

## CHAPTER 12

# Administering SQL Remote for Adaptive Server Enterprise

About this chapter      This chapter details set-up, and management issues for SQL Remote administrators using Adaptive Server Enterprise as the server for the consolidated database.

### Contents

<b>Topic</b>	<b>Page</b>
How the Message Agent for Adaptive Server Enterprise works	278
Running the Message Agent	283
Error reporting and handling	285
Adaptive Server Enterprise transaction log and backup management	286
Making schema changes	289
Using passthrough mode	290

## How the Message Agent for Adaptive Server Enterprise works

This section describes how the Message Agent for Adaptive Server Enterprise works. There are some significant differences between how the Message Agent for Adaptive Server Enterprise and the Message Agent for Adaptive Server Anywhere operate, which accommodate the different roles of the two servers.

🔗 For information on features of the Message Agent that are common to Adaptive Server Anywhere and Adaptive Server Enterprise, see "Running the Message Agent" on page 239.

Message Agent is  
ssremote

The Message Agent for Adaptive Server Enterprise is the following executable:

- ◆ On Windows operating systems, the Message Agent is *ssremote.exe*
- ◆ On UNIX operating systems, the Message Agent is *ssremote*.

### Scanning the transaction log

The Message Agent scans the Adaptive Server Enterprise transaction log in order to collect transactions to be sent to remote databases. It stores these transactions in a **stable queue**.

🔗 For more information about the stable queue, see "The stable queue" on page 279. For more information about how the Message Agent uses the stable queue, see "Message Agent operation phases" on page 280.

The SQL Remote Message Agent uses the same transaction log scanning interface as the Adaptive Server Enterprise Log Transfer manager (LTM). Adaptive Server Enterprise maintains a **truncation point**, which is an identifier for the oldest page in the transaction log needed by the replication system.

The SQL Remote Message Agent sets the truncation point as soon as transactions are scanned from the transaction log and committed in the stable queue. This allows the **dump transaction** command to reclaim space in the transaction log as soon as possible. The Message Agent does not wait until confirmation is received from remote databases before setting the truncation point.

**Message Agent must be run to reclaim log space**


The Message Agent must be run frequently enough to prevent the transaction log from running out of space. The **dump transaction** command does not reclaim space from pages after the truncation point.

Replication Server  
and SQL Remote

Using SQL Remote on an Adaptive Server Enterprise database participating in a Replication Server setup may involve other considerations. If your database has a replication agent (LTM) running against it, then you need to use the SQL Remote Open Server as an additional component. Adaptive Server Enterprise databases have replication agents running against them in the following circumstances:

- ◆ The database is participating in a Replication Server setup as a primary database, or
- ◆ The database is participating in a Replication Server setup and is using asynchronous procedure calls.

If the database is participating in a Replication Server setup as a replicate site, and no asynchronous procedure calls are being used, there is no need for the SQL Remote Open Server.

 For more information about the SQL Remote Open Server, see "Using SQL Remote with Replication Server" on page 291.

## The stable queue

The Message Agent for Adaptive Server Enterprise uses a **stable queue** to hold transactions until they can be deleted. A stable queue is a pair of database tables that hold messages that may still be needed by the replication system.

SQL Remote for Adaptive Server Anywhere does not use a stable queue.

**Stable queue not identical to Replication Server stable queue**

Sybase Replication Server also uses stable queues as storage areas for replication messages. The Replication Server and SQL Remote stable queues perform similar functions, but are *not* the same thing.

Stable queue  
location

The stable queue may be kept in the same database as the tables being replicated, or in a different database. Keeping the stable queue in a separate database complicates the backup and recovery plan, but can improve performance by putting the stable queue workload on separate devices and/or a separate Adaptive Server Enterprise server.

**Do not modify the stable queue directly**

The stable queue is maintained by and for the Message Agent. You should not modify the stable queue directly.

