CHAPTER 32

Replicating Data with Replication Server

About this chapter

This chapter describes how you can use Replication Server to replicated data between an Adaptive Server Anywhere database and other databases. Other databases in the repliation system may be Adaptive Server Anywhere databases or other kinds of database.

Contents

Before you begin

This chapter is intended for Replication Server administrators who are setting up Adaptive Server Anywhere to take part in their Replication Server installation. It assumes knowledge of Replication Server documentation, and familiarity with the Replication Server product. No attempt is made to describe Replication Server itself.

☞ For information about Replication Server, including design, commands, and administration, see your Replication Server documentation.

Introduction to replication

Data replication is the sharing of data among physically distinct databases. Changes made to shared data at any one database are replicated to the other databases in the replication system.

Data replication brings some key benefits to database users.

Data availability

Data is made available locally, rather than through potentially expensive, less reliable, and slow connections to a single central database. Data is accessible locally even in the absence of any connection to a central server, so that you are not cut off from data in the event of a failure of a long-distance network connection.

Response time

Replication improves response times for data requests for two reasons. Requests are processed on a local server without accessing some wide area network, so that retrieval rates are faster. Also, local processing offloads work from a central database server so that competition for processor time is decreased.

Sybase replication technologies

Sybase provides two replication technologies: SQL Remote and Replication Server.

- **SQL Remote** SQL Remote is designed for two-way replication involving a consolidated database and large numbers of remote databases, typically including many mobile databases. Administration and resource requirements at the remote sites are minimal, and a typical time lag between the consolidated and remote databases is on the order of hours.
- **Replication Server** Replication Server is designed for replication among relatively small numbers of data servers, with a typical time lag between primary data and replicate data of a few seconds, and generally with an administrator at each site.

Each replication technology has its own documentation. This chapter describes how to use Adaptive Server Anywhere with Replication Server.

For information about SQL Remote, see the book *Data Replication with SQL Remote*.

Replicate sites and primary sites

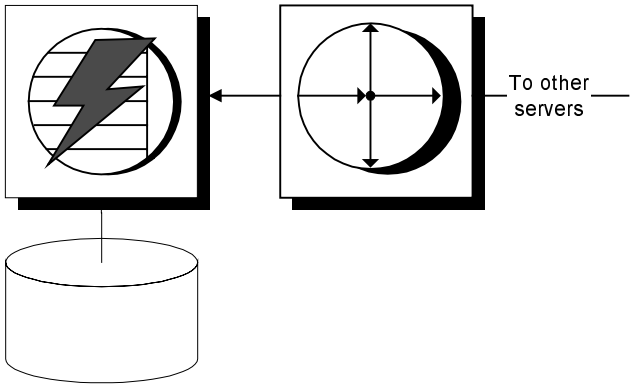
In a Replication Server installation, the data to be shared among databases is arranged in **replications**.

For each replication definition, there is a **primary site**. Changes to the data in the replication are made only at the primary site for that replication. The sites that receive the data in the replication are called **replicate sites**.

Replicate site components

You can use Adaptive Server Anywhere as a replicate site with no additional components.

The following diagram illustrates the components required for Adaptive Server Anywhere to participate in a Replication Server installation as a replicate site.



- Replication Server receives data changes from primary site servers.
- Replication Server connects to Adaptive Server Anywhere to apply the changes.
- Adaptive Server Anywhere makes the changes to the database.

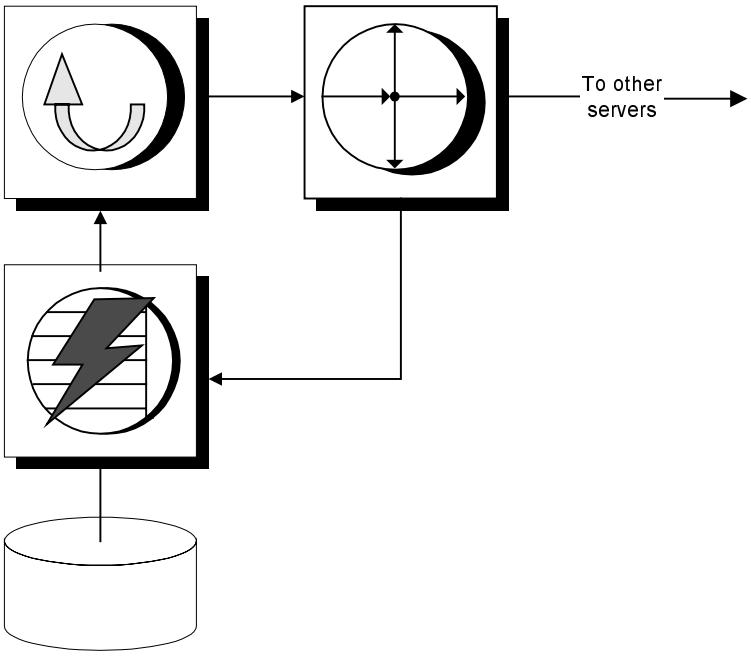
Asynchronous procedure calls

Replication Server can use asynchronous procedure calls (APC) at replicate sites to alter data at a primary site database. If you are using APCs, the above diagram does not apply. Instead, the requirements are the same as for a primary site.

Primary site components

To use an Adaptive Server Anywhere database as a primary site, you need to use the Log Transfer Manager (LTM), or Replication Agent. The LTM supports Replication Server version 10.0 and greater.

The following diagram illustrates the components required for Adaptive Server Anywhere to participate in a Replication Server installation as a primary site. The arrows in the diagram represent data flow.



