

CHAPTER 4

A Tutorial for Adaptive Server Anywhere Users

About this chapter This chapter guides you through setting up a simple replication system.

Contents

Topic	Page
Tutorial overview	38
A tutorial using Sybase Central	41
Setting up a consolidated database	44
Set up the remote database in Sybase Central	49
A tutorial using Interactive SQL and DBXTRACT	51
Set up the consolidated database	53
Set up the remote database	57
Start replicating data	59
A sample publication	63

Tutorial overview

This tutorial describes how to set up a simple SQL Remote replication system using Adaptive Server Anywhere.

Tutorial goals

In the tutorial you act as the system administrator of a consolidated Adaptive Server Anywhere 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:

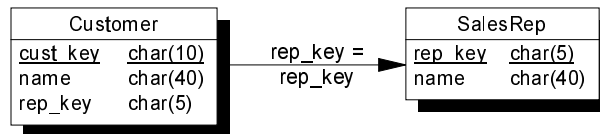
- ◆ Creating a consolidated database on your Adaptive Server Anywhere server.
- ◆ Creating a file-sharing replication system with a single Adaptive Server Anywhere remote database.
- ◆ Replicating data between the two databases.

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	<p>One row for each sales representative that works for the company. The SalesRep table has the following columns:</p> <ul style="list-style-type: none"> ◆ rep_key An identifier for each sales representative. This is the primary key. ◆ name The name of each sales representative. <p>The SQL statement creating this table is as follows:</p> <pre>CREATE TABLE SalesRep (rep_key CHAR(12) NOT NULL, name CHAR(40) NOT NULL, PRIMARY KEY (rep_key))</pre>
Customer	<p>One row for each customer that does business with the company. The Customer table includes the following columns:</p> <ul style="list-style-type: none"> ◆ 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. <p>The SQL statement creating this table is as follows:</p> <pre>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))</pre>

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.

Sybase Central or command-line utilities

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 command-line utilities: this requires typing commands individually.

Where next?


- ◆ To work through the tutorial using Sybase Central, go to "A tutorial using Sybase Central" on page 41, next.
- ◆ To work through the tutorial entering commands explicitly, go to "A tutorial using Interactive SQL and DBXTRACT" on page 51.

A tutorial using Sybase Central

The following sections are a tutorial describing how to set up a simple SQL Remote replication system in Adaptive Server Anywhere 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, or who prefer to work with command-line utilities, is presented in "A tutorial using Interactive SQL and DBXTRACT" on page 51, and contains the SQL statements executed behind the scenes by Sybase Central.

In this tutorial you act as the DBA 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.

 This tutorial assumes that you have some familiarity with Sybase Central. If you need help with Sybase Central, use the Sybase Central online Help.

Preparing for the Sybase Central replication tutorial

This section describes the steps you need to take to prepare for the tutorial. These steps include the following:

- ◆ Create the directories and databases required for the tutorial.
- ◆ Add the tables to the consolidated database.

❖ To prepare for the tutorial:

- 1 Create a directory to hold the files you make during this tutorial; for example *c:\tutorial*.

```
mkdir c:\tutorial
```

- 2 Create a subdirectory for each of the two user IDs in the replication system, to hold their messages. Create these subdirectories using the following statements at a system command line:

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

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

- 3 The tutorial uses two databases: a consolidated database named *hq.db* and a remote database named *field.db*. At this point, you should create the **hq** database with the Create Database utility in Sybase Central:
 - ◆ Start Sybase Central. You will be creating a new database so you do not have to connect to any particular existing database.
 - ◆ Click Utilities in the left panel.
 - ◆ Double-click Create Database in the right panel. The Create Database wizard is displayed.
 - ◆ Create a database with filename *c:\tutorial\hq.db*.
 - ◆ You can use the default settings for this database. Make sure you elect to maintain a transaction log. Replication cannot take place without a transaction log.

An Adaptive Server Anywhere database is simply a file, which can be copied to other locations and computers when necessary.

