CHAPTER 5

A Tutorial for Adaptive Server Enterprise Users

About this chapter

This chapter presents a tutorial in which you set up a simple SQL Remote replication system between an Adaptive Server Enterprise database and an Adaptive Server Anywhere database, from scratch.

Contents

Introduction

This chapter presents a tutorial to lead you through setting up a SQL Remote installation. The installation replicates data between an Adaptive Server Enterprise database (the consolidated database) and an Adaptive Server Anywhere database (the remote database).

Tutorial goals

In the tutorial you act as the system administrator of a consolidated Adaptive Server Enterprise database, and set up a simple replication system. The replication system consists of a simple sales database, with two tables.

The consolidated database holds all of the database, while the remote database has all of one table, but only some of the rows in the other table.

The tutorial takes you through the following steps:

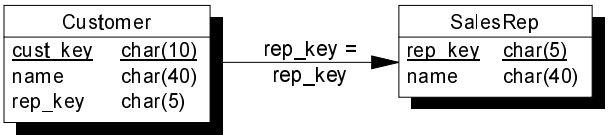
- Creating a consolidated database on your Adaptive Server Enterprise server.
- Creating a file-sharing replication system with a single Adaptive Server Anywhere remote database.
- Replicating data between the two databases.

The database

The tutorial uses a simple two-table database. One table holds information about sales representatives, and the other about customers. The tables are much simpler than you would use in a real database; this allows us to focus just on those issues important for replication.

Database schema

The database schema for the tutorial is illustrated in the figure.



Features to note include the following:

- Each sales representative is represented by one row in the SalesRep table.
- Each customer is represented by one row in the customer table.

- Each customer is assigned to a single Sales representative, and this assignment is built in to the database as a foreign key from the Customer table to the SalesRep table. The relationship between the Customer table and the SalesRep table is many-to-one.

The tables in the database

The tables are described in more detail as follows:

Table	Description

SalesRep	One row for each sales representative that works for the company. The SalesRep table has the following columns:
	♦ **rep_key** An identifier for each sales representative. This is the primary key.
	♦ **name** The name of each sales representative.
	The SQL statement creating this table is as follows:
	`CREATE TABLE SalesRep (`
	`rep_key CHAR(12) NOT NULL,`
	`name CHAR(40) NOT NULL,`
	`PRIMARY KEY (rep_key)`
	`)`

Customer	One row for each customer that does business with the company. The Customer table includes the following columns:
	♦ **cust_key** An identifier for each customer. This is the primary key.
	♦ **name** The name of each customer.
	♦ **rep_key** An identifier for the sales representative in a sales relationship. This is a foreign key to the SalesRep table.
	The SQL statement creating this table is as follows:
	`CREATE TABLE Customer (`
	`cust_key CHAR(12) NOT NULL,`
	`name CHAR(40) NOT NULL,`
	`rep_key CHAR(12) NOT NULL,`
	`FOREIGN KEY ( rep_key )`
	`REFERENCES SalesRep (rep_key ),`
	`PRIMARY KEY (cust_key)`
	`)`

Replication goals

The goals of the replication design are to provide each sales representative with the following information:

- The complete SalesRep table.
- Those customers assigned to them.

The tutorial describes how to meet this goal using SQL Remote.

Use Sybase Central or the command line

The tutorial material is presented twice. One section describes how to set up the installation using the Sybase Central management utility. The second section describes how to set up the installation using the Adaptive Server Enterprise and Adaptive Server Anywhere interactive SQL utilities: this requires typing commands individually.

Where next?

- To work through the tutorial using Sybase Central, go to "Setting up SQL Remote using Sybase Central" on page 69.
- To work through the tutorial entering commands explicitly, go to "A tutorial using isql and ssxtract" on page 79.

Setting up SQL Remote using Sybase Central

The following sections are a tutorial describing how to set up a simple SQL Remote replication system using Sybase Central.