- The Adaptive Server Anywhere database server manages the database.
- The Adaptive Server Anywhere Log Transfer Manager connects to the database. Itscans the transaction log to pick up changes to the data, and sends them to Replication Server.
- Replication Server sends the changes to replicate site databases.

A replication tutorial

This section provides a step-by-step tutorial describing how to replicate data from a primary database to a replicate database. Both databases in the tutorial are Adaptive Server Anywhere databases.

Replication Server assumed

This section assumes you have a running Replication Server. It does not describe how to install or configure Replication Server. The Replication Server documentation is the primary source of information about Replication Server.

What is in the tutorial

This tutorial describes a simple replication, where only tables are replicated. For information about replicating procedures, see "Preparing procedures and functions for replication" on page 853.

The tutorial uses a simple example of a (very) primitive office news system: a single table with an ID column holding an integer, a column holding the user ID of the author of the news item, and a column holding the text of the news item. The primary key consists of the **id** column and the **author** column.

To work through the tutorial, it is recommended that you create a directory (for example, *c:\tutorial*) to hold the files you create in the tutorial.

Set up the Adaptive Server Anywhere databases

This section describes how to create and set up the Adaptive Server Anywhere databases for replication.

You can create a database using Sybase Central or the *dbinit* command-line utility. For this tutorial, we use the *dbinit* command-line utility.

❖ **To create the primary site database:**

- From the directory you are using for the tutorial (for example *c:\tutorial*), enter the following command.

  ```
  dbinit -b -c -k primedb
  ```

  This creates a database file *primedb.db* in the current directory.

❖ **To create the replicate site database:**

- From the directory you are using for the tutorial (for example *c:\tutorial*), enter the following command.

  ```
  dbinit -b -c -k repdb
  ```

  This creates a database file *repdb.db* in the current directory.

For information on the choice of command-line switches, see "Creating a Transact-SQL-compatible database" on page 792.

What's next? The next step is to start database servers running on these databases.

Start the database servers

You need to run the primary site database server, with the primary database loaded.

❖ **To start the primary site database server:**

- Change to the tutorial directory and enter the following command line to start a network database server running the **primedb** database and using the TCP/IP network communication protocol on the default communications port (2638):

  ```
  dbsrv6 -x tcpip primedb.db
  ```

  The `-n` command line option defines the server name as **PRIMESV**. This is the name for Adaptive Server Anywhere applications. It is *not* the name used by Replication Server or other Open Client applications.

❖ **To start the replicate site database server:**

- Change to the tutorial directory and enter the following command line to start a network database server running the **repdb** database, but on a different port:

  ```
  dbsrv6 -x tcpip{port=2639} -n REPSV repdb.db
  ```

  The `-n` command line option defines the server name as **REPSV**. This is the name for Adaptive Server Anywhere applications. It is *not* the name used by Replication Server or other Open Client applications.

What's next? The next step is to make entries for each of the Adaptive Server Anywhere servers into an interfaces file, so that Replication Server can communicate with these database servers.

Set up the Open Servers in your system

You need to add a set of Open Servers to the list of Open Servers in your system.

Adding Open Servers

Open Servers are defined in your interfaces file (*sql.ini*) using the *dsedit* utility. For NetWare and UNIX users, the interfaces file is named *interfaces*, and the utility is named *sybinit*.

For full instructions on how to add definitions to your interfaces file, see "Configuring Open Servers with DSEDIT" on page 821.

Required Open Servers

For each Open Server definition you must provide a **name** and an **address**. Do not alter the other attributes of the definition. You need to add Open Server entry for each of the following:

- **The primary database** Create an entry named PRIMEDB with address as follows:
  - **Protocol** NLWNSCK
  - **Network address** localhost,2638
- **The replicate database** Create an entry named REPDB with address as follows:
  - **Protocol** NLWNSCK
  - **Network address** localhost,2639
- **The LTM at the primary database** This is needed so that the LTM can be shut down properly. Create an entry named PRIMELTM with address as follows:
  - **Protocol** NLWNSCK
  - **Network address** localhost,2640
- **Your Replication Server** This tutorial assumes you already have the Replication Server Open Server defined.

What's next?

You should confirm that the Open Servers are configured properly.

Confirm the Open Servers are configured properly

You can confirm that each Open Server is available by selecting ServerObject➤Ping Server from the DSEDIT utility.

Alternatively, you can confirm that each Open Server is configured properly by connecting to the database using an Open Client application such as the *isql* utility.

To start *isql* running on the primary site database, type

```
isql -U dba -P sql -S PRIMEDB
```

Add Replication Server information to the primary database

You need to add Replication Server tables and procedures to the primary site database for the database to participate in a Replication Server installation. You also need to create two user IDs to be used by Replication Server. The SQL command file *rssetup.sql* is provided with Adaptive Server Anywhere to carry out these tasks.

The *rssetup.sql* command file must be run on the Adaptive Server Anywhere server from the Interactive SQL utility.

❖ **To run the rssetup script:**

1. From Interactive SQL, connect to the database as user ID **DBA** using password **SQL**.
2. Run the *rssetup* script using the following command:

   ```
   read "path\rssetup.sql"
   ```

   where *path* is your Adaptive Server Anywhere installation directory.

Actions carried out by rssetup.sql

The *rssetup.sql* command file carries out the following functions:

- Creates a user named **dbmaint**, with password **dbmaint** and with DBA permissions. This is the maintenance user name and password required by Replication Server to connect to the primary site database.
- Creates a user named **sa**, with password **sysadmin** and with DBA permissions. This is the user ID used by Replication Server when materializing data.
- Adds **sa** and **dbmaint** to a group named **rs_systabgroup**.

