlog_prefix_chars parameter specifies the non-alphabetic characters that can be used in the prefix string.

pdb_xlog_prefix_chars

Non-alphabetic characters that are allowed in the database object name prefix string that identifies Replication Agent transaction log objects.

Default

_#@ (DB2 Universal Database)

_$#@ (Microsoft SQL Server)

_#$ (Oracle)

Value

A string of characters with no separators.

Comments

- The default value of the pdb_xlog_prefix_chars parameter depends on the type of primary database that the Replication Agent instance was created for. The default value is based on the standard, non-alphabetic characters allowed by each non-Sybase database.

- When you set or change the value of the pdb_xlog_prefix_chars parameter, the new value replaces any existing value; it does not add or append the new value to a previous value.
- When you use the ra_config command to set the value of the pdb_xlog_prefix parameter, any non-alphabetic characters specified on the command line are validated against the value of the pdb_xlog_prefix_chars parameter.
- Alphabetic characters a-z are always valid in the pdb_xlog_prefix parameter, and they need not be specified.
- Replication Agent does not support delimited names for transaction log objects, so you cannot use a space character in the value of the pdb_xlog_prefix parameter.
- The value you specify for the pdb_xlog_prefix_chars parameter is not validated. There are no restrictions on the characters you can include.

Note The primary data server may restrict the characters used in certain positions in a database object name. Refer to the documentation for your primary data server for more information.

pds_connection_type

The type of connectivity driver used on the primary database connection.

Default — One of the following values is set automatically when the Replication Agent instance is created.

Values

MSSQLJDBC – Replication Agent uses the Microsoft SQL Server JDBC driver to connect to the primary Microsoft SQL Server database.

ORAJDBC – Replication Agent uses the Oracle JDBC driver to connect to the primary Oracle database.

UDBODBC – Replication Agent uses the DB2 Universal Database ODBC driver to connect to the primary database in DB2 Universal Database.

Comments

- The value of the pds_connection_type parameter is set automatically at the time a Replication Agent instance is created. The specific value depends on the type of Replication Agent instance created.

  Note Sybase recommends that you do *not* change the default value of the pds_connection_type parameter.

- The value of the pds_connection_type parameter determines which of several other Replication Agent configuration parameters related to the primary database connection must also have values specified.
  - MSSQLDBC requires corresponding values for the following parameters:
    - pds_server_name
    - pds_port_number
    - pds_database_name
  - UDBODBC requires corresponding values for the following parameters:
    - pds_database_name
    - pds_datasource_name
  - ORAJDBC requires corresponding values for the following parameters:
    - pds_host_name
    - pds_port_number
    - pds_database_name
  - If using the *tnsnames.ora* file the following parameters are required:
    - pds_tns_filename
    - pds_tns_connection
  - The value of the pds_connection_type parameter is automatically when a Replication Agent instance is created.

pds_dac_port_number

Note This parameter is available only for Microsoft SQL Server.

The dedicated administrative connection (DAC) port number for Microsoft SQL Server error log.

Default: 1111

Value: A valid port number for Microsoft SQL Server.

Comments:

- This parameter is required when executing the pdb_xlog init command against a Microsoft SQL Server for the first time. It is required to support installation of changes to the *mssqlsystemresource* database.
- For information regarding the use of the pds_dac_port_number parameter, see the "Microsoft SQL Server" chapter of the *Replication Agent Primary Database Guide*, under "Initialization of the primary data server and Replication Agent" and "First-time initialization."

See also: pdb_xlog configuration parameter.

pds_database_name

The name of the primary database.

Default: <not_configured>

Value: A valid database name.

Note You cannot set pds_database_name if pds_tns_connection is set.

Comments:

- The value of the pds_database_name parameter is the name of the primary database on the primary data server.

  Note Some primary data servers may not support multiple databases in a single instance of the data server. In that case, the value of the pds_database_name parameter should be the name of the data server instance.

- See the *Replication Agent Administration Guide* for more information about setting up Replication Agent connection configuration parameters.

pds_datasource_name

Note This parameter is available only for UDB.

The data source name (DSN) specified for the ODBC driver used on the primary database connection.

Default: <not_configured>

Value: A valid ODBC data source name.

Comments:

- The value of the pds_datasource_name parameter is the data source name (DSN) of the ODBC driver on the primary database connection.
- If the value of the pds_connection_type parameter is UDBODBC, the value of the pds_datasource_name parameter must be the database alias of the primary database in the DB2 Universal Database server.
- See the *Replication Agent Administration Guide* for more information about setting up Replication Agent connection configuration parameters.

pds_host_name

The name of the primary data server host machine.

Default: <not_configured>

Value: A valid host name.

Note You cannot set pds_host_name if the pds_tns_connection is set.

Comments:

- The value of the pds_host_name parameter is the network name of the host machine on which the primary data server resides.
- See the *Replication Agent Administration Guide* for more information about setting up Replication Agent connection configuration parameters.

pds_integrated_security

Note This parameter is available only for Microsoft SQL Server.

Determines if the Replication Agent should use Windows authentication when connecting to the primary SQL Server.

