

HOUR 3

Getting to Know the SQL Server Management Studio

The Microsoft SQL Server Management Studio Express is the new interface that Microsoft has provided for management of your SQL Server database. It is the main tool that you will use when maintaining your databases and the objects they contain. In this hour you'll learn:

- ▶ The system databases that ship with SQL Server 2005 Express
- ▶ What is available under the Security node of the Management Studio
- ▶ The types of server objects that are available
- ▶ What is available under the Replication mode of Management Studio
- ▶ What is available under the Management node of the Management Studio
- ▶ How to create or attach to a SQL Server database

Microsoft SQL Server Management Studio Express

Management Studio has replaced its predecessor, Enterprise Manager. Not only is Management Studio easier to use, it provides more functionality than Enterprise Manager. It combines Enterprise Manager and Query Analyzer into one powerful tool. In the sections that follow, you will explore the various nodes available in Management Studio and learn what is available under each node.

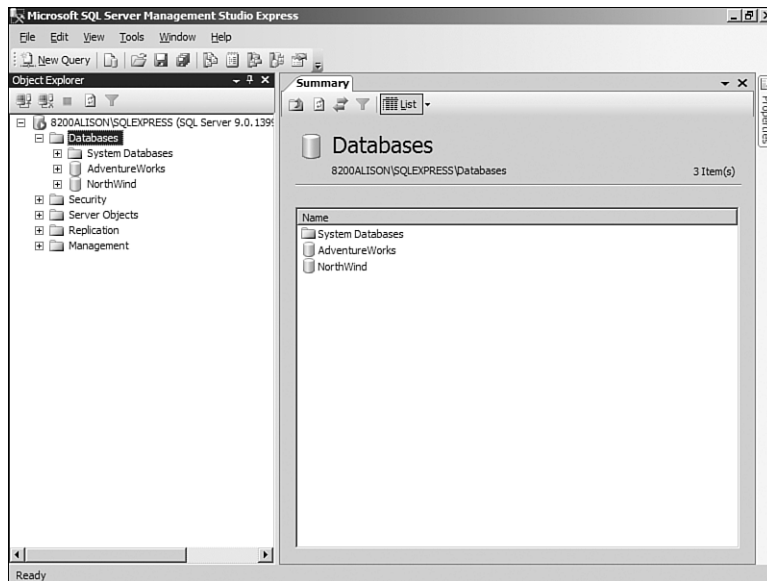
The Databases Node

The Databases node is the first node in SQL Server Management Studio. Within the Databases node are one or more subnodes. The first subnode is System Databases. There are additional subnodes for each database contained on the server (see Figure 3.1). The

sections that follow cover each of the system databases found under the System Databases subnode.

FIGURE 3.1

Within the Databases node are one or more subnodes.



The Master Database

The master database is the “database of all databases.” It keeps track of logon accounts, linked servers, system configuration settings, and more. It also contains initialization settings for SQL Server. Prior to SQL Server 2005, Master also contained tables that stored information about all the other system objects (databases, tables, views, stored procedures, triggers, functions, and more). SQL Server 2005 stores these tables in a database called Resource. Resource contains tables that track of all the objects that are associated with the SQL Server. The Resource database is hidden and should be accessed only by a Microsoft Customer Support Services (CSS) specialist to provide troubleshooting and support services to customers.

The Model Database

Model is a very special database. Anything that you place in model is automatically propagated to all the databases that you create thereafter. This means that, for example, you can add a State table to Model. That State table then appears in all the new databases that you build. You work with Model just as you work with any other database. You can include almost any object in Model. This means that you easily can propagate tables, views, stored procedures, triggers, functions, and more.

This not only provides you with standardization between databases, but provides you with a great jump start on creating the databases you need. If you modify Model, you do not affect any existing databases. All new databases will be affected by your changes.

