Contents

Foreword................................................................................................. 6
Replication Server versions covered in this edition............................... 6
How complete is 'Complete'? ................................................................. 6
How to use this Quick Reference Guide ................................................ 7
Syntax conventions................................................................................... 7
Terminology and Acronyms....................................................................... 7
Errata........................................................................................................... 8
Undocumented commands: warning & disclaimer .................................... 8
Overview & General Concepts ................................................................ 9
1 Starting points for common RepServer DBA tasks............................... 9
2 RepServer basics.................................................................................... 9
   2.1 Data model considerations for replication systems...................... 10
   2.2 Why replication systems can be so difficult.............................. 11
3 Sybase data replication products........................................................... 11
4 Products required for heterogeneous replication................................. 12
5 Licensable options & license keys......................................................... 13
6 RepServer documentation...................................................................... 13
RepAgents & Primary Databases............................................................... 15
7 Marking tables/procedures/databases for replication......................... 15
   7.1 Primary ASE databases ............................................................... 15
   7.2 Marking data for a warm standby or database repdef.................. 18
   7.3 Be careful when marking big text/image tables in A3B ......... 19
   7.4 ASE secondary truncation point vs. RSSD locator..................... 19
   7.5 Primary non-ASE databases......................................................... 20
8 Replication Agents................................................................................ 20
   8.1 ASE RepAgent configuration......................................................... 20
   8.2 ASE RepAgent traceflags (undocumented) ................................. 22
   8.3 Non-ASE RepAgents...................................................................... 23
Core Replication Functionality................................................................. 25
9 RepServer syntax basics....................................................................... 25
10 Getting RepServer status information............................................... 25
11 Datatypes, datatype classes & -translations....................................... 26
   11.1 Specifying datatypes in a repdef................................................ 27
   11.2 Datatype classes and datatype translations.............................. 27
   11.3 RSSD tables - datatypes ............................................................ 28
12 Connections to RepAgents................................................................. 29
13 Connections to primary/replicate databases..................................... 29
   13.1 RSSD tables - database connections...................................... 31
   13.2 Getting information about database connections.................... 31
   13.3 Configuration parameters for database connections............... 32
14 Replication definitions........................................................................ 35
   14.2 Table repdefs............................................................................. 36
   14.3 Schema changes for replicated tables....................................... 39
   14.4 Function repdefs (stored procedure replication)....................... 39
   14.5 Applied functions & Replication Options & Functions ............ 41
   14.6 Database repdefs......................................................................... 43
   14.7 Switching primary/replicate with database repdefs/subs .......... 46
   14.8 Warm standby vs. Database repdefs/subscriptions............... 47
15 Replicating text/image/rawobject columns....................................... 51
16 Replicating identity columns............................................................. 52
   16.1 Replicating identity column inserts ......................................... 53
   16.2 Replicating identity column updates........................................ 53
   16.3 Common problems when replicating identity columns............. 53
17 Publications & Articles....................................................................... 54
   17.1 Getting information about publications and articles................. 55
   17.2 RSSD tables - publications and articles.................................... 55
18 Subscriptions......................................................................................... 56
