New Features
Replication Agent™ 15.0

Document ID: DC00503-01-1500-01

Last revised: November 2006

This document describes the new features that are available for Replication Agent 15.0 for Linux, Microsoft Windows, and UNIX.

Oracle support

Replication Agent 15.0 supports replication from Oracle 10g, including the two new binary datatypes. It also supports reading from Oracle archive logs as well as online redo logs for both Oracle 9i and 10g.

Oracle archive logs

Three new configuration properties were created for archive log management:

- pdb_archive_path, which allows Replication Agent to specify an archive log file path
- pdb_archive_remove, which allows Replication Agent to remove archived redo logs from the specified path
- pdb_include_ archive, which allows Replication Agent to read both the archive log files and the online redo log files

For a detailed description of each of these properties, see the Sybase Replication Agent *Reference Guide*.

Oracle 10g support

Replication Agent supports replication from Oracle 10g, including the following new datatypes:

Table 1: Oracle to Sybase datatype mapping

Oracle datatype	Oracle length/range	Sybase datatype	Sybase length/range
BINARY_FLOAT	5 bytes, 32-bit single precision floating point number datatype	float	4 or 8 bytes, depending on precision
BINARY_DOUBLE	9 bytes, 64-bit single precision floating point number datatype	double	8 bytes

For more information, see the Replication Agent *Primary Database Guide*.

Universal Database version 8 support

Replication Agent supports the IBM DB2 Universal Database (UDB) version 8. For more information, see the Replication Agent *Primary Database Guide*.

Replication Server version 15.0 support

To support Replication Server 15.0, Replication Agent supports the following configuration property and datatypes.

New configuration property

The new configuration property pdb_support_large_identifier enables or disables the replication of large identifiers up to 255 characters in length with Replication Server 15.0.

Unsigned datatype support

The following table shows the unsigned datatypes now supported by Replication Agent 15.0.

Table 2: Unsigned integer replication definition datatype mapping

Replication Server 15.0 unsigned datatypes	Replication definition datatypes
unsigned bigint	NUMERIC (20)
unsigned int	NUMERIC (10)
unsigned smallint	INT
unsigned tinyint	TINYINT

For more information, see the Replication Agent *Primary Database Guide* and the Replication Agent *Administration Guide*.

SySAM 2.0

Version 2.0 of Sybase Software Asset Management (SySAM) has replaced SySAM 1.0. Version 2.0 supports required asset management capabilities and allows Sybase to improve its license management.

In SySAM 2.0, you must download and manually install licenses through the Sybase Product Download Center (SPDC). To enforce licensing, Sybase now requires a valid license for each platform that is associated with a specific Replication host or License server host.

For more information, see the Sybase Software Asset Management 2.0 *User's Guide* and the Replication Agent *Install Guide*.

Migration from version 12.6 to 15.0

To upgrade to the new Replication Agent 15.0 software from Replication 12.6, follow the process defined in Appendix A in the Sybase Replication Agent *Primary Database Guide*.

New features added in 12.6 EBFs

The following list of features were added with EBFs after the initial Replication Agent 12.6 release and are included in the current 15.0 release. For information about any of the identified features, see the 15.0 documents that are identified.

Command for migrating Replication Agent

You can use the Replication Agent ra_migrate command to migrate the installed Replication Agent version to the current version. When you apply an EBF you may be required to run this command to upgrade to the latest version of the Replication Agent.

For more information, see the Replication Agent *Reference Guide*.

Commands for creating and removing replication definitions (Oracle only)

Sybase added the following Replication Agent commands for creating and removing replication definitions:

- rs_create_repdef
- rs_drop_repdef
- pdb_auto_create _repdef

This includes commands that allow replication definitions to be automatically created when tables or procedures are marked, and to be removed when they are unmarked.

For more information, see the Replication Agent *Reference Guide*.

Configuration parameter for automatically marking tables (Oracle only)

The configuration parameter pdb_automark_tables now determines whether the Replication Agent automatically marks user tables for replication. If you set the value to true, user tables are marked during initialization and DDL replication (when replication of DDL commands is enabled).

For more information, see the Replication Agent *Reference Guide*.

Understanding sequence replication (Oracle only)

The following section describes sequence replication in Oracle.

Logging of sequence information

Individual sequence changes are not logged in the Oracle database *log* file. However, changes to Oracle sequences do impact (update) the Oracle sys.seq$ table. These changes do not occur with each new sequence value generated; instead, the sys.seq$ table is updated periodically, based on refresh activity for sequence caching or other system changes. The value for a sequence stored in the sys.seq$ table is the “next” value to be assigned *after* the existing cache of values has been exhausted.

