Microsoft SQL Server 2005

Managing SQL Server Express with SQL Server 2005 Management Studio Express Edition

SQL Server Technical Article

Writers: Eric Brown, Senior Consultant Quilogy Inc.

Published: March 2006

Applies To: SQL Server 2005

Summary: Learn how to manage SQL Server 2005 Express Edition by using the free graphical management tool, SQL Server 2005 Management Studio Express Edition (SSMSE). Developers and administrators will learn how to use SSMSE features to simplify, automate, and reduce the complexity of database support and administration.

Prerequisites:

SQL Server 2005 Express Edition

SQL Server 2005 Management Studio Express Edition November CTP

Copyright

Table of Contents

An Introduction to SQL Server Management Studio Express Edition

SQL Server 2005 Management Studio Express Edition (SSMSE) provides the database developer and administrator with a robust set of tools for working with Microsoft® SQL Server™ Express Edition. Based on the same technology and functionality as that in SQL Server Management Studio, SSMSE uses Microsoft Visual Studio® .NET 2005 user interfaces and IDE layout, so that switching between SSMSE and Visual Studio .NET 2005 easy. This familiar interface allows .NET developers to be productive more quickly.

The features in SSMSE are limited to those that are needed to manage a SQL Server 2005 Express Edition database. While you can use SSMSE to manage other editions of SQL Server 2005, you cannot manage components of the server that are not in SQL Server Express. For example, you can connect only to the relational database engine only by using SSMSE because other services are not installed with SQL Server Express.

SSMSE and SQL Server Management Studio (SSMS) cannot be installed together. If multiple editions of SQL Server are installed on your computer, you can only use SSMS and not SSMSE.

Getting started

The next sections provide a tour of the features in SSMSE. Because of its graphical user interface, many SSMSE features can be accessed simply by right-clicking. You can accomplish many tasks by using both script and dialog boxes. The beginning developer can use dialog boxes to accomplish basic tasks.

By design, SQL Server always creates a default state that is secure and functional. Changes to default settings are for the purpose of setting up the database for the unique needs of users and applications.

Assuming that SQL Server 2005 Management Studio Express Edition is installed, let’s start by connecting to an instance.

Connecting to servers

This section covers how to connect to an instance of SQL Server Express. The Connect to Server dialog box allows users to provide both logon credentials and specific connection properties. You can use it to connect directly to SQL Server Express.

For the authentication method, you can choose either SQL Server Authentication or Windows Authentication. By default, this is set to Windows Authentication.

There are two tabs on the Connect to Server dialog box: the **Login** tab and the **Connection Properties** tab. To view the **Connection Properties** tab, click the **Options** button. Use the **Connection Properties** tab (shown in Figure 1) to input authentication credentials and server name.

Figure 1: Connection Properties tab

In the Connect to Server dialog box, you make choices about which database to connect to and the network method (TCP/IP, Named Pipes, or shared memory) to use. You can use the dialog box to encrypt your connection. The dialog box has some additional settings, such as the connection time-out and network packet size.

The default network protocol is shared memory protocol. If the database resides locally and will not receive connections over the network, this is the correct protocol to use. If you are connecting to a remote instance of SQL Server Express, change this to TCP/IP.

Typically, there is no reason to change the default packet size. If you know that your connection might take more than 15 seconds to resolve, change the connection time-out until the connection works. You can press the **Test** button and incrementally change the time-out until you connect successfully.

You can set the startup option to **Open Empty Environment,** which lets SSMSE start up faster. To do this, on the **Tools** menu, select **Options**. Select **Environment**, and then **General**. After you start up SSMSE, double-click the registered server to connect to and you will not need to interact with the connection dialog box.

Organizing your database servers

The primary means for organizing multiple database servers is the Registered Servers window. The Registered Servers window lists the SQL Server instances that are currently registered in SSMSE. Once the connection to the database server is established, you see a number of windows. You can review the set of server connections (or shortcuts to servers, if you prefer to think of it that way) under the registered server in the Registered Servers window. If the Registered Servers window is not visible, from the **Views** menu, select **Registered Servers**.

You can create a server group, which can include a list of individual registered servers. In a hosting environment, where a single database server may have multiple SQL Server Express instances installed, server groups allow you to work across servers in an efficient manner. For example, Figure 2 shows a folder called Express Servers. In that folder are two servers.

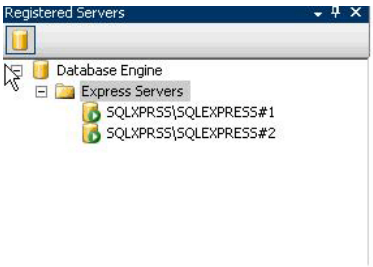

Figure 2: Registered Servers window

You can use the Registered Servers window to:

- Stop, start, pause, or restart an instance of SQL Server Express.
- Connect to a new query window.
- Connect to a new Object Explorer window.
- Open SQL Server Configuration Manager.

You can also alter the properties of the servers that are registered. To do so, right-click the server to modify, and click **Properties**.