   18.1 Subscriptions: materialization methods.................................... 56
   18.2 Subscriptions: users and permissions........................................ 58
<table>
<thead>
<tr>
<th>Page</th>
<th>Section</th>
</tr>
</thead>
<tbody>
<tr>
<td>18.3</td>
<td>Queues for subscription (de)materialization</td>
</tr>
<tr>
<td>18.4</td>
<td>Automatic materialization: subscription commands</td>
</tr>
<tr>
<td>18.5</td>
<td>Bulk/manual materialization: subscription commands</td>
</tr>
<tr>
<td>18.6</td>
<td>RSSD tables - subscriptions</td>
</tr>
<tr>
<td>18.7</td>
<td>Getting information about subscriptions</td>
</tr>
<tr>
<td>18.8</td>
<td>Subscribing to ASE 'truncate table'</td>
</tr>
<tr>
<td>18.9</td>
<td>Subscribing to bitmaps</td>
</tr>
<tr>
<td>18.10</td>
<td>Common subscription creation problems</td>
</tr>
<tr>
<td>19</td>
<td>Warm Standby</td>
</tr>
<tr>
<td>19.2</td>
<td>Warm standby commands</td>
</tr>
<tr>
<td>19.3</td>
<td>Tips for setting up a warm standby</td>
</tr>
<tr>
<td>19.4</td>
<td>Warm standby - switching active and standby</td>
</tr>
<tr>
<td>19.5</td>
<td>Switching client applications to the new active database</td>
</tr>
<tr>
<td>19.6</td>
<td>Configuration parameters for logical connections</td>
</tr>
<tr>
<td>19.7</td>
<td>Warm standby - notes and remarks</td>
</tr>
<tr>
<td>19.8</td>
<td>RSSD tables - warm standby</td>
</tr>
<tr>
<td>19.9</td>
<td>Getting information about a warm standby</td>
</tr>
<tr>
<td>20</td>
<td>Functions, Function strings &amp; Function string classes</td>
</tr>
<tr>
<td>20.1</td>
<td>Summary of function/function string concepts</td>
</tr>
<tr>
<td>20.2</td>
<td>Function string classes</td>
</tr>
<tr>
<td>20.3</td>
<td>Function string inheritance</td>
</tr>
<tr>
<td>20.4</td>
<td>Function string commands</td>
</tr>
<tr>
<td>20.5</td>
<td>Function string tables - functions, function strings &amp; -class</td>
</tr>
<tr>
<td>20.6</td>
<td>Getting information about function strings</td>
</tr>
<tr>
<td>20.7</td>
<td>Function categories</td>
</tr>
<tr>
<td>20.8</td>
<td>Table repdef-specific functions (data access)</td>
</tr>
<tr>
<td>20.9</td>
<td>Function repdef-specific functions (user-defined)</td>
</tr>
<tr>
<td>20.10</td>
<td>Class-specific functions (transaction control)</td>
</tr>
<tr>
<td>20.11</td>
<td>SQL input/output templates for function strings</td>
</tr>
<tr>
<td>20.12</td>
<td>System variables in input/output templates</td>
</tr>
<tr>
<td>20.13</td>
<td>Tricks with function strings</td>
</tr>
<tr>
<td>20.14</td>
<td>User-defined functions: obsolete commands</td>
</tr>
<tr>
<td>21</td>
<td>Error classes</td>
</tr>
<tr>
<td>21.1</td>
<td>RSSD tables - error classes</td>
</tr>
<tr>
<td>22</td>
<td>RSSD exceptions log</td>
</tr>
<tr>
<td>23</td>
<td>Routes</td>
</tr>
<tr>
<td>23.2</td>
<td>Route commands</td>
</tr>
<tr>
<td>23.3</td>
<td>RSSD tables - routes</td>
</tr>
<tr>
<td>23.4</td>
<td>Getting information about routes</td>
</tr>
<tr>
<td>23.5</td>
<td>Configuration parameters for routes</td>
</tr>
<tr>
<td>23.6</td>
<td>Direct &amp; indirect routes</td>
</tr>
<tr>
<td>23.7</td>
<td>Site-, System- and Route-tables</td>
</tr>
<tr>
<td>24</td>
<td>Configuration parameters</td>
</tr>
<tr>
<td>24.1</td>
<td>Configuration parameters - command</td>
</tr>
<tr>
<td>24.2</td>
<td>RSSD tables - configuration parameters</td>
</tr>
<tr>
<td>24.3</td>
<td>RepServer-wide configuration parameters</td>
</tr>
<tr>
<td>24.4</td>
<td>Configuration parameters for network security</td>
</tr>
<tr>
<td>25</td>
<td>Parallel DSI</td>
</tr>
<tr>
<td>25.1</td>
<td>Configuration parameters for parallel DSI</td>
</tr>
<tr>
<td>26</td>
<td>Replication System Administration</td>
</tr>
<tr>
<td>26.1</td>
<td>Installation/setup: the rs_init utility</td>
</tr>
<tr>
<td>26.2</td>
<td>Character set &amp; collation</td>
</tr>
<tr>
<td>26.3</td>
<td>Installation/wins or Windows</td>
</tr>
<tr>
<td>26.4</td>
<td>RepServer environment variables</td>
</tr>
<tr>
<td>27</td>
<td>Startup &amp; shutdown</td>
</tr>
<tr>
<td>28</td>
<td>RepServer traces (undocumented)</td>
</tr>
<tr>
<td>29</td>
<td>Disk space &amp; RepServer devices/disk partitions</td>
</tr>
<tr>
<td>29.1</td>
<td>RSSD tables - disk partitions</td>
</tr>
<tr>
<td>29.2</td>
<td>Estimating disk space requirements</td>
</tr>
<tr>
<td>30</td>
<td>Queues</td>
</tr>
<tr>
<td>30.1</td>
<td>Diagnostic &amp; maintenance commands for queues</td>
</tr>
<tr>
<td>30.2</td>
<td>Configuration parameters for queues</td>
</tr>
<tr>
<td>30.3</td>
<td>RSSD tables - queues</td>
</tr>
<tr>
<td>31</td>
<td>Special Maintenance commands</td>
</tr>
<tr>
<td>32</td>
<td>Users, VSS, passwords and permissions</td>
</tr>
<tr>
<td>32.2</td>
<td>ID server user</td>
</tr>
<tr>
<td>32.3</td>
<td>RepAgent user</td>
</tr>
<tr>
<td>32.4</td>
<td>Primary user (RSSD)</td>
</tr>
<tr>
<td>32.5</td>
<td>Maintenance user</td>
</tr>
<tr>
<td>32.6</td>
<td>Route user</td>
</tr>
<tr>
<td>32.7</td>
<td>RepServer subscription user</td>
</tr>
<tr>
<td>32.8</td>
<td>Users, passwords, permissions - Commands</td>
</tr>
</tbody>
</table>
dbcc settrunc ('ltm', 'ignore' | 'valid')