The MSDB Database

The MSDB (Microsoft Database) database is used by SQL Server, SQL Server Management Studio, and SQL Server Agent. All three of them use it to store data, including scheduling information and backup and restore history information. For example, SQL Server maintains a complete backup and restore history in MSDB. There are several ways that you can add to or modify information stored in the MSDB database. They include

- ▶ Scheduling tasks
- ▶ Maintaining online backup and restore history
- ▶ Replication

The TempDB Database

TempDB is a system database that acts as a resource to all users working with a particular instance of SQL Server. TempDB holds the following objects:

- ▶ Temporary user objects such as temporary tables, temporary stored procedures, temporary table variables, or cursors
- ▶ Internal objects used by the database engine to perform tasks such as sorting
- ▶ Row versions that are generated in data modification transactions

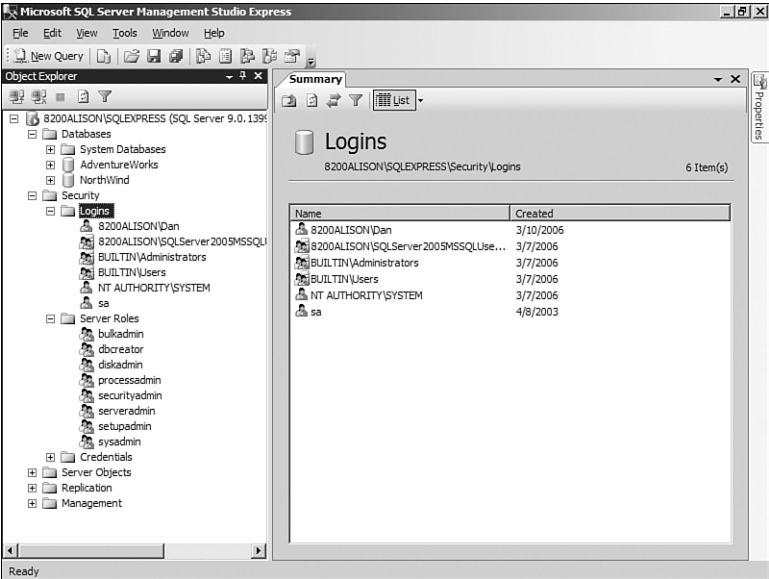
The Security Node

As its name implies, the Security Node enables you to manage SQL Server security. Using the Security Node, you can work with logins, add to and remove people from server roles, and create credentials. This chapter provides an introduction to security. For more information, see Hour 21, “SQL Server Authentication,” and Hour 22, “SQL Server Permissions Validation.”

The Logins Node

Logins represent the users and roles that have access to your system. Note in Figure 3.2 that two types of icons appear under the Logins node. One is granting a role access to the database, and the other is granting a user access to the database.

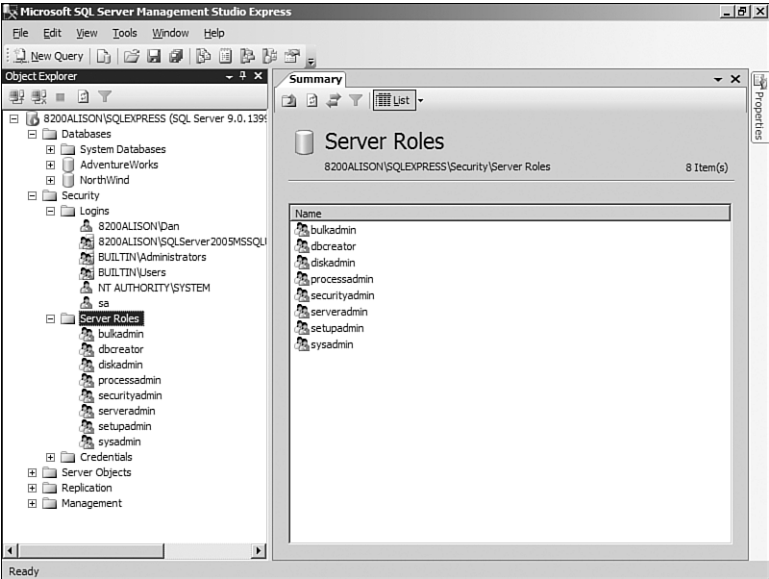
FIGURE 3.2
 Notice that Administrators is a role, and sa is a user.



The Server Roles Node

Server Roles are predefined roles, supplied by SQL Server. Each Server Role possesses a pre-defined set of rights. Figure 3.3 shows the available Server Roles. You cannot add or remove Server Roles.

FIGURE 3.3
 Each Server Role possesses a pre-defined set of rights.



The Credentials Node

Credentials are new to SQL Server 2005. A credential is a record that contains the authentication information required for SQL Server to connect to an outside resource. Most credentials are made up of a Windows login and password.

Server Objects

Server Objects refer to a set of objects used at the server level (not at the database level). These objects include Backup Devices, Linked Servers, and Server Triggers.