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 user ID **DBA** using password **SQL**.
- 2 Click the 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 a 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 require DBA 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, connect to the HQ database, and 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 the directory you have created to hold messages for the consolidated database (*hq*).

The address 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.

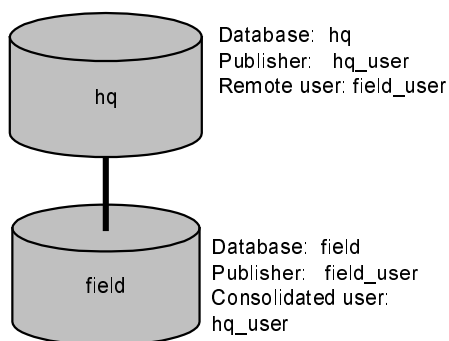
- 6 Click OK to save the message type.

Add the publisher and remote user to the database

In SQL Remote's hierarchical replication system, each database may have zero or one database immediately above it (the consolidated database) and zero or more databases immediately below it (remote databases).

In this tutorial, the current database is the consolidated database of a two-level system. It has no database above it, and only one remote database below it.

The following diagram illustrates the two databases:



For any database in a SQL Remote replication setup, there are three permissions that may be granted to identify databases on the hierarchy:

- ◆ **PUBLISH permission** Identifies the current database in all outgoing messages
- ◆ **REMOTE permission** Identifies each database receiving messages from the current database that is below it on the hierarchy
- ◆ **CONSOLIDATE permission** Identifies a database receiving messages from the current database that is directly above it on the hierarchy.

Add a database publisher user ID

Permissions can only be granted by a user with DBA authority. To carry out these examples you should connect from Sybase Central to the **hq** database as user ID **DBA**, with password **SQL**.

Any database, consolidated or remote, that distributes changes to other databases in the replication system is a publisher database. Each database in the replication system is identified by a single user ID. You set that ID for your database by adding a publisher to the database. This section describes setting permissions for the consolidated **hq** database.

First create a user ID named **hq_user**, who will be the publisher user ID.

❖ **To create the publisher:**

- 1 Click the Users & Groups folder on the left panel.
- 2 Double-click Add User. The New User Wizard is displayed.
- 3 Enter the name **hq_user**, with password **hq_pwd**, and click Next.
- 4 On the next page, ensure that the user is granted Remote DBA authority; this enables the user ID to run the Message Agent. Then click Next.
- 5 On the final page, check the box identifying this user ID as the publisher. Then click Finish to create the user.

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 database 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 Click the SQL Remote folder on the left panel, then click the Remote Users folder on the left panel.
- 2 Double-click Add Remote User on the right panel. The New Remote User wizard is displayed.

- 3 Create a remote user with user ID **field_user**, password **field_pwd**, message type **file**, and address **field**. 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 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.

- 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 On the next page ensure that the Remote DBA authority is checked, so that the user can run the Message Agent.
- 6 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.

Add publications and subscriptions

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.

❖ 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.

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 The first page informs you that you will extract from the running database **hq**.
- 4 On the next page, choose to Start Subscriptions Automatically, for user **field_user**.
- 5 Create the database as file *c:\tutorial\field.db*, using a transaction log of name *field.log* in the same directory.
- 6 Choose to extract all parts of the schema (the default setting).
- 7 Leave the other options at their default settings, and create the remote database.

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 Interactive SQL to input and replicate data.

You should connect to the **field** database as DBA and confirm that all the database objects are created. These include the **SalesRep** and **Customer** tables, the **SalesRepData** publication, and the subscription for the consolidated database.

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 59.

A tutorial using Interactive SQL and DBXTRACT

The following sections are a tutorial describing how to set up a simple SQL Remote replication system for users without Windows 95 or Windows NT 3.51 or later, who cannot run Sybase Central. It may also be useful to users of Sybase Central who prefer to use command-line tools or who want to know what Sybase Central is doing behind the scenes.

