

CHAPTER 9

Deploying and Synchronizing Databases

About this chapter This chapter describes the steps you need to take to deploy and synchronize a SQL Remote replication installation.

Contents

Topic	Page
Deployment overview	204
Test before deployment	205
Synchronizing databases	207
Using the extraction utility	209
Synchronizing data over a message system	216

Deployment overview

	<p>When you have completed the design phase of a SQL Remote system, the next step is to create and deploy the remote databases and applications.</p>
Deployment tasks	<p>In some cases, deployment is a major undertaking. For example, if you have a large number of remote users in a sales force automation system, deployment involves the following steps:</p> <ol style="list-style-type: none">1 Building an Adaptive Server Anywhere database for each remote user, with their own initial copy of the data.2 Installing the database, together with the Adaptive Server Anywhere database server, the SQL Remote Message Agent, and client application, on each user's machine.3 Ensuring that the system is properly configured, with correct user names, Message Agent connection strings, permissions, and so on. <p>In the case of large-scale deployments, remote sites are most commonly Adaptive Server Anywhere databases, and this chapter focuses on this case.</p>
Topics covered	<p>This chapter covers the following topics:</p> <ul style="list-style-type: none">◆ Creating remote databases Before you can deploy a SQL Remote system, you must create a remote database for each remote site. Most of the description focuses on creating remote Adaptive Server Anywhere databases.◆ Synchronizing data Synchronization of a database is the setting up of the initial copy of data in the remote database.

Test before deployment

Thorough testing of your SQL Remote system should be carried out before deployment, especially if you have a large number of remote sites.

When you are in the design and setup phase, you can alter many facets of the SQL Remote setup. Altering publications, message types, writing triggers to resolve update conflicts are all easy to do.

Once you have deployed a SQL Remote application, the situation is different. A SQL Remote setup can be seen as a single **dispersed database**, spread out over many sites, maintaining a loose form of consistency. The data may never be in exactly the same state in all databases in the setup at once, but all data changes are replicated as complete transactions around the system over time. Consistency is built in to a SQL Remote setup through careful publication design, and through the reconciliation of UPDATE conflicts as they occur.

Upgrading and resynchronization

Once a SQL Remote setup is deployed and is running, it is not easy to tinker with. An upgrade to a SQL Remote installation needs to be carried out with the same care as an initial deployment. This applies also to upgrading maintenance releases of the Adaptive Server Enterprise or Adaptive Server Anywhere database software. Any such software upgrade needs to be tested for compatibility before deployment.

Making changes to a database schema at one database within the system can cause failures because of incompatible database objects. The passthrough mode does allow schema changes to be sent to some or all databases in a SQL Remote setup, but must still be used with care and planning.

The loose consistency in the dispersed database means that updates are always in progress: you cannot generally stop changes being made to all databases, make some changes to the database schema, and restart.

Without careful planning, changes to a database schema will produce errors throughout the installation, and will require all subscriptions to be stopped and resynchronized. Resynchronization involves loading new copies of the data in each remote database, and for more than a few subscribers is a time-consuming process involving work interruptions and possible loss of data.

Changes to avoid on a running system

The following are examples of changes that should not be made to a deployed and running SQL Remote setup. From the list, you will see that there is a class of changes that are **permissive**, and these are generally permissible, while other changes are **restrictive**, and must be avoided.

The following changes must be avoided, except under the conditions stated:

- ◆ Change the publisher for the consolidated database.
- ◆ Make restrictive changes to tables, such as dropping a column or altering a column to not allow NULL values. Changes that include the column or including NULL entries may already be being sent in messages around the SQL Remote setup, and will fail.
- ◆ Alter a publication. Publication definitions must be maintained at both local and remote sites, and changes that rely on the old publication definition may already be being sent in messages around the SQL Remote setup.

You can make permissive changes, such as adding a new table or column, as long as you use passthrough to ensure that the new table or column exists in the remote database and in the publication at the remote database.

- ◆ Drop a subscription. This can be done only if you use passthrough deletes to remove the data at the remote site.
- ◆ Unload and reload an Adaptive Server Anywhere database.