Default: false

Value: true – Specifies that Replication Agent should connect to the primary SQL Server using Windows authentication.

false – Specifies that Replication Agent should connect to the primary SQL Server using SQL Server authentication (default).

Comments:

- The default value for pds_integrated_security is set to false when a Replication Agent instance is created. In this default setting, you must configure the pds_password parameter, and the primary SQL Server must be configured to allow SQL Server authentication.
- Set this value to true to have the Replication Agent connect to the primary SQL Server using Windows authentication. The Windows environment and the primary SQL Server must be configured to use Windows authentication. For more information, see the Microsoft SQL Server section in the *Replication Agent Primary Database Guide*.

pds_password

The password that Replication Agent uses for primary data server access.

Default: "" (empty string)

Value: A valid password.

Comments:

- The value of the pds_password parameter is the password for the user login name that the Replication Agent uses to access the primary data server.
- The value of the pds_password parameter is encrypted in the Replication Agent instance configuration file.
- See the *Replication Agent Administration Guide* for more information about setting up Replication Agent connection configuration parameters.

pds_port_number

The client port number for the primary data server.

Default: 1111

Value

A valid port number on the primary data server host machine.

Note You cannot set pds_port_number if pds_tns_connection is set.

Comments

- The value of the pds_port_number parameter is the client port number for the primary data server.
- See the *Replication Agent Administration Guide* for more information about setting up Replication Agent connection configuration parameters.

pds_retry_count

The number of times the Replication Agent tries to establish a connection to the primary database.

Default

5

Value

An integer from 0 to 2,147,483,647.

Comments

- The value of the pds_retry_count parameter is the number of times that Replication Agent will try to establish a network connection to the primary database after a connection failure.
- Sybase recommends a setting of 5 for this parameter.
- See the *Replication Agent Administration Guide* for more information about setting up Replication Agent connection configuration parameters.

pds_retry_timeout

The number of seconds the Replication Agent waits between retry attempts to connect to the primary database.

Default

10

Value

An integer from 0 to 3600.

Comments

- The value of the pds_retry_timeout parameter is the number of seconds that the Replication Agent will wait between retry attempts to establish a network connection to the primary database after a connection failure.
- See the *Replication Agent Administration Guide* for more information about setting up Replication Agent connection configuration parameters.

pds_server_name

The server name of the primary data server.

Default: <not_configured>

Value: A valid server name.

Comments:

- The value of the pds_server_name parameter is the server name of the primary data server.
- See the *Replication Agent Administration Guide* for more information about setting up Replication Agent connection configuration parameters.

pds_tns_connection

Note This parameter is available only for Oracle.

The Oracle connection name that identifies the primary database connection in the Oracle *tnsnames.ora* file.

Default: <not configured>

Value: A valid primary database connection name from the Oracle *tnsnames.ora* file specified by the pds_tns_filename configuration parameter.

Comments:

- Setting of the configuration parameter overrides settings of the configuration parameters pds_host_name, pds_database_name, and pds_port_number.
- This configuration parameter is required when the Oracle data server instance to be replicated is part of a Real Application Cluster (RAC) configuration.

See also: pds_tns_filename configuration parameter

pds_tns_filename

Note This parameter is available only for Oracle.

The fully-qualified file name identifying the Oracle *tnsnames.ora* file that contains connection parameters for the primary Oracle data server.

Default	<not configured>
Value	A valid Oracle *tnsnames.ora* file that contains the connection parameters to the primary Oracle data server. This file normally resides in the *ORACLE_HOME\network\admin* directory.
Comments	• When configured, Replication Agent will use the connection information specified in the interfaces file to connect to the primary database and the pds_host_name and the pds_port_number are ignored. The pds_tns_connection name should be configured to the entry name in the Sybase interfaces file when pds_tns_filename is configured. • Setting of the configuration parameter is required when the Oracle data server instance to be replicated is part of a Real Application Clusters (RAC) configuration.
	Warning! The Replication Agent process must have *read* permission to this file. Access failures will prevent the Replication Agent from connecting to the Oracle server.
See also	pds_tns_connection configuration parameter

pds_username

The user login name that the Replication Agent uses for primary data server access.

Default	<not_configured>
Value	A valid user name.
Comments	• The value of the pds_username parameter is the login name that the Replication Agent uses to log in to the primary data server. This login name must be defined in the primary data server, with appropriate privileges or authority in the primary database. • The Replication Agent uses this login to access primary database objects and to create, remove, and manage its transaction log objects in the primary database. • See the *Replication Agent Administration Guide* for more information about setting up Replication Agent connection configuration parameters.

ra_retry_count

The number of times the Replication Agent attempts to restart replication after a failure.

Default: 2

Value: An integer greater than 0.

Comments:

- The value of the ra_retry_count parameter is the number of times that the Log Transfer Manager component will try to get the Replication Agent instance back into *Replicating* state after a failure or error causes the instance to go to *Admin* state.
- When a network connection fails, the Replication Agent attempts to re-establish the connection, using the values stored in its connection configuration parameters for that connection.
- If the Replication Agent is unable to re-establish a connection after the number of retries specified in the pds_retry_count or rs_retry_count parameter, then the Replication Agent instance goes to *Admin* state and the Log Transfer Manager component attempts to return the Replication Agent instance to *Replicating* state, based on the settings of the ra_retry_count and ra_retry_timeout parameters.

ra_retry_timeout

The number of seconds the Replication Agent waits between attempts to restart replication after a failure.

