

CHAPTER 3

SQL Server Management Studio

SQL Server Management Studio (SSMS) is a new integrated application that provides access to most of the graphical tools you can use to perform administrative and development tasks on SQL Server 2005. SSMS is a replacement for the Enterprise Manager, Query Analyzer, and Analysis Manager that were available in SQL Server 2000. Microsoft has consolidated all those tools into one, with a focus on providing a tool that suits the needs of both developers and database administrators (DBAs).

SSMS is a complicated tool that provides an entry point to almost all of SQL Server's functionality. The functionality that is accessible from SSMS is entirely too much to cover in one chapter. The aim of this chapter is to give a basic overview of SSMS, with a concentration on features that are new to SQL Server 2005. Other chapters in this book focus on the components of SSMS and provide more detailed coverage.

What's New in SSMS

SSMS is an entirely new environment for SQL Server 2005. It encapsulates many of the features previously available in other tools and also offers many new features. The bulk of these new features can be grouped into four major categories: environmental changes, integrated management, enhanced query authoring, and enhanced project management.

The environmental changes are changes that have occurred to the graphical application. SSMS has a new look and feel, and it offers some significant change to the way that

IN THIS CHAPTER

- ▶ What's New in SSMS
- ▶ The Integrated Environment
- ▶ Administration Tools
- ▶ Development Tools

windows are managed within the application. The application was rewritten in .NET and has features that are more like the development environment found in Visual Studio. Many windows in SSMS are dockable, can be pinned, and can be set to Auto Hide. In addition, many of the management dialog boxes are now modal, which means they can stay open while you open other windows within the application.

The new integrated management features stem from a consolidation of management tools. SSMS now contains management functionality that was contained in SQL Server 2000's Enterprise Manager, Analysis Manager, SQL Server Service Manager, Query Analyzer, and other tools. The functionality from these tools has been integrated into one environment that shares common Help, a summary window that displays useful information, an Object Explorer tree for easy navigation, and a myriad of other tools that can be accessed from one central location.

The changes related to query authoring are also based on a consolidation of functionality that was contained in several different tools in previous versions of SQL Server. Scripts that were previously created with Query Analyzer or Analysis Services can now be authored in SSMS. A new SSMS window named the Query Editor is an editing tool for the creation of SQL Server scripts. It brings with it many of the great features from the prior tools, such as color coding, syntax checks, and performance analysis, along with some new features, such as Dynamic Help, an XML editor, enhanced templates, and the ability to write scripts without being connected to the database.

The last category of changes in SSMS relates to managing the files or scripts you create when working with SQL Server 2005. SSMS provides a tool to organize scripts, connections, and other, related files into projects. These projects can also be grouped to form a solution. Once again, this functionality is based on the Visual Studio application development environment and the way it organizes development files into projects and solutions. As with Visual Studio, these files can also be managed with source control in SSMS. SSMS provides links to Visual SourceSafe, which allows you to secure the files and manage version control.

This chapter further explores the new features in SSMS. It first examines the features at the environmental level, focusing on how SSMS behaves and how to best utilize the environment. Next, it looks at the administrative tools and what changes have been made to help you better manage your SQL Server environment. Finally, this chapter looks at the development tools that are available with SSMS and the changes that have been made to improve your SQL Server development experience.

The Integrated Environment

Those who have been working with SQL Server for a long time may remember the SQL Enterprise Manager that came with SQL Server 6.5. In some respects, with SSMS, Microsoft has moved back to the paradigm that existed then. Like the SQL Server 6.5 Enterprise Manager, SSMS provides an integrated environment where developers and DBAs alike can perform the database tasks they need. Say goodbye to Query Analyzer, Analysis Manager, and a number of other desperate tools and say hello to SSMS, which provides "one-stop shopping" for most of your database needs.

Window Management

Figure 3.1 shows a sample configuration for the SSMS main display. The environment and the windows that are displayed are completely customizable, with the exception of the document window area. Figure 3.1 shows the document window area displaying the Summary page. The Summary page is the default, but other pages, such as a Query Editor window, can take the focus in this tab-oriented section of the SSMS display.

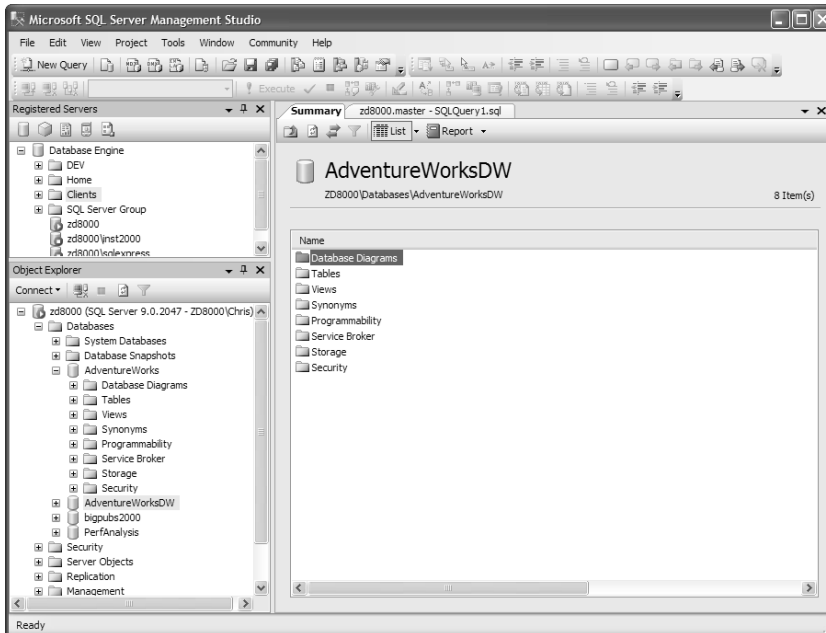


FIGURE 3.1 The SSMS main display.

The dialogs that form the rest of the SSMS display are referred to as *components* and include the Registered Servers and Object Explorer windows that are shown in Figure 3.1, as well as a number of other components that can be displayed via the View menu found at the top of the SSMS display. You can configure each of the component windows in a number of ways; for example, you can have them float, or you can hide, dock, Auto Hide, or display them as tabbed documents in the document window area.

The configuration that you choose for your SSMS display depends on the type of work you do with SQL Server as well as the type of person you are. The Auto Hide feature causes the component window to shrink to a tab along the left or right side of the display. When you mouse over the tab, the window automatically expands and stays expanded as long as the mouse cursor remains in the component window area. Auto Hide helps maximize the working real estate that is available in the document window for query development and the like. Docking many windows can clutter the screen, but it allows you to view many different types of information all at once. This is a matter of personal preference, and SSMS has made it very easy to change.

TIP

You can reposition the component windows by dragging and dropping them to the desired locations. When you are in the middle of a drag and drop, rectangular icons with arrows are displayed at different locations on the SSMS window surface. If you mouse over one of these arrowed icons to select the window location, you see the window destination highlighted. If you release your mouse button while the destination is highlighted, the window docks in that position.

Some users at first ignore the arrow icons and keep hovering the window over the location where you want the window to go. Hovering the window over the desired location does not allow you to effectively dock it. You should save yourself some time and aggravation and use the arrow icons for drag-and-drop positioning.

The other big changes to the SSMS window environment include non-modal windows that are sizable. The change to non-modal windows allows you to perform multiple tasks at once without needing to open another instance of the SSMS application. Enterprise Manager users of SQL Server 2000 were forced to open another instance of the application during many administrative tasks in order to continue with other work. With SSMS, you can launch a backup with the Back Up Database dialog box and then continue working with the Object Explorer or other components in SSMS while the backup is running. This is a great timesaver and helps improve overall productivity.

The ability to size the dialog boxes is another user-friendly change that may seem minor but is quite handy on certain windows. For example, the SQL Server 2000 Enterprise Manager Restore dialog had a fixed size. Viewing the backup set information in this relatively small (nonsizable) dialog box was a challenge. The Restore dialog in SQL Server 2005's SSMS can contain a slew of information related to the backup sets available for restore. The ability to size the windows allows for much more information to be displayed.

The tabbed document window area provides some usability improvements as well. This area, as described earlier, is fixed and is always displayed in SSMS. Component windows can be displayed in this area, along with windows for the Query Editor, diagrams, and other design windows. If desired, you can change the environment from a tabbed display to multiple-document interface (MDI) mode. In this mode, each document is opened in its own window within the document window. The MDI mode manages windows like the SQL Server 2000 Query Analyzer and may be more user-friendly for some people. You can change to MDI mode by selecting Tools, Options and then selecting MDI Environment from the General page.

One particularly useful window that can be displayed in the document window is the Summary page. This new window displays information relative to the node that is selected in the Object Explorer and includes options to produce detailed reports and graphs. The Summary page is displayed in the document window by default when SSMS is launched, but you can also display it by pressing F7 or choosing Summary from the View menu.

TIP

In SQL Server 2000, you could select multiple objects for scripting by selecting the items from the Object Explorer tree in Enterprise Manager. You cannot use the Object Explorer tree to do this with SQL Server 2005, and this has generated some confusion. The solution is the Summary page, which provides a means for doing multiple selections of the objects it displays. You can hold down the Ctrl key and click only those items that you want to script. After you have selected the items you want, you simply right-click one of the selected items and choose the preferred scripting method. This also works with scheduled jobs that are displayed in the Summary page. SQL Server 2000 did not offer this capability.