(ASE command) With 'ignore', the secondary truncation point in the transaction log of the current database is deactivated (also known as disabled or removed). Warning: this is a drastic, last-resort option that will stop all replication from this database, and may cause transactions to be lost. Don't do this unless you fully understand the implications!

With 'valid', the secondary truncation point is re-established. The RSSD procedure rs_zeroltm (see below) should be run before restarting the RepAgent.

With 'begin' or 'end', the secondary truncation point is moved to the start or end of the transaction log, respectively. With 'pageid', sets the secondary truncation point to the specified page; this should be done only in special troubleshooting procedures.

To execute this dbcc settrunc, RepAgent must not be running in this database.

dbcc dbrepair ( database_name, 'ltmignore' )

(ASE command) Deactivates the secondary truncation point in the specified database. This can be used instead of dbcc settrunc when the database is not accessible.

dbcc settrunc ('ltm', 'gen_id', db_generation_no)

(ASE command) Sets the database generation number for the current database. See section 41 (p.125) for more information.

rs_zeroltm primary_dataserver, primary_database

(RSSD procedure) Resets the locator for the specified primary database in the RSSD table rs_locater. Before running rs_zeroltm, the RepAgent should be stopped. When the RepAgent is restarted after running rs_zeroltm, it sends the current secondary truncation point in the transaction log to RepServer, which stores it in rs_locater. rs_zeroltm should be used when the secondary truncation point in the primary transaction log, or the primary database's generation number, has been changed explicitly by the DBA (with dbcc settrunc). After running rs_zeroltm, the RSSD table rs_locater is resynchronized with the transaction log when the RepAgent is restarted.

7.5 Primary non-ASE databases

In non-ASE primary databases, data is marked for replication with a heterogeneous RepAgent, using RepAgent commands like pdb_setreptable. See section 8.3 (p.23).

8 Replication Agents

A replication agent ('RepAgent') is the component that picks up the primary transactions to be replicated from the primary database, and sends these to RepServer.

8.1 ASE RepAgent configuration

In ASE 11.5+, RepAgents are threads inside ASE (for ASE pre-11.5, an external RepAgent, called LTM (Log Transfer Manager; an old name for RepAgent) must be used instead. LTMs are not covered in this book; see the RepServer documentation). The ASE procedures in this section should be executed in the primary ASE database. When RepServer has no connection for a particular database, but the ASE RepAgent tries to connect anyway, the ASE RepAgent will not start and ASE error 9261 will appear in the ASE errorlog ('aborting due to an unrecoverable communications or Replication Server error'). To resolve this, either create a database connection in RepServer (see section 13; p.29), or disable the ASE RepAgent.

truncation point currently resides, depending on the following column.

- "secondary trunc state" - if 1, the secondary truncation point is valid (= active); if 0, no secondary truncation point is currently maintained.
- "db rep stat" - an integer consisting of status bits (see the documentation)
- "generation id" - the database generation ID
- "database id" - the database ID
- "database name" - the database name
- "ltl version" - the log transfer language (LTL) version used by the RepAgent
formats used by different databases such as ASE, Oracle, DB2, etc. Datatype definitions are also referred to as 'user-defined datatypes' (UDDs) in the RepServer documentation; note that this is unrelated to the ASE concept of user-defined datatypes (as created with `sp_addtype`). RepServer datatype definitions can be used to define 'datatype translations'; these are needed when the primary and replicate database are of a different type (for example, when replicating from a primary Oracle database to replicate ASE and DB2 databases). For most datatypes, RepAgent and RepServer perform the necessary conversion automatically; for some datatypes, like date/time data, the translation can be defined explicitly.

This datatype conversion can be done on two levels:

- For an individual primary column (or parameter) in a repdef. For each column (or parameter), a datatype must be specified. By adding the `map to datatypes` clause to the column declaration in the repdef (see above), all incoming values for that column will be translated to the specified datatype before being replicated further through the replication system. The official RepServer documentation calls this 'column-level translations'.

- For all data replicated to a replicate database. By creating datatype classes and datatype translations through the predefined SQL scripts mentioned below, a set of datatype translations is attached to a function string class. For a replicate database. To maintain that function string class, all data values replicated to that database are converted by the DSI according to those translations (only for those datatypes for which a translation has been defined). The official RepServer documentation calls this 'class-level translations'.

The two types of translations can be mixed. Column-level translations (`map to in the repdef`) are applied before class-level translations (performed by the DSI).

To initialize datatype classes, datatype definitions and datatype translations for heterogeneous replication, certain predefined SQL scripts, located in the `$SYBASE/$SYBASE_REP/scripts` directory, must be executed in the RSSD. For example, when replicating from a primary ASE database to a replicate Oracle database, the scripts `hds_oracle_funcstrings.sql`, `hds_oracle_udds.sql` and `hds_clt_ase_to_oracle.sql` should be run (you may need to edit some of these scripts with the actual name of your RSSD database).

In addition, some further setup steps are required. See the Replication Server Configuration Guide (p.13), chapter Installing and Implementing Heterogeneous Datatype Support for full details.

### 11.3 RSSD tables - datatypes

Datatype classes are stored in the RSSD table `rs_classes`, with `rs_classes.classid = 'D'`. For each datatype class, the RSSD table `rs_datatype` contains datatype definitions for that class. The RSSD table `rs_translation` contains datatype translations for a particular function string class (the so-called 'class-level translations'). The RSSD table `rs_columns` contains the datatype conversions specified in the repdef with `map to` (the so-called 'column-level translations'), in columns `declared_dtid` (the declared datatype) and `published_dtid` (the published datatype).

The RSSD `rs_*` procedures do not display any information about datatype translations. Instead, query the RSSD tables directly. (Note: the exact relationship between datatype classes and function string classes remains unclear; in fact, there may not even be a direct relationship - the RepServer documentation just doesn't contain much information about this topic.)