Default: 10

Value: An integer greater than 0.

Comment: The value of the ra_retry_timeout parameter is the number of seconds that the Log Transfer Manager component will wait between its attempts to get the Replication Agent instance back into *Replicating* state after a failure causes the instance to go to *Admin* state.

rasd_backup_dir

Note This parameter is available only for Oracle and Microsoft SQL Server.

The directory path for Replication Agent System Database (RASD) backup files.

Default

The path to the RASD *backup* directory created automatically when the Replication Agent instance was created. For example:

- On Microsoft Windows platforms:

  %SYBASE%\RAX-15_2*inst_name*\repository\backup

  where:

  - *%SYBASE%* is the path to the Replication Agent installation directory.
  - *inst_name* is the name of the Replication Agent instance.

- On UNIX platforms:

  $SYBASE/RAX-15_2/*inst_name*/repository/backup

  where:

  - *$SYBASE* is the path to the Replication Agent installation directory.
  - *inst_name* is the name of the Replication Agent instance.

Value

A valid path on the Replication Agent host machine.

Comments

- When you create a Replication Agent instance, an RASD *backup* directory is created automatically as part of the instance directory structure. The default value of the rasd_backup_dir parameter points to that directory.
- If you specify any valid path as the value of the rasd_backup_dir parameter, Replication Agent places its RASD backup files in that directory during RASD backup operations, and it looks in that directory for the RASD backup files during restore operations.

rasd_database

Note This parameter is available only for Oracle and Microsoft SQL Server.

The directory path for the Replication Agent System Database (RASD) database file.

Default

The path to the RASD database file created automatically when the Replication Agent instance was created. For example:

- On Microsoft Windows platforms:

  %SYBASE%\RAX-15_2*inst_name*\repository*inst_name*.db

  where:

  - *%SYBASE%* is the path to the Replication Agent installation directory.
  - *inst_name* is the name of the Replication Agent instance.

- On UNIX platforms:

  $SYBASE/RAX-15_2/*inst_name*/repository/*inst_name*.db

  where:

  - *$SYBASE* is the path to the Replication Agent installation directory.
  - *inst_name* is the name of the Replication Agent instance.

Value

A valid path and RASD database file name on the Replication Agent host machine.

Comments

- When you create a Replication Agent instance, the *repository* directory and the RASD database file are created automatically. The default value of the rasd_database parameter points to the RASD database file in that directory.
- If you specify any valid path and RASD database file name as the value of the rasd_database parameter, the Replication Agent instance looks in that directory for its RASD database file, with the file name you specified.

rasd_mirror_tran_log

Note This parameter is available only for Oracle and Microsoft SQL Server.

Enables or disables Replication Agent System Database (RASD) transaction log mirroring.

Default

false

Values

true – enables mirroring the RASD transaction log to another file.

false – disables mirroring of the RASD transaction log.

Comment

Setting the value of the rasd_mirror_tran_log parameter to true provides additional recovery options in the event of a device failure on the Replication Agent host machine.

rasd_trace_log_dir

Note This parameter is available only for Oracle and Microsoft SQL Server.

The directory path for the Replication Agent System Database (RASD) trace log file.

Default

The path to the *repository* directory created automatically when the Replication Agent instance was created. For example:

- On Microsoft Windows platforms:

  %SYBASE%\RAX-15_2*inst_name*\repository

  where:

  - *%SYBASE%* is the path to the Replication Agent installation directory.
  - *inst_name* is the name of the Replication Agent instance.

- On UNIX platforms:

  $SYBASE/RAX-15_2/*inst_name*/repository

  where:

  - *$SYBASE* is the path to the Replication Agent installation directory.
  - *inst_name* is the name of the Replication Agent instance.

Value

A valid path on the Replication Agent host machine.

Comments

- When you create a Replication Agent instance, the *repository* directory is created automatically as part of the instance directory structure. The default value of the rasd_trace_log_dir parameter points to that directory.
- If you specify any valid path as the value of the rasd_trace_log_dir parameter, the Replication Agent instance writes its RASD trace log file in that directory.

rasd_tran_log

Note This parameter is available only for Oracle and Microsoft SQL Server.

The directory path for the Replication Agent System Database (RASD) transaction log file.

Default

The path to the RASD transaction log file created automatically when the Replication Agent instance was created. For example:

- On Microsoft Windows platforms:

```
%SYBASE%\RAX-15_2\inst_name\repository\inst_name.log
```

where:

- *%SYBASE%* is the path to the Replication Agent installation directory.
- *inst_name* is the name of the Replication Agent instance.

- On UNIX platforms:

```
$SYBASE/RAX-15_2/inst_name/repository/inst_name.log
```

where:

- *$SYBASE* is the path to the Replication Agent installation directory.
- *inst_name* is the name of the Replication Agent instance.

Value

A valid path on the Replication Agent host machine.