The reports that are available on the Summary page are often overlooked. Part of the reason for this may be that the reports are not available for every node in the Object Explorer tree. Top-level nodes in the tree are where most of the reports are found. For example, if you select a database in the Object Explorer tree and view the Summary page, you see a Report icon that is enabled on the toolbar at the top of the Summary page. If you click the drop-down arrow next to that icon, you find a list of reports that are available for creation. These reports include Disk Usage, Backup and Restore Events, Top Transactions by Age, and a host of others. Graphs are included with some reports, and you can export or print all these reports. Figure 3.2 shows an example of the Disk Usage report for the AdventureWorks database.

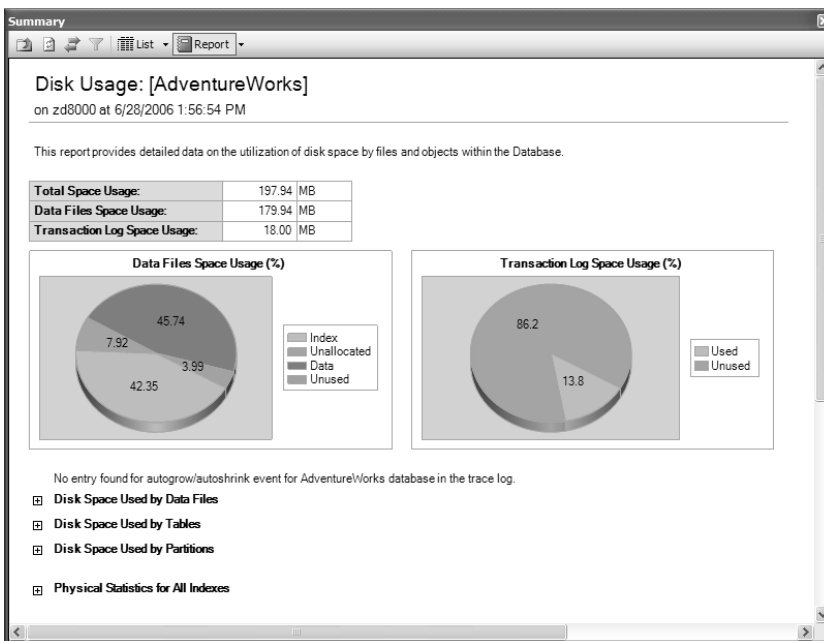


FIGURE 3.2 A Disk Usage summary report.

The graphs are easy to read, and some sections of the report can be expanded to provide more detail. Bullets at the bottom of a report are nodes that can be expanded. For example, the bullets *Disk Space Used by Data Files* and *Disk Space Used by Table* at the bottom of Figure 3.2 can be expanded.

Integrated Help

SSMS offers an expanded set of help facilities as well as improved integration into the application environment. The Help sources have been expanded to include both local and online resources. Local help is similar to the Help resources available in past versions and references files that are installed on your machine during the installation process. Local help includes the local SQL Server Books Online resources. Local help files are static and get updated only if another documentation installation is run on the local machine.

Online help is new to SQL Server 2005 and provides access to content that is not static and can be updated with the very latest changes. Three default online resources are provided by default:

- ▶ **MSDN Online**—MSDN Online contains the latest version of the MSDN documentation, including the latest quarterly releases.
- ▶ **Codezone Community**—Codezone Community includes a set of third-party websites that have partnered with Microsoft and provide a wealth of information from sources outside Microsoft.
- ▶ **Questions**—The Questions option allows you to search the forum archives for answers to questions that others have already asked. It also allows you to post your own questions.

The help resources you use on your machine are configurable. You can choose to search online resources first, followed by local help, or you can choose an option that searches local help resources first, followed by online resources. You can also choose specific Codezone online resources to search, or you can eliminate the search of all online resources. Figure 3.3 shows the online help Options window, which allows you to configure your Help options. You access this dialog by selecting *Tools, Options*.

The Help resources you select are used when you search for content within the Help facility. When you use both local and online resources options, you see results from multiple locations in your search results. Figure 3.4 shows a sample Books Online Document Explorer window with results from a search on “Management Studio.” Notice that the panel on the right side of the window lists entries under *Local Help*, *MSDN Online*, *Codezone Community*, and *Questions*. Each of these sections contains search results that you can access by simply clicking on that area. The number of search results for each section is displayed in parentheses after the section name.

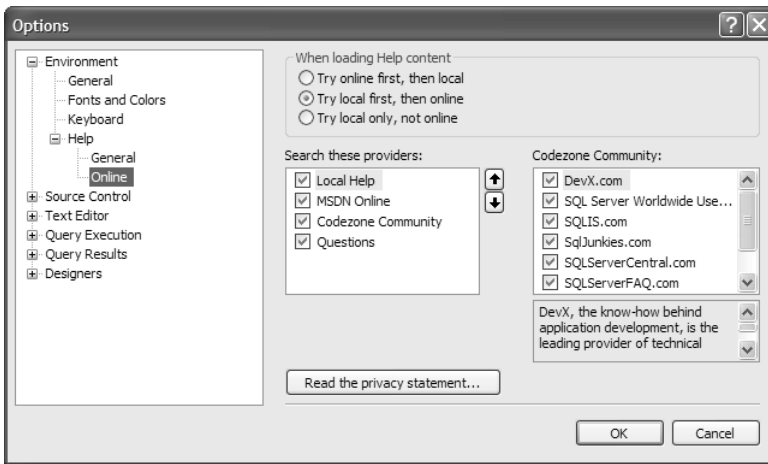


FIGURE 3.3 Setting Help options.

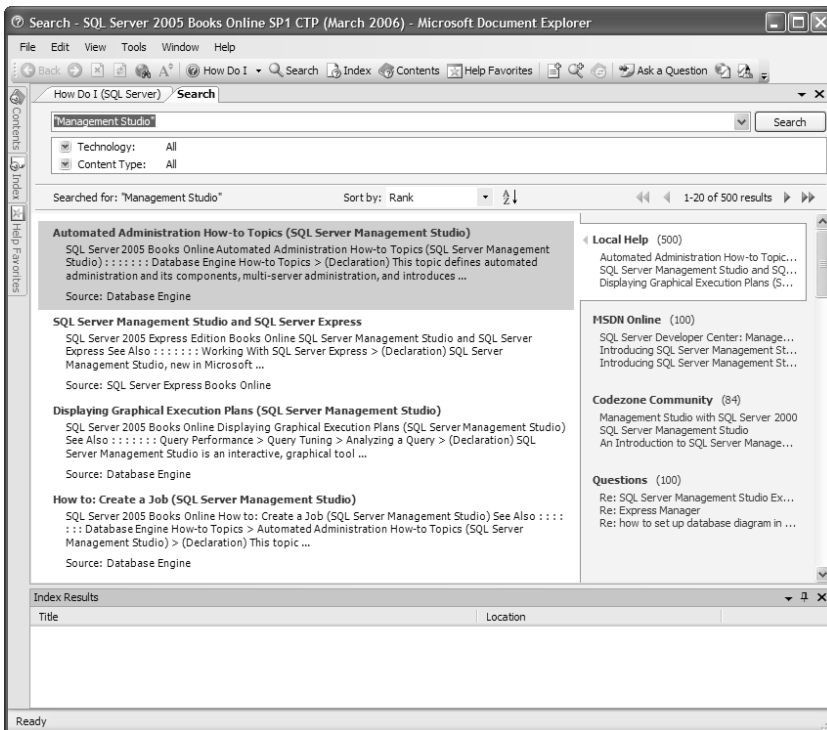


FIGURE 3.4 A Books Online search.

One other significant change to the help facilities in SSMS is the addition of Dynamic Help. Dynamic Help is a carryover from the Visual Studio environment. It is a help facility that automatically displays topics in a Help window that are related to what you are doing in SSMS. For example, if you are working in a query window and type the word `SELECT` to start your query, the Dynamic Help window displays several topics related to the `SELECT` statement. If you are working in the Object Explorer, it displays Help topics related to the Object Explorer.

Dynamic Help is one of the component windows that you can dock or position on the SSMS surface. To use Dynamic Help, you select Help, Dynamic Help. Figure 3.5 shows an example of the SSMS environment with the Dynamic Help window docked on the right side of the window. The Dynamic Help topics in this example are relative to the `SELECT` keyword that is typed in the Query Editor window in the middle of the screen.