**Stable queue tables**

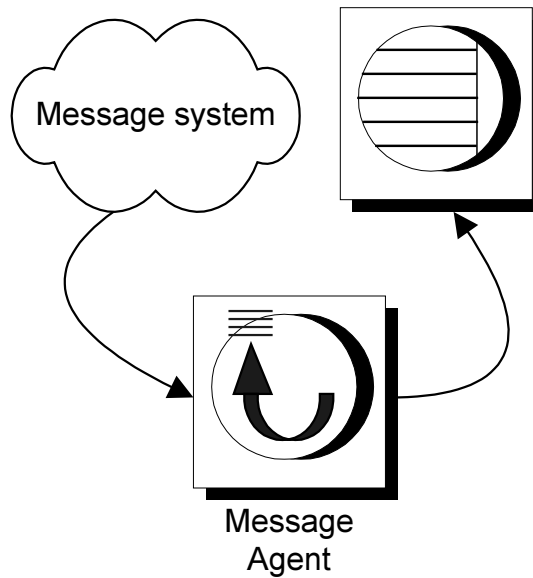
The stable queue consists of two tables. The **sr\_transaction** table has one row for each transaction in the stable queue, and the supplementary table **sr\_queue\_state** has a single row, which stores persistent global information about the state of the stable queue.

🔗 For a description of each of the columns of these tables, see "Stable Queue tables" on page 347.

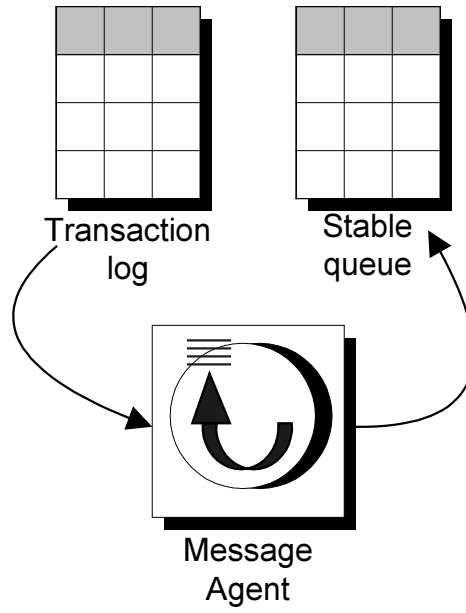
## Message Agent operation phases

The Message Agent has the following phases of execution:

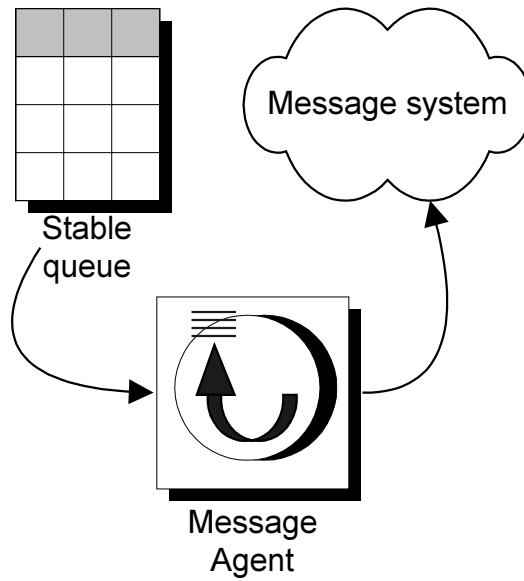
- ◆ **Receiving messages** During this phase, the Message Agent receives incoming messages and applies them to the Adaptive Server Enterprise server.



- ◆ **Populating the stable queue** During this phase the Message Agent scans the Adaptive Server Enterprise transaction log into the stable queue.




- ◆ **Sending messages** During this phase, the Message Agent builds outgoing messages from the stable queue.




The transactions remain in the stable queue until confirmation has been received from all remote databases. When confirmation is received, the transactions are automatically removed from the stable queue by the Message Agent.

The Message Agent does not scan the transaction log for the stable queue.

 For information on running multiple copies of the Message Agent to carry out these tasks, see "Running multiple Message Agents" on page 283.