Comments

- When you create a Replication Agent instance, the *repository* directory and RASD transaction log file are created automatically. The default value of the rasd_tran_log parameter points to that transaction log file.
- If you specify any valid path and RASD transaction log file name as the value of the rasd_tran_log parameter, the Replication Agent instance looks in that directory for its RASD transaction log file, with the name you specified.

rasd_tran_log_mirror

Note This parameter is available only for Oracle and Microsoft SQL Server.

The directory path for the Replication Agent System Database (RASD) transaction log file mirror.

Default

The path to the RASD transaction log file mirror in the *tran_log_mirror* directory created automatically when the Replication Agent instance was created. For example:

- On Microsoft Windows platforms:

%SYBASE%\RAX-15_2*inst_name*\repository\tran_log_mirror*inst_name*.log

where:

- *%SYBASE%* is the path to the Replication Agent installation directory.
- *inst_name* is the name of the Replication Agent instance.

- On UNIX platforms:

$SYBASE/RAX-15_2/*inst_name*/repository/tran_log_mirror/*inst_name*.log

where:

- *$SYBASE* is the path to the Replication Agent installation directory.
- *inst_name* is the name of the Replication Agent instance.

Value

A valid path on the Replication Agent host machine.

Comment

If you specify any valid path and transaction log file name as the value of the rasd_tran_log_mirror parameter, the Replication Agent instance looks in that directory for its RASD transaction log file mirror, with the name you specified.

rs_charset

The character set used in communication with the primary Replication Server.

The Replication Agent default character set must be set to match the primary database's character set. The value of the rs_charset parameter must be set to match Replication Server's character set. If they differ, Replication Agent will do character set conversion before sending data to Replication Server.

If the character set on your Replication Agent is different from 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 on setting the RA_JAVA_DFLT_CHARSET environment variable, see Chapter 2 of the *Replication Agent Administration Guide*.

Note Setting this parameter to anything other than the character set of the primary Replication Server causes Replication Agent to do character set conversion before sending data to Replication Server. This will degrade the Replication Agent performance.

Default

Defaults to empty string ("").

Value

Any valid Sybase character set supported by the Java VM on the Replication Agent host machine.

Note If Replication Agent can connect to Replication Server 15.0.1 or later, the rs_charset in Replication Agent is ignored and the rs_charset in Replication Server is used.

Comments

- Sybase recommends that you use the exact same value as that of the RS_charset parameter in the Replication Server configuration (*.cfg*) file which is located at: *$SYBASE/REP-15_0/install/<instance>.cfg*. For example, *iso_1*.
- Sybase recommends that you configure the primary data server and primary Replication Server to use the same character set.

  Note If rs_charset is not set at the time you try to resume replication, Replication Agent returns an error.

  When the Replication Agent instance is created, the rs_charset parameter is set to its default value "" (empty string).
- If you specify a valid character set for the value of the rs_charset parameter, the Replication Agent instance sends replicated transaction data from the primary database to the primary Replication Server in that character set.
- If you do *not* specify a valid character set name for the value of the rs_charset parameter (including the default rs_charset value ""), the Replication Agent instance will not allow you to resume replication.

- If the values of the rs_charset and the system default character set are valid but *not* the same value, Replication Agent converts the replicated transaction data from the system-defined database character set to the Replication Server character set before sending it to the primary Replication Server.
- See the *Replication Agent Administration Guide* for more information about setting up Replication Agent connection configuration parameters.

rs_host_name

The name of the primary Replication Server host machine.

Default: <not_configured>

Value: A valid host name.

Comments:

- The value of the rs_host_name parameter is the name of the host machine for the primary Replication Server.
- See the *Replication Agent Administration Guide* for more information about setting up Replication Agent connection configuration parameters.

rs_packet_size

The network packet size on the connection to the primary Replication Server.

Default: 2048

Value: An integer from 2048 to 8192.

Comments:

- The value of the rs_packet_size parameter is the maximum size (in bytes) of the network packets handled by the TCP/IP network protocol.
- The Replication Agent rs_packet_size parameter is equivalent to the Replication Server rs_packet_size parameter.
- When the network packet size is smaller, more packets must be processed to transmit a given amount of data to the Replication Server. When the network packet size is larger, more system resources are consumed to process the packets.
- The optimum value of the rs_packet_size parameter is based on the nature of the typical data replicated. If the typical operation is very large, a larger packet size is more efficient.

- A larger value of the rs_packet_size parameter is more efficient when the value of the lti_batch_mode parameter is true.
- See the *Replication Agent Administration Guide* for more information about setting up Replication Agent connection configuration parameters.

rs_password

The password that Replication Agent uses for Replication Server access.

Default: "" (empty string)

Value: A valid password.

Comments:

- The value of the rs_password parameter is the password for the user login name that Replication Agent uses to log in to the primary Replication Server.
- The value of the rs_password parameter is encrypted in the Replication Agent instance configuration file.
- See the *Replication Agent Administration Guide* for more information about setting up Replication Agent connection configuration parameters.

rs_port_number

The client port number of the primary Replication Server.

Default: 1111

Value: A valid port number on the Replication Server host machine.

