CHAPTER 20

Administering and Managing Log Shipping

Log shipping is one of four SQL Server 2005 high-availability alternatives. Other SQL Server 2005 high-availability alternatives include database mirroring, failover clustering, and peer-to-peer replication. Database mirroring and peer-to-peer replication are new technologies introduced with SQL Server 2005 and are discussed in other chapters within the book.

What's New for Log Shipping in Service Pack 2?

The `SQLLogShip.exe` application is now supported in SQL Server 2005 Server Pack 2. This command-line application performs all the different log shipping processes such as initiating the transaction log backup on the primary server, copying the transaction log backups to the staging area, and restoring the backups to the secondary server.

For additional information on how to use the `SQLLogShip.exe` application to manage log shipping, see the section "Managing Log Shipping from the Command Line" later in this chapter.

SQL Server 2005 Log Shipping Overview

Log shipping offers increased database availability and database protection by maintaining a warm standby database on another instance of SQL Server 2005. Unlike database mirroring, a log shipping failover is not done automatically because you must perform several manual steps to successfully complete a failover. Failover steps include manually redirecting clients from the primary database to the secondary database after manually bringing the secondary database online.

Note

It is important to understand that some data loss might occur when a failover is performed and the original database is no longer available. Because transaction log backups are shipped and restored to the secondary servers on a schedule, any changes made between the time of the last "shipped" transaction log and the failure are lost because these changes have not been applied to the secondary server.

If data loss is unacceptable, other SQL Server high-availability options should be considered.

In previous versions of SQL Server, log shipping was commonly used for geographical fault tolerance because geographical failover clustering was often expensive and impractical. Now, with the introduction of the new SQL Server 2005 database mirroring technology, the use of log shipping for this role has been reduced.

In SQL Server 2005, log shipping is now more commonly used in conjunction with other SQL Server high-availability options such as clustering and mirroring. Log shipping is also useful when a delay in the restoration is desirable. In this scenario, the time between the backup and restore process is purposely delayed to provide recovery from catastrophic database operations.

With log shipping, the primary database handles client activity, whereas the secondary database copies and restores transaction log backups. SQL Server log shipping is simply composed of several SQL Server Agent jobs that back up the transaction log from a database on the primary server; each transaction log backup is then copied and restored in the correct order to a database on a secondary server. The restore process on the secondary server is kept "open," allowing additional transaction log backups to be applied.

Figure 20.1 depicts the process used by log shipping.

Many database administrators find similarities between database mirroring and log shipping. However, key elements make these technologies different. For example, unlike database mirroring, the log shipping primary server does not continuously send and commit data to the secondary servers; instead, data is sent on a defined schedule. This increases the chance of data loss when a log shipping failover occurs, while mirroring can be configured for synchronous and asynchronous operation. In addition, log shipping does not provide an automatic failover or failback mechanism, whereas database mirroring can be configured to automatically fail over and fail back the database. Log shipping does provide the ability to configure multiple secondary servers, whereas database mirroring can have only a single secondary database.

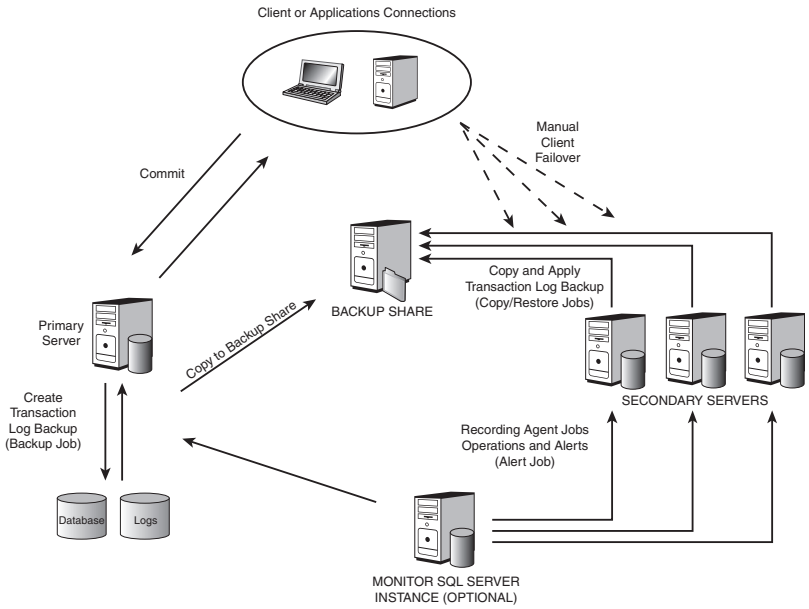

FIGURE 20.1
Overview of log shipping.

Understanding Log Shipping Terminology

To effectively administer and manage log shipping, you need to understand the terminology and components that make up this solution. To prepare you for administrative and management tasks, this section details some of the log shipping terms used throughout this chapter:

- **Primary Server**—The primary server contains the source database used in the log shipping configuration.
- **Secondary Server**—The secondary server contains the replica database. This database is a warm standby copy of the source database on the primary server.
- **Monitor Server**—This optional server records various details about the log shipping activity between the primary and secondary servers. This information includes backup operations, restore operations, and any alerts that have been generated.
- **Backup Job**—The backup job resides on the primary server; this job is used to back up the transaction logs and store them in the backup

share. The backup share can be a network location, the secondary server or local drive on the primary server.

- **Copy Job**—The copy job resides on each secondary server; this job is used to copy the transaction log backups from the backup location to the staging area. The staging area is usually a local drive on the secondary server.
- **Restore Job**—The restore job resides on each secondary server; this job is used to restore the transaction log backups that were copied to the local staging area.