Passwords and user IDs

While the hard-wired user IDs (**dbmaint** and **sa**) and passwords are useful for test and tutorial purposes, you should change the password and perhaps also the user IDs when running databases that require security. Users with DBA permissions have full authority in a Adaptive Server Anywhere database.

The user ID **sa** and its password must match that of the system administrator account on the Replication Server. A NULL password is not currently allowed by Adaptive Server Anywhere.

Permissions

The *rssetup.sql* script carries out a number of operations, including some permissions management. The permissions changes made by *rssetup.sql* are outlined here. *You do not have to make these changes yourself.*

For replication, you should ensure that the **dbmaint** and **sa** users can access this table without explicitly specifying the owner. To do this, the table owner user ID must have group membership permissions, and the **dbmaint** and **sa** users must be members of the table owner group.

To grant group permissions, you must have DBA authority.

For example, if the table is owned by user **DBA**, you should grant group permissions to the DBA user ID:

```
GRANT GROUP
TO DBA
```

You should then grant the **dbmaint** and **sa** users membership in the DBA group. To grant group membership, you must either have DBA authority or be the group ID.

```
GRANT MEMBERSHIP
IN GROUP "DBA"
TO dbmaint ;

GRANT MEMBERSHIP
IN GROUP "DBA"
TO sa ;
```

Create the table for the primary database

In this section, we create a single table in the primary site database, using *isql*. First, if you are not already connected to the primary site database, do so:

```
isql -U dba -P sql -S PRIMEDB
```

Once connected, you can create a table in the database:

```
CREATE TABLE news (
  ID int,
  AUTHOR char( 40 ) DEFAULT CURRENT USER,
  TEXT char( 255 ),
  PRIMARY KEY ( ID, AUTHOR )
)
go
```

Identifier case sensitivity
In Adaptive Server Anywhere, all identifiers are case insensitive. In Adaptive Server Enterprise, identifiers are case sensitive by default. Even in Adaptive Server Anywhere, ensure the case of your identifiers matches in all parts of the SQL statement to avoid incompatibility with Adaptive Server Enterprise.

In Adaptive Server Anywhere, user IDs and passwords are case insensitive in case-insensitive databases and case sensitive in case sensitive databases.

For **news** to act as part of a replication primary site, you must set the REPLICATE flag to ON for the table using an ALTER TABLE statement:

```
ALTER TABLE news
REPLICATE ON
go
```

This is equivalent to running the **sp_setreplicate** or **sp_setreptable** procedure on the table in Adaptive Server Enterprise. REPLICATE ON can not be set in a CREATE TABLE statement.

Add Replication Server information to the replicate database

You should run the *rssetup.sql* command file on the replicate database in exactly the same manner as it was run on the primary database.

You should also ensure that the **dbmaint** and **sa** users can access this table without explicitly specifying the table owner.

These tasks are the same as those carried out on the primary database. For a description of how to carry them out, see "Add Replication Server information to the primary database" on page 838.

Create the tables for the replicate database

The replicate site database needs to have tables created to hold the data it receives. You should create these tables at this point. As long as the database elements are in place, no extra statements are required for them to act as a replicate site in a Replication Server installation. In particular, you do not need to set the REPLICATE flag to ON; this is required only at the primary site.

Replication Server allows replication between tables and columns with different names. As a simple example, however, create a table in the replicate database identical in definition to that in the primary database (except for the REPLICATE flag, which is not set to ON in the replicate database). The table creation statement for this table is:

```
CREATE TABLE news (
   ID int,
   AUTHOR char( 40 ) DEFAULT CURRENT USER,
   TEXT char( 255 ),
   PRIMARY KEY ( ID, AUTHOR )
)
go
```

For the tutorial, the CREATE TABLE statement must be *exactly* the same as that at the primary site.

You must ensure that the users **dbmaint** and **sa** can access this table without specifying the owner name. Also, these user IDs must have SELECT and UPDATE permissions on the table.

Set up Replication Server

You need to carry out the following tasks on the Replication Server:

- Create a connection for the primary site data server.
- Create a connection for the replicate site data server.
- Create a replication definition.
- Create a subscription to the replication.

This section describes each of these tasks. It also describes starting the Adaptive Server Anywhere LTM.

Create a connection for the primary site

Using *isql*, connect to Replication Server and create a connection to the primary site Adaptive Server Anywhere database.

The following command creates a connection to the **primedb** database on the **PRIMEDB** Open Server.

```
create connection to PRIMEDB.primedb
set error class rs_sqlserver_error_class
set function string class rs_sqlserver_function_class
set username dbmaint
set password dbmaint
with log transfer on
go
```

If you have changed the **dbmaint** user ID and password in the *rssetup.sql* command file, make sure you replace the **dbmaint** username and password in this command.

The **primedb** database name is not actually used by Replication Server; instead, the database name is read from the command line of the **PRIMEOS** Open Server. You must, however, include a database name in the CREATE CONNECTION statement to conform to the syntax.

☞ For a full description of the create connection statement, see your *Replication Server Commands Reference*.

Create a connection for the replicate site.

Using *isql*, connect to Replication Server and create a connection to the replicate site Adaptive Server Anywhere database.

The following command creates a connection to the **repdb** database on the **REPDB** Open Server.

```
create connection to REPDB.repdb
set error class rs_sqlserver_error_class
set function string class rs_sqlserver_function_class
set username dbmaint
set password dbmaint
go
```

This statement differs from that for the primary site server in that there is no **with log transfer on** clause.