Backup Devices

Backup devices include the tapes and disks that you use to back up or restore your SQL Server. When creating a backup, you must designate the backup device that you want to use (see Figure 3.4). You select from a list of backup devices that you have created.

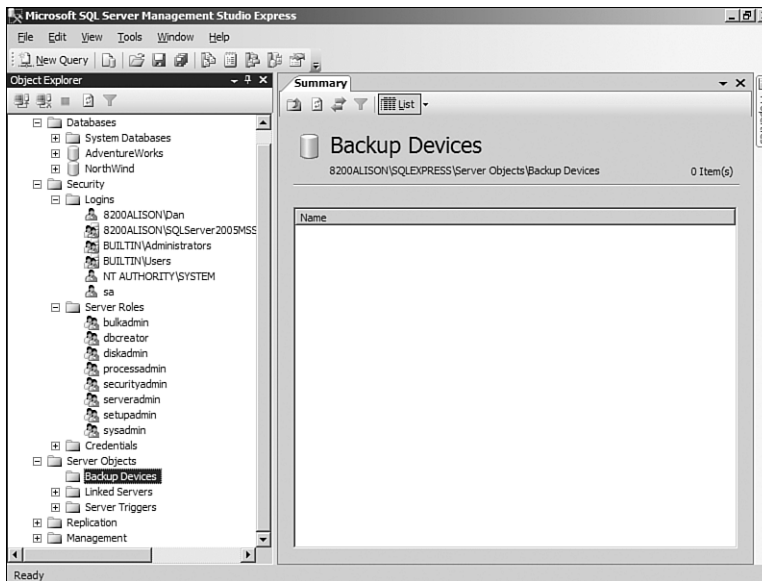


FIGURE 3.4
When creating a backup, you must first designate the backup device you want to use.

Linked Servers

Linked servers enable you to work with other SQL Servers, as well as databases other than SQL Server databases, right from within Management Studio. This offers a few advantages:

- ▶ The capability to get remote server access
- ▶ The capability to issue distributed queries, updates, commands, and transactions on heterogeneous data sources across the enterprise
- ▶ The capability to address diverse data sources in a similar manner

Server Triggers

Server triggers are DDL (Data Definition Language) triggers. They execute in response to changes being made to the structure of the database. They are great for both auditing and regulating database operations. For example, if SQL Server determined that there were more than a specified number of records in a table, it would not allow users to remove fields from the table.

The Replication Node

Data replication is the capability of a system to make copies of its data and application objects automatically in remote locations. You easily can propagate any changes to the original or data changes to the copies to all the other copies. Data replication enables users to make changes to data offline at remote locations. SQL Express synchronizes changes to either the original or the remote data with other instances of the database.

The original database is referred to as the *design master*. You can make changes to definitions of tables or other application objects only at the design master. You use the design master to make special copies called *replicas*. Although there is only one design master, replicas can make other replicas. The process of the design master and replicas sharing changes is referred to as *synchronization*.

To see an example of data replication at work, imagine that you have a team of salespeople who are out on the road all day. At the end of the day, each salesperson logs on to one of the company's servers through Terminal Services. The replication process sends each salesperson's transactions to the server. If necessary, the process sends any changes to the server data to the salesperson.

Management

The Management node contains tools that help you to manage your SQL Server. These tools include the capability to view both the SQL Server Logs and the Activity Monitor.

SQL Server Logs

SQL Server Express 2005 adds entries for certain system events to the SQL Server Error Log and to the Microsoft Windows application log. You can use these logs to identify the sources of problems. Using the SQL Server Management Studio Log File Viewer, you can integrate SQL Server, SQL Server Agent, and the Windows logs into a single list, making it easy to review all related events.

Activity Monitor

You use the Activity Monitor component of SQL Server Management Studio to get information about users' connections to the database engine and the locks that they hold. The Activity Monitor has three pages. The Process Info page contains information about the connections. The Locks by Process page sorts the locks by the connection. The Locks by Object page sorts the locks by the object name.