Environmental Considerations That Affect Log Shipping

The following environmental characteristics should be identified when configuring and managing log shipping, along with potential issues that may arise:

- **Network Speed**—The available network bandwidth between each server affects the speed the transaction log backup can be copied from the primary server to the backup share and from the backup share to the secondary server. The time to copy the backup files cannot exceed the backup interval. If the copy time exceeds the backup interval, log shipping will become backlogged and eventually fail.
- **Server Performance**—The performance of the server needs to be considered. Both the time to back up the transaction log on the primary server and restoration of the transaction log backups on the secondary servers cannot exceed the defined backup interval. If the server cannot restore backups fast enough, log shipping will become backlogged and eventually fail.
- **Drive Capacity**—Enough storage capacity is required to store the transaction log backup made by the backup job and the transaction log copy made by the copy job. The retention period for these files can be adjusted but should provide enough history to recover from an interruption. For example, creating a backup every 15 minutes generates 96 backups each day.
- **Monitor Server Placement**—The placement of the optional monitor server is important because this server is responsible for recording backups, restores, and alerts in a central location.

When you're considering these factors, it is important to combine the total end-to-end transition. For example, if the transaction log backup is executed

every 15 minutes and takes 10 minutes to complete, and it takes 15 minutes to copy and restore the backup to the secondary server, the next backup job will start before the first iteration of the log shipping process can complete. In this scenario, a backlog of transaction log backups will accumulate in the backup share.

When a backlog occurs, the synchronization gap between the primary and secondary database will continue to grow. The transaction log backups are removed based on a predefined schedule. When the synchronization gap between the primary and secondary servers grows beyond the transaction log retention period, the backups are removed before the secondary database has a chance to copy and restore them, ultimately causing the log shipping process to fail.

When SQL Server 2005 Log Shipping Is Desirable

Some of the key driving factors for implementing log shipping with SQL Server 2005 include

- A controlled delay between the primary database and secondary databases is desirable.
- Automatic client failover is not necessary.
- An additional replication of an existing database mirror is necessary.
- The data must be replicated to multiple secondary servers.
- A need exists to host a replica database for reporting purposes.

The first thing organizations should do when considering a high-availability solution is to identify the gaps between the current and desired states of their business and then determine if log shipping fulfills their high-availability business goals or another high-availability solution is more appropriate.

Log Shipping Design Considerations

The following describes common design considerations that should be explored before configuring log shipping:

- **Security**—When you want to configure log shipping, the sysadmin role is required on each SQL Server that will take part in the implementation. This includes the primary server, each secondary server, and the monitor server. The agent account used to run the backup job on the primary server must be able to read and write to the backup location. The agent account on the secondary server used to copy the

backups to the staging area must have read access to the backup share and read/write access to the staging area.

- **SQL Server Version**—SQL Server 2005 Standard, Workgroup, and Enterprise Editions can be used for log shipping. All servers must be configured with the same case-sensitivity settings.
- **Recovery Mode**—The recovery mode of the source database on the primary server must be configured as full or bulk-logged. The simple recovery model commits data to the database and then removes transaction log files; the Full and Bulk-logged recovery models truncate the transaction log until after a successful transaction log backup. Because the transaction log is a key part of log shipping, the simple recovery model cannot be used.
- **Backup Storage Location**—The backup storage location is used to store the transaction log backups created by the backup job. In a large-scale environment, it is highly recommended to host the transaction log backups on a fault-tolerant server independent of the log shipping primary or secondary servers. However, in a small environment or test environment, it may be acceptable to store the data on local storage attached to the primary server. The account used to execute the backup job must have read/write access to this location.
- **Staging Storage Location**—The staging storage location is used to store the transaction log backups copied from the backup location. It is common to store these backups on local storage on the secondary server if enough drives are available. If adequate drives are unavailable, the staging area can be hosted on a different server. The account used to execute the copy job on each secondary server must have read/write access to this location and read access to the backup storage location.
- **Monitor Server**—The monitor server is optional and is normally hosted on a SQL Server that is not part of the log shipping implementation. This server is important because it provides a central location for the log shipping status and alert messages. In the event of a failure, the monitor server can be used to determine when the last backup occurred. If a monitor server is not specified, each server in the log shipping implementation monitors itself.

Combining Log Shipping with Other SQL Technologies

In most cases, other SQL Server high-availability alternatives and technologies can be combined with log shipping. Log shipping often complements

failover clustering or database mirroring because these technologies provide fast and automatic failover.

Log shipping complements these technologies by providing an additional layer of availability. For example, an organization may implement database mirroring between two offices to support hot failover of critical business data. Log shipping is often implemented in this scenario to make an additional replica of this data to a remote disaster recovery site.

The following sections describe how log shipping interacts with other SQL Server 2005 technologies.

Log Shipping and Failover Clustering

Log shipping can be combined with failover clustering to achieve maximum availability, business continuity, and disaster recovery. For example, CompanyABC may have a failover cluster supporting business data in the primary locations. In this case, database mirroring may be used to create a hot standby of the database in another office, while log shipping is used to create a third copy in a remote disaster recovery site.

Log Shipping and Database Mirroring

One of the limitations of database mirroring compared to log shipping is that database mirroring can have only one mirrored server associated with each principal, whereas log shipping can have multiple standby servers. The two technologies can be combined if there is a need to ship the principal database logs to a remote location other than where the mirror resides. In addition, log shipping databases can be used for reporting, whereas mirror databases cannot unless a snapshot is used.

> **Note**
>
> Log shipping needs to be reinitialized on the mirror SQL Server instance in the event of a failure and role change.
>
> To ensure log shipping is continuous after a mirror has failed over, simply create the primary/secondary relationship twice—once on the primary database and once on the mirror database. Because only one database is active at a time, the log shipping jobs never overlap.

Log Shipping and Replication

Log shipping can also be combined with database replication if necessary. However, replication must be manually reconfigured after a log shipping

failover. Alternatively, if the failover is permanent, the secondary server can be renamed to allow replication to continue.