You do not need to enter SQL statements if you are using Sybase Central to administer SQL Remote. A tutorial for those who do not have access to Sybase Central, who prefer to use command-line utilities, or who wish to understand the individual commands executed, is presented in "A tutorial using isql and ssxtract" on page 79. The commands described in that section are executed behind the scenes by Sybase Central.

First steps

Create a login name and password

To work through the tutorial, you must have system administrator privileges on an Adaptive Server Enterprise server. The tutorial assumes that your login name is the two-letter word **sa** and that this login name has a password of **sysadmin**.

Create a database

Create a database named **hq** on your Adaptive Server Enterprise with sufficient space to hold the tables and data required by the tutorial database. A space of 10 Mb is sufficient.

❖ **To create a database from Sybase Central:**

1. Connect to the Adaptive Server Enterprise from Sybase Central.
2. Open the Databases folder, and double-click Add database in the right pane.
3. Enter the name **hq** on the first page of the Wizard.
4. Follow the instructions in the Wizard.

☞ For information on how to create databases and assign space to them, see your Adaptive Server Enterprise documentation.

Install SQL Remote

You need to install SQL Remote into the **hq** database.

❖ **To install SQL Remote into the HQ database:**

1. Open the **hq** database container, on the left pane of Sybase Central.
2. Open the SQL Remote folder, double-click Setup SQL Remote, and follow the instructions. For this tutorial, you should install the stable queue in the **hq** database.

If your TEMPDB database is too small, you may have to add space to it for this to work.

For a full description of how to install SQL Remote, see "Setting Up SQL Remote" on page 29.

Create directories for messages

Make a directory for the files created in this tutorial. For example:

```
mkdir c:\tutorial
```

You should create a directory for each of the two users of the replication system under your tutorial directory:

```
mkdir c:\tutorial\hq
mkdir c:\tutorial\field
```

The next step is to add a pair of tables to the consolidated database.

❖ **To add tables to the consolidated database:**

1. Connect to the **hq** database from Sybase Central, as a system administrator.
2. Click the User Tables folder of the **hq** database.
3. Double-click Add Table, and use the Table Editor to create a table named **SalesRep** with the following columns:

Key	Column	Data Type	Size/Prec
Primary key	**rep_key**	char	5
	name	char	40

You do not need to use the Advanced Properties window. The columns are created by default not allowing NULL.

4. Double-click Add Table again, and use the Table Editor to create a table named **Customer** with the following columns:

Key	Column	Data Type	Size/Prec
Primary key	**cust_key**	char	10
	name	char	40
	rep_key	char	5

Again, you do not need to use the Advanced Properties window. The columns are created by default not allowing NULL.

5 Open the Foreign Keys folder of the Customer table container, and double-click Add Foreign Key. Using the Wizard, add a foreign key to the **rep_key** column of the **SalesRep** table. You can use the default settings for this foreign key.

You are now ready for the rest of the tutorial.

Setting up the consolidated database

This section of the tutorial describes how to prepare the consolidated database of a simple replication system.

Preparing a consolidated database for replication involves the following steps:

1 Create a message type to use for replication.

2 Grant PUBLISH permissions to a user ID to identify the source of outgoing messages.

3 Grant REMOTE permissions to all user IDs that are to receive messages.

4 Create a publication describing the data to be replicated.

5 Create subscriptions describing who is to receive the publication.

You should have system administrator authority to carry out these tasks.

Add a SQL Remote message type

All messages sent as part of replication use a message type. A message type description has two parts:

- A message link supported by SQL Remote. In this tutorial, we use the FILE link.
- An address for this message link, to identify the source of outgoing messages.

Adaptive Server Anywhere databases already have message types created, but you need to supply an address for the message type you will use.

❖ **To add an address to a message type:**

1 From Sybase Central, open the **hq** database container.

2 Click the SQL Remote folder on the left panel.

3 Double-click the Message Types folder on the right panel.

4 Double-click the **file** message type.

5 Enter a publisher address to provide a return address for remote users. Enter *hq*; the directory you have created to hold messages for the consolidated database.