If an Adaptive Server Anywhere 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.

An Adaptive Server Enterprise database can be unloaded and reloaded as long as the system is quiet and the transaction log is fully scanned. The **page_id** and **row_id** rows in the **sr_queue_state** table of the stable queue must be reset.

Synchronizing databases

What is synchronization?

SQL Remote replication is carried out using the information in the transaction log, but there are two circumstances where SQL Remote deletes all existing rows from those tables of a remote database that form part of a publication, and copies the publication's entire contents from the consolidated database to the remote site. This process is called **synchronization**.

When to synchronize

Synchronization is used under the following circumstances:

- ◆ When a subscription is created at a consolidated database a synchronization is carried out, so that the remote database starts off with a database in the same state as the consolidated database.
- ◆ If a remote database gets corrupt or gets out of step with the consolidated database, and cannot be repaired using SQL passthrough mode, synchronization forces the remote site database back in step with the consolidated site.

How to synchronize

Synchronizing a remote database can be done in the following ways:

- ◆ **Use the database extraction utility** This utility creates a schema for a remote Adaptive Server Anywhere database, and synchronizes the remote database. This is generally the recommended procedure.
- ◆ **Manual synchronization** Synchronize the remote database manually by loading from files, using the PowerBuilder pipeline, or some other tool.
- ◆ **Synchronize over the message system** Synchronize the remote database via the message system using the SYNCHRONIZE SUBSCRIPTION statement (Adaptive Server Anywhere) or `sp_subscription 'synchronize'` procedure (Adaptive Server Enterprise).

Caution

Do not execute SYNCHRONIZE SUBSCRIPTION or `sp_subscription 'synchronize'` at a remote database.

Mixed operating systems and database extraction

In many installations, the consolidated server will be running on a different operating system than the remote databases.

Adaptive Server Anywhere databases can be copied from one file or operating system to another. This allows you flexibility in how you carry out your initial synchronization of databases.

Example

For example, you may be running an Adaptive Server Enterprise server on a UNIX system that holds the consolidated database, but wish to deploy remote databases on laptop computers running Windows 95.

In this circumstance, you have several options for the platforms on which you extract the database, including the following, assuming you have the requisite software:

- ◆ Run the extraction utility on UNIX to create the reload script and data files. Copy the script and data files to a Windows 95 machine. Create the Adaptive Server Anywhere databases and load them up with the schema and data on Windows 95.
- ◆ Run the extraction utility on UNIX to create the reload script and data files. Create the Adaptive Server Anywhere databases and load them up with the schema and data on the same UNIX platform, and then copy the database files onto Windows 95 machines for deployment.
- ◆ Run the extraction utility on Windows 95, and carry out all database creation and other tasks on the Windows 95 operating system.

Notes on synchronization and extraction

- ◆ Extracting large numbers of subscriptions, or synchronizing subscriptions to large, frequently-used tables, can slow down database access for other users. You may wish to extract such subscriptions when the database is not in heavy use. This happens automatically if you use a SEND AT clause with a quiet time specified.
- ◆ Synchronization applies to an entire subscription. There is currently no straightforward way of synchronizing a single table.
- ◆ For Adaptive Server Enterprise, Sybase Central does not display subscriptions as started until the Message Agent first runs against the database.

Using the extraction utility

The extraction utility is an aid to creating remote Adaptive Server Anywhere databases. It cannot be used to create remote Adaptive Server Enterprise databases.

Running the extraction utility

The extraction utility can be accessed in the following ways:

- ◆ From Sybase Central.
- ◆ As a command-line utility. This is the *dbextract* command-line utility (Adaptive Server Anywhere), or the *ssextract* command-line utility (Adaptive Server Enterprise).

Caution

Do not run the Message Agent while running the extraction utility. The results are unpredictable.

Creating a database from the reload files

The command-line utility unloads a database schema and data suitable for building a remote Adaptive Server Anywhere database for a named subscriber. It produces a SQL command file with default name *reload.sql* and a set of data files. You can use these files to create a remote Adaptive Server Anywhere database.