By default, no datatype classes are defined in the RSSD tables `rs_classes` and `rs_datatype` (except for the standard class `rs_rs_dt_class`). Other datatype classes, like `rs_oracle_dt_class` exist as an empty class by default: a row exists in `rs_classes`, but no corresponding rows exist in `rs_datatype`. The SQL scripts in `$SYBASE/$SYBASE_REP/scripts` insert rows into these RSSD tables, thus creating the datatype definitions and translations for these classes.

In principle, it should be possible to create your own datatype classes and corresponding translations. Because no RepServer commands exist for this purpose, it would require manually inserting rows into the RSSD tables `rs_classes`, `rs_datatype` and `rs_translation`. Unfortunately, exactly how to define such rows is mostly undocumented, but the SQL scripts mentioned above, and the RSSD table descriptions in the Replication Server Reference Manual (p.13), may be a source of inspiration. Creating your own datatype classes and -translations is probably not officially supported by Sybase.
Notes:

- Connections to the primary and replicate master databases must be created with rs_init as for any other database.
- An ASE RepAgent will be running in the primary master database, so its transaction log may need to be enlarged to ensure it does not fill up.
- The primary master database must be marked for replication with sp_reptostandby 'master', 'all'.
- The ASE RepAgent in the primary master database must be configured with ‘send warm standby xacts’ = true (p.21).
- The master database can be replicated either as a warm standby or as a database replication definition and corresponding subscriptions.
- When using a database repdef for the ASE master database, the repdef properties replicate dll and replicate system procedures must both be enabled (or the DDL commands and sp_ calls above, respectively, will not be replicated).
- Materialization of the replicate master database should be done manually.
- Replication of the master database does not replicate the actual suid/srid values for new logins and roles. Rather, it relies on identical contents for syslogins and syssrvroles on both sides so that the algorithms will pick the same values for new logins or roles.
- Since passwords for logins and roles are replicated as ciphertext, cross-platform synchronization is not supported (though it may actually work in some cases).
- Ensure that the sp_ procedures listed above are executed from within the master database if they are to be replicated.

Workaround in ASE pre-15.0.1

In ASE pre-15.0.1, common solutions for synchronizing data in the master database are to modify system stored procedures like sp_addlogin and sp_password to simulate replication of their execution. Such a mechanism must be set up by the DBA. For example, a system stored procedure can execute its counterpart in the replicate database server as an RPC. Alternatively, the modified stored procedure can write a row to a normally replicated user table in a user database which executes the corresponding system stored procedure on the replicate side (with a customized function string; see page 82 for an example).

14.10 ‘send standby’ option in repdefs

send standby can be specified as an option in table repdefs (send standby all columns; send standby replication definition columns) and function repdefs (send standby all parameters; send standby replication definition parameters). This option applies only to a warm standby as well as when database replication is combined with table or function repdefs.

The absence or presence of send standby affects how primary data is replicated:

- Replicated columns: for a table repdef without a send standby option, or with only send standby (which is identical to send standby all columns), all columns of the primary table are replicated. With send standby replication definition columns, only the columns declared in the repdef are replicated.
- Primary key columns: for a table repdef with a send standby option (no matter which one), only the primary key columns of this repdef are used in the where-clause of replicate SQL update/delete statements. Without a send standby option, the union of all primary key columns for all repdefs for this table is used in the where-clause. Completeness note 1: without a repdef, all table columns are used in the where-clause of replicate SQL statements. Completeness note 2: when multiple repdefs exist for a table, only one repdef can have a send standby option enabled at a time).
- Replicated parameters: for a function repdef without a send standby option, or with only send standby (which is identical to send standby all parameters), all parameters of the primary stored procedure are replicated. With send standby replication definition parameters, only the parameters declared in the repdef are replicated.

For a warm standby only, the send standby options in the repdefs can be overridden by the configuration parameter send_standby_repdef_cols for the logical connection (p.69).

The current send standby setting for a repdef is displayed by the RSSD procedure rs_helprep (under ‘Used by Standby’).
rs_helprepdb [ replicate_dataserver, replicate_database ]
(RSSD procedure) Without any parameters, displays a list of all replicate databases with subscriptions to primary data in the current RepServer (either table/function/database repdefs or publications). When no route is involved, this is the list of all replicate databases in this RepServer. With a dataserver/database name, displays that database only when such subscriptions exist. To display the actual repdefs, publications or subscriptions, use rs_helprep/rs_helpsub rs_helppubsub or rs_hdledbrep/rs_hdledbsub.

18.8 Subscribing to ASE ‘truncate table’

The ASE T-SQL statement truncated table removes all rows from a table in a fast and efficient way. In ASE 15.0, truncated table...partition...truncates a specific partition. Because these row deletions are not written to the transaction log, they cannot be replicated. However, the truncate table statement itself can be replicated instead. To subscribe to truncate table statements on the primary table, specify the clause subscribe to truncate table in create/define subscription (p.59). By default (i.e. without this clause) truncate table replication is disabled. To change the status of truncate table replication for existing subscriptions, use sysadmin apply_truncate_table (see below).