The address (**hq**) for a file link is a directory in which files containing the message are placed. It is taken relative to the SQLRemote environment variable or registry entry. As you have not set this value, the address is taken relative to the directory from which the Message Agent is run. *You should run the Message Agent from your tutorial directory for the addresses to be interpreted properly*.

For information about setting the SQLRemote value, see "Setting message type control parameters" on page 231.

6 Click OK to save the message type.

Create the necessary users and permissions

A set of users and permissions are required for SQL Remote installations. In this tutorial, the following are required:

- A publisher user name.
- A remote user or subscriber.

This section describes the steps you need to take to create each user and assign them the necessary permissions.

❖ **To create the publisher:**

1 From Sybase Central, open the container for the **hq** database.

2 Open the Users folder, and double-click Add User.

3 Create a user named **hq_user**, mapping the login name to an available login name on your server. In this tutorial, a password of **hq_pwd** is assumed.

4 Make this user the publisher of the HQ database. Open the SQL Remote folder, and double-click Set Publisher. Select **hq_user** from the list of users to set it as the publisher.

A database can have only one publisher. You can find out who the publisher is at any time by opening the SQL Remote folder.

Add a remote user

Each remote database is identified in the consolidated database by a user ID with REMOTE permissions. Whether the remote database is a personal server or a network server with many users, it needs a single user ID to represent it to the consolidated database.

In a mobile workgroup setting, remote users may already be users of the consolidated database, and so no new users would need to be added; although they would need to be set as remote users.

When a remote user is added to a database, the message system they use and their address under that message system need to be stored along with their database user ID.

❖ **To add a remote user:**

1. Double click the SQL Remote folder on the left panel, then click the Remote Users folder on the left panel.
2. If you do not have a login name that you can use for the remote user, open the logins folder directly under the server container, and add a login. The name is unimportant.

   Double-click Add Remote User on the right panel. The New Remote User wizard is displayed.
3. Create a remote user with name **field_user**. For the message type and address, select the FILE type and the corresponding address you are using for this user (such as *field*).

   As with the publisher address, the address of the remote user (**field**) is a directory relative to the SQLRemote environment variable or registry entry. As you have not set this value, the address is taken relative to the directory from which the Message Agent is run. You should run the Message Agent from your tutorial directory for the addresses to be interpreted properly.

   For information about setting the SQLRemote value, see "Setting message type control parameters" on page 231.
4. On the next page, ensure that the Send Then Close option is checked. (In many production environments you would not choose Send Then Close, but it is convenient for this tutorial.)
5. When you have finished all the entries, click Finish to create the remote user.

You have now created the users who will use this system.

Create the publication and subscription

This section describes how to add a publication to a database, and how to add a subscription to that publication for a user. The publication replicates all rows of the table **SalesRep** and some of the rows of the **Customer** table.

The first step is to mark the **SalesRep** and **Customer** tables for SQL Remote replication. Marking a table for SQL Remote replication enables it to be included in publications.

❖ **To mark the tables for SQL Remote replication:**

1 Open the SQL Remote folder in the **hq** database.

2 Open the Remote Tables folder, and double-click Add Remote Table.

3 Select **SalesRep** from the list of tables. You can leave the Conflict Resolution fields as they are. Click Apply to mark the table for SQL Remote replication.

4 Select **Customer** from the list of tables. Again, you can leave the Conflict Resolution fields as they are. Click OK to mark the table for SQL Remote replication.

❖ **To add a publication:**

1 Click the Publications folder in the SQL Remote folder.

2 Double-click Add Publication. The Publication Wizard is displayed.

3 Name the publication **SalesRepData** on the first page of the Wizard.

4 On the next page, click Add Table and select **SalesRep** from the list. Leave All Columns selected, and press OK to add the table.

Then click Add Table again, and select Customer from the list. Again, leave All Columns selected. Click the Subscribe Restriction tab, and choose to Subscribe by the column **rep_key**. Click OK to add the table to the publication.