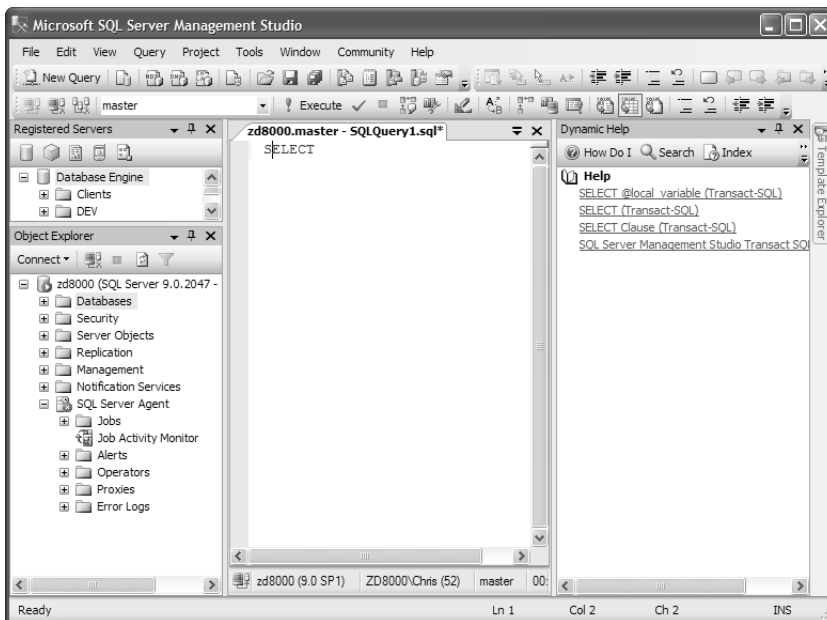


FIGURE 3.5 Dynamic Help.

Administration Tools

The tools that are available with SSMS can be broadly categorized into tools that are used for administering SQL Server and tools that are used for developing or authoring new SQL Server objects. As a matter of practice, developers use some of the administrative tools, and administrators use some of the development tools.

SSMS comes with an expanded set of tools to help with SQL Server administrative tasks. It builds on the functionality that was available in the SQL Server 2000 Enterprise Manager and adds some new tools and functionality to help ease the administrative burden.

Using Registered Servers

Registered servers is a new concept in SQL Server 2005 and represents a new division between managing servers and registering servers. With the SQL Server 2000 Enterprise Manager, the Microsoft Management Console (MMC) tree was displayed on the left side of the Enterprise Manager screen, and it contained servers that had been registered via that tree. Any registered servers or groups were listed in the tree, along with any of the associated objects.

With SQL Server 2005, registered servers are managed and displayed in the Registered Servers component window. The objects associated with these registered servers are displayed in the Object Explorer rather than in the Registered Servers window.

Figure 3.6 shows an example of the Registered Servers window, with several server groups and their associated registered servers. You can add new groups any time; this window offers a handy way of organizing the servers you work with.

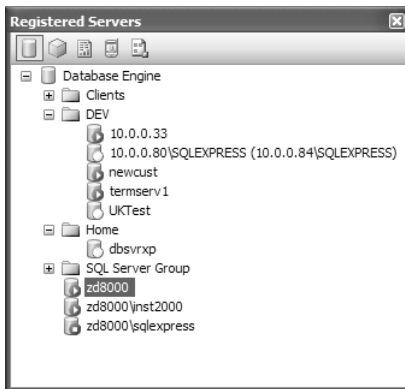


FIGURE 3.6 The Registered Servers window.

The servers listed in Figure 3.6 are all database engine servers. These server types are the conventional SQL Server instances, like those you could register in the SQL Server 2000 Enterprise Manager. You can also register several other types of servers. The icons across the top of the Registered Servers window indicate the types of servers that can be registered. In addition to database engine servers, you can also register servers for Analysis Services, Reporting Services, SQL Server Mobile, and Integration Services. The Registered Servers window gives you one consolidated location to register all the different types of servers that are available in SQL Server 2005. You simply click the icon associated with the appropriate server type, and the registered servers of that type are displayed in the Registered Servers tree.

NOTE

The SQL Server 2005 Registered Servers window enables you to register servers that are running SQL Server 2000 and SQL Server 7.0 as well. You can manage all the features of SQL Server 2000 with SQL Server 2005 tools. You can also have both sets of tools on one machine. The SQL Server 2000 and SQL Server 2005 tools are compatible and function normally together.

The SQL Server 2000 Enterprise Manager and Query Analyzer cannot be used to manage SQL Server 2005. You can connect the Query Analyzer to a SQL Server 2005 instance and run queries, but the Object Explorer and other tools are not compatible with SQL Server 2005.

When a server is registered, you have several options available for managing the server. You can right-click the server in the Registered Servers window to start or stop the related server, open a new Object Explorer window for the server, connect to a new query window, or export the registered servers to an XML file so that they can be imported on another machine.

TIP

The import/export feature can be a real timesaver, especially in environments where many SQL servers are managed. You can export all the servers and groups that are registered on one machine and save the time of registering them all on another machine. For example, you can right-click the Database Engine node, select Export, and then choose a location to store the XML output file. Then, all you need to do to register all the servers and groups on another machine is move the file to that machine and import the file.

Using Object Explorer

The Object Explorer window that existed in the SQL Server 2000 Query Analyzer has been integrated into SSMS. It has the same tree-like structure that was present in SQL Server 2000 but contains some significant improvements over its predecessor. The most significant feature for those folks managing a large number of database objects is the ability to populate the Object Explorer tree asynchronously. This may not hit home for folks who deal with smaller databases, but those who waited on the synchronous population of Object Explorer in SQL Server 2000 will be excited. The Object Explorer tree in SSMS displays immediately and allows navigation in the tree and elsewhere in SSMS while the population of the tree is taking place.

The Object Explorer is adaptive to the type of server that it is connected to. For a database engine server, the databases and objects such as tables, stored procedures, and so on are displayed in the tree. If you connect to an Integration Services server, the tree displays information about the packages that have been defined on that type of server. Figure 3.7 shows an example of the Object Explorer with several different types of SQL Server servers displayed in the tree. Each server node has a unique icon that precedes the server name, and the type of server is also displayed in parentheses following the server name.

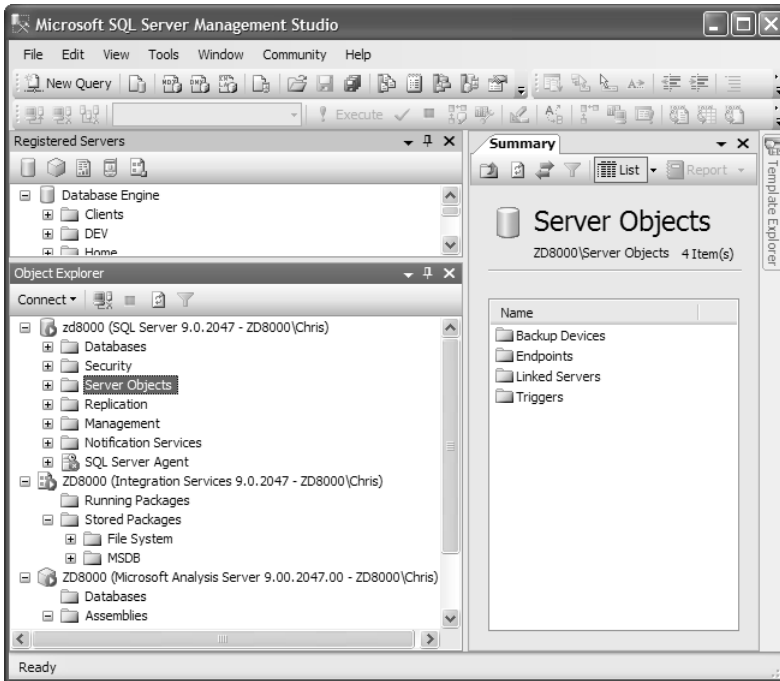


FIGURE 3.7 Multiple server types in Object Explorer.

The objects displayed in the Object Explorer tree can be filtered in SQL Server 2005. The number of filters is limited, but those that are available can be helpful. For example, you can filter the tables that are displayed in Object Explorer based on the name of the table, the schema that it belongs to, or the date on which it was created. Again, for those who deal with large databases and thousands of database objects, this feature is very helpful.

Administrators also find the enhanced scripting capabilities in the Object Explorer very useful. The scripting enhancements are centered mostly on the administrative dialog boxes. These dialogs now include a script button that allows you to see what SSMS is doing behind the scenes to effect your changes. In the past, the Profiler could be used to gather this information, but it was more time-consuming and less integrated than what is available now.

Figure 3.8 shows an example of an administrative dialog, with the scripting options selected at the top. You can script the commands to a new query window, a file, the Windows Clipboard, or a job that can be schedule to run at a later time.

Aside from these features, many of the features and much of the functionality associated with the Object Explorer is similar to what was found in SQL Server 2000. Keep in mind that there are some additional nodes in the Object Explorer tree and that some of the objects are located in different places. There is now a separate node for the SQL Server Agent that contains scheduled jobs and related objects. Linked servers are now located under the Server Objects node, and several new additions are available in the Management node that were found elsewhere in the prior version.