Editing of reload.sql may be needed

The database extraction utility is intended to assist in preparing remote databases, but is not intended as a black box solution for all circumstances. You should edit the *reload.sql* command file as needed when creating remote databases.

❖ **To create a remote database from the reload file:**

- 1 Create an Adaptive Server Anywhere database using Sybase Central or using the *dbinit* utility.
- 2 Connect to the database from the Interactive SQL utility, and run the *reload.sql* command file. The following statement entered in the Interactive SQL command window runs the *reload.sql* command file:

```
read path\reload.sql
```

where *path* is the path of the reload command file.


When used from Sybase Central, the extraction utility carries out the database unloading task, in the same way that *dbxtract* or *ssxtract* does, and then takes the additional step of creating the new database.

The extraction utility does not use a message system. The reload file (*ssxtract/dbxtract*) or database (from Sybase Central) is created in a directory accessible from the current machine. Synchronizing many subscriptions over a message link can produce heavy message traffic and, if the message system is not completely reliable, it may take some time for all the messages to be properly received at the remote sites.

Before extracting a database

You must complete the following tasks before using the extraction utility at a consolidated database.

- ◆ You must have created message types for replication.
- ◆ You must have added a publisher user ID to the database.
- ◆ You must have added remote users to the database.
- ◆ You must have added the publication to the database.
- ◆ You must have created a subscription for the remote users.

 For a description of how to carry out these steps, see the tutorial in the chapter "A Tutorial for Adaptive Server Anywhere Users" on page 37.

When you use the extraction utility to create a remote database, the user for which you are creating the database receives the same permissions they have in the consolidated database. Further, if the user is a member of any groups on the consolidated database, those group IDs are created in the remote database with the permissions they have in the consolidated database.

Using the extraction utility from Sybase Central

For full information on using the extraction utility from Sybase Central, see the Sybase Central online Help. This section describes one way to extract a database for a remote user from the current consolidated database.

- ❖ **To extract a database for a remote user:**
 - 1 Click the Remote Users folder on the left panel, which is in the SQL Remote folder. The right panel displays the remote users.

- 2 Right-click the remote user for whom you wish to extract a database, and select Extract Database from the popup menu. The Extraction Wizard is displayed.
- 3 Follow the instructions in the Wizard.

For more information

For information about the extraction utility options, available as command-line options or as choices presented by the Database Extraction Wizard, see "Extraction utility options" on page 317.

Designing an efficient extraction procedure

It is very inefficient to create a large number of remote databases by running the extraction utility for each one. You can make the process much more efficient. This section describes one way of making the process more efficient.

There are several potential causes of inefficiency in a large-scale extraction process:

- ◆ The extraction utility extracts one database at a time, including the schema and data for each user. Commonly, many users share a common schema, and only the data differs. The brute force method of running the extraction utility for each user repeats large amounts of work unnecessarily. Extracting schema and data separately can help with this problem.
- ◆ Running from Sybase Central, the extraction utility creates a new database for each user. If subscribers share a common schema, you could create a single database, with schema but no data, and copy the file.
- ◆ By default, the extraction utility runs at isolation level zero. If you are extracting a database from an active server, you should run it at isolation level 3 (see "Extraction utility options" on page 317) to ensure that data in the extracted database is consistent with data on the server.

Running at isolation level 3 may hamper others' turnaround time on the server because of the large number of locks required. It is recommended that you run the extraction utility when the server is not busy, or run it against a copy of the database.

An efficient approach to extracting many databases

One approach that avoids these problems is as follows:

- 1 Make a copy of the consolidated database, and at the same time start the subscriptions from the live database. Messages will now start being sent to subscribers, even though they have no database and will not receive them yet.

To start several subscriptions within a single transaction, use the REMOTE RESET statement (Adaptive Server Anywhere) or **sp_remote** procedure (Adaptive Server Enterprise).