5 Complete the Wizard to create the publication.

Add a subscription

Each user ID that is to receive changes to a publication must have a **subscription** to that publication. Subscriptions can only be created for a valid remote user. You need to add a subscription to the **SalesRepData** publication for the remote database user **field_user**.

❖ **To add a subscription:**

1 Double-click the Publications folder, which is in the SQL Remote folder, so that the **SalesRepData** publication is displayed in the *left* panel.

2 Click the Remote Users folder so that remote users are displayed in the *right* panel.

3 Drag the **field_user** user from the right panel onto the **SalesRepData** publication in the left panel. The Create Subscription window is displayed. Enter a value of **rep1** in the With Value box. The value **rep1** is the **rep_key** value we will give to the user **field_user** in the **SalesRep** table.

At this stage, the subscription is not **started**—that is, no data will be exchanged. The subscription is started by the database extraction utility.

You have now set up the consolidated database.

Set up the remote database in Sybase Central

The remote database needs to be created and configured in order to send and receive messages and participate in a SQL Remote setup.

Like the consolidated database, the remote database needs a publisher (in this case, the **field_user** user ID) to identify the source of outgoing messages, and it needs to have **hq_user** identified as a user with consolidated permissions. It needs the **SalesRepData** publication to be created and needs a subscription created for the **hq_user** user ID.

The remote database also needs to be **synchronized** with the consolidated database; that is, it needs to have a current copy of the data in order for the replication to start. In this case, there is no data in the publication as yet.

The database extraction utility enables you to carry out all the steps needed to create a remote database complete with subscriptions and required user IDs.

You need to extract a database from the consolidated database for remote user **field_user**.

❖ To extract a database:

1. Click the Remote Users folder, which is in the SQL Remote folder.
2. Right-click the **field_user** remote user, and select Extract Database from the popup menu. The Extraction Wizard is displayed.
3. Use the user ID and password that you have used to create the tables and users in the database.
4. On the next page, check Start Subscriptions Automatically, for user **field_user**. Also, check Create New Remote Database. Adaptive Server Anywhere must be installed for Create New Remote Database to be available.
5. Create the database as file *c:\tutorial\field.db*, using a transaction log in the same directory.
6. Choose to extract all parts of the schema (the default setting). Leave the other options at their default settings, and create the remote database.

You should connect to the **field** database as DBA and confirm that all the database objects are created.

❖ To connect to and inspect the remote database:

1. From Sybase Central, choose Tools➤Connect➤Sybase Adaptive Server Anywhere.

2 Provide the user ID **DBA** and the password **SQL**. These must be typed in upper case, as the database was created as case sensitive. Select the *field.db* file, and connect.

3 Open the database container, and confirm that the tables and user names are present.

In a proper SQL Remote setup, the remote database **field** would need to be loaded on to the computer using it, together with an Adaptive Server Anywhere server and any client applications required. For this tutorial, we leave the database where it is and use *isql* to input and replicate data.

What next?

The system is now ready for replication.

For the next step, inserting and replicating data, see the section "Start replicating data" on page 88.

A tutorial using isql and ssxtract

The following sections are a tutorial describing how to set up a simple SQL Remote replication system. This version of the tutorial does not use Sybase Central.

This tutorial describes the stored procedures used to configure and manage SQL Remote. It also describes how to run the *ssxtract* command-line utility to extract remote databases from a consolidated database and the Message Agents to send information between the databases in the replication system.

In this tutorial you act as the administrator of the consolidated database, and set up a simple replication system using the file-sharing message link. The simple example is a primitive model for a sales-force automation system, with two tables. One contains a list of sales representatives, and another a list of customers. The tables are replicated in a setup with one consolidated database and one remote database. You can install this example on one computer.

First steps

Create a login name and password

To work through the tutorial, you must have system administrator privileges on an Adaptive Server Enterprise server. The tutorial assumes that your login name is the two-letter word **sa** and that this login name has a password of **sysadmin**.

The tutorial uses the Adaptive Server Enterprise *isql* utility. With the login name and password as given above, you can connect to your Adaptive Server Enterprise server using the following command line:

```
isql -S server-name -U sa -P sysadmin
```

where *server-name* is the name of the Adaptive Server Enterprise to which you connect.