If you have changed the **dbmaint** user ID and password in the *rssetup.sql* command file, make sure you replace the **dbmaint** username and password in this command.

Create a replication definition

Using *isql*, connect to Replication Server and create a replication definition. The following statement creates a replication definition for the news table on the **primedb** database:

```
create replication definition news
with primary at PRIMEDB.primedb
( id int, author char(40), text char(255) )
primary key ( id, author )
go
```

For a full description of the CREATE REPLICATION DEFINITION statement, see your Replication Server Commands Reference.

Configure and start the Adaptive Server Anywhere LTM

For replication to take place, the Adaptive Server Anywhere LTM must be running against the primary site server. Before you start the Adaptive Server Anywhere LTM, it must be properly configured by editing an LTM configuration file.

A sample configuration file for the **primedb** database is as follows. If you are following the examples, you should make a copy of this file as *prime.cfg*:

```
#
# Configuration file for 'PRIMELTM'
#
SQL_server=PRIMEDB
SQL_database=primedb
SQL_user=sa
SQL_pw=sysadmin
RS_source_ds=PRIMEDB
RS_source_db=primedb
RS=your_rep_server_name_here
RS_user=sa
RS_pw=sysadmin
LTM_admin_user=dba
LTM_admin_pw=sql
LTM_charset=cp850
scan_retry=2
APC_user=sa
APC_pw=sysadmin
SQL_log_files=C:\TUTORIAL
```

If you have changed the user ID and password in the *rssetup.sql* command file from **sa** and **sysadmin**, you should use the new values in this configuration.

To start the Adaptive Server Anywhere LTM running on the primary site server use the following command:

```
dbltm -S PRIMELTM -C prime.cfg
```

The connection information is in *prime.cfg*. In this command line, PRIMELTM is the server name of the LTM.

You can find usage information about the Adaptive Server Anywhere LTM by typing the following statement:

```
dbltm -?
```

The Adaptive Server Anywhere LTM for Windows NT can be run as an NT service. For information on running programs as services, see "Running the server outside the current session" on page 18.

Create a subscription for your replication

Using *isql*, connect to Replication Server and create a subscription for the replication.

The following statement creates a subscription for the news replication defined in "Create a replication definition" on page 842 with replicate site as the **repdb** database.

```
create subscription NEWS_SUBSCRIPTION
for news
with replicate at REPDB.repdb
go
```

Your installation is now complete. You should try replicating data to confirm that the setup is working properly.

Enter data at the primary site for replication

You can now replicate data from the primary to the replicate database. As an example, connect to the primary database using the *isql* utility, and enter a row in the **news** table.

```
insert news (id, text)
values (1, 'Test news item.' )
commit
go
```

The Adaptive Server Anywhere LTM sends only committed changes to the Replication Server. The data change is replicated next time the LTM polls the transaction log.

Tutorial complete

You have now completed the tutorial. The following section describes in more detail the steps you have carried out.

Configuring databases for Replication Server

Each Adaptive Server Anywhere database that participates in a Replication Server installation needs to be configured before it can do so. Configuring the database involves the following tasks:

- Creating a Replication Server-compatible database.
- Selecting a secure user ID for the maintenance user and the name used by Replication Server when materializing data.
- Setting up the database for Replication Server.
- Configuring the language and character set, where necessary.

This section discusses all but the first of these tasks.

For a discussion of how to create a database that is compatible with Replication Server, see "Creating a Transact-SQL-compatible database" on page 792.

Configuring the LTM

Each primary site Adaptive Server Anywhere database requires an LTM to send data to Replication Server. Each primary or replicate site Adaptive Server Anywhere database requires an Open Server definition so that Replication Server can connect to the database.

For information on configuring the LTM, see "Configuring the LTM" on page 854.

Setting up the database for Replication Server

Once you have created your Adaptive Server Anywhere database, and created the necessary tables and so on within the database, you can set the database up ready for use with Replication Server. You do this using a setup script supplied with the Adaptive Server Anywhere Replication Agent product. The script is named *rssetup.sql*.

When you need to run the setup script

The setup script needs to be run at any Adaptive Server Anywhere database that is taking part in a Replication Server installation, whether as a primary or a replicate site.

What the setup script does

The setup script carries out the following functions:

- Creates user IDs required by Replication Server when connecting to the database.
- Creates a set of stored procedures and tables used by Replication Server. The tables begin with the characters **rs_**, and the procedures begin with the characters **sp_**.

Included in the procedures are some that are important for character set and language configuration.

Prepare to run the setup script

Replication Server uses a special data server **maintenance user** login name for each local database that contains replicated tables. This allows Replication Server to maintain and update the replicated tables in the database.

The maintenance user

The setup script creates a maintenance user with name **dbmaint** and password **dbmaint**. The maintenance user is granted DBA permissions in the Adaptive Server Anywhere database, which allows it full control over the database. For security reasons, you should change the maintenance user ID and password.

❖ **To change the maintenance user ID and password:**

1 Open the *rssetup.sql* setup script in a text editor.

2 Change all occurrences of the **dbmaint** user ID to the new maintenance user ID of your choice. The user ID occurs in the following places at the top of the setup script file:

```
grant connect to dbmaint identified by dbmaint
grant dba to dbmaint
grant group to dbmaint
grant connect to sa identified by sysadmin
grant dba to sa
grant membership in group dbmaint to sa
go

connect dbmaint identified by dbmaint
go
```

3 Change all occurrences of the **dbmaint** password to the new maintenance user password of your choice. The password occurs in the following places at the top of the setup script file:

```
grant connect to dbmaint identified by dbmaint
grant dba to dbmaint
grant group to dbmaint
grant connect to sa identified by sysadmin
grant dba to sa
grant membership in group dbmaint to sa
go

connect dbmaint identified by dbmaint
go
```

The materialization user ID

When Replication Server connects to a database to materialize and initial copy of the data in the replication, it does so using the Replication Server system administrator account.