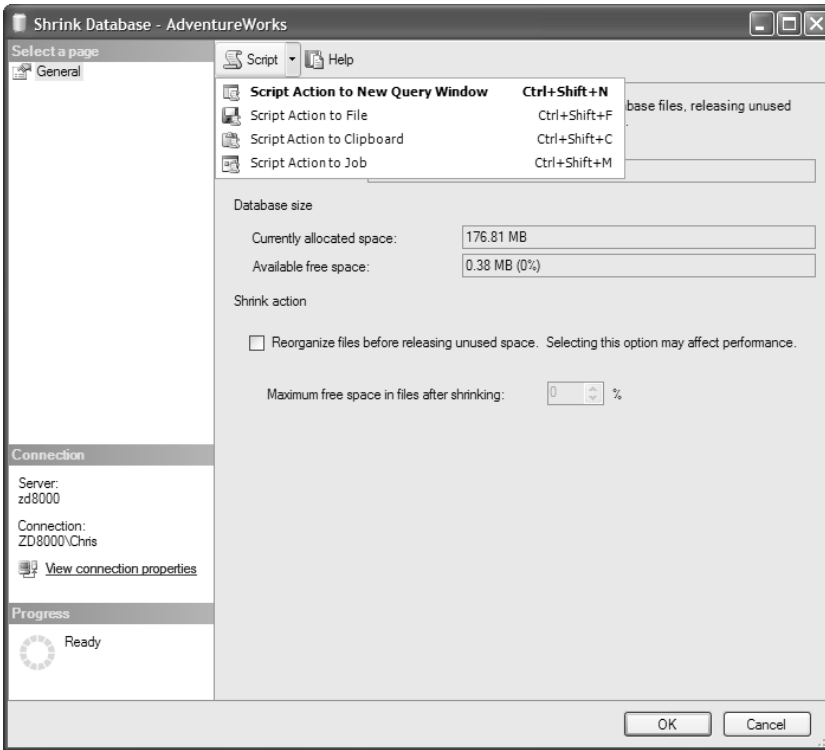


FIGURE 3.8 Scripting from administrative dialogs.

Using Activity Monitor

The functionality that was previously available in the `Current Activity` node of Enterprise Manager has been ported to a new tool named the Activity Monitor. The Activity Monitor is a non-modal window that is launched from the Object Explorer tree. To access it, you open the `Management` node of the tree and double-click the Activity Monitor node. Figure 3.9 shows an example of the Activity Monitor window.

The default `Process Info` page lists current processes on the database server. By default, the system processes are not listed, but you can add them to the display by using the filtering capabilities in the application. If you click the `Filter` button, a `Filter Settings` window like the one shown in Figure 3.10 is displayed. You can set the `Show System Processes` value to `True` to display all the processes, and you can adjust any of the other filter values to display the desired set of processes.

The other two pages in the Activity Monitor display information about locks on the server. You can display locks by server process ID (SPID) or select locks based on a specific database object. This information is similar to what is retrieved when you run the `sp_lock` system stored procedure or the `sys.dm_tran_locks` dynamic management view.

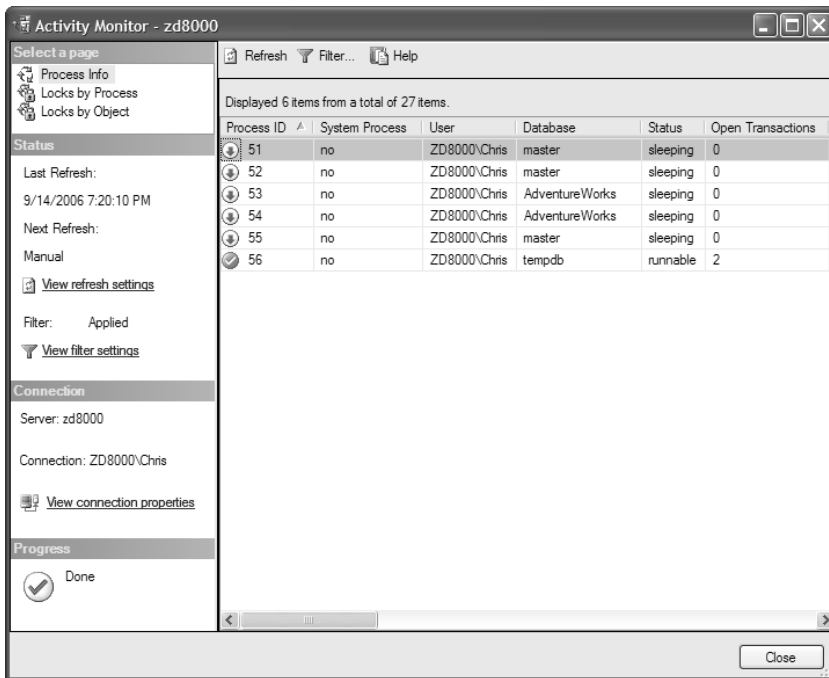


FIGURE 3.9 Process info in the Activity Monitor.

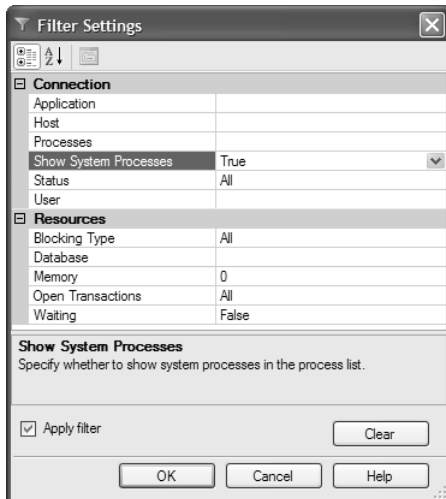


FIGURE 3.10 Filter settings in the Activity Monitor.

NOTE

Each of the pages in the Activity Monitor has an abundance of columns that display useful information. The default window is not nearly big enough to display all the information. You can use the scrollbar to navigate to the columns that are hidden. To rearrange the order of the columns, you simply drag a column header to the desired location.

One of the most impressive features of the Activity Monitor is its ability to refresh the display automatically. You can click the View Refresh settings option on the left side of the screen to adjust the refresh rate. You can select the Auto Refresh Every option and select the number of seconds between refreshes to have the screen automatically refresh.

Using Log File Viewer

The Log File Viewer is another non-modal window that is new to SQL Server 2005. Like the Activity Monitor, it houses information that was previously displayed in the document window in the SQL Server 2000 Enterprise Manager. It can display log files that are generated from several different sources, including Database Mail, SQL Server Agent, SQL Server, and Windows NT.

The Log File Viewer can be launched from the related node in the SSMS Object Explorer. For example, you can select the Management node and expand SQL Server Error Logs. If you double-click one of the error logs listed, a new Log File Viewer window is launched, displaying the SQL Server log file entries for the log type selected (see Figure 3.11).

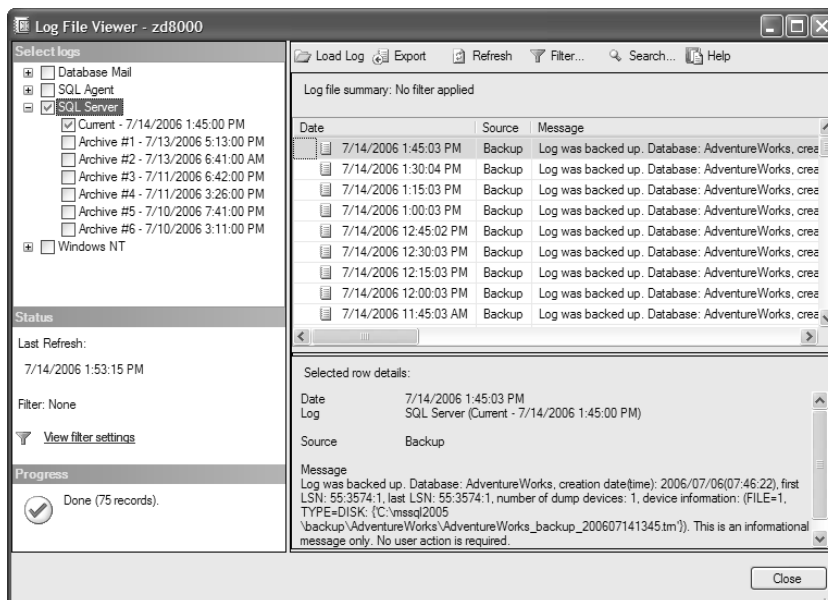


FIGURE 3.11 SQL Server logs displayed in the Log File Viewer.

NOTE

By default, entries are shown in the SQL Server Log File Viewer from newest to oldest. This is different from the default order in the SQL Server 2000 Enterprise Manager, which displayed the log file entries from oldest to newest.

One of the first things you notice when you launch the Log File Viewer is that there is a tree structure at the top-left corner of the screen that shows the log files you are viewing. You can see that there are four different log types available: Database Mail, SQL Agent, SQL Server, and Windows NT. You can choose to display multiple log files within a given log type (for example, the current SQL Server log and Archive #1) or you can select logs from different sources. For example, you can display all the current log entries for SQL Server and the current log entry for the SQL Server Agent.

When multiple logs are selected, you can differentiate between the rows shown on the right side of the Log File Viewer by looking at the Log Source column and the Log Type column. The Log Source values match up with the names that are shown in the tree structure where the log was selected. The Log Type column shows the type of log, such as SQL Agent or SQL Server. Rows from the different log types are displayed together and sorted according to the date on which the row was created. The sort order cannot be changed.

TIP

You can rearrange the order of the columns shown in the Log File Viewer. You simply click the column header and drag the column to the desired location. When viewing rows for more than one log type or multiple logs, it is best to drag the Log Type and Log Source columns to a location that is easily viewed so that you can distinguish between the entries.

Other noteworthy features in the Log File Viewer include the ability to filter and load a log from an external source. You can filter on dates, users, computers, the message text, and the source of the message. You can import log files from other machines into the view by using the Load Log facility. This works hand-in-hand with the Export option, which allows you to export the log to a file. These files can be easily shared so that others can review the files in their own Log File Viewer.