Ensure that you have an appropriate login ID and can connect to your server before starting this tutorial.

Create a database

Create a database named **hq** on your Adaptive Server Enterprise server with sufficient space to hold the tables and data required by the tutorial database. A space of 4 Mb is sufficient.

❖ **To create a database:**

1 Using *isql*, connect to the server as a user with system administrator privileges:

```
isql -S server-name -U sa -P sysadmin
```

2 Use the master database:

```
use master
go
```

3 Create a database named **hq**. In this example, we use a 5 Mb database with a 5 Mb log, on two different devices:

```
create database hq
on database_device = 5
log on log_device = 5
go
```

☞ For more information on how to create databases and assign space to them, see your Adaptive Server Enterprise documentation.

Install SQL Remote

You need to install SQL Remote into the **hq** database.

❖ **To install SQL Remote into the hq database:**

1 If the system administrator login name you are using does not have the **hq** database as the default database, make a backup copy of the *ssremote.sql* script from your installation directory, and add the following two lines to the beginning of the script:

```
use hq
go
```

2 Change to the tutorial directory. Then, using *isql*, connect to the server using the **hq** database, and run the *ssremote.sql* script from your SQL Remote installation directory. The following command should be entered all on one line:

```
isql -S server-name -U sa -P sysadmin -i
ssremote.sql
```

3 If the system administrator login name you are using does not have the **hq** database as the default database, make a backup copy of the *stableq.sql* script from your installation directory, and add the following two lines to the beginning of the script:

```
use hq
go
```

4 Using *isql*, connect to the server using the **hq** database, and run the *stableq.sql* script from your SQL Remote installation directory. The following command should be entered all on one line:

```
isql -S server-name -U sa -P sysadmin -i stableq.sql
```

Create directories for messages

Create a directory to hold the files from this tutorial. For example:

```
mkdir c:\tutorial
```

You should create a directory for each of the two users of the replication system under your parent directory for this tutorial:

```
mkdir c:\tutorial\hq
mkdir c:\tutorial\field
```

The next step is to add a pair of tables to the consolidated database.

❖ To add tables to the consolidated database:

1 Connect to the **hq** database from *isql*, as a system administrator.

2 Use the **hq** database:

```
use hq
go
```

3 Create the **SalesRep** table with the following statement:

```
create table SalesRep (
   rep_key char(12) not null,
   name char(40) not null,
   primary key (rep_key) )
go
```

4 Create the **Customer** table with the following statement:

```
create table Customer (
   cust_key char(12) not null,
   name char(40) not null,
   rep_key char(12) not null,
   primary key (cust_key) )
go
```

5 Alter the **Customer** table to add a foreign key to the **SalesRep** table:

```
alter table Customer
add foreign key
( rep_key ) references SalesRep
go
```

You are now ready for the rest of the tutorial.

Setting up the consolidated database

This section of the tutorial describes how to prepare the consolidated database of a simple replication system.

Preparing a consolidated database for replication involves the following steps:

1 Create a message type to use for replication.

2 Grant PUBLISH permissions to a user ID to identify the source of outgoing messages.

3 Grant REMOTE permissions to all user IDs that are to receive messages.

4 Create a publication describing the data to be replicated.

5 Create subscriptions describing who is to receive the publication.

You should have system administrator authority to carry out these tasks.

Create the message links and addresses

In this tutorial, messages are exchanged using the shared file link. You must create a FILE message type supplying the address of the consolidated database publisher.

❖ **To create the message type:**

♦ Execute the **sp_remote_type** stored procedure, using HQ as the address of the consolidated database publisher:

```
sp_remote_type file, hq
go
```

The address (**hq**) for a file link is a directory in which files containing the message are placed. It is taken relative to the SQLRemote environment variable or registry entry. As you have not set this value, the address is taken relative to the directory from which the Message Agent is run. You should run the Message Agent from your tutorial directory for the addresses to be interpreted properly.