> **Note**
> Due to the requirements and manual work, database mirroring is often a better choice when high availability for replicated databases is necessary.

Administering Log Shipping

The following sections provide step-by-step tasks used to administer log shipping. These tasks include several prerequisites that you should understand because they affect the success of the log shipping solution. The examples provided throughout these sections simulate a log shipping implementation between two servers while using a third server to act as the monitor.

The AdventureWorks database on INSTANCE01 on the primary server SQL01 will be shipped to INSTANCE01 on the secondary server SQL03. The SQL Agent service on each server is running under the SQL.Service domain credentials. Table 20.1 shows the log shipping configuration.

Table 20.1 **Server Names and Role**

Role	SQL Server Instance	Location
Primary Server	SQL01\INSTANCE01	San Francisco
Secondary Server	SQL03\INSTANCE01	New York
Monitor Server	SQL02\INSTANCE01	New York

Configuring the Log Shipping Storage Locations

Follow these steps to configure the backup storage location. In this example, the transaction log backups will be stored on a local drive on the primary server. The SQL Agent service account or proxy account on the primary server must be granted access to the folders created in this procedure.

1. On the primary server, navigate to the D:\ drive.
2. Create a folder called **`LSBackup`**.
3. Create a folder called **`AdventureWorks`** in LSBackup.
4. Right-click the AdventureWorks folder and select "Sharing and Security."

5. Select Share This Folder and accept the default name of AdventureWorks.
6. Click the Permissions button.
7. Select the Everyone group and click Remove.
8. Click the Add button, type **SQL.Service**, and then click OK.
9. Enable Change permissions, and then click OK.
10. Select the Security tab.
11. Click the Add button, type **SQL.Service**, and then click OK.
12. Enable Modify permissions and then click OK to close the properties window.

Follow these steps to configure the staging storage location. In this example, the backups will be stored on a local drive on the secondary server. The SQL Agent service account or proxy account on the secondary server must be granted access to the folders created in this procedure.

1. On the secondary server, navigate to the D:\ drive.
2. Create a folder called **LSCopy**.
3. Create a folder called **AdventureWorks** in LSCopy.
4. Right-click the AdventureWorks folder and select Properties.
5. Select the Security tab.
6. Click the Add button, type **SQL.Service**, and then click OK.
7. Enable Modify permissions and then click OK to close the properties window.

Configuring the Database Recovery Model

SQL Server databases can support different types of recovery models. To support log shipping, the database recovery model must be set to full or bulk-logged. The Full and Bulk-logged recovery models do not automatically truncate the transaction log; the transaction log is truncated only after a successful transaction log backup is performed.

Follow these steps to configure the AdventureWorks database to use the Full recovery model:

1. From the test server (SQL01), choose Start, All Programs, Microsoft SQL Server 2005, SQL Server Management Studio.
2. Select Database Engine from the Server Type drop-down; then enter the server and instance name (**SQL01\INSTANCE01**).

3. Select Windows Authentication from the Authentication drop-down menu and then click the Connect button.
4. A connection to the database engine is made. If the Object Explorer pane is not visible, press the F8 button.
5. From within the Object Explorer pane, expand Databases. The AdventureWorks database should be listed.
6. Right-click the AdventureWorks database and select Properties. The Database Properties window opens.
7. Select the Options page. Select Full from the Recovery model drop-down menu and then click OK.

The AdventureWorks database now uses the Full recovery model. The following code can also be used to change the recovery model of the AdventureWorks database:

```
ALTER DATABASE AdventureWorks SET RECOVERY FULL
```

Note

On a heavily used database, the transaction log can grow very quickly and can consume a large amount of space. It is important to allocate enough space to store the transaction log, transaction log backup, and transaction log backup copies.

For additional information on database recovery models, see Chapter 17, "Backing Up and Restoring the SQL Server 2005 Environment" (online).

Implementing Log Shipping

After the prerequisites have been configured and the recovery model has been selected, the log shipping configuration can begin. Follow these steps to start the configuration of log shipping on the AdventureWorks database:

1. From the primary server (SQL01), choose Start, All Programs, Microsoft SQL Server 2005, SQL Server Management Studio.
2. Select Database Engine from the Server Type drop-down; then enter the server and instance name (**`SQL01\INSTANCE01`**).
3. Select Windows Authentication from the Authentication drop-down menu and then click the Connect button.

4. A connection to the database engine is made. If the Object Explorer pane is not visible, press the F8 button.
5. From within the Object Explorer pane, expand Databases; the AdventureWorks database should be listed.
6. Right-click the AdventureWorks database and select Properties. The Database Properties window opens.
7. Select the Transaction Log Shipping page. Select the Enable This as a Primary Database in a Log Shipping Configuration option.

Configuring the Transaction Log Backup Settings

The next step in the process is to configure the transaction log backup settings. The following options are available for the log shipping backup:

- **Network Path to Backup Folder**—This option is required and defines a network path to store the transaction log backups.
- **Local Path to Backup Folder**—This option defines the local path to store the transaction log backups and needs to be configured only if the transaction log backups will be stored locally.
- **Backup Retention**—The Delete Files Older Than option defines how long the transaction log backups are kept before being deleted. It is important to store enough backups to recover from small interruptions in the log shipping process. For additional information on recovering, see the section "Recovering from Log Shipping Interruptions" later in this chapter.
- **Alert Threshold**—The Alert If No Backup Occurs Within option defines the threshold that must be reached before an alert is generated when a successful backup does not occur.
- **Backup Job Schedule**—The backup schedule defines the reoccurring pattern for the transaction log backup.

> **Note**
>
> It is important to ensure the transaction log for the database is not backed up by any other maintenance plans.
>
> Normally, when the transaction log is successfully backed up, it is truncated. If a different maintenance plan truncates the transaction log before being shipped to the secondary server, the log shipping process will fail.