Comments:

- The value of the rs_port_number parameter is the client port number of the primary Replication Server.
- See the *Replication Agent Administration Guide* for more information about setting up Replication Agent connection configuration parameters.

rs_replicate_owner_required

Indicates if the owner is always included with the replicate table clause when generating replication definitions.

Default: true

Value	true – the owner is always included in the replicate table clause. false – the owner is not included in the replicate table clause unless the table is marked with the owner mode value set to on.
Comments	For additional information, see rs_create_repdef command.

rs_retry_count

The number of times the Replication Agent will retry establishing a connection to the primary Replication Server.

Default	5
Value	An integer greater than 0.
Comments	• The value of the rs_retry_count parameter is the number of times that Replication Agent will try to establish a network connection to the Replication Server after a connection failure. • Sybase recommends a setting of 5 for this parameter. • See the *Replication Agent Administration Guide* for more information about setting up Replication Agent connection configuration parameters.

rs_retry_timeout

The number of seconds the Replication Agent waits between attempts to connect to the primary Replication Server.

Default	10
Value	An integer greater than 0.
Comments	• The value of the rs_retry_timeout parameter is the number of seconds that the Replication Agent will wait between its retry attempts to establish a network connection to the primary Replication Server after a connection failure. • See the *Replication Agent Administration Guide* for more information about setting up Replication Agent connection configuration parameters.

rs_source_db

The name of the database identified in the Replication Server primary database connection.

Default: <not_configured>

Value: A valid database name.

Comments:

- The value of the rs_source_db parameter is the name of the primary database by which the primary Replication Server recognizes the primary database transaction log.
- The value of the rs_source_db parameter must match the name of the database specified in the Replication Server create connection command for the primary database.
- See the *Replication Agent Administration Guide* for more information about setting up Replication Agent connection configuration parameters.

rs_source_ds

The name of the data server identified in the Replication Server primary database connection.

Default: <not_configured>

Value: A valid server name.

Comments:

- The value of the rs_source_ds parameter is the name of the primary data server by which the primary Replication Server recognizes the primary database transaction log.
- The value of the rs_source_ds parameter must match the name of the data server specified in the Replication Server create connection command for the primary database.
- The value of the rs_source_ds parameter should *not* be the same as the name of the Replication Agent instance.
- See the *Replication Agent Administration Guide* for more information about setting up Replication Agent connection configuration parameters.

rs_ticket_version

Determines whether Replication Agent records the primary database time or the primary database date and time into the rs_ticket marker.

Default: 1

Value:
- 1 – Replication Agent records only the primary database time.
- 2 – Replication Agent records both the primary database date and time.

Comments:
- If the value is set to 1, Replication Agent records only the primary database time into rs_ticket marker. For example, 13:20:19.368.
- If the value is set to 2, Replication Agent records both the primary database date and time into rs_ticket marker. For example, 12/14/07 13:20:19.368.

See also: rs_ticket command.

rs_username

The user login name that the Replication Agent uses for Replication Server access.

Default: <not_configured>

Value: A valid user name.

Comments:
- The value of the rs_username parameter is the user login name that Replication Agent uses to log in to the primary Replication Server.
- The value of the rs_password parameter is the password for the login name specified by the rs_username parameter.
- The user login name that Replication Agent uses to log in to the Replication Server must have connect source permission in the Replication Server.
- See the *Replication Agent Administration Guide* for more information about setting up Replication Agent connection configuration parameters.

rssd_charset

The character set used in communication with the RSSD of the primary Replication Server.

Default	"" (empty string)
Value	Any valid Sybase character set supported by the Java VM on the Replication Agent host machine.
Comments	• The value of the rssd_charset parameter must match (or be compatible with) the RSSD character set. The RSSD character set is usually the same as the Replication Server default character set identified by the Replication Server rs_charset configuration parameter. • If you specify a valid character set for the value of the rssd_charset parameter, the Replication Agent instance communicates with the RSSD using that character set. • If you do *not* specify a valid character set name for the value of the rssd_charset parameter (including the default rssd_charset value ""), the Replication Agent communicates with the RSSD using the RSSD charset. • The rssd_charset parameter does not need to be set if the Replication Agent use_rssd parameter is set to false. • See the *Replication Agent Administration Guide* for more information about setting up Replication Agent connection configuration parameters.

rssd_database_name

The database name of the RSSD of the primary Replication Server.

Default	<not_configured>
Value	A valid database name.
Comments	• The value of the rssd_database_name parameter is the database name of the RSSD of the primary Replication Server. • The rssd_database_name parameter does not need to be set if the Replication Agent use_rssd parameter is set to false. • See the *Replication Agent Administration Guide* for more information about setting up Replication Agent connection configuration parameters.

rssd_host_name

The name of the machine on which the RSSD of the primary Replication Server resides.

Default	<not_configured>
Value	A valid host name.
Comments	• The value of the rssd_host_name parameter is the name of the host machine on which the RSSD of the primary Replication Server resides. • The rssd_host_name parameter does not need to be set if the Replication Agent use_rssd parameter is set to false. • See the *Replication Agent Administration Guide* for more information about setting up Replication Agent connection configuration parameters.

rssd_password

The password that the Replication Agent uses for access to the RSSD of the primary Replication Server.