- 2 Extract the remote databases from the copy of the database. As the database is a copy, there are no locking and concurrency problems. For a large number of remote databases, this process may take several days.
- 3 As each remote database is created, it is out of date, but its user can receive and apply messages that have been being sent from the live consolidated database, to bring themselves up to date.

This solution interferes with the production database only during the first step. The copy must be made at isolation level three if the database is in use, and uses large numbers of locks. Also, the subscriptions must be started at the same time that the copy is made. Any operations that take place between the copy and the starting of the subscriptions would be lost, and could lead to errors at remote databases.

Limits to using the extraction utility

While the extraction utility is the recommended way of creating and synchronizing remote databases from a consolidated databases, there are some circumstances where it cannot be used, and you must synchronize remote databases manually. This section describes some of those cases.

- ◆ **Cannot create Adaptive Server Enterprise remote databases** The extraction utility can only be used for Adaptive Server Anywhere remote databases.
- ◆ **Additional tables at the remote database** Remote databases can have tables not present at their consolidated database as long as these tables do not take part in replication. Of course, the extraction utility cannot extract such tables from a consolidated database.
- ◆ **Adaptive Server Enterprise/Adaptive Server Anywhere differences** Some features in Adaptive Server Enterprise are not present in Adaptive Server Anywhere. The extraction utility carries out a mapping onto similar features, but the mapping is not complete.

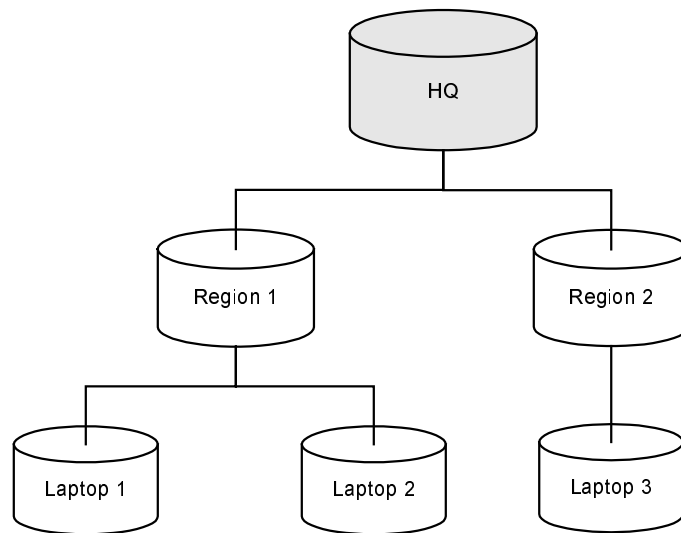
↪ For more information on Adaptive Server Enterprise/Adaptive Server Anywhere issues, see "Using the extraction utility for Adaptive Server Enterprise" on page 214.

- ◆ **Extracting procedures and views** By default, the extraction utility extracts all stored procedures and views from the database. While some of these views and procedures are likely to be required at the remote site, others may not be required—they may refer only to parts of the database that are not included in the remote site.

After running the extraction utility, you should edit the reload script and remove unnecessary views and procedures.

- ◆ **Using the extraction utility in multi-tiered setups** To understand the role of the extraction utility in multi-tiered arrangements, consider a three-tiered SQL Remote setup.

This setup is illustrated in the following diagram.



From the consolidated database at the top level, you can use the extraction utility to create the second-level databases. You can then add remote users to these second-level databases, and use the extraction utility from each second-level database to create the remote databases. However, if you have to re-extract the second-level databases from the top-level consolidated database, you will delete the remote users that were created, along with their subscriptions and permissions, and will have to rebuild those users. The exception is if you resynchronize data only, in which case you can use the extraction utility to replace the data in the database, without replacing the schema.

Using the extraction utility for Adaptive Server Enterprise

The extraction utility for Adaptive Server Enterprise takes an Adaptive Server Enterprise database schema, and produces an Adaptive Server Anywhere database. There are several limitations and techniques specific to this tool.

Adaptive Server Enterprise features unsupported in Adaptive Server Anywhere

There are some features in Adaptive Server Enterprise that are either not supported or are only partially supported in Adaptive Server Anywhere. The extraction utility handles some of these features partially, and some not at all.