Creating a SQL Server Database

Before you can build tables, views, stored procedures, triggers, functions, and other objects, you must create the database in which they will reside. A database is a collection of objects that relate to one another. An example would be all the tables and other objects necessary to build a sales order system. To create a SQL Server database, follow these steps:

1. Right-click the Databases node and select New Database. The New Database dialog appears (see Figure 3.5).
2. Enter a name for the database.
3. Enter a path for the database (see Figure 3.6).
4. Click to select the Options page and change any options as desired (see Figure 3.7).
5. Click OK to close the New Database dialog and save the new database. The database now appears under the list of databases (see Figure 3.8) under the Databases node of SQL Server Management Studio. If the database does not appear, right-click the Databases node and select Refresh.

FIGURE 3.5
 The New Database dialog enables you to create a new database.

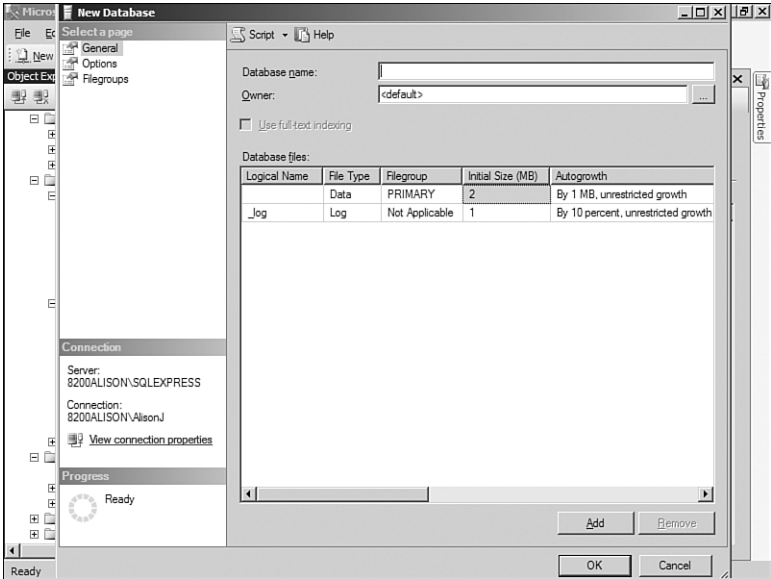
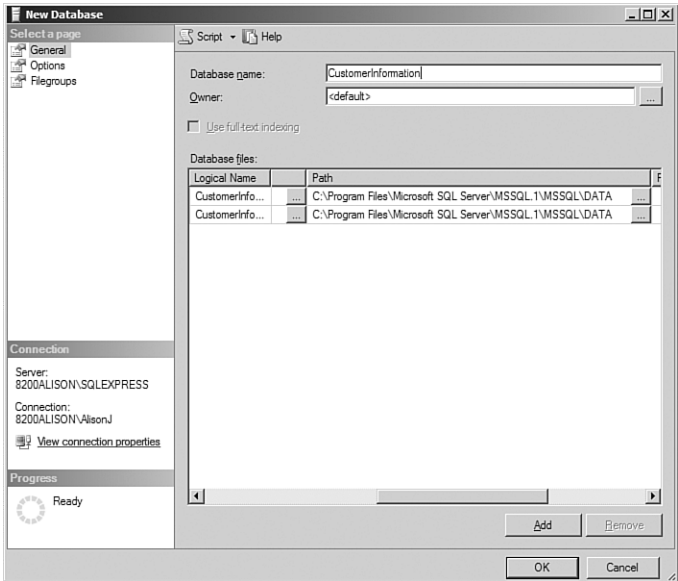


FIGURE 3.6
 You can opt to accept the default path, or you can designate a path for the database.



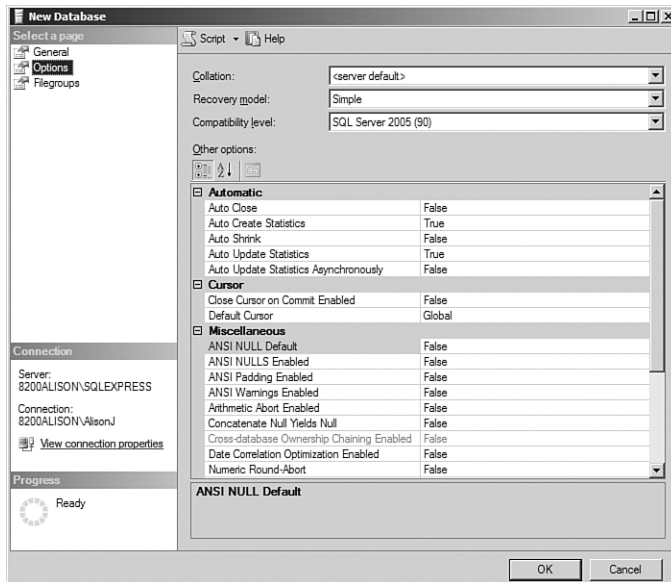


FIGURE 3.7
The Options page of the New Database dialog enables you to set custom options for the database.