Example: create subscription Accounts_sub for Accounts with replicate at PROD.report_db subscribe to truncate table

Notes:
- all subscriptions for a particular replicate table must have identical settings for subscribing to truncate table.
- In a warm standby, truncate table replication is enabled by default; this can later changed with alter logical connection (see below).
- For a subscription to a database repdef, truncate table replication is disabled by default (but note that it was incorrectly enabled by default in early versions of 12.6). It can be enabled only by including subscribe to truncate table in create/define subscription. To change the setting for truncate table replication, drop & recreate the database subscription.
- The current status of truncate table replication for a subscription's replicate table or replicate database can be displayed in the replicate RSSD only with the RSSD procedures rs_helpdbsub, rs_helpsub or rs_hdledbsub (p.62).
- The actual status information is stored in the RSSD table rs_subscriptions in the replicate RSSD; when rs_subscriptions.status & 1073741824 > 0, truncate table replication is enabled, and disabled otherwise.

sysadmin apply_truncate_table, replicate_dataserver, replicate_database, 
{ replicate_table_owner_name | ' ', replicate_table_name, { 'on' | 'off' } } 
Executed at the replicate RepServer, enables ('on') or disables ('off') replication of the ASE truncate table statement for all existing subscriptions for the specified replicate table in the replicate database. The replicate table owner's name must be specified only if this was also done in the repdef; otherwise, specify '' (= two single quotes). sysadmin apply_truncate_table takes effect immediately.

Example: sysadmin apply_truncate_table PROD, report_db, DailySales_tab, 'on'

alter logical connection to logical_dataserver.logical_database
set send_truncate_table to { { 'on' | 'off' } } 
In a warm standby, enables ('on', default) or disables ('off') replication of the ASE truncate table statement from the active database to the standby database. The change takes effect once the connection to the standby database is resumed (it is good practice to suspend the connection first). To determine the current setting for a logical connection, use the RSSD procedure rs_hdlecfg.db, which is included in the electronic supplement to this book (p.8); in 15.0.1. admin config can also be used (p.90).

18.9 Subscribing to bitmaps

RepServer supports the special datatype rs_address, which can be used for subscribing to bitmaps (i.e. specific bits in a column or parameter). This can be useful when individual bits have a functionally relevant meaning, for example in status fields where a group of on/off indicators is packed together in one 4-byte integer.

Warning: before using bitmap subscriptions, please pay attention to the following. With bitmap subscriptions, a primary update will not be replicated if only rs_address columns are updated (unless a change in the bitmap would cause a row to migrate
rs_helpclass [ function_class_name ]
(RSSD procedure) Without any parameter, displays a list of all function string classes (as well as the error classes) in the current RSSD, including their primary RepServer. With parameter, displays only the specified class. To display information about class-specific function strings in the class, use rs_helpclassfstring (>p.76).

**admin show_function_classes**
Displays a list of all function string classes in this RepServer, including their parent classes (if any), and their level of inheritance. Does not display the primary RepServer for each class.

**alter connection to dataserver.database**
**set function string class [ to ] function_class_name**
Changes the function string class for the specified database connection. When a function is translated to an actual SQL statement for this database, function strings from this class will be used as templates. The change takes effect once the connection is resumed (it is good practice to suspend the connection before changing a parameter). When a connection is created, a function string class is defined with create connection, but this can later be changed with alter connection.

In most cases, function string classes are used for replicate databases. The function string class for a primary database is relevant only during subscription (de)materialization, when function strings for rs_select and rs_select_with_lock are used (>p.74).

Example:
```
alter connection to PROD.prod_db
set function string class to MyFunkyClass
```

To determine the function string class currently configured for a connection, run this query in the RSSD:
```
select dsname, dbname, classname 'Function string class'
from rs_databases d, rs_classes c
where d.funcclassid = c.classid
```

### 20.3 Function string inheritance

Hierarchical relationships can exist between individual function string classes, facilitating a feature known as 'function string inheritance'. From this perspective, the following types of function string classes can be distinguished:

- 'parent class' - a function string class from which another class is derived;
- 'derived class' - a function string class which has a parent class;
- 'base class' - a function string class which has no parent class.

A derived class inherits all function strings from its parent class; hence the name 'function string inheritance':

- When altering a function string in a parent class, this modification is immediately inherited by all derived classes for this parent class where this function string was not overridden (see next bullet).
- When creating a function string in a derived class, this overrides the inherited function string from the parent class (but does not affect the function string in the parent class itself).

When creating or modifying a function string, this takes effect immediately on the local RepServer; when there is an outgoing route, the modification takes effect in the destination RepServer once it has arrived in the destination RSSD.

Function string inheritance can have multiple levels: a parent class can also be a derived class at the same time. Parent classes can be established or changed with create function string class or alter function string class.

When creating a base class (i.e. a function string class without a parent class), all function strings must be explicitly created for the base class (with create function string); otherwise, any attempt to access a database with this function class will cause errors in the DSI.

**Notes:**
- rs_sqlserver_function_class (the default class used for ASE connections) cannot be used as a parent class; use rs_default_function_class or another function class instead.
- Function strings cannot be created or modified for rs_default_function_class or rs_db2_function_class; instead, create a derived class and create customized function strings in that class.
- Function string inheritance is dynamic (unlike the static class initialization for
rs_whereclauses
Contains information about where-clauses for articles (not for subscriptions). The actual where-clause is stored in rs_rules.