Development Tools

SSMS delivers an equally impressive number of enhancements for database developers. These new tools are based on tools such as Query Analyzer that were available in prior versions of SQL Server. They deliver the same functional value available in prior releases and offer enhancements that address some of the shortcomings.

The Query Editor

The Query Editor sits at the top of the list for new development tools in SSMS. The Query Editor, as its name indicates, is the editing tool for writing queries in SSMS. It contains

much of the functionality that was contained in SQL Server 2000's Query Analyzer. The ability to write Transact-SQL (T-SQL) queries, execute them, return results, generate execution plans, and many of the other features you may be familiar with in Query Analyzer are also available with the Query Editor.

One main difference with the Query Editor is that it has been integrated into the SSMS environment. In SQL Server 2000, the Query Analyzer was a separate application with its own independent interface. In SQL Server 2005, SSMS houses the query-editing capabilities along with all the administrative capabilities.

NOTE

The biggest upside to the integration of the query-editing tool into the SSMS environment is that you can find almost anything you need to administer or develop on your SQL Server database in one spot. There is no need to jump back and forth between applications. One possible downside, however, is that SSMS may be much more than some database developers need.

Clicking the New Query button, opening a file, and selecting the Script to File option from a list of database objects in the Object Explorer are just a few of the ways to launch the Query Editor. Figure 3.12 shows the Query Editor window with a sample SELECT statement from the AdventureWorks database. The figure shows the Query Editor window displayed on the right side of the screen and the Object Explorer on the left side.

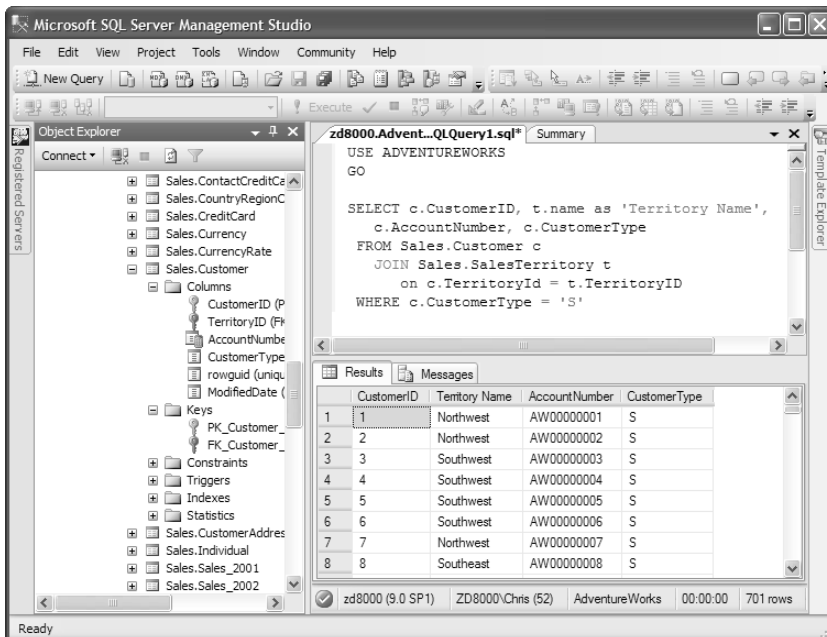


FIGURE 3.12 The Query Editor window in SSMS.

The basic editing environment within the Query Editor is similar to Query Analyzer. The top portion of the Query Editor window contains the query. The bottom portion contains the results of an executed query. The results can be displayed as text, displayed in a grid format, or output as XML. However, in the Query Editor, windows are by default managed differently than with Query Analyzer. Multiple Query Editor windows are displayed in a tabbed format; in comparison, Query Analyzer displayed a separate window for each query.

TIP

The tabbed document display has some advantages, but you can set an option in SSMS that causes the Query Editor to behave much like the Query Analyzer. To do this, you select Tools, Options to launch the Options dialog. The default page has a section named Environmental Layout. If you choose the MDI Environment option, you set SSMS in MDI mode instead of the tabbed layout.

Query Editor Types

The Query Editor in SQL Server 2005 enables you to develop different types of queries. You are no longer limited to database queries based on SQL. You can use the Query Editor to develop all types of SQL Server Scripts, including those for SQL Server Analysis Services (SSAS) and SQL Server Mobile Edition. The SSAS queries come in three different flavors: multidimensional expressions (MDX), data mining expressions (DMX), and XML for analysis (XMLA). Only one selection exists for creating SQL Server Mobile Edition scripts.

You see these new query options when you create a new query. When you select New from the SSMS menu, you can choose what type of query to create. You use the Database Engine Query choice to create a T-SQL query against the database engine. The other new query options correspond to SSAS and SQL Server Mobile Edition. The toolbar on SSMS also has icons that correspond to each type of query that can be created.

Each query type has a code pane that works much the same way across all the different types of queries. The code pane, which is the topmost window, color-codes the syntax that is entered, and it has sophisticated search capabilities and other advanced editing features that make it easy to use. The features that are new to SQL Server 2005 and apply to all the editor types include line numbering, bookmarks, hyperlinks in the comments, and a color-coded indicator that is shown in front of each line that has changed since the script was opened.

Other code pane features are available only for certain types of queries. IntelliSense, which automatically completes syntax and arguments, is available for all queries except database engine queries. Squiggles, which are wavy lines that appear below a word in the editor to indicate possible syntax errors, are available with MDX, DMX, and XML queries. The MDX, DMX, and XML editors also offer code outlining, which enables you to expand and collapse code segments to make it easier to review code.

Disconnected Editing

New to SQL Server 2005 is the ability to use the code editor without a database connection. When creating a new query, you can choose to connect to a database or select Cancel to leave the code pane disconnected. To connect to the database at a later time, you can right-click in the code pane window and select the Connect option. You can also disconnect the Query Editor at any time or choose the Change Connection option to disconnect and connect to another database all at once.

Along with disconnected editing are some changes to the Windows behavior that are worth noting. The biggest changes relate to the behavior of query windows that are currently open at the time that a file is opened for editing. With SQL Server 2000 Query Analyzer, the currently selected window would be populated with the contents of the file that you were opening. Prior to this replacement, a prompt would be displayed that asked whether you wanted to save your results. If the query window was empty, the contents would be replaced without the prompt for saving.

With SQL Server 2005, a new query window is opened every time a new file is opened. The new window approach is faster but can lead to many more open windows in the document window. You need to be careful about the number of windows/connections you have open. Also, you need to be aware that the tabbed display shows only a limited number of windows. Additional connections can exist even if their tabs are not in the active portion of the document window.

Editing SQLCMD Scripts in SSMS

SQLCMD is a command-line utility that is new to SQL Server 2005. You can use it for ad hoc interactive execution of T-SQL statements and scripts. It is basically a replacement for the ISQL and OSQL commands that were used in prior versions of SQL Server. (OSQL still works with SQL Server 2005, but ISQL has been discontinued.)

What's new to SSMS is the ability to write, edit, and execute SQLCMD scripts within the Query Editor environment. The Query Editor in SSMS treats SQLCMD scripts in much the same way as other scripts. The script is color-coded and can be parsed or executed. This is possible only if you place the Query Editor in SQLCMD mode, which you do by selecting Query, SQLCMD Mode or selecting the SQLCMD mode icon from the SSMS toolbar.

Figure 3.13 shows a sample SQLCMD script in SSMS that can be used to back up a database. This example illustrates the power and diversity of a SQLCMD script that utilizes both T-SQL and SQLCMD statements. It uses environment variables that are set within the script. The script variables DBNAME and BACKUPPATH are defined at the top of the script with the SETVAR command. The BACKUP statement at the bottom of the script references these variables, using the convention $\$(variablename)$, which substitutes the value in the command.

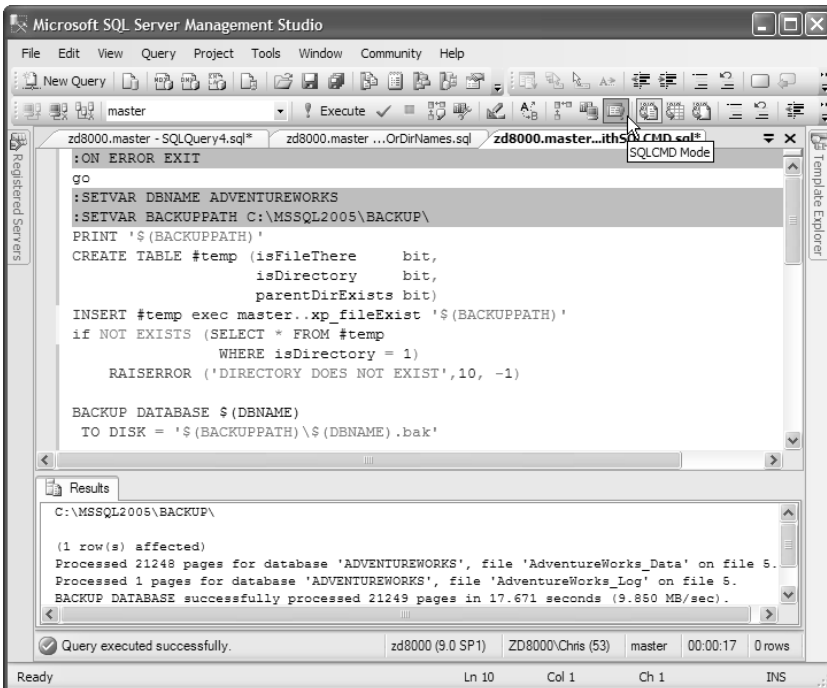


FIGURE 3.13 Editing a SQLCMD script in SSMS.