Making the most of Object Explorer

Object Explorer is your home base for working with a database. When you click the plus symbol next to a database, only those objects that are directly related to the database are shown. The user privileges for the login that is used to connect to the database dictate which objects are available. If you are not connected initially to a database, you can connect by clicking the connection button in the upper-left corner of the Object Explorer window.

Figure 3 shows the various objects that are available for us to work with. The System Databases folder, under the Databases folder, separates system databases because SQL Server uses them to manage database processes.

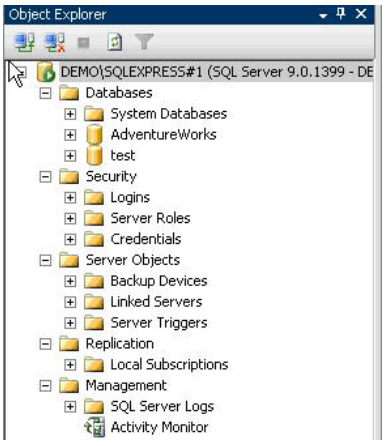

Figure 3: Object Explorer

From within Object Explorer, you can click any database icon and see another set of folders. In Figure 4, the database has its own set of database diagrams, tables, views, synonyms, and more.

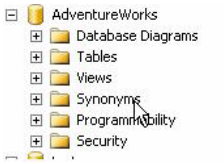

Figure 4: Database-level features

To display any objects stored in a folder, click the folder. The Programmability folder contains stored procedures, functions, database triggers, assemblies, user-defined types, rules, and defaults. The Types folder in the Programmability folder contains both system and user-provided data types. If you create a data type, you manage it from within this folder.

One of the best ways to become acquainted with SSMSE is to use it to create a database and set up a real-world administration system.

Creating a database

There are multiple ways to create a database in SQL Server 2005. My personal favorite is to right-click the Databases folder and select **New Database**. This displays the New Database dialog box. This dialog box provides an easy way to specify database settings. There are three tabs in the dialog box. The **General** tab supplies the database name text box. The **Options** tab contains settings such as auto-shrink, auto-close, cursor behavior, and recovery and state values. For most purposes, you can leave these values at their default settings. The standard values work for the most common database usage scenarios.

Figure 5 is an example New Database dialog box.

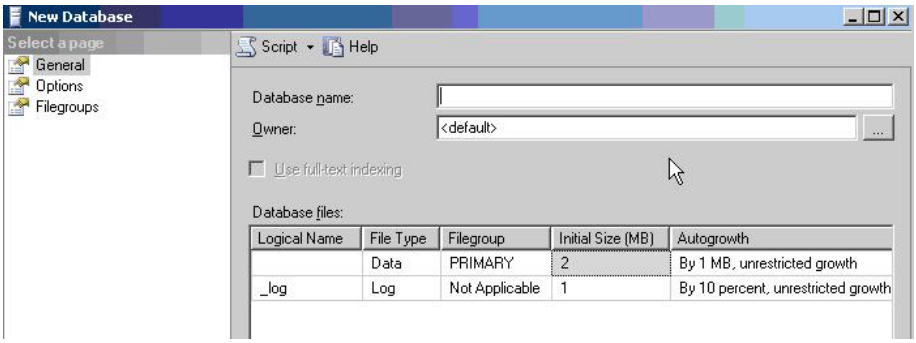

Figure 5: New Database dialog box

On the **Filegroups** tab, you provide filegroup allocations.

The New Database dialog box is nonmodal and provides the option to script all the settings to a Query Editor window, a file, or the Clipboard. To view these options, click the **Script** button at the top of the dialog box. Or, simply click **OK** and the database will be created.

Note that the default owner for a new database is the logged-in user who is creating the database. You can enter the name of another user login to be the owner of the database.

Creating tables

With our database in place, we can create tables. If you are designing a table structure from scratch, you can use either the Database Designer or Table Designer and create the database visually via a database diagram or by right-clicking on the Table folder and selecting **New Table**.

You can work with the database diagram to visually build the relationships between tables. When you save the database diagram, the tables and objects are created. Be warned—you cannot revert to a previous version when working in a database diagram.

You can also use a built-in template, accessed from Template Explorer. To view these, from the **View** menu, select **Templates**. Templates are boilerplate files containing SQL scripts that help you create objects in the database. Templates are reviewed in Using Template Explorer later in this paper.

The most direct method for creating a table is to right-click the Tables folder under your user database and select **Create table**. This starts the Table Designer. The designer provides a straightforward method for creating database objects.

As shown in Figure 6, three values are needed to create a table. These are **Column Name**, **Data Type**, and **Allow Nulls**. Name the columns by using a name that illustrates what the column contains. For example, a column containing a first name might be named **Fname**. Some developers recommend that the name should also contain information about nullability and data type.