This tutorial describes the SQL statements for managing SQL Remote, which can be run from Interactive SQL. It also describes how to run the *dbxtract* command-line utility to extract remote databases from a consolidated database.

In this tutorial you act as the DBA 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.

Preparing for the replication tutorial

This section describes the steps you need to take to prepare for the tutorial. These steps include the following:

- ◆ Create the directories and databases required for the tutorial.
- ◆ Add a table to the consolidated database.

❖ To create the databases and directories for the tutorial:

- 1 Create a directory to hold the files you make during this tutorial; for example *c:\tutorial*.

```
mkdir c:\tutorial
```

- 2 The tutorial uses two databases: a consolidated database named *hq.db* and a remote database named *field.db*. Change to the tutorial directory and create these databases using the following statements at a command line:

```
dbinit hq.db  
dbinit field.db
```

The database initialization tool for Windows 3.x is *dbinitw* rather than *dbinit*. Under Windows 3.x, you should execute the commands from the Program Manager File ► Run menu or you could make an icon for each command.

- 3 Create a subdirectory for each of the two user IDs in the replication system. Create these subdirectories using the following statements at a command line:

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

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

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

- 1 Connect to *hq.db* from Interactive SQL as user ID **DBA**, using password **SQL**.
 - ◆ Type **CONNECT** in the Interactive SQL Command window, and click **Execute**.
 - ◆ Enter the user ID **DBA** and the password **SQL**, and click **OK**.
 - ◆ If the **hq** database is already running, type **hq** in the Database Name box. If it is not running, enter the Database File *c:\tutorial\hq.db*. Then click **OK** to connect.

- 2 Enter the following **CREATE TABLE** statement to create the **SalesRep** table:

```
CREATE TABLE SalesRep (
    rep_key CHAR(12) NOT NULL,
    name CHAR(40) NOT NULL,
    PRIMARY KEY ( rep_key )
);
```

- 3 Enter the following **CREATE TABLE** statement to create the **Customer** table:

```
CREATE TABLE Customer (
    cust_key CHAR(12) NOT NULL,
    name CHAR(40) NOT NULL,
    rep_key CHAR(12) NOT NULL,
    FOREIGN KEY REFERENCES SalesRep,
    PRIMARY KEY ( cust_key )
);
```

You are now ready for the rest of the tutorial.

Set up the consolidated database

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

You require DBA authority to carry out this task.

Create 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.

❖ To create the message type:

- ◆ Create the file message type using the following statement:

```
CREATE REMOTE MESSAGE  
TYPE file  
ADDRESS 'hq'
```

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.

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

Grant PUBLISH and REMOTE at the consolidated database

In the hierarchical replication system supported by SQL Remote, each database may have one consolidated database immediately above it in the hierarchy and many databases immediately below it on the hierarchy (remote databases).

PUBLISH permission identifies the current database for outgoing messages, and the REMOTE permission identifies each database receiving messages from the current database.

GRANT PUBLISH
to identify outgoing
messages

Permissions can only be granted by a user with DBA authority. To carry out these examples you should connect using the Interactive SQL utility to **hq** as user ID **DBA**, with password **SQL**.

Each database that distributes its changes to other databases in the replication system is a publisher database. Each database in the replication system that publishes changes to a database is identified by a single user ID. You set that ID for your database using the GRANT PUBLISH statement. This section describes setting permissions for the consolidated database (*hq.db*).

❖ **To create a publisher for the database:**

- ◆ Connect to the database using Interactive SQL, and type the following statement:

```
GRANT CONNECT
TO hq_user
IDENTIFIED BY hq_pwd ;

GRANT PUBLISH TO hq_user ;
```

You can check the publishing user ID of a database at any time using the CURRENT PUBLISHER special constant:

```
SELECT CURRENT PUBLISHER
```

GRANT REMOTE
for each database
to which you send
messages

Each remote database is identified using the GRANT REMOTE statement. 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 this would require no extra action on the part of the DBA.

The GRANT REMOTE statement identifies the message system to be used when sending messages to the recipient, as well as the address.