For information about setting the SQLRemote value, see "Setting message type control parameters" on page 231.

With the message type defined, you can now make the necessary users.

Create the necessary users and permissions

A set of users and permissions are required for SQL Remote installations. In this tutorial, the following are required:

- A remote user or subscriber, with name **field_user**.
- A publisher user name, called **hq_user**.

This section describes the steps you need to take to create each user and assign them the necessary permissions.

❖ **To create the publisher:**

1 Add a login called **hq_user**, with **hq** as the default database and with system administrator access:

```
exec sp_addlogin hq_user, hq_pwd, hq
go
exec sp_role 'grant', sa_role, hq_user
go
```

2 Add the login name as a user to the HQ database:

```
use hq
go
exec sp_adduser hq_user
go
```

3 Make this user the publisher of the HQ database:

```
exec sp_publisher hq_user
go
```

Add a remote user

Each remote database is identified in the consolidated database by a user ID with REMOTE permissions. Whether the remote database is a single-user server or a database server with many users, it needs a single user ID to represent it to the consolidated database.

In a mobile workgroup setting, remote users may already be users of the consolidated database, and so no new users would need to be added; although they would need to be set as remote users.

When a remote user is added to a database, the message system they use and their address under that message system need to be stored along with their database user ID.

❖ **To create the subscriber:**

1 If you do not have a login name that you can use for the remote user, add a login:

```
exec sp_addlogin field_user, field_pwd, hq
```

```
go
```

2 Add a user to the **hq** database:

```
exec sp_adduser field_user
go
```

3 Grant the user remote permissions. Execute the **sp_grant_remote** stored procedure, using **field_user** as the user name, **file** as the message type, and the appropriate directory as the address:

```
exec sp_grant_remote field_user, file, field
go
```

As with the publisher address, the address of the remote user (**field**) is a directory relative to the SQLRemote environment variable or registry entry. As you have not set this value, the address is taken relative to the directory from which the Message Agent is run. You should run the Message Agent from your tutorial directory for the addresses to be interpreted properly.

For information about setting the SQLRemote value, see "Setting message type control parameters" on page 231.

Create the publication and subscription

The remaining task is to define the data to be replicated. To do this, you must first create a publication, which defines the available data, and then create a subscription for **field_user**, which defines the data that user is sharing.

In Adaptive Server Enterprise, they are created with the **sp_create_publication** procedure, which creates an empty publication, and the **sp_add_article** procedure, which adds articles to the procedure. Also, each table must be marked for replication before it can be included in a publication.

❖ To create the publication:

1 Create an empty publication:

```
exec sp_create_publication SalesRepData
go
```

2 Mark both the **SalesRep** table and the **Customer** table for publication:

```
exec sp_add_remote_table SalesRep
go
exec sp_add_remote_table Customer
go
```

3 Add the whole **SalesRep** table to the **SalesRepData** publication:

```
exec sp_add_article SalesRepData, SalesRep
go
```

4 Add the Customer table to the **SalesRepData** publication, using the **rep_key** column to partition the table. The following statement should be typed all on one line, except for the `go`:

```
exec sp_add_article SalesRepData, Customer, NULL,
'rep_key'
go
```

Add a subscription

Each user ID that is to receive changes to a publication must have a **subscription** to that publication. Subscriptions can only be created for a valid remote user. You need to add a subscription to the **SalesRepData** publication for the remote database user **field_user**.

❖ **To create a subscription:**

1 Create a subscription to **SalesRepData** for **field_user**, with a subscription value of **rep1**:

```
exec sp_subscription 'create', SalesRepData,
field_user, 'rep1'
go
```

At this stage, the subscription is not **started**—that is, no data will be exchanged. The subscription is started by the database extraction utility.

Extract the remote database

There are three stages to producing a remote Adaptive Server Anywhere database:

- Extract the schema and data into a set of files. You do this using the *ssxtract* command-line utility.
- Create an Adaptive Server Anywhere database.
- Load the schema and data into the database.