34 Configuration (.cfg) file
The RepServer configuration file (referred to as .cfg file in this book) is an ASCII file read by RepServer during startup to determine the server name, login and password for the ID server and RSSD server. Because the .cfg file contains passwords, it should be protected properly on file-system level.

By default, the .cfg file is located in $SYBASE/$SYBASE_REP/install and named repserver_name.cfg. The actual location of the .cfg file is passed to RepServer with the -C command line parameter in the RUN_servername file (p.100).

The .cfg file contains some rather self-explanatory keywords. It is not recommended to edit the .cfg file manually, unless you understand what you're doing. When the .cfg file does not exist or contains invalid keywords, RepServer will not start.

When the .cfg file contains encrypted passwords, these passwords can only be modified through rs_init (p.98).

When a line starts with #, the entire line is a comment.

(15.0) For RepServers with an embedded RSSD that have outgoing routes, the ERSSD RepAgent also has a .cfg file, by default located in $SYBASE (p.86).

35 Replication system monitoring features
RepServer comes with features for monitoring the status of the various servers in the replication system (not to be confused with the monitoring counters in section 40.3, which are about the internal workings of a single RepServer). The abbreviations of the various features can be confusing, so here's a quick overview; details are in the sections below:

• RMP - Replication Manager Plug-in for Sybase Central, Java version (12.6)
• RMS - Replication Monitoring Services (15.0)
• RSP - RepServer Plug-in for Sybase Central, non-Java version (pre-15.0)
• RSM - Replication Server Manager (pre-15.0)

35.1 Replication Manager Plug-in (RMP) for Sybase Central
As of 12.6, the 'Replication Manager Plug-in' (RMP) for the Java version of Sybase Central can be used to perform most RepServer DBA tasks through the Sybase Central GUI; this runs on all Java-enabled platforms. RMP includes support for creating repdefs, subscriptions, warm standby etc., reverse-engineering these, resuming/ suspending connections, etc. In 12.6, RMP support is limited to a few core RepServer areas, but in 15.0 almost the full range of RepServer functionality is covered.

In 15.0, RMP replaces RSP, the RepServer Plug-in (35.3) for the non-Java version of Sybase Central.

RMP is located in $SYBASE/RMP-xx_y (e.g. RMP-15_0). If it is not automatically registered with Sybase Central, you can register it manually through Tools Æ Plug-ins Æ Register and browse to $SYBASE/$SYBASE_RMP/lib/RMPlugin.jar.

35.2 Replication Monitoring Services (RMS) (15.0)
RMS (Replication Monitoring Services) is a 3-tier utility for monitoring replication systems, involving the Unified Agent Framework (UAF) and the Replication Manager Plug-in (RMP; 35.1) for Sybase Central. In 15.0, RMS replaces the pre-15.0 RMS server (35.4).

RMS monitors the states of RepServers and dataservers, and can automatically execute user-created shell scripts when certain events occur. With the Replication Manager Plug-in, most RepServer DBA tasks can be performed, and errorlog files for RepServers and dataservers can be viewed remotely (in fact, this plug-in can be handy for monitoring errorlogs of remote ASE servers, even without using RepServer at all).

In addition to monitoring RepServers, RMS can also monitor ASE and IQ servers, non-ASE RepAgents, OpenServers, Direct Connect servers, etc.

RMS can be accessed through the Replication Manager Plug-in for Sybase Central, but also through an ASCII command-line interface. To start and stop RMS, the UAF agent must be started/stopped. For more information about using RMS, see the product manual What's new in Replication Server.
A

abort switch ..............67
activate subscription ......61
active database (in warm standby) .......65
active-standby pair .....65
add partition ..........101
address, rs_ (datatime) ......26, 63
ad-hoc SQL statements, replicating ...82
admin config ..........90
admin cwd ............26
admin disk space .....25, 101
admin echo ...........26
admin get_generation .....20, 126
admin health ..........25
admin log name ..........26
admin logical_status .....26, 71
admin pid .............26
admin quiesce_check .....116
admin quiesce_force_rsi ....88
admin rssd_name ..........26, 109
admin security_property .....95
admin security_setting .....95
admin set_log_name .......26
admin show_connections ......32, 88
admin show_function_classes ......73
admin show_route_versions ......89
admin show_site_version ......89
admin sqm_readers .....103
admin statistics ......125
admin statistics, flush_status ......125
admin statistics, reset ......125
admin stats ............123
admin stats tps / cps / bps .....23
admin stats, backlog .....123
admin stats, cancel ...123
admin stats, reset ......123
admin stats, status ......123
admin stats_config_connection .....136
admin stats_config_module .....127
admin stats_config_route .....125
admin stats_intrusive_counter ......25
admin time ............26
admin translate ......26
admin version .........26, 89
admin who .............25
admin who, dist ........29
admin who, dsi ........32, 126
admin who, rsi .........88
admin who, sqm ..........102, 126
admin who, stat .......103
admin who, dsi_down ....25
admin who, rsi_up ......25
alias to db, maintenance user ......98
allow connections ......29, 118
alter applied function replication def. ...41
alter connection ..........30
alter connection (function class) ......73
alter database replication definition ......45
alter function ..........83
alter function replication definition ......41
alter function string ......75
alter function string class ......72