Follow these steps to continue the configuration of log shipping by establishing the log shipping backup options:

1. From within the Transaction Log Shipping page of the database properties, select the Backup Settings button.
2. The Backup Setting window opens, allowing configuration of the log shipping backup.
3. Enter **`\\SQL01\AdventureWorks`** in the Network Path to Backup Folder field.
4. Enter **`D:\LSBackup\AdventureWorks`** in the local path field.
5. Keep the default 72 hours for the Delete Files Older Than option.
6. Keep the default 1 hour for the Alert If No Backup Occurs Within option.

Figure 20.2 shows how the Transaction Log Backup Settings window should look after being configured using the preceding steps.

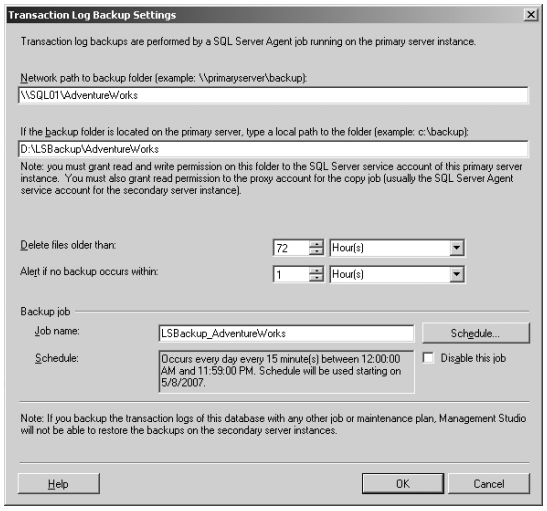

FIGURE 20.2
Transaction Log Backup Settings window.

At this point, you can configure the schedule for the log shipping backup by clicking the Schedule button. See the section "Managing Log Shipping Jobs" later in this chapter for additional details on configuring the transaction log backup schedule.

For the purposes of this exercise, accept the default schedule. The default schedule will run the backup job at 15-minute intervals throughout the day. Click OK to return to the Transaction Log Shipping page.

Adding Secondary Database

The sysadmin role is required on the server being added as the log shipping secondary. Follow these steps to add SQL03\INSTANCE01 as the secondary server:

1. From within the Transaction Log Shipping page of the database properties, click the Add button.
2. When the Secondary Database Setting window opens, click the Connect button.
3. Enter the server and instance name (**`SQL03\INSTANCE01`**).
4. Select Windows Authentication from the Authentication drop-down menu and then click the Connect button.
5. The secondary database field is prepopulated to match the name of the source database. This name is acceptable in most instances but can be changed if necessary.

Before the backups can be shipped to the secondary server, the secondary server must be initialized. The initialization process simply restores a full copy of the source database to the target server. This restoration process leaves the database in a loading state ready to restore additional transaction log backups.

Initializing the Secondary Database

On the Initialization tab, the following options used to prepare the database on the secondary server are available:

- **Generate Full Backup and Restore**—The option titled Yes, Generate a Full Backup of the Primary Database and Restore It into the Secondary Database kicks off the backup immediately following the completion of the configuration page and restores it to the secondary server.
- **Restore Existing Backup**—The option titled Yes, Restore an Existing Backup of the Primary Database into the Secondary Database can be used to avoid latency of copying the initial backup over the network if a recent backup of the database is close to the secondary server.

- **Do Not Initialize**—The option titled No, the Secondary Database Is Initialized should be used only if a recent copy of the source database has already been created and is in a state ready for additional restores. This option is commonly used when recovering from a log shipping interruption and during a failover/failback scenario.

Follow these steps to configure the database initialization options and continue the log shipping implementation:

1. Select the option Yes, Generate a Full Backup of the Primary Database and Restore It into the Secondary Database.
2. Click the Restore Options button. The path to the database and transaction log on the secondary server can be specified.
3. Enter **`D:\Program Files\Microsoft SQL Server\MSSQL.1\MSSQL\Data`** for the folder for the data files.
4. Enter **`L:\Program Files\Microsoft SQL Server\MSSQL.1\MSSQL\Data`** for the folder for the log files. Click OK.

> **Note**
> Make sure the path to each folder exists; otherwise, the log shipping configuration will fail.

The paths for the database and transaction log were configured to match the path for the AdventureWorks database on the primary server. It is a best practice to standardize the path of the database on each server.

Creating the Copy Files Job

The next step in the process is to configure the Copy Files job settings. This job is created on the secondary server and is used to copy the transaction log backups from the backup share to the staging folder. The Copy Files tab contains options used to configure the copy job.

> **Note**
> The target folder is normally in local storage on the secondary SQL Server. If the log backups are being shipped to a failover cluster, a shared storage location is highly recommended as the transaction log staging area.

The following options are available on the Copy Files tab:

- **Destination Folder**—This is the storage location for the copied transaction log backups. This folder is usually kept in local storage on the secondary server and in large-scale environments should be placed on a different set of disks than the database and transaction log.
- **Backup Retention**—This option defines how long the transaction log backups are kept in the staging folder. The backup copies are kept for 72 hours by default. This option should be adjusted as necessary to keep the destination location from running out of space while providing enough backups to recover from minor interruptions.
- **Schedule**—The schedule controls how often the copy job copies new backup files in the source folder to the staging area. The schedule of the copy file job is set to run all day at 15-minute intervals, as a default. The schedule of this job does not need to match the backup job and is commonly configured at a higher interval than the backup job. See the section "Managing Log Shipping Jobs" later in this chapter for additional details on managing transaction log schedules.