Extracting the schema and data

With all the information included, the next step is to extract an Adaptive Server Anywhere database for user **field_user**. The following command line (entered all on one line, from the tutorial directory) carries out this procedure:

```
ssxtract -v -c "eng=server-name;
dbn=hq;uid=sa;pwd=sysadmin" C:\tutorial\field field_user
```

The command-line arguments have the following meaning.

- **-v** Verbose mode. For development work, this provides additional output.
- **-c** Connection string argument. The connection string is supplied in double quotes following the `-c`.
- **eng=server-name** Specifies the server to which the extraction utility is to connect.
- **dbn=hq** Specifies the database on the server to use; in this case **hq**.
- **uid=sa** The login ID to use to log on to the database.
- **pwd=sysadmin** The password to use to log on to the database.
- **C:\tutorial\field** The directory in which to place files holding the data.
- **field_user** The user ID for which to extract the database.

For more information on extraction utility command-line switches, see "The extraction command-line utility" on page 315

Running this command produces the following files:

- **Reload script** The reload script is named *reload.sql*, and is placed in the current directory.
- **Data files** Files containing data to load into the database. In this case, these files are empty.

Creating an Adaptive Server Anywhere database

You can create an Adaptive Server Anywhere database using the *dbinit* command-line utility. A simple Adaptive Server Anywhere database is a file, unlike Adaptive Server Enterprise databases.

You should create the Adaptive Server Anywhere database so that it is compatible with Adaptive Server Enterprise database behavior, unless you have set options in your Adaptive Server Enterprise server that are different from the default.

❖ **To create a database file named field.db:**

- Enter the following command from the *c:\tutorial\field* directory:

```
dbinit -b -c -k field.db
```

The `-b` switch forces use of blank padding in string comparisons. The `-c` switch enforces case sensitivity for string comparisons. The `-k` switch makes the system catalog more compatible with Adaptive Server Enterprise.

Loading the data into the database

You can load the data into the database using the Adaptive Server Anywhere Interactive SQL utility or the *rtsql* command-line utility. *rtsql* is an alternative to Interactive SQL for batch processes only, and is provided for the runtime database.

❖ **To load the data into the database using Interactive SQL:**

1. Start an Adaptive Server Anywhere server running on the **field** database:

   ```
   dbeng6 field.db
   ```

2. Connect to the server using the Interactive SQL utility:

   ```
   dbisql -c "eng=field;dbn=field;uid=DBA;pwd=SQL"
   ```

   The user ID and password must be entered in upper case, as the Adaptive Server Anywhere database was created as case-sensitive.

3. Load the data using the READ command:

   ```
   READ C:\TUTORIAL\RELOAD.SQL
   ```

❖ **To load the data into the database as a batch process:**

1. Start an Adaptive Server Anywhere server running on the **field** database:

   ```
   dbeng6 field.db
   ```

2. Run the script from Interactive SQL:

   ```
   dbisql -c "eng=field;dbn=field;uid=DBA;pwd=SQL"
   reload.sql
   ```

   The user ID and password must be entered in upper case, as the Adaptive Server Anywhere database was created as case-sensitive.

What next?

The system is now ready for replication.

For the next step, inserting and replicating data, see the section "Start replicating data" on page 88.

Start replicating data

You now have a replication system in place. In this section, data is replicated from the consolidated database to the remote database, and from the remote to the consolidated database.

Enter data at the consolidated database

In this section we enter data into the **SalesRep** and **Customer** tables at the consolidated (Adaptive Server Enterprise) database, and replicate this data to the Adaptive Server Anywhere database.