alter logical connection ..........63, 66
alter partition ..........101
alter queue
  set sqm_cache_enable ....105
  set sqm_cache_size ......105
  set sqm_page_size ......105
  set_xact_with_large_msg .....105
alter replication definition ........37
alter request function replication data. ...41
alter route .............87
alter user ..........108
always_replicate (text/image data) ....51
applied functions ......39, 41
apply_truncate_table, sysadmin ......63
articles, for publications ......54
articles, rs_ (RSSD table) .....56, 112
ASA (SQL Anywhere) for ERSSD ....110
ASA, for ERSSD:
  miscellaneous commands .....111
  startup/stop commands ......111
ASE environment variables ......99
ASE errorlog, location ......127
ASE RepAgent ..........20
ASE Replicator ..........12
ASE system databases, replicating ....10
assign action ..........83
atomic materialization ......56
atomic materialization, incremental ......56
autocorrection ..........38
audit database repdefs ......38
audit function strings ......38
and warm standby ......38
automatic materialization ......57, 59

B

backlog, in queues ......123
base class ...........73
Basics, RepServer ......9
batch (config) ..........32
batch_begin (config) ......32
batch_end, rs_ (function) ......78
batch_start, rs_ (function) ......78
begin, rs_ (function) ......78
bitmap subscriptions ......63
bulk materialization ......57, 61
byte_order (config) ......92

caching RSSD tables ..........121
capacity, rs_ (RSSD proc) ......102
cappable, rs_ (RSSD table) .....102, 112
change password, RepServer user ......108
character set in replication system ......99
check publication ......54
check subscription .....60, 62
check_repl, rs_ (function) ......78
classes, rs_ (RSSD table) ......28, 75, 85, 112
class-level translations (datatime) ......28
cm_fadeout_time (config) ......92
cm_max_connections (config) ......92
column-level translations (datatime) ......28
columns, rs_ (RSSD table) ......112
command_rexy (config) ......32
comments, syntax ......25

Index

#temporary tables (ASE) ......10
& (rs_address bitmask operator) ......64
.config file (configuration file) ......115

function strings .................................. 74
conflict resolution ................................ 82
definition ........................................ 71
input/output templates ......................... 79
modifiers (for variables) ...................... 79
restoring default ................................ 75
various tricks .................................... 81
functions
   definition ...................................... 71
   function class-specific ..................... 77
   function repdef-specific (user-def'd) .... 77
   table repdef-specific ....................... 76
functions, rs_ (RSSD table) .............. 75, 113

generation numbers ................................ 125
   DSI ............................................. 126
   primary database ................................ 126
get charset, rs_ (function) ................... 78
get generation, admin ......................... 126
get lastcommit, rs_ (function) .............. 78
get lastcommit, rs_ (proc) .................... 31
get sortorder, rs_ (function) .............. 77
get texptr, rs_ (function) ................... 77
get_thread_seq, rs_ (function) ................ 78
get_thread_seq_noholdlock, rs_ (func) .... 78
gettext, dbcc (ASE cmd) ...................... 19
grant .............................................. 108

ha failover (config) ............................. 92
heartbeat......................................... 117, 132
Help: replication doesn’t work! .......................... 128
help rep agent, sp_ (ASE proc) ................. 21
helpcfg, rs_ (RSSD proc) ..................... 91, 92, 105, 122, 124
helpcfg db, rs_ (RSSD proc) ...... 63, 69, 91
helpcfg route, rs_ (RSSD proc) .......... 91
helpclass, rs_ (RSSD proc) .................. 73, 84
helpclassstring, rs_ (RSSD proc) ....... 76
helpcounter, rs_ (RSSD proc) ............ 124, 125
helpdb, rs_ (RSSD proc) .................... 31
helpdbrep, rs_ (RSSD proc) ................ 51
helpdbsub, rs_ (RSSD proc) ............... 62
helperror, rs_ (RSSD proc) .................. 84
helpexception, rs_ (RSSD proc) ............ 85
helpstring, rs_ (RSSD proc) ............... 76
helpfunc, rs_ (RSSD proc) .................... 76
helppartition, rs_ (RSSD proc) ........... 81
helppub, rs_ (RSSD proc) ..................... 91
helppubsub, rs_ (RSSD proc) ............ 55, 62
helprep, rs_ (RSSD proc) .................... 51
helprepdb, rs_ (RSSD proc) .............. 31, 51, 63
helpreptable, rs_ (RSSD proc) ........ 51
helproute, rs_ (RSSD proc) ............ 38, 98
helpsub, rs_ (RSSD proc) ................... 98
helperuser, rs_ (RSSD proc) .............. 198
heterogeneous RepAgents ..................... 23
heterogeneous replication ................. 198
hibernate off, sysadmin ...................... 106
hibernate on, sysadmin ....................... 106
hibernation mode ................................ 106