The Adaptive Server Anywhere database must have a user ID and password that match the Replication Server system administrator user ID and password. A NULL password is not allowed by Adaptive Server Anywhere.

The setup script assumes a user ID of sa and a password of sysadmin for the Replication Server administrator. You should change this to match the actual name and password.

❖ **To change the system administrator user ID and password:**

1. Open the *rssetup.sql* setup script in a text editor.
2. Change all occurrences of the **sa** user ID to match the Replication Server system administrator user ID. The user ID occurs in the following places at the top of the setup script file:

```
grant connect to dbmaint identified by dbmaint
grant dba to dbmaint
grant group to dbmaint
grant connect to sa identified by sysadmin
grant dba to sa
grant membership in group dbmaint to sa
go
```

3. Change all occurrences of the **sa** password to match the Replication Server system administrator password. The password has the initial setting of **sysadmin** and occurs in the following place at the top of the setup script file:

```
grant connect to dbmaint identified by dbmaint
grant dba to dbmaint
grant group to dbmaint
grant connect to sa identified by sysadmin
grant dba to sa
grant membership in group dbmaint to sa
go
```

Run the setup script

Once you have modified the setup script to match the user IDs and passwords appropriately, you can run the setup script to create the Replication Server procedures and tables in the Adaptive Server Anywhere database.

❖ **To run the setup script:**

1. Start the Adaptive Server Anywhere database on a Adaptive Server Anywhere database engine or server.
2. Start the Interactive SQL utility, and connect to the database as a user with DBA authority. When a Adaptive Server Anywhere database is created, it has a user with user ID **DBA** and password **SQL**, which has DBA authority.
3. Run the script by entering the following command in the Interactive SQL command window:

   ```
   read path\rssetup.sql
   ```

   where *path* is the path to the setup script.

Character set issues

Each Adaptive Server Anywhere database is assigned a specific collation (character set and sort order) when it is created. Replication Server uses a different set of identifiers for character sets and sort orders.

When Replication Server connects to a database, it runs a procedure named **sp_serverinfo**, which provides information to Replication Server about the server. Among those pieces of information are the character set and the sort order.

As Replication Server expects to receive an Open Client/Open Server character set and sort order when it connects, the Adaptive Server Anywhere **sp_serverinfo** procedure uses a translation table to convert the Adaptive Server Anywhere collation into the equivalent Open Client/Open Server names, for provision to Replication Server.

❖ **To find the collation sequence used by a Adaptive Server Anywhere database:**

- Connect to the database from Sybase Central, right click on the database icon in the left pane, and select Properties from the pop-up menu. The collation is listed on the Extended Information tab. Or:
- Run the *dbinfo* command-line utility. For example, the following command provides information about a database named **asademo** running on a server named **asademo**:

  ```
  dbinfo -c "uid=dba;pwd=sql;eng=asademo;dbn=asademo"
  ```

Automatically supported collations

The setup script includes a set of mappings from Adaptive Server Anywhere collations to Open Client/Open Server character sets and sort orders. These mappings are as follows:

Adaptive Server Anywhere collation	Open Server character set	Open Server sort order (case sensitive)	Open Server sort order (case insensitive)
default	cp850	dictionary_cp850	nocase_850
850	cp850	dictionary_cp850	nocase_850
SJIS	sjis	bin_sjis	bin_sjis
EUC_JAPAN	eucjis	bin_eucjis	bin_eucjis

Adding a language mapping

If you wish to use a character set and sort order other than those listed in the table, you must add a correspondence between the Adaptive Server Anywhere collation and the Open Client/Open Server character set and sort order to the Adaptive Server Anywhere database.

❖ **To add a character set mapping to a database:**

♦ Run the **sp_add_rs_collation** procedure. The arguments are as follows:

Parameter	Description
@collation_name	Adaptive Server Anywhere collation name
@cs_name	Open Client/Open Server character set name
@so_name_case	Open Client/Open Server sort order (case sensitive)
@so_name_caseless	Open Client/Open Server sort order (case insensitive)

A final argument, @user_defined, has the default setting of 'Y'. You should not supply a @user_defined parameter.

☞ For a full description of Adaptive Server Anywhere collations, see "Database Collations and International Languages" on page 289.

☞ For more information on character set issues, see "Language and character set issues" on page 856.

Removing a language mapping

If you wish to change a language mapping, you need to first remove the existing mapping and then add a new one.

❖ **To remove a language mapping:**

♦ Run the **sp_add_rs_collation** procedure.

The **sp_add_rs_collation** procedure takes a single argument, *@collation_name*, which is the Adaptive Server Anywhere collation name.

Using the LTM

As the Adaptive Server Anywhere LTM relies on information in the Adaptive Server Anywhere transaction log, you must take care not to delete or damage the log without storing backups. A transaction log mirror is recommended.

For more information about transaction log management, see the section "Transaction log and backup management" on page 859.

You cannot substitute a Adaptive Server Anywhere LTM for a Adaptive Server Enterprise LTM since the transaction logs are of a different format.

The Adaptive Server Anywhere LTM supports replication of inserts, updates, and deletes, as well as replication of Transact-SQL-dialect stored procedures.