❖ **To add a remote user:**

- ◆ Connect to the database using Interactive SQL, and enter the following statements:

```
GRANT CONNECT TO field_user
IDENTIFIED BY field_pwd ;

GRANT REMOTE TO field_user
TYPE file ADDRESS 'field' ;
```

The address string is the directory used to hold messages for **field_user**, enclosed in single quotes. 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.

Create publications and subscriptions

A publication is created using a CREATE PUBLICATION statement. This is a data definition language statement, and requires DBA authority. For the tutorial, you should connect to the **hq** database as user ID **DBA**, password **SQL**, to create a publication.

Set up a publication at the consolidated database

Create a publication named **SalesRepData**, which replicates all rows of the table **SalesRep**, and some of the rows of the table **Customer**.

❖ To create the publication:

- ◆ Connect to the database from Interactive SQL, and enter the following statement:

```
CREATE PUBLICATION SalesRepData (
    TABLE SalesRep,
    TABLE Customer SUBSCRIBE BY rep_key
)
```

Set up a subscription

Each user ID that is to receive changes to the publication must have a subscription. The subscription can only be created for a user who has REMOTE permissions. The GRANT REMOTE statement contains the address to use when sending the messages.

❖ To create the subscription:

- ◆ Connect to the database from Interactive SQL, and enter the following statement:

```
CREATE SUBSCRIPTION
TO SalesRepData ('rep1')
FOR field_user ;
```

The value **rep1** is the **rep_key** value we will give to the user **field_user** in the **SalesRep** table.

The full CREATE SUBSCRIPTION statement allows control over the data in subscriptions; allowing users to receive only some of the rows in the publication. For more information, see "CREATE SUBSCRIPTION statement" on page 360.

The CREATE SUBSCRIPTION statement identifies the subscriber and defines what they receive. However, it does not synchronize data, or start the sending of messages.

Set up the remote database

The remote database needs to be configured in order to send and receive messages and participate in a SQL Remote setup. Like the consolidated database, the remote database needs a CURRENT PUBLISHER to identify the source of outgoing messages, and it needs to have the consolidated database identified as a subscriber. The remote database also needs the publication to be created and needs a subscription created for the consolidated database. 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.

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

Extract the remote database information

Leave the **hq** database running, and change to the tutorial directory.

Type the following command at the system command line (all on one line) to extract a database for the user **field_user** from the consolidated database:

```
dbextract -v -c "dbn=hq;uid=dba;pwd=sql" c:\tutorial
field_user
```

In Windows 3.x, you should run the following command either from File►Run in Program Manager or from an icon:

```
dbxtracw -v -c "dbn=hq;uid=dba;pwd=sql" c:\tutorial
field_user
```

The `-v` command-line switch produces more verbose output. This is useful during development.

This command assumes the **hq** database is currently running on the default server. If the database is not running, you should enter a database file parameter in the connection string:

```
dbf=hq.db
```

instead of the **dbn** database name parameter.

☞ For details of the *dbextract* utility and its options, see "The extraction command-line utility" on page 315.

The *dbextract* command creates a SQL command file named *reload.sql* in the current directory and a data file in the *c:\tutorial* directory. It also starts the subscriptions to the remote user.

The next step is to load these files into the remote database.

Load the remote database information

❖ To load the database information:

- 1 From the tutorial directory, connect to the remote database *field.db* from Interactive SQL as user ID **DBA** and password **SQL**.

You can do this by typing `CONNECT` in the Interactive SQL command window, and then entering `DBA`, `SQL`, and the database file `c:\tutorial\field.db` in the connection dialog box.

- 2 Once connected, run the *reload.sql* command file:

```
READ C:\tutorial\reload.sql
```

The *reload.sql* command file carries out the following tasks:


- ◆ Creates a message type at the remote database.
- ◆ Grants `PUBLISH` and `REMOTE` permissions to the remote and consolidated database, respectively.
- ◆ Creates the table in the database. If the table had contained any data before extraction, the command file would fill the replicated table with a copy of the data.
- ◆ Creates a publication to identify the data being replicated.
- ◆ Creates the subscription for the consolidated database, and starts the subscription.