ID server ........................................ 106
ID server user .................................. 106

name of ID server ................................ 106
id_msg_confidentiality (config) .......... 95
id_msg_integrity (config) .................. 95
id_msg_origin_check (config) ............ 95
id_msg_replay_detection (config) ....... 95
id_msg_sequence_check (config) .......... 95
id_mutual_auth (config) .................... 95
id_security Mechanism (config) ........... 95
id_server (config) ............................ 92
id_unified_login (config) ................. 95
identifiers ...................................... 95
identifiers, in primary database ........... 95
identity columns, replicating ............... 52
ids, rs_ (RSSD table) .................... 113
ids, rs_ (RSSD table) ..................... 113
ignore loss ...................................... 116, 128
image/text data, mark for replication ....... 18
image/text/rawobject data ................ 79
incremental atomic materialization ....... 56
indexes in replicate database ............... 11
indirect routes .................................. 87, 89
info, rs_ (table) .................................. 31
Information Liquidity Model ............... 12
inheritance, function string classes ....... 73
init, rs_ (utility) ............................... 98
init_erroarctions, rs_ (RSSD proc) ....... 84
init_sqm_write_delay (config) ............ 92
init_sqm_write_max_delay (config) ....... 92
initial RepServer username/password .... 106
initialize_threads, rs_ (function) .......... 78
initializing a new RepServer ............... 98
input templates (function strings) ....... 79
insert, rs_ (function) ......................... 79
insert_sqm_write_max_delay (config) ..... 92
insert_sqm_write_delay (config) ........... 92
initialize, rs_ (utility) ..................... 98
initialize new RepServer ................... 98
input templates (function strings) ....... 79
insert, rs_ (function) ......................... 76
intermediate RepServer (routes) .......... 85
Internet, RepServer resources on ... 14

I origin, rs_ (variable) ....................... 80
LAN RepAgents .................................. 23
language templates (function strings) .... 79
large transactions (DSI) .................. 95, 96
Last Seg Block (queues) .................... 102
last_text_chunk, rs_ (variable) .......... 81
lastcommit, rs_ (table) ..................... 31, 117
lastfn, rs_ (variable) ....................... 117
licenseable options, license keys ...... 13
license manager, license file ............. 13
login continuation character (\) ........... 25
multi license manager) ...................... 13
local queue ID (LQID) ....................... 104
locate, in RSSD table rs_locater .......... 19
locate, rs_ (RSSD table) .................. 20, 113
locking yourself out .......................... 131
log transfer
   resume ........................................ 29
   suspend ........................................ 29
   log Transfer Language (LTL) .......... 12
   log_first_tran, sysadmin ................. 105
   log_name, admin ............................. 26
   log_system_name (non-ASE RepAgent) ... 24
   log_system_name (ASE RepAgent) ...... 24
   log System_name (non-ASE RepAgent) ... 24
   logical_status, admin ..................... 71
   loss detection ................................ 118, 119
   LQID ........................................... 104
   LTM ............................................ 20
   LTM locator (non-ASE RepAgent) ....... 19, 24
   LTM truncation point ....................... 19

G

H
troubleshooting
  connectivity problems.................. 127
  disk partitions full.................. 129
  general steps.......................... 127
  transaction log full (ASE)............ 128
  trunc_reset, rs_ (function)............ 78
  trunc_set, rs_ (function)............... 78
  truncate table replication
    enable/disable........................ 63
    subscribing................................ 63
  truncate table partition............... 63, 79
  with database repdefs.................. 45
  truncate, rs_ (function)................ 76, 79
  truncation point, secondary........... 19
  tvalues, rs_ (RSSD table)............... 114

U
  Undocumented commands, warning &
  disclaimer.................................. 8
  unified_login (config)................... 95
  unique primary key, why required...... 10
  unit of replication...................... 9
  update, rs_ (function).................... 76
  update_lastcommit, rs_ (proc)......... 31
  update_threads, rs_ (function)......... 78
  upgrading routes........................ 89
  use_batch_markers (config)............... 34
  use_security_services (config)......... 94
  use_ssl (config).......................... 94
  user, user_raw (variable modifier).... 80
  username for RepServer, default....... 106
  username/password, forgotten........... 131
  users
    ID server user.......................... 106
    maintenance user...................... 107
    primary user (RSSD).................... 107
    RepAgent user........................... 106
    RepServer user........................... 106
    route user................................ 108
    subscription user...................... 58, 108
    users, rs_ (RSSD table).................. 114

V
  validate publication..................... 54
  validate subscription................... 62
  varchar_truncation (config)............ 94
  version.................................. 132
  version, admin........................... 89
  version, rs_ (RSSD table).............. 114

W
  wait for create standby................ 66, 133
  wait for delay........................... 133
  wait for switch.......................... 67, 133
  wait for time............................ 133
  warm standby................................ 65
  and autocorrection...................... 38
  DDL replication.......................... 18
  function strings......................... 70
  getting status information............... 71
  list of replicated DDL stmts/procs...... 77
  marking data for replication........... 78
  ‘multiple warm standby’ (MWSA)........ 43
  setup steps................................ 67
  switch active, detailed steps.......... 68
  switching applications to standby...... 69
  vs. database redfail...................... 47
  where-clause
    display with rs_helppub............... 55
    display with rs_helpsub.............. 62
    for a subscription.................... 59
    for an article.......................... 55
    whereclauses, rs_ (RSSD table)....... 56, 115
  who, admin................................ 25
  Windows
    registry key, startup parameters...... 100
    service, for RS.......................... 100
    writetext, rs_ (function)............. 77
    writetext_log, rs_ (variable)......... 81
    writing strings to RS errorlog......... 26

Z
  zeroltm, rs_ (RSSD proc)............... 20, 126