SQLCMD scripts that are edited in SSMS can also be executed within SSMS. The results are displayed in the results window of the Query Editor window, just like any other script. After you test a script, you can execute it by using the SQLCMD command-line utility. The SQLCMD command-line utility is a very powerful tool that can help automate script execution. For more information on using SQLCMD in SSMS, refer to the Books Online topic “Editing SQLCMD Scripts with Query Editor.” The SQLCMD command-line utility is discussed in more detail in Chapter 4, “SQL Server Command-Line Utilities.”

Regular Expressions and Wildcards in SSMS

SSMS has a robust search facility that includes the use of regular expressions. Regular expressions provide a flexible notation for finding and replacing text, based on patterns within the text. Regular expressions are found in other programming languages and applications, including the Microsoft .NET Framework. The regular expressions in SSMS work in much the same way as these other languages, but there are some differences in the notation.

The option to use regular expressions is available whenever you are doing a find or replace within an SSMS script. You can use the find and replace option in the code pane or the results window. You can use the Find and Replace option from the Edit menu or use press either the Ctrl+F or Ctrl+H shortcut keys to launch the Find and Replace dialog

box. Figure 3.14 shows an example of the Find and Replace dialog that utilizes a regular expression. This example is searching for the text `Customer`, preceded by the `@` character and not followed by the `Id` characters. This kind of search could be useful for searching a large stored procedure where you want to find the customer references but don't want to see the variables that contain *customer* in the first part of the variable name.

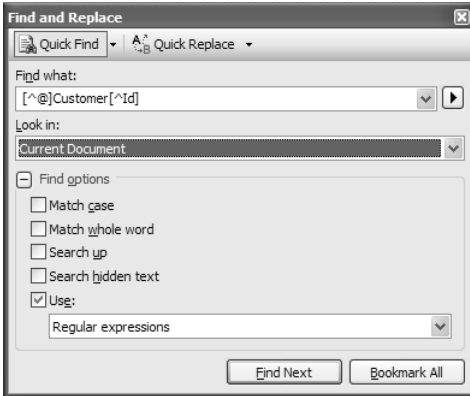


FIGURE 3.14 A find and replace with regular expressions.

You use regular expressions only when the Use check box in the Find and Replace dialog is selected. When this option is selected, you can choose either Regular Expressions or Wildcards. Wildcard searches work much the same way in SSMS as they do in file searches. For example, if you wanted to find any references to the word *zip*, you could enter `*zip*` in the Find What text box. The wildcard options are limited but very effective for simple searches.

Regular expressions have a much more extensive number of available search options. When you choose the option to use regular expressions, the arrow button is enabled to the right of the text box where you enter your search text. If you click this button, you are given an abbreviated list of regular expression characters that you can use in your searches. A brief description of what each character represents in the search is listed next to the character. For a complete list of characters, you can choose the Complete Character List option at the bottom of the list. This option brings you to the Books Online topic “How to: Search with Regular Expressions,” which gives a comprehensive review of all the characters.

Enhanced Performance Output

The Query Editor in SSMS has an extended set of options available for capturing and distributing performance-related data. It contains many of the familiar performance features that you may have grown accustomed to in SQL Server 2000 Query Analyzer—plus more.

Changes in the collection of performance data include a new Execution Plan tab that is displayed in the results window, along with the Results and Messages tab. The Execution Plan tab can be populated with two different types of plans: estimated plans and actual plans. The actual execution plan is a new display for SQL Server 2005; it shows the plan that was used in generating the actual query results. The actual plan is generated along with the results when the Include Actual Execution Plan option is selected. This option can be selected from the SSMS toolbar or from the Query menu. Figure 3.15 shows an example of an actual execution plan generated for a query against the AdventureWorks database. It uses the familiar treelike structure that was also present in SQL Server 2000, but the display has been enhanced for SQL Server 2005. The ToolTips that are displayed when you mouse over a node in the execution plan include additional information; you can see that information in a more static form in the Properties window if you right-click the node and select Properties. The icons in the graphical plan have changed, and the display is generally easier to read in SQL Server 2005.

Microsoft SQL Server Management Studio

File Edit View Query Project Tools Window Community Help

AdventureWorks

zdB000.Advent...QLQuery8.sql* Table - Product...BillOfMaterials zdB000.Advent...QLQuery7.sql*

```

SELECT e.employeeID, e.gender, *
FROM HumanResources.Employee e
JOIN HumanResources.EmployeePayHistory p
  on e.employeeId = p.employeeId
where e.employeeId = 197

```

Results Messages Execution plan

Query 1: Query cost (relative to the batch): 100%

```

SELECT e.employeeID, e.gender, * FROM HumanResources.Employee e JOIN HumanResources.Employee...

```

SELECT

Cost: 0

Nested Loops (Inner Join) Cost: 0

Clustered Index Seek [AdventureWorks].[HumanResources].[...] Cost: 50

Clustered Index Seek [AdventureWorks].[HumanResour] Cost: 50

Clustered Index Seek

Scanning a particular range of rows from a clustered index.

Physical Operation	Clustered Index Seek
Logical Operation	Clustered Index Seek
Actual Number of Rows	1
Estimated I/O Cost	0.003125
Estimated CPU Cost	0.0001581
Estimated Operator Cost	0.0032831 (50%)
Estimated Subtree Cost	0.0032831
Estimated Number of Rows	1
Estimated Row Size	403 B
Actual Rebinds	0
Actual Rewinds	0
Ordered	True
Node ID	1

Object

[AdventureWorks].[HumanResources].[Employee]
[PK_Employee_EmployeeID] [e]

Seek Predicates

Prefix: [AdventureWorks].[HumanResources].[Employee].EmployeeID = (197)

Query executed successfully. zdB000 (9.0 SP1) ZD

Ready

FIGURE 3.15 Displaying an actual execution plan in Query Editor.

NOTE

The Manage Indexes and Manage Statistics options that were available in the SQL Server 2000 Query Analyzer are not present in the Query Editor in SQL Server 2005. Those options in Query Analyzer were accessible by right-clicking a node in the query plan. You can use the Database Engine Tuning Advisor (DTA) in SQL Server 2005 to analyze the Query Editor statements or open the Table Designer to manage the indexes on a specific table.

Query plans generated in the Query Editor are easier to distribute in SQL Server 2005. You have several options for capturing query plan output so that you can save it or send it to someone else for analysis. If you right-click an empty section of the Execution Plan window, you can select the Save Execution Plan As option, which allows you to save the execution plan to a file. By default, the file has the extension `.sqlplan`. This file can be opened using SSMS on another machine to display the graphical output.

The query plan can also be output in XML format and distributed in this form. You make this happen by using the `SET SHOWPLAN_XML ON` option. This option generates the estimated execution plan in a well-defined XML document. The best way to do this is to turn off the display of the actual execution plan and execute the `SET SHOWPLAN_XML ON` statement in the code pane window. Next, you set the Query Editor to return results in grid format and then execute the statements for which you want to generate a query plan. If you double-click the grid results, they are displayed in the SSMS XML editor. You can also save the results to a file. If you save the file with the `.sqlplan` extension, the file will display the graphical plan when opened in SSMS.

Using the Query Designer in the Query Editor

A graphical design tool is now accessible from the Query Editor window where you write your queries. This is a great option that was missing in prior versions of SQL Server. With SQL Server 2000, you could access a graphical query designer by opening a table in Enterprise Manager and selecting Query, but this option was disconnected from the Query Analyzer environment, where the queries were authored.

With SQL Server 2005, you can right-click in the Query Editor window and choose Design Query in Editor. A dialog box appears, allowing you to add tables to the graphical query designer surface. The tables that are selected are shown in a window that allows you to select the columns you want to retrieve. Columns that are selected appear in a `SELECT` statement that is displayed at the bottom of the Query Designer window. Figure 3.16 shows an example of the Query Designer window that contains two tables from the AdventureWorks database. The two tables selected in this figure are related, as indicated by the line between them.

The T-SQL statements are generated automatically as you select various options on the Query Designer screen. If you select Sort Type, an `ORDER BY` clause is added. If you choose an alias for a column, it is reflected in the T-SQL. If tables are related, the appropriate joins are generated.