## Running the Message Agent

 This section describes how to run the Message Agent for Adaptive Server Enterprise. For information on features of the Message Agent that are common to Adaptive Server Anywhere and Adaptive Server Enterprise, see "Running the Message Agent" on page 239.

### The Message Agent and replication security

In the tutorials earlier in this book, the Message Agent was run using a user ID with system administrator permissions. The operations in the messages are carried out from the user ID specified in the Message Agent connection string; by using a system administrator user ID, you can be sure that the user has permissions to make all the changes.

In practice, you will not use such a user ID, but the Message Agent needs to run using a user ID with replication role. You can grant replication role with the following statement:

```
sp_role 'grant', replication_role, user_name
```

### Running multiple Message Agents

The three phases of Message Agent operation are described in the section "Message Agent operation phases" on page 280. To summarize, these phases are:

- ◆ Receiving messages.
- ◆ Scanning the transaction log.
- ◆ Sending messages.

You may wish to run separate copies of the Message Agent to carry out these different phases. You can specify which phases a given Message Agent is to execute on the Message Agent command line.

Specifying which phases to execute

The command-line options are as follows:

- ◆ **Receive** The `-r` command-line switch instructs the Message Agent to receive messages while it is running. To cause the Message Agent to shut down after receiving available messages, use the `-b` switch in addition to `-r`.
- ◆ **Scan log** The `-i` command-line switch instructs the Message Agent to scan the transaction log into the stable queue while it is running.

- ◆ **Send** The `-s` command-line switch instructs the Message Agent to send messages while it is running.
- ◆ **Multiple phases** If none of `-r`, `-i`, or `-s` is specified, the Message Agent executes all three phases. Otherwise, only the indicated phases are executed.

There are several circumstances where you may wish to run multiple Message Agents.

- ◆ **Ensuring the transaction log does not run out of space** It is important that the transaction log not be allowed to become full. For this reason, you must scan the transaction log frequently enough to ensure that all entries required by SQL Remote are placed in the stable queue. Therefore, you may want to run a Message Agent that scans the transaction log continuously, even if you are only receiving and sending messages in batch mode.
- ◆ **Mixing operating systems** If you wish to use a message link supported under Windows 95 or NT, you must use a Message Agent on that platform to send and receive messages. You can do this, while running the log scanning on a UNIX machine, by running two copies of the Message Agent.

How Message Agents are synchronized

The operations of two or more Message Agents are synchronized by a table called **sr\_marker**. This table has a single column called **marker**, of data type **datetime**.

When the Message Agent wants to wait for transactions to be scanned into the stable queue, it updates **sr\_marker** and waits for it to work its way through the system. The column in **sr\_queue\_state** is also called **marker**, and contains the most recent marker to be scanned from the transaction log.



## Error reporting and handling

This section describes how errors are reported and handled by the Message Agent.

### Default error handling

The default action taken by the Message Agent when an error occurs is to record the fact in its log output. The Message Agent sends log output to a window or a log file recording its operation. By default, log output is sent to the window only; the `-o` command-line option sends output to a log file as well.

The Message Agent log includes the following:

- ◆ Listing of messages applied.
- ◆ Listing of failed SQL statements.
- ◆ Listing of other errors.

UPDATE conflicts  
are not errors

UPDATE conflicts are not errors, and so are not reported in the Message Agent output.


### Implementing error handling procedures

SQL Remote allows you to carry out some other process in addition to logging a message if an error occurs. The `Replication_error` database option allows you to specify a stored procedure to be executed by the Message Agent when an error occurs. By default no procedure is executed .

The procedure must have a single argument of type CHAR, VARCHAR, or LONG VARCHAR. The procedure is called twice: once with the error message and once with the SQL statement that causes the error.

While the option allows you to track and monitor errors in replication, you must still design them out of your setup: this option is not intended to resolve such errors.

For example, the procedure could insert the errors into a table with the current time and remote user ID, and this information can then replicate back to the consolidated database. An application at the consolidated database can create a report or send e-mail to an administrator when errors show up.