Default	"" (empty string)
Value	A valid password.
Comments	• The value of the rssd_password parameter is the password for the user login name that the Replication Agent uses to access the RSSD of the primary Replication Server. • The value of the rssd_password parameter is encrypted in the Replication Agent instance configuration file. • The rssd_password parameter need not be set if the Replication Agent use_rssd parameter is set to false. • See the *Replication Agent Administration Guide* for more information about setting up Replication Agent connection configuration parameters.

rssd_port_number

The client port number of the Replication Server System Database (RSSD) of the primary Replication Server.

Default	1111
Value	A valid port number on the RSSD host machine.
Comments	• The value of the rssd_port_number parameter is the client port number of the RSSD data server.

- The rssd_port_number parameter need not be set if the Replication Agent use_rssd parameter is set to false.
- See the *Replication Agent Administration Guide* for more information about setting up Replication Agent connection configuration parameters.

rssd_username

The user login name that the Replication Agent uses to access the RSSD of the primary Replication Server.

Default: <not_configured>

Value: A valid user login name in the RSSD data server.

Comments:

- The value of the rssd_username parameter is the user login name that the Replication Agent uses to access the RSSD.
- The rssd_username parameter need not be set if the Replication Agent use_rssd parameter is set to false.
- See the *Replication Agent Administration Guide* for more information about setting up Replication Agent connection configuration parameters.

scan_sleep_increment

The number of seconds to add to each wait interval before scanning the transaction log, after a previous scan yields no transaction to be replicated.

Default: 5

Value: An integer from 0 to 3600.

Comments:

- The value of the scan_sleep_increment parameter is the number of seconds added to each wait interval before the Log Reader component scans the log device for a transaction to be replicated, after a previous scan yields no such transaction.
- The number of seconds specified by the scan_sleep_increment parameter is added to each wait interval, until the wait interval reaches the value specified by the scan_sleep_max parameter.

- For optimum Replication Agent performance, the value of the scan_sleep_increment parameter should be balanced with the average number of operations in the primary database over a period of time. In general, better performance results from reading more operations from the transaction log during each Log Reader scan.
- With a primary database that is less frequently updated, increasing the value of the scan_sleep_increment parameter may improve overall performance.
- If the database is continuously updated, the value of the scan_sleep_increment parameter may not be significant to Replication Agent performance.

scan_sleep_max

The maximum wait interval between Log Reader transaction log scans.

Default: 60

Value: An integer from 5 to 86400.

Comments:

- The value of the scan_sleep_max parameter is the maximum number of seconds that can elapse before the Log Reader component scans the transaction log for a transaction to be replicated, after a previous scan yields no such transaction.
- For reduced replication latency in an infrequently updated database, Sybase recommends lower number settings for the scan_sleep_max parameter.
- If the primary database is continuously updated, the value of the scan_sleep_max parameter is not significant to Replication Agent performance.

skip_ltl_errors

Determines whether the Replication Agent ignores Log Transfer Language (LTL) error messages.

Warning! Using the skip_ltl_errors parameter incorrectly may cause data inconsistencies between the primary and replicate databases.

Default	false
Values	true – enables skipping LTL errors to continue replication. false – disables skipping LTL errors.
Comments	• If the skip_ltl_errors configuration parameter is set to true, the Replication Agent instance logs any LTL error messages returned by the Replication Server, along with the offending LTL commands, and then it continues processing transaction log records. • If the skip_ltl_errors configuration parameter is set to false, the Replication Agent instance stops all of its replication processing and goes to *Admin* state when it receives an LTL error message and the error is unrecoverable. • The skip_ltl_errors parameter is intended for troubleshooting only, with assistance from Sybase Technical Support.

structured_tokens

Determines whether the Replication Agent uses LTL structured tokens.

Default	true
Values	true – enables LTL structured tokens. false – disables LTL structured tokens.
Comments	• If the structured_tokens configuration parameter is set to true, the Log Transfer Interface (LTI) component uses LTL structured tokens when it generates LTL commands. • Using structured tokens in the LTL can significantly improve overall replication system performance. • Using structured tokens in the LTL can improve Replication Server performance, especially when non-Sybase datatypes in the primary database must be translated by Replication Server. • To replicate columns that have one or more spaces in the column name, you must set the value of the structured_tokens parameter to true.

truncation_interval

Specifies a time interval between automatic truncations of the Replication Agent transaction log.

Warning! If you configure automatic truncation, For UDB, the Replication Agent silently deletes the primary database log files, or for Oracle the Replication Agent deletes the archive log files that it no longer needs. For more information, see the *Replication Agent Primary Database Guide*.

Default: 0

Value: An integer from 0 to 720.

Comments:

- The value of the truncation_interval parameter is the number of minutes between automatic transaction log truncations.
- Automatic transaction log truncation based on the value of the truncation_interval parameter takes place only when the value of the truncation_type parameter is interval.
- The maximum truncation interval is 720 minutes, or 12 hours.
- If the value of the truncation_interval parameter is 0 (zero) and the value of the truncation_type parameter is interval (the default values for both parameters), automatic truncation is disabled.
- To truncate the transaction log manually, use the pdb_truncate_xlog command.