Follow these steps to configure the Copy Files tab:

1. Select the Copy Files tab on the Secondary Database Settings dialog box.
2. Enter **`D:\LSCopy\AdventureWorks`** in the Destination Folder field.
3. Accept the default 72-hour retention period and default schedule.

Figure 20.3 shows how the Copy Files tab should look after being configured using the preceding steps.

> **Note**
>
> The transaction log backups can be used for a point-in-time recovery of the database.

The default schedule executes the copy job every 15 minutes. With this schedule, a total of (72*60/15)=288 transaction log backups will be stored in the staging folder.

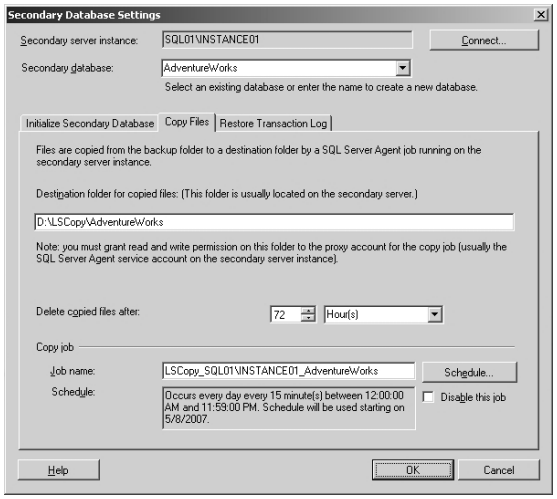

FIGURE 20.3
Copy Files tab settings.

Defining the Restoration Options

The next step in the process is to configure the Restore Transaction Log job settings. This job is also created on the secondary server and is used to restore the transaction log backups located in the staging folder. The following options are available for the restoration process:

- **No Recovery Mode**—This option restores the transaction log backups with the NORECOVERY option. The NORECOVERY option keeps the restore process open, allowing additional transaction log backups to be applied.
- **Standby Mode**—This option is similar to the NORECOVERY option except that it allows the data on the secondary database to be queried for reporting purposes.
- **Delay Restoring Backup**—An optional delay can be configured. This option can be used to control when the transaction log backups are restored. A delay may be desirable in some situations when the warm standby server is used to recover from a point in time before a catastrophic operation occurred.
- **Alert Threshold**—An alert threshold can be configured to generate a SQL alert when a backup hasn't been performed within a specific time frame.

Select the Restore Transaction Log tab on the Secondary Database Settings dialog box. Accept the default option and then click OK to complete the configuration.

> **Note**
>
> When using a log shipping server for reporting purposes, ensure that the restore job and reporting jobs don't overlap because this can cause the reports to fail or show inaccurate data. Consider using replication as a great alternative for maintaining a real-time copy of the database for reporting.

The default options on this tab specify the no recovery mode, the delay option is set to 0, and the alert setting is set to 45 minutes. This configuration offers an adequate starting point, but these options can be fine-tuned to achieve maximum effectiveness for the environment.

Adding a Monitor Server

The final step in the configuration process is to define the monitor server. The monitor server is used to track the status of the backup and restoration process. If either the backup or restore exceeds the specified threshold, an alert can be generated.

> **Note**
>
> The monitor server must be defined during the initial configuration of log shipping. To add a monitor server after log shipping has been established, remove and re-create the log shipping configuration.

Use the following procedure to add the server SQL02\INSTANCE01 as the monitor server:

1. From within the Transaction Log Shipping page of the database properties, enable the Use a Monitor Server Instance option.
2. Click the Settings button. When the Log Shipping Monitor Setting window opens, click the Connect button.
3. Enter the server and instance name (**`SQL02\INSTANCE01`**).
4. Select Windows Authentication from the Authentication drop-down menu and then click the Connect button.

Note

The monitor role should be hosted on a server other than the primary or secondary server.

These log shipping configuration options are available:

- **Monitor Connections**—The agent that executes each of the log shipping jobs must be able to connect to the SQL Server. You can set the Monitor Connections option to impersonate the SQL Server agent proxy account or use a specific account.
- **History Retention**—This is the amount of time old log shipping transactions are kept in the monitor server database.
- **Alert Job**—The alert job name can be changed; however, the job schedule is hard-coded and doesn't normally need to be modified.

Click OK to accept the default settings and return to the Transaction Log Shipping page. Figure 20.4 shows how the Transaction Log Shipping options page should look after being configured using the preceding steps.

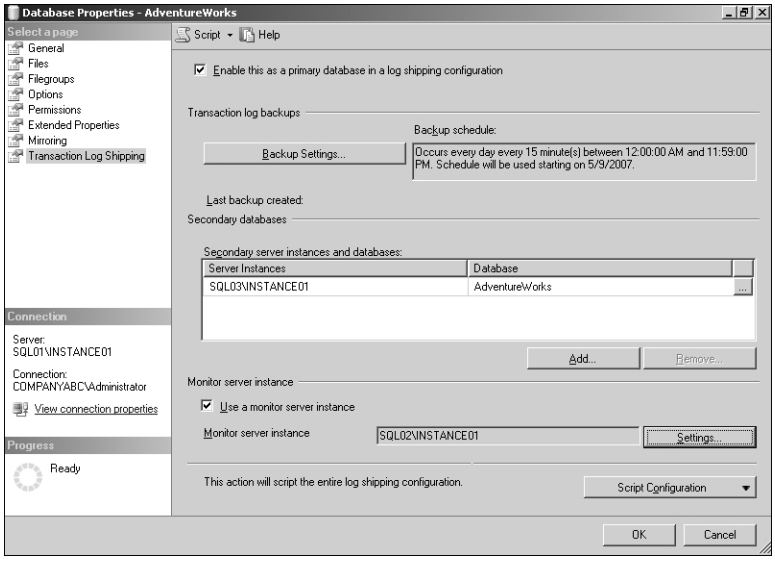

FIGURE 20.4
Transaction Log Shipping properties.