While connected to the **field** database as `DBA`, confirm that the tables are created by typing

```
SELECT * FROM SalesRep ;  
  
SELECT * FROM Customer ;
```

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 59 .

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

First, enter some data into the consolidated database.

❖ **To enter data at the consolidated database:**

- 1 Connect to the consolidated database **hq** from the Interactive SQL utility as user ID **DBA**, with password **SQL**.

- 2 Enter and commit two rows into the **SalesRep** table:

```
INSERT INTO SalesRep (rep_key, name)
VALUES ('rep1', 'Field User') ;
INSERT INTO SalesRep (rep_key, name)
VALUES ('rep2', 'Another User') ;
COMMIT ;
```

- 3 Enter and commit two rows into the **Customer** table:

```
INSERT INTO Customer (cust_key, name, rep_key)
VALUES ('cust1', 'Ocean Sports', 'rep1' ) ;
INSERT INTO Customer (cust_key, name, rep_key)
VALUES ('cust2', 'Sports Plus', 'rep2' ) ;
COMMIT ;
```

- 4 Confirm that the data is entered by viewing the data in the tables:

```
SELECT *
FROM SalesRep;
```

```
SELECT *
FROM Customer;
```

The next step is to send the relevant rows to the remote 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 *dbremote* program is the Message Agent for Adaptive Server Anywhere.

❖ **To send the data to the remote database:**

- 1 From a command prompt, change to your tutorial directory. For example,

```
> c:
> cd c:\tutorial
```

- 2 Enter the following statement at the command line to run the Message Agent against the consolidated database:

```
dbremote -c "dbn=hq;uid=dba;pwd=sql"
```

In Windows 3.x, you should run the following command either from File►Run in Program Manager or from an icon:

```
dbremotw -c "dbn=hq;uid=dba;pwd=sql"
```

with the tutorial directory as the working directory. These command lines assume that the **hq** database is currently running on the default server.

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

- 3 Click Shutdown on the Message Agent window to stop the Message Agent when the messages have been sent. The Message Agent window displays the message Execution completed when all processing is complete.

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 data at the remote database:**

- 1 From a command prompt, change to your tutorial directory. For example,

```
> c:
> cd c:\tutorial
```

- 2 Enter the following statement at the command line to run the Message Agent against the **field** database:

```
dbremote -c "dbn=field;uid=dba;pwd=sql"
```

In Windows 3.x, you should run the following command either from File►Run in Program Manager or from an icon:

```
dbremotw -c "dbn=field;uid=dba;pwd=sql"
```


These command lines assume that the **field** database is currently running on the default server.

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

- 3 Click Shutdown on the Message Agent window to stop the Message Agent when the messages have been processed. The Message Agent window displays the message Execution completed when all processing is complete.

The Message Agent window displays status information while running. This information can be output to a log file for record keeping in a real setup. You will see that 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 **SalesRep** table by typing the following statement:

```
SELECT * FROM Customer
```

You will also 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 "dbn=field;uid=dba;pwd=sql"
```

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

5 With the *hq.db* database running, run *dbremote* to receive the message at the consolidated database:

```
dbremote -c "dbn=hq;uid=dba;pwd=sql"
```

6 Connect to the consolidated database and display the **Customer** table:

```
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.

A sample publication

The command file *salespub.sql* contains a set of statements that creates a publication on the sample database. This publication illustrates several of the points of the tutorial, in more detail.

❖ **To add the publication to the sample database:**

- 1 Connect to the sample database from Interactive SQL.
- 2 Type **READ *path*\scripts\salespub.SQL**, where *path* is your Adaptive Server Anywhere installation directory.

The *salespub.sql* publication adds columns to some of the tables in the sample database, creates a publication and subscriptions, and also adds triggers to resolve update conflicts that may occur.