The Adaptive Server Enterprise LTM sends data changes to the Replication Server before they are committed. The Replication Server holds the changes until a COMMIT statement arrives. The Adaptive Server Anywhere LTM sends only committed changes to Replication Server. For long transactions this may lead to some added delay in replication, as all changes have to be sent to Replication Server before being distributed.

Configuring tables for replication

Adaptive Server Anywhere does not support the **sp_setreplicate** system procedure. Instead, a table is identified as a primary data source using the ALTER TABLE statement with a single clause:

```
ALTER TABLE table-name
SET REPLICATE ON
```

The effects of setting REPLICATE ON for a table

Setting REPLICATE ON causes extra information to be placed into the transaction log. whenever an UPDATE, INSERT, or DELETE action is carried out on the table. This extra information is used by the Adaptive Server Anywhere Replication Agent to submit the full pre-image of the row, where required, to Replication Server for replication.

Even if only some of the data in the table is to be replicated, all changes to the table are submitted to Replication Server. It is Replication Server's responsibility to distinguish the data that is to be replicated from that which is not.

When a row is updated, inserted, or deleted, the pre-image of the row is the contents of the row before the action, and the post-image is the contents of the row after the action. For INSERTS, only the post-image is submitted (the pre-image is empty). For DELETES, the post-image is empty and the pre-image only is submitted. For UPDATES, both the pre-image and the updated values are submitted.

The same data types are supported for replication as in Replication Server 10.0.

Data type	Description (Open Client/Open Server type)
Exact integer data types	int, smallint, tinyint
Exact decimal data types	Decimal, numeric
Approximate numeric data types	float (8-byte), real
Money data types	money, smallmoney
Character data types	char(n), varchar(n), text
Date and time data types	datetime, smalldatetime
Binary data types	binary(n), varbinary(n), image
Bit data types	bit

Notes

- Adaptive Server Anywhere supports data of zero length that is not NULL: however, non-null long varchar and long binary data of zero length is replicated to a replicate site as NULL.

If a primary table has columns with unsupported data types, you can replicate the data if you create a replication definition using a compatible supported data type. For example, to replicate a DOUBLE column, you could define the column as FLOAT in the replication definition.

Side effects of setting REPLICATE ON for a table

There can be a replication performance hit for heavily updated tables. You could consider using replicated procedures if you experience performance problems that may be related to replication traffic, as replicated procedures send only the call to the procedure instead of each individual action.

As setting REPLICATE ON causes extra information to be sent to the transaction log, this log grows faster than for a non-replicating database.

Minimal column replication definitions

The Adaptive Server Anywhere LTM supports the Replication Server replicate minimal columns feature. This feature is enabled at Replication Server.

For more information on replicate minimal columns, see your Replication Server documentation.

Preparing procedures and functions for replication

Stored procedures can be used to modify the data in tables; updates, inserts, and deletes are executed from within the procedure.

Replication Server can replicate procedures as long as they satisfy certain conditions. The first statement in a procedure must carry out an update in order for the procedure to be replicated. See your Replication Server documentation for a full description of how Replication Server replicates procedures.

Adaptive Server Anywhere supports two dialects for stored procedures: the Watcom-SQL dialect, based on the draft ISO/ANSI standard, and the Transact-SQL dialect. You can use either dialect in writing stored procedures for replication.

Function APC format

The Adaptive Server Anywhere LTM supports the Replication Server **function APC** format. To make use of these functions, set the configuration parameter **rep_func** to **on** (the default is **off**).

The LTM interprets all replicated APC's as either table APC's or function APC's. A single Adaptive Server Anywhere database cannot have some function APC's combined with other table APC's.

☞ For more information about replicate functions, see your Replication Server documentation.

SQL Statements for controlling procedure replication

A procedure can be configured to act as a replication source using the ALTER PROCEDURE statement.

The following statement makes the procedure **MyProc** act as a replication source.

```
ALTER PROCEDURE MyProc
REPLICATE ON
```

The following statement prevents the procedure **MyProc** from acting as a replication source.

```
ALTER PROCEDURE MyProc
REPLICATE OFF
```

These statements have the same effect as running **sp_setreplicate** or **sp_setrepproc 'table'** on the procedure in Adaptive Server Enterprise. The **sp_setrepproc 'function'** syntax is not supported.

The effects of setting REPLICATE ON for a procedure

When a procedure is used as a replication data source, extra information is sent to the transaction log any time the procedure is called.

Asynchronous procedures

Asynchronous procedures are procedures that are called at a replicate site database in order to update data at a primary site database. The procedure carries out no action at the replicate site, but the call to the procedure is replicated to the primary site, where a procedure of the same name is executed: this is called an asynchronous procedure call (APC). The changes made by the APC are then replicated from the primary to the replicate database in the usual manner.

For information about APCs, see your Replication Server documentation.

The APC_user and APC support

Support for APC's in Adaptive Server Anywhere is different from that in Adaptive Server Enterprise. In Adaptive Server Enterprise, each APC is executed using the user ID and password of the user who called the procedure at the replicate site. In Adaptive Server Anywhere, the password is not stored in the transaction log, and so is not available at the primary site. To work around this difference, a single user ID with associated password is entered into the LTM configuration file, and this user ID (the **APC_user**) is used to execute the procedure at the primary site. The APC_user must, therefore, have appropriate permissions at the primary site for each APC that may be called.

Configuring the LTM

LTM behavior is controlled by modifying the LTM **configuration file**, which is a plain text file that can be created and edited using a text editor.