Now that the log shipping configuration is complete, click OK on the properties page to apply the configuration to the primary, secondary, and monitor servers.

After you click OK, the database is backed up and restored on the secondary server. Then the SQL jobs used to create backups are scheduled on the source server, the copy job and restore jobs are created on the destination server, and the alert job is created on the monitor server.

Managing Log Shipping

The following sections describe common log shipping management tasks. These tasks include recovery steps that should be taken when the primary and secondary databases become unsynchronized, along with steps on how to fail over the database to a secondary server and then fail back to the original database after the primary server has been repaired.

Recovering from Log Shipping Interruptions

Log shipping interruptions can result in the log shipping process stopping until the problem is corrected. When the log shipping process is stopped, the database on the secondary server is no longer synchronized with the source data. It is important to correct these interruptions quickly to prevent transaction log backups in the backup share from being removed before they can be shipped to the secondary server. If the transaction log backups in the backup share are removed before being shipped to the secondary server, the log shipping configuration must be re-created and the secondary database must be reinitialized.

> **Note**
>
> Different factors should be considered when configuring the transaction log retention period. For example, the 72-hour default retention period allows easy recovery from a failure that may occur on Friday evening, without your needing to reinitialize the database on the secondary server when returning to work the following week.

Common problems are usually the result of inadequate disk space. If the transaction log backup job runs every 15 minutes and 72 hours of transaction log backups are retained, 288 transaction log backups are generated. These backups are stored in both the backup share and staging folders.

Additional problems usually occur if one of the jobs used in the log shipping process stops working, but all the transaction log backups are available. In this scenario, the problem can be corrected, and the log shipping process will catch up during the next execution cycle. For example, if a security setting was changed on the backup share and the agent on the secondary no longer has the appropriate permissions needed to access the share, the log shipping process stops. When the permissions are corrected, the next log shipping job cycle will copy and restore all transaction log backups from where it left off before the problem occurred.

In a worst-case scenario, a transaction log backup is lost or otherwise unable to be restored to the secondary server. In this scenario, the log shipping configuration must be rebuilt, and the database must be reinitialized. Follow these steps to remove log shipping from the database:

1. From the primary server (SQL01), choose Start, All Programs, Microsoft SQL Server 2005, SQL Server Management Studio.
2. Select Database Engine from the Server Type drop-down; then enter the server and instance name (**`SQL01\INSTANCE01`**).
3. Select Windows Authentication from the Authentication drop-down menu and then click the Connect button.
4. A connection to the database engine is made. If the Object Explorer pane is not visible, press the F8 button.
5. From within the Object Explorer pane, expand Databases; the AdventureWorks database should be listed.
6. Right-click the AdventureWorks database and select Properties. The Database Properties window opens.
7. Select the Transaction Log Shipping page. Uncheck the Enable This as a Primary Database in a Log Shipping Configuration option.

After log shipping has been removed, the log files in the backup share and the staging area can also be removed because they cannot be applied to the database out of order and are most likely unusable. The database on the secondary server is not removed automatically and should be manually removed before re-creating log shipping.

Managing Log Shipping Jobs

The following options are available when configuring the schedule properties for the backup, copy, and restore log shipping jobs:

- **Name and Schedule Type**—The name of the backup job should be unique and allow easy identification of the job function. The schedule type for the log shipping backup job should be set to Reoccurring because the job must back up the transaction log backups repeatedly.
- **Frequency**—The frequency of the log shipping job is normally set to Daily. This option is adequate for most implementations of log shipping because the occurrence pattern option essentially keeps the job running at predefined intervals throughout the day.
- **Occurrence Pattern**—The occurrence pattern executes each transaction log shipping job at predefined intervals throughout the day. The default reoccurrence pattern is set to run the job every 15 minutes from 12:00:00 a.m. to 11:59:00 p.m. The reoccurrence pattern should be configured according the organization's Service Level Agreements (SLAs).
- **Duration**—The duration of the log shipping agent jobs is commonly set to No End Date because keeping the target servers updated is key.

> **Note**
>
> The size of the transaction log backups, speed of the network, and length of time the restoration takes all play a significant role in the planning and scheduling process to prevent a backlog from occurring.
>
> Increasing the frequency of transaction log backups reduces the amount of data lost during a failure while directly increasing the network and system resources consumed.

The log shipping job schedules can be changed during the log shipping configuration or after log shipping has been implemented. The backup job is located on the primary server, and the copy and restore jobs are located on the secondary servers.

To change the log shipping jobs, navigate to SQL Server Agent, Jobs from within SQL Server Management Studio. Double-click one of the log shipping jobs and select the Schedule options page to change the schedule for the job.

Failing Over to a Secondary Server

There are two basic scenarios for initiating a log shipping failover from the primary to the secondary server. The first type of failover occurs when both the primary and secondary server are up and running and the database on

both systems is in a usable state. This scenario is referred to as a *controlled* failover and is usually performed to test the log shipping failover process.

The other scenario occurs when the primary server or database has failed and is unavailable, and the secondary database must be brought online. In this scenario, because the primary database is not available, the work performed between the last transaction log backup and the time of the failure is lost and must be manually re-created.

The process for the two scenarios is almost the same. However, the controlled failover scenario has some additional steps to ensure that no data is lost during the transition.

Preparing for a Controlled Failover

You can use the following procedure to initiate a controlled failover to the secondary database. This procedure is used to eliminate the possibility of data loss but is available only when the primary server is still online and the database is available.

> **Note**
>
> Before starting a controlled failover, ensure no users or applications are using the database.