 For information on setting the `REPLICATION_ERROR` option, see "SQL Remote options" on page 323.

# Adaptive Server Enterprise transaction log and backup management

You must protect against losing transactions that have been replicated to remote databases. If transactions are lost that have already been replicated to remote databases, the remote databases will be inconsistent with the consolidated database. In this situation, you may have to re-extract all remote databases.

## Protecting against media failure on the transaction log

Media failure on the transaction log can cause committed transactions to be lost. If the transaction log has been scanned and these transactions have already been sent to subscriber databases, then the subscribing databases contain transactions that are lost from the publishing database, and the databases are in an inconsistent state.

Why the transaction log is needed

The transaction log is needed, even after the entries have been scanned into the stable queue, to guard against media failure on the database file. If the database is lost, it must be recovered to a point that includes every transaction that may have been sent to remote databases.

This recovery is done by restoring a database dump and loading transaction dumps to bring the database up to date. The last transaction dump restored is the dump of the active transaction log at the time of the failure.

Protecting against transaction log loss

There are two ways of protecting against inconsistency arising from media failure on the transaction log:

- ◆ **Mirror the transaction log** When a device is mirrored, all writes to the device are copied to a separate physical device.
- ◆ **Only replicate backed-up transactions** There is a command-line switch for the Message Agent that prevents it from sending transactions until they are backed up.

Mirroring the transaction log

The only way to protect against media failure on the transaction log is by mirroring the transaction log.

Disk mirroring can provide nonstop recovery in the event of media failure. The **disk mirror** command causes an Adaptive Server Enterprise database device to be duplicated—that is, all writes to the device are copied to a separate physical device. If one of the devices fails, the other contains an up-to-date copy of all transactions.

For information on disk mirroring in Adaptive Server Enterprise, see the chapter "Mirroring Database Devices", in the Adaptive Server Enterprise *System Administration Guide*.

Replicating only backed-up transactions

The Message Agent also provides a command line switch (`-u`) that only sends transactions that have been backup up. In Adaptive Server Enterprise, this means transactions complete before the latest **dump database** command or **dump transaction** command.

Choosing an approach

The goal of the strategy is to reduce the possibility of requiring re-extraction of remote databases to an acceptable level. In large setups, the possibility must be as close to zero as possible, as the cost of re-extraction (in terms of down time) is very high.

- ◆ The Message Agent `-u` command-line switch can be used instead of transaction log mirroring when recovery of all transactions in a consolidated database is not needed and mirroring is considered too expensive. This may be true in small setups or setups where there are no local users on the consolidated database.
- ◆ The Message Agent `-u` command-line switch can also be used in addition to mirroring to provide additional protection against total site failure or double media failure.

## Stable queue recovery issues

Keeping the stable queue in a separate database complicates backup and recovery, as consistent versions of the two databases have to be recovered.

Normal recovery automatically restores the two databases to a consistent state, but recovery from media failure takes some care. When restoring database dumps and transaction dumps, it is important to recover the stable queue to a consistent point.

Two procedures in the stable store database are provided to help with recovery from media failure:

- ◆ **sp\_queue\_dump\_database** This procedure is called whenever a dump database is scanned from the transaction log.
- ◆ **sp\_queue\_dump\_transaction** This procedure is called whenever a dump transaction is scanned from the transaction log.

You can modify these stored procedures to issue **dump database** and **dump transaction** commands in the stable store database.

## Transaction log management

The Adaptive Server Enterprise **log transfer** interface allows the Message Agent to scan the Adaptive Server Enterprise transaction log. When this interface is being used, it sets a **truncation point** in the transaction log. The truncation point prevents Adaptive Server Enterprise from re-using pages in the transaction log before they have been scanned by SSREMOTE. For this reason, DUMP TRANSACTION will not necessarily release transaction log pages that are before the oldest open transaction. DUMP TRANSACTION will not release transaction log pages beyond the "truncation point".