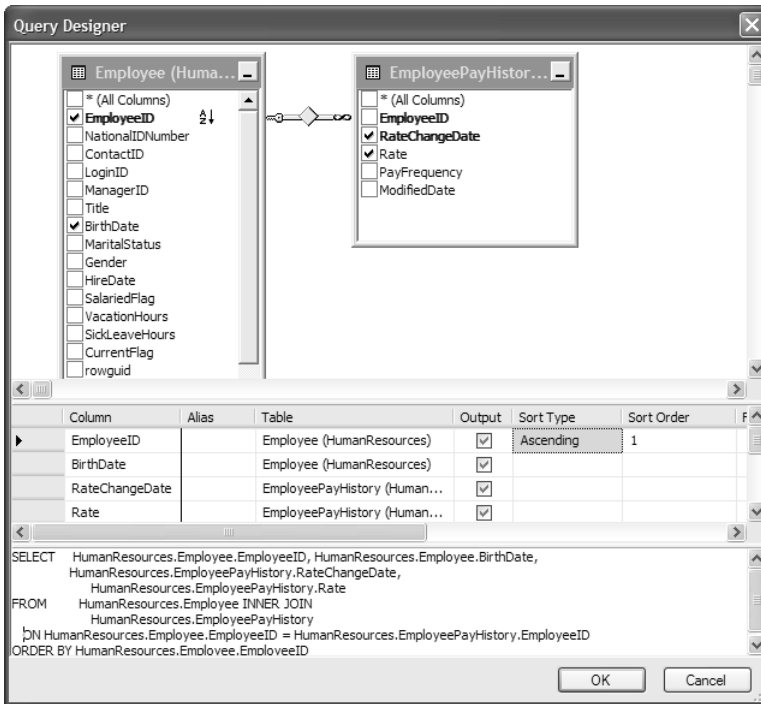


FIGURE 3.16 Designing queries in the Query Editor.

When you click OK on the Query Designer window, the related T-SQL is automatically placed in the Query Editor window. You can edit the T-SQL as needed or use it as is. You can imagine the time savings you can achieve by using this tool.

TIP

The Query Designer has a very impressive feature that allows you to view a T-SQL query visually. If you copy a valid T-SQL statement, open the Query Designer, and paste the T-SQL into the SQL pane at the bottom of the Query Designer, it tries to resolve the T-SQL into a graphical display. The tables in the FROM clause are shown in the designer panel, and information related to the selected columns is listed as well. The Query Designer cannot resolve all T-SQL statements and may fail to generate a visual display for some complex T-SQL.

Managing Projects in SSMS

Project management capabilities like those available in Visual Studio are now available in SSMS. Queries, connections, and other files that are related can be grouped into projects. A project or set of projects is further organized or grouped as a solution. This type of organization is the same as in the Visual Studio environment.

Projects and solutions are maintained and displayed with the Solution Explorer. The Solution Explorer contains a tree-like structure that organizes the projects and files in the solution. It is a component window within SSMS that you launch by selecting View, Solution Explorer. Figure 3.17 shows an example of the Solution Explorer. The solution in this example is named `EmployeeUpgrade`, and it contains two projects, named `Phase1` and `Phase2`. Each project contains a set of connections, a set of T-SQL scripts, and a set of miscellaneous files.

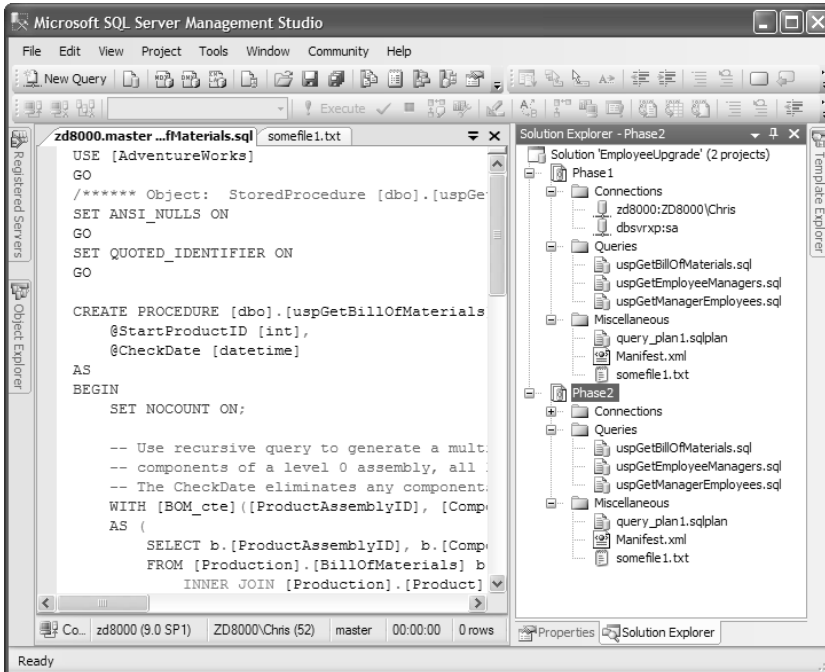


FIGURE 3.17 Solutions and projects listed in the Solution Explorer.

The first thing to do when using the project management capabilities in SSMS is to add a project. To do this, you select File, New, and when the New dialog appears, you select project to add a new project. It is a bit odd, but you must create the project before you can create the solution. When adding the new project, you are given a choice of the type of project, and you must select either SQL Server Scripts, Analysis Services Scripts, or SQL Mobile Scripts. Each one of these project types is geared toward the respective SQL Server technology.

After the project is added, you can add the related connections and files. To add a new connection, you simply right-click the Connections node. The Connections entries allow you to store SQL Server connection information that relates to the project you are working on. For example, you could have a connection to your test environment and

another connection to the production environment that relates to the project. When a connection is included in the project, you can double-click it, and a new query window for that connection is established.

SQL script files are added to a project in a similar fashion to connections: You right-click the **Queries** node and select the **New Query** option. A new Query Editor window appears, allowing you to enter the T-SQL commands. Any T-SQL script is viable for this category, including those that relate to database objects such as stored procedures, triggers, and tables.

You can also add existing files to a project. To do this, you right-click the project node, select **Add**, and then select **Existing Item**. The file types listed in the drop-down at the bottom of the **Add Existing Item** dialog include SQL Server files (*.sql), SQL deadlock files (*.xd1), XML files (*.xml), and execution plan files (*.sqlplan). SQL Server files are added, by default, to the **Queries** node. All the other file types are added to the **Miscellaneous** node. The connection entries are not stored in a separate file but are contained in the project file itself.

Integrating SSMS with Source Control

SSMS has the capability to integrate database project files into a source control solution. Source control provides a means for protecting and managing files. Source control applications typically contain features that allow you to track changes to files, control and track who uses the files, and provide a means for tagging the files with a version stamp so that the files can be retrieved at a later time, by version.

SSMS can integrate with a number of different source control applications. Visual SourceSafe is Microsoft's basic source control solution, but other source control applications can be used instead. The source control client application must be installed on the machine on which SSMS is running. When this is complete, you can set the source control application that SSMS will use within SSMS. To do this, you select **Tools, Options** and navigate to the **Source Control** node. The available source control clients are listed in the **Current Source Control Plug-in** drop-down.

The link between SSMS and the source control application is the database solution. After a solution has been created, it can be added to source control. To add a solution to a source control application, you open the **Solution Explorer** and right-click the solution or any of the projects in the solution. You then see the **Add Solution to Source Control** option. You must then log in to the source control application and select a source control project to add the solution to.

When the solution has been added to a source control application, all the related projects and project files are added as well. The projects and files that are in the source control application have additional options available in the **Solution Explorer**. Figure 3.18 shows a sample solution that has been added to a source control application. A subset of the source control options that are available when right-clicking project files are shown in this figure as well.

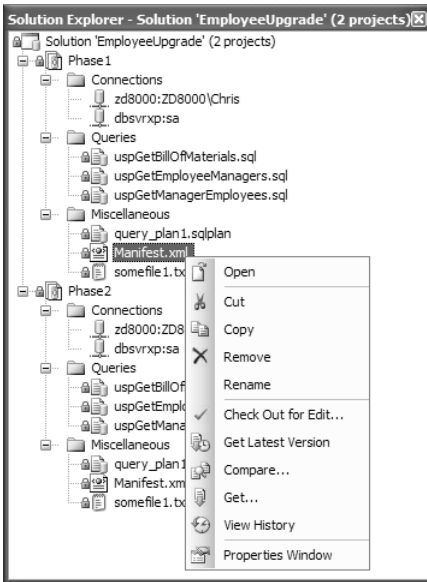


FIGURE 3.18 Source control options in the Solution Explorer.

The options that relate to source control are listed toward the bottom of the options list. The options that are available depend on the status of the selected file. For example, if a file has been checked out, additional options are displayed that relate to checking the file back in. The following are some of the common source control options:

- ▶ **Check Out for Edit**—This option allows you get a copy of the file from the source control application so that you can modify the file. When you check out the file, the source control provider can keep track of the user who has checked out the file, and it can also prevent other users from checking the file out.
- ▶ **Check In**—This option copies the locally modified file into the source control solution. The file must first be checked out for editing before you can use the Check In option. A new version for the file is established, and any prior versions of the file are retained as well.
- ▶ **Get Latest Version**—This option gets a read-only copy of the latest version of the project file from the source control application. The file is not checked out with this option.
- ▶ **Compare**—This option enables you to compare version of source control files. The default comparison that is shown is between the file in the source control application and the local file on your machine.
- ▶ **Get**—This option is similar to the Get Latest Version option, but it retrieves a read-only copy of the file. With this option, a dialog box appears, allowing you to select the file(s) that you want to retrieve.

- ▶ **View History**—This option lists all versions of the files that have been checked into the source control application. The History dialog box has many options that you can use with the different versions of the file. You can view differences between versions of the files, view the contents of a specific version, generate reports, or get an older version of the file.
- ▶ **Undo Checkout**—This option changes the checkout status in the source control application and releases the file to other source control users. Any changes that were made to the local copy of the file are not added to the source control version.