❖ **To enter data at the Adaptive Server Enterprise database:**

1 Connect to the Adaptive Server Enterprise server from *isql*:

```
isql -S server-name -U sa -P sysadmin
```

2 Ensure you are using the **hq** database, and enter a series of rows:

```
use hq
go

insert into SalesRep (rep_key, name)
values ('rep1', 'Field User')
go

insert into SalesRep (rep_key, name)
values ('rep2', 'Another User')
go

insert into Customer (cust_key, name, rep_key)
values ('cust1', 'Ocean Sports', 'rep1')
go

insert into Customer (cust_key, name, rep_key)
values ('cust2', 'Sports Plus', 'rep2')
go

commit
go
```

Ocean Sports is assigned to **Field User**, and **Sports Plus** is assigned to **Another User**. You must commit the changes, as SQL Remote replicates only committed changes.

Having entered the data at the consolidated database, you now need to send the relevant rows to the remote Adaptive Server Anywhere database.

Send data from the consolidated database

To send the rows to the remote database, you must run the Message Agent at the consolidated database. The *ssremote* program is the Message Agent for Adaptive Server Enterprise.

❖ **To replicate the data from Adaptive Server Enterprise:**

♦ Enter the following statement (on a single line) at the command line to run the Message Agent against the consolidated database:

```
ssremote -c "eng=server-
name;dbn=hq;uid=sa;pwd=sysadmin"
```

3 Click Shutdown on the Message Agent window to stop the Message Agent when the messages have been sent.

Receive data at the remote database

To receive the insert statement at the remote database, you must run the Message Agent, *dbremote*, at the remote database.

❖ **To receive the data at Adaptive Server Anywhere:**

1 With the database server running, receive the data using the Message Agent for Adaptive Server Anywhere:

```
dbremote -c "eng=field;dbn=field;uid=DBA;pwd=SQL"
```

☞ For more information on *dbremote* command line switches, see "The Message Agent" on page 306.

2 Click Shutdown on the Message Agent window to stop the Message Agent when the messages have been processed.

The Message Agent window displays status information while running. This information can be output to a log file for record keeping in a production setup.

The Message Agent first receives a message from **hq**, and then sends a message. This return message contains confirmation of successful receipt of the replication update; such confirmations are part of the SQL Remote message tracking system that ensures message delivery even in the event of message system errors.

Verify that the data has arrived

You should now connect to the remote **field** database using Interactive SQL, and inspect the **SalesRep** and **Customer** tables, to see which rows have been received.

❖ **To verify that the data has arrived:**

1 Connect to the field database using Interactive SQL.

2 Inspect the **SalesRep** table by typing the following statement:

```
SELECT * FROM SalesRep
```

You will see that the **SalesRep** table contains both rows entered at the consolidated database. This is because the **SalesRepData** publication included all the data from the **SalesRep** table.

3 Inspect the **Customer** table by typing the following statement:

```
SELECT * FROM Customer
```

You will see that the **Customer** table contains only row (Ocean Sports) entered at the consolidated database. This is because the **SalesRepData** publication included only those customers assigned to the subscribed Sales Rep.

Replicate from the remote database to the consolidated database

You should now try entering data at the remote database and sending it to the consolidated database. Only the outlines are presented here.

❖ **To replicate data from the remote database to the consolidated database:**

1 Connect to the **field** database from Interactive SQL.

2 INSERT a row at the remote database. For example

```
INSERT INTO Customer (cust_key, name, rep_key)
VALUES ('cust3', 'North Land Trading', 'rep1')
```

3 COMMIT the row.:

```
COMMIT;
```

4 With the *field.db* database running, run *dbremote* to send the message to the consolidated database.

```
dbremote -c "eng=field;dbn=field;uid=DBA;pwd=SQL"
```

(For Windows 3.x, run the *dbremotw* equivalent.)

5 Run *ssremote* to receive the message at the consolidated database:

```
ssremote -c "eng=server-
name;dbn=hq;uid=sa;pwd=sysadmin"
```

6 Connect to the consolidated database and display the **Customer** table. This now has three rows:

```
SELECT *
FROM Customer
```

cust_key	name	rep_key
cust1	Ocean Sports	rep1
cust2	Sports Plus	rep2
cust3	North Land Trading	rep1

In this simple example, there is no protection against duplicate entries of primary key values. SQL Remote does provide for such protection. For information, see the chapters on SQL Remote Design.