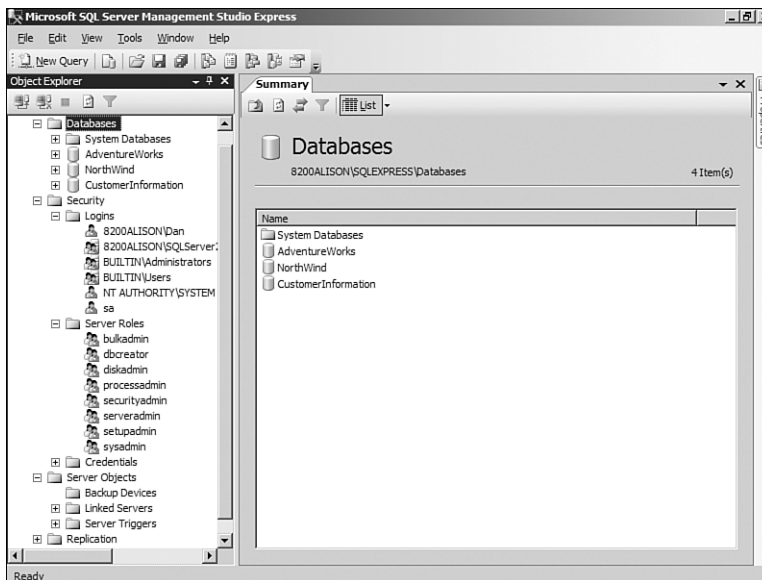
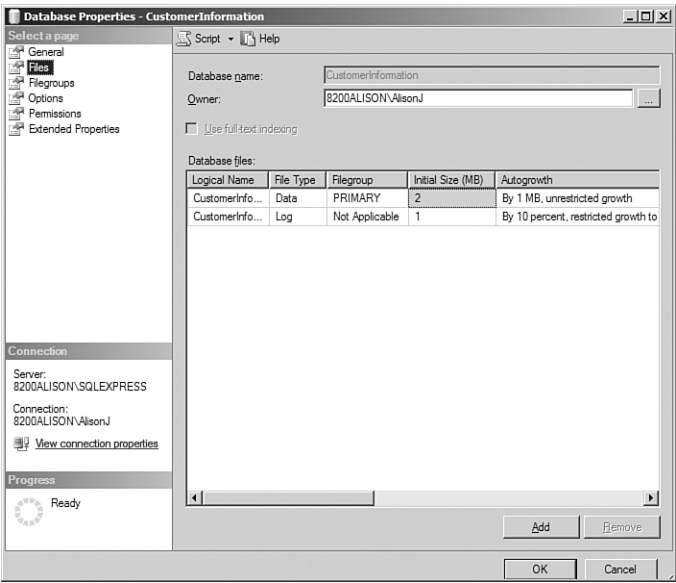


FIGURE 3.8
The new database appears under the list of databases in the Databases node.

Files

In the previous section you created a new SQL Server database. You accepted all the default options available on the General page of the New Database dialog. Many important options are available on the General page. They include the Logical Name, File Type, Filegroup, Initial Size, Autogrowth, Path, and File Name (see Figure 3.9).

FIGURE 3.9
 Several important features are available on the General page of the New Database dialog.



The logical name is the name that SQL Server will use to refer to the database. It is also the name that you will use to refer to the database when writing programming code that accesses it.

The File Type is Data or Log. As its name implies, SQL Server stores data in data files. The file type of Log indicates that the file is a transaction log file.

The initial size is very important. You use it to designate the amount of space you will allocate initially to the database.

By the Way

I like to set this number to the largest size that I ever expect the data database and log file to reach. Whereas disk space is very cheap, performance is affected every time that SQL Server needs to resize the database.

Related to the initial size is the Autogrowth option. When you click the Build button (ellipse) to the right of the currently selected autogrowth option, the Change Autogrowth dialog appears (see Figure 3.10).