☞ For a full description of Adaptive Server Enterprise/Adaptive Server Anywhere compatibility, see the part *Transact-SQL Compatibility*, in the *Adaptive Server Anywhere User's Guide*.

Features not supported in *ssxtract* include the following:

- ◆ **Grouped procedures** Adaptive Server Anywhere does not support procedure groups, and they are not extracted by *ssxtract*.
- ◆ **Named constraints and defaults** Adaptive Server Anywhere does not support named constraints and named defaults. Any such objects are extracted directly as constraints and defaults that apply to a single object, and the name is lost.
- ◆ **Roles** *ssxtract* extracts roles using the Adaptive Server Anywhere concept of groups. It creates a group with the named role, and assigns users to it.
- ◆ **Passwords** If the user for whom a database is being extracted does not have an entry in SYSLOGINS, no password is extracted. If the user does have a login ID, a dummy password is extracted.
- ◆ **NCHAR, NVARCHAR** These data types are extracted as CHAR and VARCHAR, with NULLS allowed.
- ◆ **timestamp columns** Although Adaptive Server Anywhere does provide a timestamp column, it is a different data type from that of Adaptive Server Enterprise. Timestamp columns are not extracted.

Customizing the system tables

The objects that are to be loaded into an Adaptive Server Anywhere database are described in the system catalog. The extraction utility for Adaptive Server Enterprise first creates a set of Adaptive Server Anywhere system tables in TEMPDB, and fills them with data from the Adaptive Server Enterprise catalog. It then unloads this set of tables to provide the reload script that in turn builds an Adaptive Server Anywhere database.

There may be cases where you wish to change the content of the Adaptive Server Anywhere system tables held in TEMPDB. SQL Remote provides a place for you to do that.

The stored procedure that creates and fills the Adaptive Server Anywhere system objects in TEMPDB is called **sp_populate_sql_anywhere**. As its final operation, this procedure calls a procedure called **sp_user_extraction_hook**. This procedure, by default, does nothing. If you wish to customize the extraction procedure, you can do so by writing a suitable **sp_user_extraction_hook** procedure.

Synchronizing data over a message system

Creating subscriptions

A subscription is created at a consolidated Adaptive Server Anywhere database using the **sp_subscription** procedure with a first argument of **create**.

Creating a subscription defines the data to be received. It does not synchronize a subscription (provide an initial copy of the data) or start (exchange messages) a subscription.

Synchronizing subscriptions

Synchronizing a subscription causes the Message Agent to send a copy of all rows in the subscription to the subscriber. It assumes that an appropriate database schema is in place. At an Adaptive Server Anywhere consolidated database, subscriptions are synchronized using the SYNCHRONIZE SUBSCRIPTION statement. At an Adaptive Server Enterprise consolidated database, subscriptions are synchronized using the **sp_subscription** procedure with a first argument of **synchronize**.

When synchronization messages are received at a subscriber database, the Message Agent replaces the current contents of the database with the new copy. Any data at the subscriber that is part of the subscription, and which has not been replicated to the consolidated database, is lost. Once synchronization is complete, the subscription is started by the Message Agent using the START SUBSCRIPTION statement or **sp_subscription** procedure with a first argument of **start**.

Large volume of messages may result

Synchronizing databases over a message system may lead to large volumes of messages. In many cases, it is preferable to use the extraction process to synchronize a database locally without placing this burden on the message system.

Synchronizing subscriptions during operation

If a remote database becomes out of step with the consolidated database, and cannot be brought back in step using the SQL passthrough capabilities of SQL Remote, synchronizing the subscription forces the remote database into step with the consolidated database by copying the rows of the subscription from the consolidated database over the contents at the remote database.

Data loss on synchronization

Any data in the remote database that is part of the subscription, but which has not been replicated to the consolidated database, is lost when the subscription is synchronized. You may wish to unload or back up the remote database using Sybase Central or, for Adaptive Server Anywhere, the *dbunload* utility before synchronizing the database.