Other source control options are available via the Source Control menu in SSMS. You select an item in the Solution Explorer and then select File, Source Control. You can use this menu to check the status of a file by using the SourceSafe Properties option, set source control properties, launch the source control application, and perform other source control operations.

Using SSMS Templates

Templates provide a framework for the creation of database objects in SSMS. They are essentially boilerplate files that help generate scripts for common database objects. They can speed up the development of these scripts and help enforce consistency in the generation of the underlying database objects.

SQL Server 2005 has expanded the features available for generating templates. One substantial change is the addition of the Template Explorer. The Template Explorer is a component window that is available in SSMS and replaces the Template tab that was available in the SQL Server 2000 Query Analyzer. Figure 3.19 shows the Template Explorer and the available SQL Server template folders. Separate templates also exist for Analysis Services and SQL Server Mobile Edition. You can view them by selecting the related icon at the top of the Template Explorer.

You access the available templates by expanding the template folder in the Template Explorer tree. For example, if you expand the Index folder, you see six different types of index templates. If you double-click one of the templates, a new Query Editor window appears, populated with the template script. Figure 3.20 shows the template script that is displayed when you open the Create Index Basic template.

The template script contains template parameters that have the following format within the script:

```
<parameter_name, data_type, value>.
```

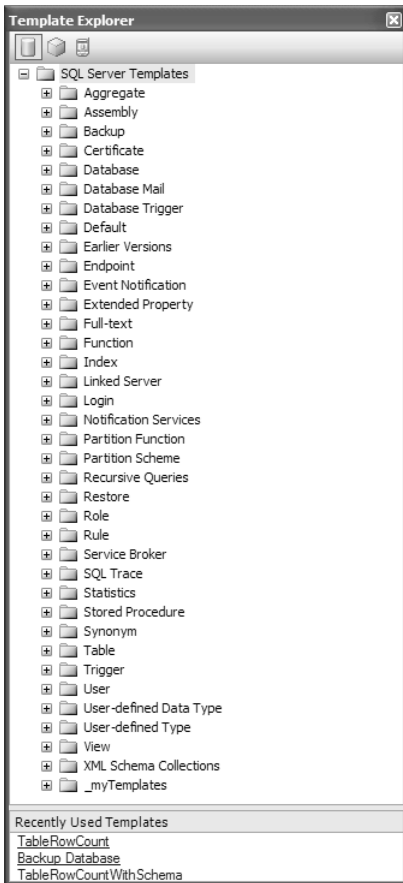


FIGURE 3.19 The SSMS Template Explorer.

You can manually replace these parameters in the script, or you can use the Specify Values for Template Parameters option from the Query menu to globally replace the parameters in the script with the desired values. Selecting Query, Specify Values for Template Parameters launches the Specify Values for Template Parameters dialog box, which enables you to enter the parameter values (see Figure 3.21).

TIP

When you use the Specify Values for Template Parameters option, some parameters may be missed if the parameter text has been altered. For example, if you add a carriage return after *parameter_name*, the Parameters dialog box does not list that parameter. It is best to leave the template script unchanged before you specify values for the parameters. You should make changes to the script after the values have been specified.

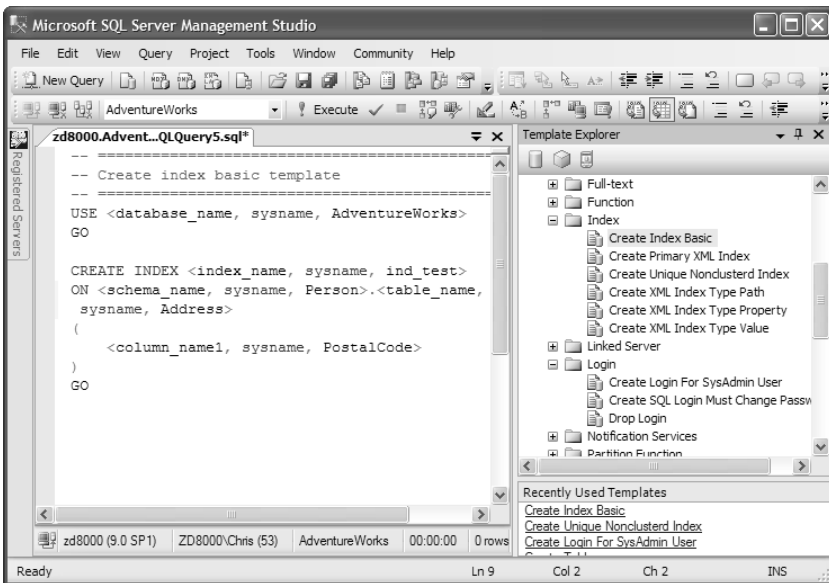


FIGURE 3.20 The template script for creating a basic index.

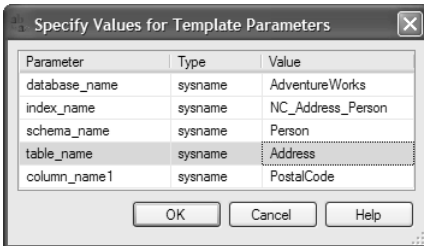


FIGURE 3.21 The Specify Values for Template Parameters dialog box.

After you have entered the parameter values and clicked OK, the values are reflected in the script. For example, the values shown in Figure 3.21 for the basic index template result in the following script:

```
-- =====
-- Create index basic template
-- =====

USE AdventureWorks
GO
CREATE INDEX NC_Address_Person
ON Person.Address
(
  PostalCode
)
GO
```

You also have the option of creating your own custom templates. These templates can contain parameters just like those that are available with the default templates. You can also create your own template folder that will be displayed in the Template Explorer tree. To create a new template folder, you right-click the SQL Server Templates node in the Template Explorer tree and select New, Folder. A new folder appears in the tree, and you can specify a new folder name. Figure 3.22 shows the Template Explorer with a set of custom templates found under the `_mytemplates` folder. The code pane in this figure shows the contents of a new custom template named `sys.objectSelectWithParameters`. This custom template contains two parameter declarations: `object_type` and `modify_date`. When you select the Specify Values for Template Parameters options for this custom template, you have the opportunity to change the values, just as you can with the default templates.

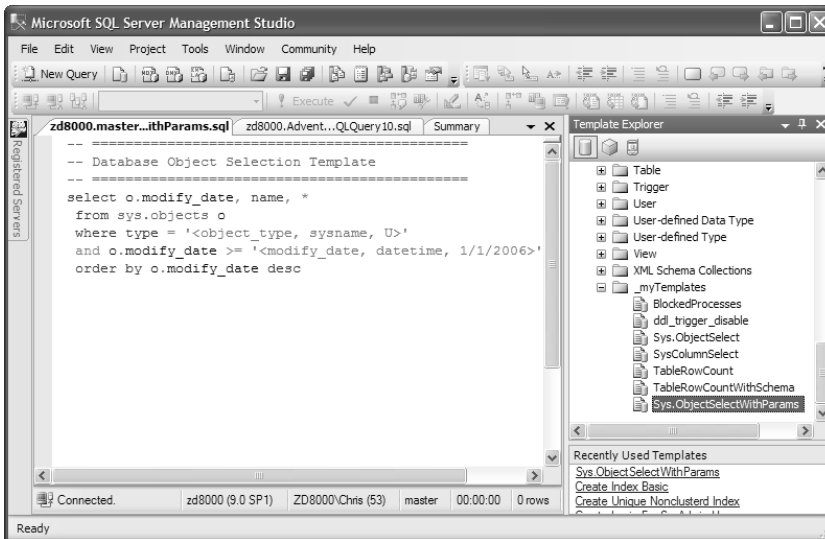


FIGURE 3.22 A custom template example.

NOTE

When you double-click a template in the Template Explorer tree, you create a script that is based on the template. Changes made to the script do not affect the template; they affect only the script that was generated from the template. To change the actual template, you need to right-click the template and select Edit. When you have completed your changes, you need to make sure to save the template.

Also, keep in mind that there is no requirement to have parameters in your templates. Templates are handy tools for accessing any code snippet you might use. After the code snippet is added as a template, you can open a new Query Editor window based on the template or simply drag and drop the template from the Template Explorer to an existing Query Editor window, and the code for the template is pasted into the window.

Summary

The number of tools and features available in SSMS is extensive and can be daunting when you first enter the environment. Remember that you can customize this environment and hide many of the windows that are displayed. You can start with a fairly simple SSMS configuration that includes the Object Explorer and a Query Editor window. This allows you to accomplish a majority of your SQL Server tasks. As you become more familiar with the environment, you can introduce new tools and features to help improve your overall productivity.

The discussion of SSMS does not end with this chapter. Further details related to SSMS are covered throughout this book. You can use the new features described in this chapter as a starting point and look to other chapters for more detailed discussion of database features that are accessible through SSMS.

Chapter 4 looks at the SQL Server utilities that can be run from the command prompt. These tools allow you to perform some of the same tasks that are available in SSMS. The ability to launch these utilities from the command line can be useful when you're automating tasks or accessing SQL Server when graphical user interface tools such as SSMS are not available.