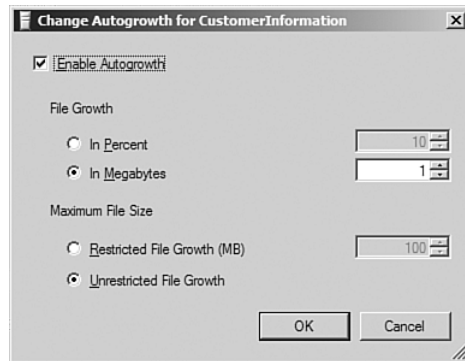


FIGURE 3.10
The Change Autogrowth dialog enables you to designate options that affect how the database file grows.

The first question is whether you want to support autogrowth at all. Some database designers initially make their databases larger than they ever think they should be and then set autogrowth to false. They want an error to occur so that they will be notified when the database exceeds the allocated size. The idea is that they want to check things out to make sure that everything is okay before allowing the database to grow to a larger size.

The second question is whether you want to grow the file in percentage or in megabytes. For example, you can opt to grow the file 10% at a time. This means that if the database reaches the limit of 5,000 megabytes, then 10% growth would grow the file by 500 megabytes. If instead the file growth was fixed at 1,000 megabytes, the file would grow by that amount regardless of the original size of the file.

The final question is whether you want to restrict the amount of growth that occurs. If you opt to restrict file growth, you designate the restriction in megabytes. Like the Support Autogrowth feature, when you restrict the file size, you essentially assert that you want to be notified if the file exceeds that size. With unrestricted file size, the only limit to file size is the amount of available disk space on the server.

File Groups

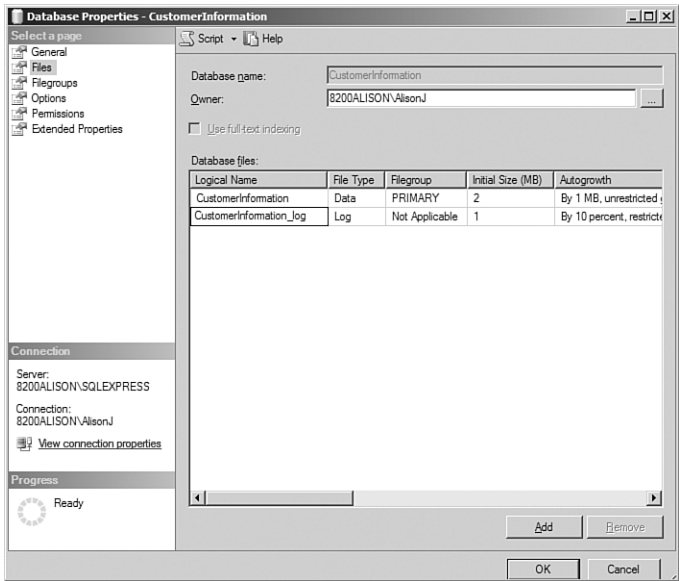
One great feature of SQL Server is that you can span a database's objects over several files, all located on separate devices. By doing this you improve the performance of the database because multiple hardware devices can access the data simultaneously.

The Transaction Log

SQL Server uses the transaction log to record *every* change that is made to the database. In the case of a system crash, you use the transaction log, along with the most recent backup file, to restore the system to the most recent data available. The transaction log supports the recovery of individual transactions, the recovery of all incomplete transactions when SQL Server is once again started, and the rolling back of a restored database, file, filegroup, or page forward to the point of failure. Specifying information about the transaction log is very similar to doing so for a database. Follow these steps:

1. While creating a new database, notice that you can also enter information about the log file (see Figure 3.11). To begin, enter a logical name for the database. I recommend that you use the logical name of the database along with the suffix `_log`.

FIGURE 3.11
 While creating a new database, you can also enter information about the log file.



2. Specify the initial size of the log file.
3. Indicate how you want the log file to grow.
4. Designate the path within which you wish to store the database.
5. Continue the process of creating the database file.

Do not move or delete the transaction log unless you are fully aware of all the possible ramifications of doing so.