The LTM configuration file contains information needed by the LTM, such as the Adaptive Server Anywhere server it is transferring a log from, the Replication Server it is transferring the log to. A valid configuration file is required in order to run the LTM.

Creating a configuration file

You must create a configuration file, using a text editor, before you can run the LTM. The `-C` LTM command-line specifies the name of the configuration file to use, and has a default of *dbltm.cfg*.

Configuration file format

The format of the LTM configuration file is the same as the Replication Server configuration file format, which is described in your *Replication Server Administration Guide*. In summary:

- The configuration file contains one entry per line.
- An entry consists of a configuration parameter, followed by the = character, followed by the value:

  Entry=*value*

- Lines beginning with a # character are comments, and are ignored by the LTM.
- The configuration file cannot contain leading blanks.
- Entries are case sensitive.

☞ For the full list of available configuration file parameters, see "The LTM configuration file" on page 94 of the book *Adaptive Server Anywhere Reference Manual*.

Example configuration file

- The following is a sample Adaptive Server Anywhere LTM configuration file.

```
# This is a comment line
# Names are case sensitive.
SQL_user=sa
SQL_pw=sysadmin
SQL_server=PRIMESV
SQL_database=primedb
RS_source_ds=PRIMESV
RS_source_db=primedb
RS=MY_REPSERVER
RS_user=sa
RS_pw=sysadmin
LTM_admin_user=DBA
LTM_admin_pw=SQL
LTM_charset=cp850
scan_retry=2
SQL_log_files=e:\logs\backup
APC_user=sa
APC_pw=sysadmin
```

Replicating transactions in batches

Effects of buffering transactions

The LTM allows buffering of replication commands to Replication Server. Buffering the replication commands leads in general to two main effects:

- **Longer lag before replication** As transactions are not sent to Replication Server by the LTM until the buffer is full, there may be a longer delay between a transaction being committed in Adaptive Server Anywhere and being replicated to Replication Server. This delay will be more noticeable on low volume installations, where transactions may wait in the buffer for some time until the next transaction is committed.

- **Higher throughput** Buffering the transactions and sending them in batches results in fewer messages being sent, and can significantly increase overall throughput, especially on high volume installations.

How batch mode works

By default, the LTM buffers transactions. The buffer is flushed (the transactions sent to Replication Server) when any of the following conditions is met:

- **Maximum number of commands** The **batch_ltl_siz** parameter sets the maximum number of LTL (log transfer language) commands that are stored in the buffer before it is flushed. The default setting is 200.
- **Maximum memory used** The **batch_ltl_mem** parameter sets the maximum memory that the buffer can occupy before it is flushed. The default setting is 256 K.
- **Transaction log processing completed** If there are no more entries in the transaction log to be processed (that is, the LTM is up to date with all committed transactions), then the buffer is flushed.

Turning off buffering

You can turn off buffering of transactions by setting the **batch_ltl_cmds** configuration parameter to **off**:

```
batch_ltl_cmds=off
```

Language and character set issues

Language and character set issues are an important consideration in many replication sites. Each database and server in the system uses a specific **collation** (character set and sorting order) for storing and ordering strings. Adaptive Server Anywhere character set support is carried out in a different manner to character set support in Adaptive Server Enterprise and other Open Client/Open Server based applications.

This section describes how to configure the Adaptive Server Anywhere LTM such that data in a Adaptive Server Anywhere database can be shared with Replication Server and hence with other databases.

The LTM automatically uses the default Open Client/Open Server language, sort order, and character set. You can override these defaults by adding entries to the LTM configuration file.

Adaptive Server Anywhere collations

Each Adaptive Server Anywhere database is assigned a specific collation when it is created.

❖ **To find the collation sequence used by a database:**

- Connect to the database from Sybase Central, right click on the database icon in the left pane, and select Properties from the pop-up menu. The collation is listed on the Extended Information tab. Or:
- Run the *dbinfo* command-line utility. For example, the following command provides information about the Adaptive Server Anywhere sample database:

```
dbinfo c:\asa60\sademo.db
```

If no collation sequence is specified when the database is created, then the 850LATIN1 collation sequence is used.

☞ For a full description of Adaptive Server Anywhere collations, see "Database Collations and International Languages" on page 289.

Open Client/Open Server collations

Adaptive Server Enterprise, Replication Server, and other Open Client/Open Server applications share a common means of managing character sets.

☞ For information on Open Client/Open Server character set support, see the chapter "Configuring Character Sets, Sort Orders, and Message Language" in the *Adaptive Server Enterprise Administration Guide*. For more information about character set issues in Replication Server, see the chapter "International Replication Design Considerations" in the *Replication Server Design Guide*.

This section provides a brief overview of Open Client/Open Server character set support.

Internationalization files

Files that support data processing in a particular language are called **internationalization files**. Several types of internationalization files are supplied with Adaptive Server Enterprise and other Open Client/Open Server applications.

There is a directory named charsets under your Sybase directory. Charsets has a set of subdirectories: one for each character set available to you. Each character set contains a set of files, as described in the followning table

File	Description
charset.loc	Character set definition files that define the lexical properties of each character such as alphanumeric, punctuation, operand, upper or lower case.
*.srt	Defines the sort order for alpha-numeric and special characters.
*.xlt	Terminal-specific character translation files for use with utilities.

Character set settings in the LTM configuration file

There are three settings in the LTM configuration file that refer to character set issues:

- **LTM_charset** The **character set** for the LTM to use. You can specify any Sybase-supported character set.
- **LTM_language** The **language** used by the LTM to print its messages to the error log and to its clients. You can specify any language to which the LTM has been localized that is compatible with the LTM character set.