To start the process, disable the existing log shipping jobs on the primary and secondary servers. You can do this by navigating to SQL Server Agent, Jobs from within SQL Server Management Studio. Right-click each log shipping job and select Disable. Based on the steps found in Implementing Log Shipping, the backup job on the primary server is called LSBackup_AdventureWorks, whereas the copy and restore jobs on the secondary server are called LSCopy_SQL01\INSTANCE01_AdventureWorks and LSCopy_SQL01\INSTANCE01_AdventureWorks.

Place the database in single user mode. You can do this by right-clicking the database from within SQL Server Management Studio and selecting Properties. When the Properties window opens, select the Options page. Change the Restrict Access option located at the bottom of the list to SINGLE_USER and then click OK.

The next step is to back up the primary database with the NORECOVERY option. This places the database in a state ready for the controlled "failback" process. This also backs up any changes that occurred between the current

time and the time of the last transaction log backup. Use the following code to back up the database with the NORECOVERY option:

```
USE MASTER
BACKUP LOG [AdventureWorks]
TO  DISK = N'D:\LSBackup\AdventureWorks\AdventureWorks_Final.trn'
WITH  NORECOVERY
GO
```

The final changes are backed up and stored in a file called `AdventureWorks_Final` in the LSBackup\AdventureWorks folder.

The next step is to synchronize the primary and secondary databases, essentially making sure they are exactly the same by restoring the final transaction log backup.

The first part of this step simply involves manually running the copy and restore jobs on the secondary server. This restores any transaction log backups in the backup share that have not been applied to the database on the secondary server. To manually run the copy and restore jobs, from within SSMS on the secondary server (SQL03), right-click the LSCopy job and select Start Job at Step. After the LSCopy job has completed successfully, right-click the LSRestore job and select Start Job at Step.

The second part of this step involves manually copying and applying the `AdventureWorks_Final.trn` backup file to the database on the secondary server. You can use the following code to commit the last transaction log to the database on the secondary server:

```
USE MASTER
RESTORE LOG [AdventureWorks]
FROM DISK = N'D:\LSCopy\AdventureWorks\AdventureWorks_Final.trn'
WITH NORECOVERY
GO
```

This code assumes the `AdventureWorks_Final.trn` file has been copied to the LSCopy\AdventureWorks folder on the secondary server.

The next step is to bring the secondary database online.

Bringing the Secondary Database Online

This section details the actual process used to bring the database on the secondary server online. These steps are a continuation of the controlled failover process. If the controlled failover process is not available, this is the

starting point for a catastrophic failure that has caused the primary database and/or server to become unavailable.

To bring the database on the secondary server online, run the following command from the secondary server:

```
USE MASTER
RESTORE DATABASE AdventureWorks WITH RECOVERY
```

The database is now online and ready to be used by clients. The next step in the process is to redirect any application or clients that use the database to the new server. This step can be different for each front-end application being used.

Failing Back to the Primary Server

Failing back to the original primary server is actually easy. When a failover was originally initiated, the secondary server that was brought online is now the primary server. To fail back to the original server, simply establish log shipping from the active database on the "new" primary server back to the "original" server. The "original" primary server then acts as the secondary server.

After log shipping is established, a controlled failover can be initiated, and the original server will once again be the primary server hosting the active database.

The only configuration option that changes is in the Secondary Database Settings dialog box: Select the No, the Secondary Database Is Initialized option.

Managing Log Shipping from the Command Line

SQL Server 2005 Service Pack 2 now supports the `SQLLogShip.exe` application. Table 20.2 shows each option available with this program.

Table 20.2 `SQLLogShip.exe` **Options**

Option	Description
`-server <instance>`	Specifies the instance of SQL Server where the operation will run.
`-backup <p_id>`	Performs a backup operation for the primary database whose primary ID is specified by `<p_id>`.

Table 20.2 **continued**

Option	Description
`-copy <s_id>`	Performs a copy operation to copy backups from the specified secondary server for the secondary database, or databases, whose secondary ID is specified by <*s_id*>.
`-restore <s_id>`	Performs a restore operation on the specified secondary server for the secondary database, or databases, whose secondary ID is specified by <*s_id*>.
`-verboselevel <level>`	Specifies the level of messages added to the log shipping history. The level can be set from 0 to 4, with 0 logging no output and 4 logging all messages.
`-logintimeout <value>`	Specifies the amount of time in seconds allotted for attempting to log in to the server instance. The default is 15 seconds.
`-querytimeout <value>`	Specifies the amount of time in seconds allotted for starting the specified operation.

The backup, copy, and restore option cannot be combined and must be executed on the appropriate server. For example, the backup option must be run on the primary server, and the copy/restore option must be run on the secondary server.

The primary ID for the backup command can be obtained by querying the log_shipping_primary_database's system table. The secondary ID for the copy and restore commands can be obtained by querying the log_shipping_ secondary system table.

Monitoring and Troubleshooting Log Shipping

The following sections describe the different administration and maintenance tables, stored procedures, and reports that can be used to troubleshoot and monitor log shipping.

It is important to become familiar with how to use each of these items because they help keep the high-availability solution running efficiently without issue.

Viewing Log Shipping Reports

Log shipping reports can be viewed from the primary, secondary, and monitor servers. However, viewing the reports from the monitor server is

most effective because the monitor server contains records from both the primary and secondary servers. Viewing the log shipping report from the primary and secondary servers shows only half the data.

Follow these steps to view log shipping reports:

1. From the test server (SQL01), choose Start, All Programs, Microsoft SQL Server 2005, SQL Server Management Studio.
2. Select Database Engine from the Server Type drop-down; then enter the server and instance name (**`SQL01\INSTANCE01`**).
3. Select Windows Authentication from the Authentication drop-down menu and then click the Connect button.
4. A connection to the database engine is made. If the Object Explorer pane is not visible, press the F8 button.
5. From within the Object Explorer pane, right-click the server name (SQL02\INSTANCE01) at the top of the tree.
6. Select Reports, Standard Reports, and Transaction Log Shipping Status from the context menu.
7. The transaction log shipping report is displayed.