As an example, a newly created sequence starts with a value of 1, increments by 1, and has a cache value of 20. (These are all default values that you can customize.) The value stored in the sys.seq$ record for this new sequence is 21. This indicates that, after the existing cache of 20 numbers is used, the "next" value to be used is 21. The record in sys.seq$ does not change until the sequence value hits 21. At that time, Oracle caches the next 20 values for the sequence, and the sys.seq$ record is updated to 41. It is this value (41), which is recorded as a change to the sequences sys.seq$ record, that will be used for replication. The key point is to realize that not every individual sequence update is recorded in the log, and therefore is not available for replication.

Replicating sequence changes

When a sequence is marked for replication, changes to that sequence against sys.seq$ are captured and sent to Replication Server in the form of parameters passed to a procedure. The procedure (rs_update_sequence) must be installed at the standby site as part of system setup, as well as a function replication definition for that procedure. At the standby site, an implementation of rs_update_sequence will increment a same-named sequence until its value is equal to the value at the primary site. Sybase provides scripts with installation to create the rs_update_sequence stored procedure and function replication definition. The scripts' locations are as follows:

```
$SYBASE/RAX-15_0/scripts/
oracle_create_replicate_sequence_proc.sql
$SYBASE/RAX-15_0/scripts/
oracle_create_rs_sequence_repdef.sql
```

Performance considerations

Compared to the performance of incrementing a sequence at the primary database, particularly where sequence values are cached, the effort to increment the same sequence at the standby site may be less efficient. The stored procedure must dynamically determine the sequence to increment and then loop internally, incrementing the sequence until the primary value has been reached. (The loop is required because there is no way to assign a specific value to a sequence.)

Because the name of the sequence is passed as a parameter, Oracle cannot pre-compile the procedure for efficiency. With the addition of the looping activity, the performance of the solution may impact some environments where a large number of highly used sequences is the norm.

Sequence replication alternatives

If the performance of sequence replication is a concern, there are other alternatives to replication that support primary and standby use of the same sequence. The following alternatives are currently suggested by Oracle and others who are interested in providing sequence coordination between multiple sites:

- Assuming the sequence is being used to generate primary key values, the sequence at each site can be concatenated with something unique to the site. For example, use a sequence number concatenated with the database name, site name, or something similar. This technique allows each site to maintain a unique "range" of sequence of numbers; as a result, there would be no value in sending (replicating) changes of one site's range to another site.
- Similar to concatenating, each site can obtain a different range of numbers by having different starting points, or increment values, for the same sequence. For example, the sequence at one site can start at 1 and increment by 2 to generate odd numbers (1, 3, 5), while the other site starts at 2 and generates even numbers (2, 4, 6). Again, each site would have a unique range and avoid any need for replication.
- A third option is available to standby solutions, where the standby site is for read-only and does not access the sequence value until failover. Rather than continually being replicated, the value of the sequence at the standby site can be updated as part of the failover tasks. After failover and before the standby allows connection to client applications, a script or procedure can query the last-used sequence value (based on the last table to use it for a primary key) and update or redefine the sequence once, based on that calculated value

For additional information for sequence replication, see the Sybase Replication Agent Primary *Database Guide*.

Replication Agent permissions (Oracle only)

Replication Agent for Oracle uses the pds_username command to connect to Oracle. You must grant the Oracle permissions that are currently identified in the Sybase Replication Agent *Primary Database Guide*, as well as the following new additions:

- create sequence
- select on sys.cdef$

- select on sys.con$
- select on sys.user$
- select on sys.coltype$
- select on sys.seq$

Specifying the owner of a table in a replication definition

Table marking behavior has changed when you specify the owner of a table in a replication definition. Now, you must always use the owner keyword if you want to enable the SEND OWNER mode. When marking, if you do not specify the replicate owner, it defaults to the primary owner.

In the following examples, the leading letter indicates either primary (p) or replicate (r):

- Example 1:

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

  The table in the replication definition will be:

  ```
  powner.rtable
  ```

- Example 2:

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

  The table in the replication definition will be:

  ```
  rowner.rtable
  ```

- Example 3:

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

  The table in the replication definition will not be owner-qualified:

  ```
  rtable
  ```

For additional information, see the Replication Agent *Reference Guide*.

Support for Microsoft SQL Server 2005 JDBC driver

(For Microsoft SQL Server only) Replication Agent now supports the SQL Server 2005 JDBC driver that you must download from Microsoft. With this, Replication Agent supports Microsoft SQL Server 2000 and 2005. Also, for SQL Server 2005, Replication Agent supports Windows authentication when connecting to the primary database. For additional information, see the Sybase Replication Agent *Primary Database Guide*.