  The Adaptive Server Anywhere LTM has been localized to (US) English and Japanese.
- **LTM_sortorder** The **sort order** that the LTM uses to compare user names. You can specify any Open Client/Open Server sort order that is compatible with the LTM's character set.

Notes

Character set In an Open Client/Open Server environment, an LTM should use the same character set as the data server and Replication Server attached to it.

Adaptive Server Anywhere character sets are differently specified than Open Client/Open Server character sets. Consequently, the requirement is that the Adaptive Server Anywhere character set be compatible with the LTM character set.

Language Valid settings are those mapped in the *locales.dat* file in the *locales* subdirectory of the Sybase release directory. However, the LTM output messages in the user interface are currently available in English and Japanese.

Sort order All sort orders in your replication system should be the same. The default is the default entry for your platform in the *locales.dat* file in the *locales* subdirectory of the Sybase release directory.

Example

- The following settings are valid for a Japanese installation:

```
LTM_charset=SJIS
LTM_sortorder=bin_sjis
LTM_language=Japanese
```

Transaction log and backup management

One of the differences between the Adaptive Server Enterprise LTM and the Adaptive Server Anywhere LTM is that while the Adaptive Server Enterprise LTM depends on a temporary recovery database for access to old transactions, the Adaptive Server Anywhere LTM depends on access to old transaction logs. There is no temporary recovery database for the Adaptive Server Anywhere LTM.

Replication depends on access to operations in the transaction log, and for Adaptive Server Anywhere primary site databases access to old transaction logs is sometimes required. This section describes how to set up backup procedures at a Adaptive Server Anywhere primary site to ensure proper access to old transaction logs.

Consequences of lost transaction logs

It is crucial to have good backup practices at Adaptive Server Anywhere primary database sites. A lost transaction log could mean having to resynchronize replicate site databases. At primary database sites, a transaction log mirror is recommended. For information on transaction log mirrors and other backup procedure information, see the Adaptive Server Anywhere *User's Guide*.

The LTM configuration file contains a directory entry, which points to the directory where backed up transaction logs are kept. This section describes how you can set up a backup procedure to ensure that such a directory is kept in proper shape.

Backup utility options

One of the options to the Backup utility is to rename the transaction log on backup and restart. For the *DBBACKUP* command-line utility, this is the `-r` command-line switch. It is recommended that you use this option when backing up the consolidated database and remote database transaction logs.

To see how this option works, consider a database named *primedb.db*, in directory *c:\prime*, with a transaction log in directory *d:\primelog\primedb.log*. Backing up this transaction log to a directory *e:\primebak* using the rename and restart option carries out the following tasks:

1. Backs up the transaction log, creating a backup file *e:\primebak\primedb.log*.
2. Renames the existing transaction log to *d:\primelog\primedb.lnn*, where **nn** is the lowest available integer, starting at **00**.
3. Starts a new transaction log, as *d:\primelog\primedb.log*.

After several backups, the directory *d:\primelog* will contain a set of sequential transaction logs. The log directory should not contain any transaction logs other than the sequence of logs generated by this backup procedure.

You can also run the LTM pointing to the directory where backup copies are held. However, the backup utility makes backups to the same file name each time by default, so you will need to ensure that old logs are renamed before subsequent backups.

Using the DELETE_OLD_LOGS option

The DELETE_OLD_LOGS Adaptive Server Anywhere database option is set by default to OFF. If it is set to ON, then the old transaction logs will be deleted automatically by the LTM when Replication Server no longer needs access to the transactions. This option can help to manage disk space in replication setups.

You can set the DELETE_OLD_LOGS option either for the PUBLIC group or just for the maintenance user. The following statement sets the option for the PUBLIC group:

```
SET OPTION PUBLIC.DELETE_OLD_LOGS = 'ON'
```

☞ For more information, see "DELETE_OLD_LOGS option" on page 154 of the book *Adaptive Server Anywhere Reference Manual*.

The Unload utility and replication

If a database is participating in replication, it cannot be unloaded and reloaded without re-synchronizing the database. Replication is based on the transaction log, and when a database is unloaded and reloaded, the old transaction log is no longer available. For this reason, good backup practices are especially important when participating in replication.

Replicating an entire database

Adaptive Server Anywhere provides a short cut for replicating an entire database, instead of having to set each table in the database as a replicated table.

A PUBLIC database option called REPLICATE_ALL can be set using the SET OPTION statement. You can set a whole database to be replicated using the command:

```
SET OPTION PUBLIC.Replicate_all='ON'
```

You require DBA authority to change this and other PUBLIC option settings. You must restart the database for the new setting to take effect. The REPLICATE_ALL option has no effect on procedures.

☞ For more information, see "REPLICATE_ALL option" on page 171 of the book *Adaptive Server Anywhere Reference Manual*.

Stopping the LTM

The LTM can be shut down from the user interface in Windows NT, and in other circumstances must be shut down by issuing a command.

❖ **To stop the LTM in Windows NT, when the LTM is not running as a service:**

`♦ Click SHUTDOWN on the user interface.

❖ **To stop the LTM by issuing a command:**

1 Connect to the LTM from *isql* using the LTM_admin_user login name and password in the LTM configuration file. The user ID and password are case sensitive.

2 Stop the LTM using the SHUTDOWN statement.

Example

♦ The following statements connect *isql* to the LTM PRIMELTM, and shut it down:

```
isql -SPRIMELTM -UDBA -PSQL
1> shutdown
2> go
```