See also: pdb_archive_path, pdb_archive_remove, and truncation_type configuration parameters

truncation_type

Configures transaction log truncation behavior of the Replication Agent.

Warning! For UDB, the Replication Agent deletes the UDB primary database log files and for Oracle the Replication Agent deletes the archived log files that it no longer needs. For more information, see the *Replication Agent Primary Database Guide*.

Default: locator_update

Values

command – Replication Agent truncates the transaction log only when the pdb_truncate_xlog command is invoked.

When the value of the truncation_type parameter is command, the only way you can truncate the transaction log is by invoking the pdb_truncate_xlog command. No automatic truncation takes place when the value of the truncation_type parameter is command.

interval – Replication Agent automatically truncates the transaction log when determined by a configurable interval of time.

locator_update – Replication Agent automatically truncates the transaction log whenever it receives a new LTM Locator value from the primary Replication Server.

When the value of the truncation_type parameter is locator_update, the transaction log is automatically truncated when Replication Agent receives a new LTM Locator from the primary Replication Server.

Comments

Note For Oracle and UDB, truncation of the archive log files that Replication Agent no longer needs from the pdb_archive_path directory is performed only if the pdb_archive_remove parameter is true.

- Regardless of the value of the truncation_type parameter, you can truncate the Replication Agent transaction log manually at any time by invoking the pdb_truncate_xlog command.
- If the value of the truncation_interval parameter is 0 (zero) and the value of the truncation_type parameter is interval (the default values for both parameters), automatic truncation is disabled.
- Replication Agent receives a new LTM Locator based on the values of the lti_update_trunc_point parameter.

See also

pdb_archive_path, pdb_archive_remove, and truncation_interval configuration parameters

use_rssd

Determines whether the Replication Agent uses replication definitions.

Default

true

Values

true – enables using replication definitions.

false – disables using replication definitions.

Comments

- If the value of the use_rssd parameter is true, the Replication Agent instance connects to the Replication Server System Database (RSSD) to retrieve replication definitions for the primary database automatically whenever it goes from *Admin* state to *Replicating* state (for example, when the resume command is invoked).
  - Each time it retrieves replication definitions, Replication Agent stores the information in a cache. Replication Agent uses replication definitions stored in its cache when it generates Log Transfer Language (LTL) commands.
  - If the Log Transfer Interface (LTI) component encounters an operation on a database object for which it does not have a cached replication definition, Replication Agent reconnects to the RSSD to update its replication definition cache.
  - If a replication definition still cannot be found for the operation, the Replication Agent instance suspends all of its replication operations and goes to *Admin* state.
- Replication Agent can use information in table and function replication definitions (that is, replication definitions for individual primary database objects) stored in the RSSD to generate more efficient LTL, and thus improve throughput in the LTI component and the Replication Server.

  Accessing replication definitions in the RSSD enables the LTI component to improve performance by:
  - Omitting column names in LTL. When columns are sent in the order specified in the replication definition, column images can be sent without column names (headings), which reduces LTL overhead.
  - Omitting unneeded columns in LTL. When columns are sent as specified in the replication definition, images for unchanged columns need not be sent, which reduces LTL overhead.
  - Sending data for each column in the datatype specified by the replication definition. This allows data to be handled more efficiently all the way through the replication system.
  - Sending database object names in the same character case as defined in the replication definition.
- If the value of the use_rssd parameter is false, none of the previously described performance improvements are possible. In that case, the Replication Agent sends all data as a char datatype in the LTL.

Glossary

This glossary describes Replication Agent terms.

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.
- SQL Anywhere (formerly Adaptive Server Anywhere) manages relational databases with a small DBMS footprint, which is ideal for embedded applications and mobile device applications.

See also **DBMS** and **RDBMS**.

atomic materialization

A materialization method that copies subscription data from a primary database to a standby 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. Bulk copying provides an alternative to using SQL insert and select commands to transfer data.

bulk materialization

A materialization method whereby subscription data in a standby 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** 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. See also **transaction**. Contrast with **rollback**.

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 different 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 **disk replication** and **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 also 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**, **DBMS**, and **RDBMS**.

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 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 database management system, 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 standalone data server system. Compare with **RDBMS**.

disaster recovery A method or process used to restore the critical business functions interrupted by a catastrophic event. A disaster recovery (or business continuity) plan defines the resources and procedures required for an organization to recover from a disaster, based on specified recovery objectives.

ERSSD An abbreviation for Embedded Replication Server System Database, which manages replication system information for a Replication Server. See also **Replication Server**.

failback A procedure that restores the normal user and client access to a primary database, after a failover procedure switched access from the primary database to a standby database. See also **failover**.

failover A procedure that switches user and client access from a primary database to a standby database, particularly in the event of a failure that interrupts operations at the primary database, or access to the primary database. Failover is an important fault-tolerance feature for systems that require high availability. See also **failback**.

function A Replication Server object that represents a data server operation such as insert, delete, or begin transaction. Replication Server distributes operations to standby 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 heterogeneous replication, in which the primary and standby 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, Replication Agent, and Replication Server. See also **Open Client** and **Open Server**.