**Watch
Out!**

Attaching to an Existing Database

There are times when someone will provide you with a database that you want to work with on your own server. That database may be a SQL Server 2005 database, or might even be a SQL Server 2000 database. To work with an existing database, all you have to do is attach to it. Here's the process:

1. Right-click the Databases node and select Attach. The Attach Databases dialog appears (see Figure 3.12).

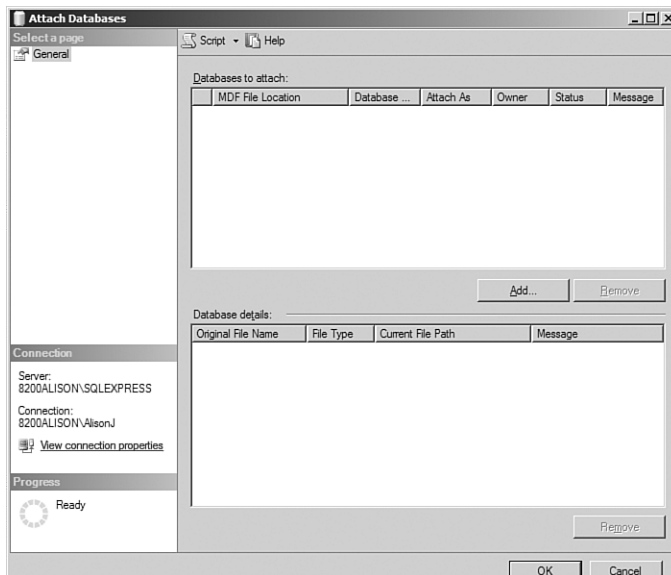
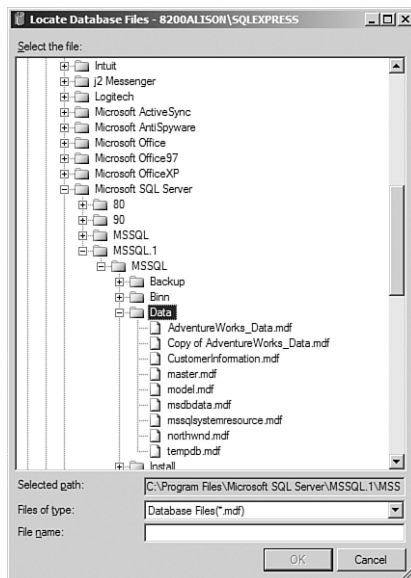


FIGURE 3.12
The Attach Databases dialog enables you to attach to existing .mdf database files.

2. Click Add. The Locate Database Files dialog appears (see Figure 3.13).
3. Locate and select the .mdf to which you want to attach.
4. Click OK to close the Locate Database Files dialog.
5. Click OK to close the Attach Databases dialog. The database appears in the list of user databases under the Databases node of SQL Server Management Studio.

FIGURE 3.13

The Locate Database Files dialog enables you to select the database to which you want to attach.



Summary

It is important that you are comfortable with the SQL Server Management Studio Express. It is the tool that you will use the most when managing SQL Server databases. In this hour you learned about the various nodes available within SQL Server Management Studio Express and what functions you can perform under each node. Finally, you learned how to create and attach to a SQL Server database.

Q&A

Q. Explain what the Master database is used for.

A. The Master database is the database of all databases. It keeps track of all the login accounts, linked servers, system configuration settings, and more. It also contains initialization settings for SQL Server.

Q. Explain how you utilize the Model database.

A. Model is a very special database. Any objects that you place in Model propagate automatically to any new databases that you create. This provides standardization as well as rapid application development.

Q. *Describe MSDB and what it does.*

A. MDSB stores data, including scheduling information and backup and restore history information.

Workshop

Quiz

1. What is a linked server?
2. What is a DDL trigger?
3. TempDB is one of the system databases (true/false).
4. It is always okay to delete a log file (true/false).
5. What is a Server Role?
6. Credentials are made up of what?
7. Name two types of backup devices.

Quiz Answers

1. A linked server enables you to work with other SQL Servers as well as databases other than SQL Server databases, right from within Management Studio.
2. A DDL trigger executes in response to changes made to the structure of the database.
3. True.
4. False.
5. A server role is pre-defined by SQL Server. It possesses a pre-defined set of rights.
6. Credentials are made up of a login and a password.
7. Tape and disk.

Activities

Create a new SQL Server database. View it in the Object Explorer. Expand the Security Node. View the existing logins. Expand the Server Roles node and take a look at the list of Server Roles. Expand the Server Objects Node. Explore the Backup Devices, Linked Servers, and Server Triggers nodes. Expand the Management node. Take a look at both the SQL Server Logs and at the Activity Monitor.