### Initializing the truncation point

The SQL Remote setup script (*ssremote.sql*) initializes the truncation point with the following command

```
dbcc settrunc( 'ltm', 'valid' ).
```

The truncation point can be reset with the following command

```
dbcc settrunc( 'ltm', 'ignore' ).
```

This command tells Adaptive Server Enterprise to ignore the truncation point, allowing transaction log pages beyond the truncation point to be released for reuse. You should only use this command when you are no longer interested in SQL Remote replication with the database and you want to be able to reclaim space in the transaction device with DUMP TRANSACTION commands. Continuing to run SQL Remote after ignoring the truncation point will fail to replicate any transactions that were in transaction log pages that were not scanned by the Message Agent and were freed by DUMP TRANSACTION.

## Making schema changes

Schema changes to tables being replicated by SQL Remote must be made on a **quiet** system. A quiet system means the following:

- ◆ **No transactions being replicated** There can be no transactions being replicated that modify the tables that are to be altered. All transactions that modify tables being altered must be scanned from the transaction log into the stable queue before the schema is altered. This is performed by running the Message Agent normally, or using the `-i -b` switches. After the Message Agent completes, you can make the schema change..
- ◆ **Message Agent** The Message Agent must be shut down when the schema change is being made.
- ◆ **SQL Remote Open Server** If you are using the SQL Remote Open Server, it must be shut down when the schema change is being made.

Schema changes include changes to publications, such as adding articles or modifying articles. However, creating or dropping subscriptions, and adding or removing remote users do not need to be done on a quiet system.

In the Adaptive Server Enterprise transaction log, there is no information recording table structure changes: the SQL Remote log scanning process gets the table structure from the Adaptive Server Enterprise system tables. Consequently, the Message Agent cannot scan an operation from the transaction log that happened against the old table structure.

Information stored in the stable queue before the schema change uses the old table definitions and information stored after the schema change uses the new table definitions.

Passthrough mode can be used at the same time as the schema change to make sure that schema changes at remote databases occur in the correct sequence.

## Using passthrough mode

	<p>The publisher of the consolidated database can directly intervene at remote sites using a passthrough mode, which enables standard SQL statements to be passed through to a remote site.</p>
Determining recipients of passthrough statements	<p>Passthrough destinations are determined by <b>sp_passthrough_user</b> and <b>sp_passthrough_subscription</b>. Executing either of these procedures determines a set of recipients for any subsequent passthrough statements.</p> <p>Executing either <b>sp_passthrough_user</b> and <b>sp_passthrough_subscription</b> adds to the current list of recipients. The <b>sp_passthrough_stop</b> procedure resets passthrough (that is, resets the list of recipients to be empty).</p> <p>In Adaptive Server Enterprise, <b>sp_passthrough</b> never executes statements in the Adaptive Server Enterprise server.</p>
Passthrough statements	<p>To cause passthrough SQL statements to replicate, you call <b>sp_passthrough</b>.</p> <p>Due to the VARCHAR(255) limitation in Adaptive Server Enterprise, you should build a long SQL statement up in pieces. Calls to <b>sp_passthrough_piece</b> will build up a single SQL statement. Calling <b>sp_passthrough</b> with the last piece will cause the built up statement to replicate.</p>
Notes on using passthrough mode	<ul style="list-style-type: none"><li>◆ You should always test your passthrough operations on a test database with a remote database subscribed. You should never run untested passthrough scripts against a production database.</li><li>◆ You should always qualify object names with the owner name. PASSTHROUGH statements are not executed at remote databases from the same user ID. Consequently, object names without the owner name qualifier may not be resolved correctly.</li></ul>

## Schema modifications

The Adaptive Server Enterprise log transfer interface does not contain information about the number of columns and data types of the columns in a table. SSREMOTE gets this information directly from the Adaptive Server Enterprise system tables. For this reason, altering a table and then scanning operations that happened before the ALTER TABLE will lead to errors. SSREMOTE must set the "truncation point" beyond all operations on replicated tables before schema changes can be made. Operations on replicated tables need to be prevented between SSREMOTE running and the schema changes being made.