Java An object-oriented programming language developed by Sun Microsystems. A platform-independent, “write once, run anywhere” programming language.

Java VM — The Java Virtual Machine. The Java VM (or JVM) is the part of the Java Runtime Environment (JRE) that is responsible for interpreting Java byte codes. See also **Java** and **JRE**.

JDBC — An abbreviation for Java Database Connectivity. JDBC is the standard communication protocol for connectivity between Java clients and data servers. See also **data server** and **Java**.

JRE — An abbreviation for Java Runtime Environment. The JRE consists of the Java Virtual Machine (Java VM or JVM), the Java Core Classes, and supporting files. The JRE must be installed on a machine to run Java applications, such as the Replication Agent. See also **Java VM**.

LAN — An abbreviation for "local area network," a computer network located on the user's premises that covers 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 standby database. Specifically, latency is the time elapsed between committing an original transaction in the primary database and committing the replicated transaction in the standby database.

In disk replication, latency is the time elapsed between a disk write operation that changes a block or page on a primary device and the disk write operation that changes the replicated block or page on a replicate device.

See also **disk replication** and **transaction replication**.

LOB — An abbreviation for large object, a type of data element that is 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 standby database. See also **Log Reader** and **Log Transfer Manager**.

Log Transfer Manager — An internal component of the Replication Agent that interacts with the other Replication Agent internal components to control and coordinate Replication Agent operations. See also **Log Reader** and **Log Transfer Interface**.

maintenance user A special user login name in the standby database that Replication Server uses to apply replicated transactions to the database. See also **Replication Server**.

materialization The process of copying the data from a primary database to a standby database, initializing the standby database so that the Replication Agent system can begin replicating transactions. See also **atomic materialization**, **bulk materialization**, and **nonatomic 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 standby databases. Contrast with **atomic materialization**. See also **bulk materialization**.

ODBC An abbreviation for Open Database Connectivity, an industry-standard communication protocol for clients connecting to data servers. See also **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 standby database. See also **inbound queue** 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 **Replication Agent**, **primary database**, and **Replication Server**.

primary database The database that contains the data to be replicated to another database (the standby database) through a replication system. The primary database is the database that is the source of replicated data in a replication system. Sometimes called the active database. Contrast with **standby database**. See also **primary data**.

primary key The column or columns whose data uniquely identify each row in a table.

primary site — The location or facility at which primary data servers and primary databases are deployed to support normal business operations. Sometimes called the active site or main site. See also **primary database** and **standby site**.

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**, **replicated transaction**, 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 — In a replication system, a state in which all updates have been propagated to their destinations. Some Replication Agent and Replication Server commands require that you first quiesce the replication system.

In a database, a state in which all data updates are suspended so that transactions cannot change any data and the data and log devices are stable.

This term is interchangeable with quiesced and in quiesce. See also **quiesce**.

RASD — An abbreviation for Replication Agent System Database. 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 Replication Command Language, the command language used to manage Replication Server.

RDBMS — An abbreviation for relational database management system, 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. See also **SQL**.

replicated data — A set of data that is replicated from a primary database to a standby database by a replication system. See also **primary database**, **replication system**, and **standby database**.

replicated transaction — A primary transaction that is replicated from a primary database to a standby database by a transaction replication system. See also **primary database**, **primary transaction**, **standby 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 standby database. See also **primary 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 Replication Server System Database, which 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 **disk replication** and **transaction replication**.

rollback An instruction to a database to back out of the changes requested in a unit of work (called a transaction). Contrast with **commit**. See also **transaction**.

SQL An abbreviation for Structured Query Language, a nonprocedural 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 standby 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**.

standby data The data managed by a standby database, which is the destination (or target) of a replication system. See also **data replication** and **standby database**.

standby database — A database that contains data replicated from another database (the primary database) through a replication system. The standby database is the database that receives replicated data in a replication system. Sometimes called the replicate database. Contrast with **primary database**. See also **standby data**.

standby site — The location or facility at which standby data servers and standby databases are deployed to support disaster recovery, and normal business operations during scheduled downtime at the primary site. Sometimes called the alternate site or replicate site. Contrast with **primary site**. See also **standby database**.

subscription — A request for Replication Server to maintain a replicated copy of a table, or a set of rows from a table, in a standby database at a specified location. See also **replication definition** and **Replication Server**.

table — In a relational DBMS, 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 for the table. See also **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 **SQL**.

transaction log — Generally, the log of transactions that affect the data managed by a data server. Replication Agent reads the transaction log to identify and acquire the transactions to be replicated from the primary database. See also **Replication Agent**, **primary database**, and **Replication Server**.

transaction replication — A data replication method that copies data-changing operations from a primary database transaction log to a standby database. See also **data replication** and **disk replication**.

transactional consistency — A condition in which all transactions in the primary database are applied in the standby database, in the same order that they were applied in the primary database.

WAN — An abbreviation for “wide area network,” a system of local-area networks (LANs) connected together with data communication lines. Contrast with **LAN**.

Index

D

E

F

G

H

I

J

L

M

N

O

P

Q

R

S

T

U

V

W