Figure 20.5 shows a sample log shipping report run from the monitor server.

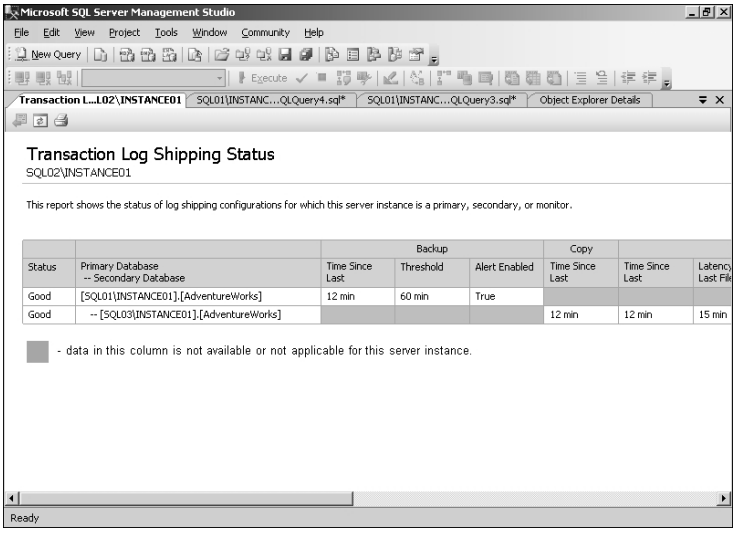

FIGURE 20.5
Transaction log shipping report.

The report shows the backup and restore configuration, the current status of each log shipping server, along with the times the last backup and copy operation took place.

Querying Log Shipping Tables for Status

The tables listed in Table 20.3 store information about log shipping and can be used to monitor status and troubleshoot issues.

Table 20.3 **Log Shipping Tables**

Table Name	Description
log_shipping_monitor_alert	Stores alert job ID.
log_shipping_monitor_error_detail	Stores error details for log shipping jobs.
log_shipping_monitor_history_detail	Contains history details for log shipping agents.
log_shipping_monitor_primary	Stores one monitor record for the primary database in each log shipping configuration, including information about the last backup file and last restored file that is useful for monitoring.
log_shipping_monitor_secondary	Stores one monitor record for each secondary database, including information about the last backup file and last restored file that is useful for monitoring.

To use these tables, for example, you could execute a standard SQL query. Use the following code to locate all log shipping errors, sorted by the most recent:

```
USE msdb
SELECT * FROM log_shipping_monitor_error_detail
ORDER BY log_time Desc
```

Using Log Shipping Stored Procedures

The following stored procedures can be used to assist in troubleshooting and monitoring log shipping activity:

- **sp_help_log_shipping_monitor_primary**—Returns monitor records for the specified primary database from the log_shipping_monitor_primary table. This stored procedure can be run on monitor servers or primary servers.

- **sp_help_log_shipping_monitor_secondary**—Returns monitor records for the specified secondary database from the log_shipping_monitor_secondary table. This stored procedure can be run on the monitor server or secondary servers.
- **sp_help_log_shipping_alert_job**—Returns the job ID of the alert job. This stored procedure can be run on the monitor server or primary and secondary servers if no monitor is defined.
- **sp_help_log_shipping_primary_database**—Retrieves primary database settings and displays the values from the log_shipping_primary_databases and log_shipping_monitor_primary tables. This stored procedure can be run on the primary server.
- **sp_help_log_shipping_primary_secondary**—Retrieves secondary database names for a primary database. This stored procedure can be run on the primary server.
- **sp_help_log_shipping_secondary_database**—Retrieves secondary database settings from the log_shipping_secondary, log_shipping_secondary_databases, and log_shipping_monitor_secondary tables. This stored procedure can be run on secondary servers.
- **sp_help_log_shipping_secondary_primary**—This stored procedure retrieves the settings for a given primary database on the secondary server. This stored procedure can be run on secondary servers.

Summary

SQL Server 2005 log shipping is almost identical to log shipping in previous versions of SQL Server. However, the role of log shipping has been reduced because new technologies such as database mirroring provide substantial enhancements. Because these new technologies have limits, log shipping is commonly used to complement other high-availability technologies in scenarios such as replicating business-critical data to a disaster recovery site.

Best Practices

The following best practices can be taken from this chapter:

- Store enough backups to recover from interruptions in the log shipping process. This ensures that transaction log backups are not removed before the problem can be corrected.

- Use the transaction log backups for a point-in-time recovery of the database in the event of data corruption.
- On a heavily used database, transactions can be generated very quickly, and the log can consume a large amount of space. Be sure to allocate enough space to store the transaction log.
- Configure the backup and restore reoccurrence pattern according to acceptable loss and environmental limits. A more aggressive reoccurrence pattern lowers the amount of data lost during a catastrophic failure.
- The size of the transaction log backups, speed of the network, and length of time the restoration takes all play a significant role in planning and scheduling the log shipping solution.
- Make sure a backlog in the log shipping process does not occur.
- Ensure the transaction log for the database is not backed up by any other maintenance plans because other maintenance plans can truncate the transaction log before the changes are shipped to the secondary servers.
- Log shipping is most practical when multiple secondary servers are necessary. If only a single secondary server is being used, use database mirroring over log shipping.
- Apply transaction log backups in sequential order to the secondary database.
- Define the monitor server during the initial configuration of log shipping. To add a monitor server after log shipping has been established, remove and re-create the log shipping configuration.
- Standardize the path of the database on each server.
- When using a log shipping server for reporting purposes, ensure the restore job and reporting jobs don't overlap because this can cause the reports to fail or show inaccurate data. Consider using replication as a great alternative for maintaining a real-time copy of the database for reporting.