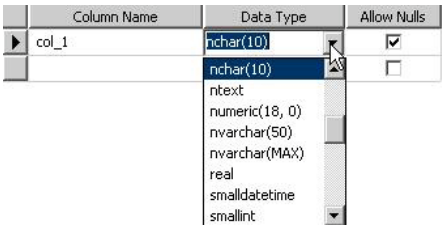

Figure 6: Table Designer

The **Allow Nulls** column specifies whether the field or column can be empty.

Column metadata other than column name, data type, and nullability are entered in the Column Properties pane at the bottom of the Table Designer window.

There are many data types available in SQL Server 2005. The most common types user are **varchar**, **integer**, **money**, and **datetime**. When designing a table, it is important to have a working knowledge of data types. Each data type stores values differently. Each data type, because of its storage mechanism, has different characteristics which affect query performance. You will find it useful to understand data types for application development. Selecting the right data type allows for greater flexibility and better quality of data.

For a complete discussion on data types, see the related topics on the Microsoft Developer Network (MSDN) Web site (http://msdn1.microsoft.com/en-us/default.aspx) and SQL Server Books Online (http://msdn2.microsoft.com/en-us/library/ms187594.aspx).

Creating a database diagram

A database diagram is a visual representation of the tables and the relationships between tables in a database. Database diagrams are useful when working with

complicated databases. I use database diagrams as a way to speed up development. I print the diagram and use it as a reference for writing Transact-SQL code.

Note that anything that can be done in the Table Designer can also be done in the Database Designer. Column metadata can be modified in the Database Designer in the same way it can be modified in the Table Designer.

To create a database diagram, navigate to the new folder called Database Diagrams under the specific database objects in Object Explorer. Right-click the folder and select **New Database Diagram**. The database diagram designer appears.

Database diagrams provide a visual basis for reading the table objects in a database. The following list summarizes what can be accomplished via the database diagram tool.

- Add existing tables. After you add a table, within the diagram, click **Add Related Tables** and the table appears in the diagram. This is helpful for discovering the relationship between tables, since the primary and foreign key relationships are used to retrieve related tables.
- Create new tables. To create a new table, right-click within the diagram and select **New table**.
- Adjust the layout of the database diagram. This includes setting the page layout, adding text comments, and hiding or displaying the relationships between tables. You can zoom in and out of the diagram, export the diagram, and copy the diagram to the Clipboard.
- Change the structure of the database. This includes adding and removing indexes, check constraints, and XML indexes.

Creating a view

You can think of a *view* as a stored version of a query. You can use a view to generate basic reports of data. For example, you have an e-commerce application that takes orders. You can use a view to access summary data such as a count of the orders or a sum of the products sold.

The easiest way to create view is to use the View Designer. To start the View Designer, right-click the Views folder and click **New View**. The View Designer opens in the Query Editor window as illustrated in Figure 7.

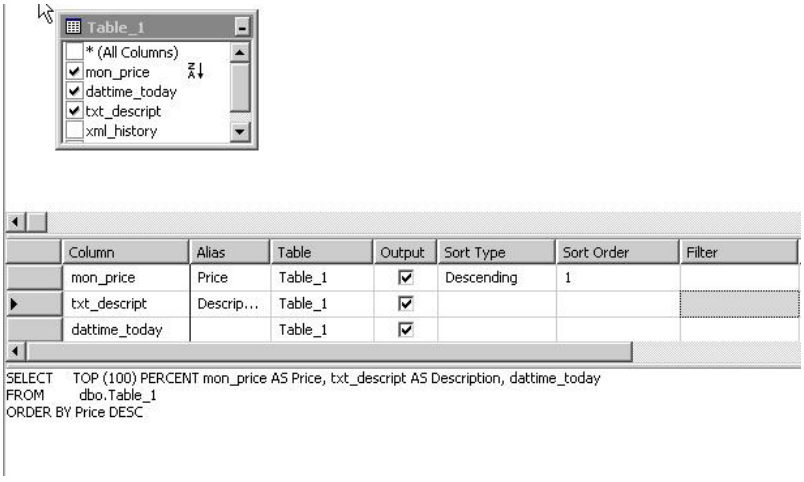

Figure 7: Creating a view with the View Designer

You will notice four panes in the View Designer:

- Diagram pane
- Criteria pane
- SQL pane
- Results pane

The top pane is the Diagram pane. Clicking the check mark adds the columns to the Column pane and to the view definition.

Within the Column pane, you can create an alias for a column. This is useful if the column name is not user friendly or you want to keep the physical column name secret. Aliases also allow code to be more legible. Aliases are also useful in queries when the tables all have synthetic primary keys. For example, you can use an alias to call the **id** column from the employee table **employee_id** and the **id** column from the department table **department_id** in your view.

The **Output** check box indicates that the column will return results to the window.

There are two types of sorting—descending and ascending. The sort order allows for multiple sorts so that you can organize complex results.

To include or exclude specific values in the results, use a filter. An example might be a filter that excludes null prices.

When you interactively design a view, it is helpful to look at the data that will be returned by the view. You can do this by clicking on the red exclamation point button to run the SQL in the view. The results are shown in the Results pane at the bottom of the designer.

Writing scripts by using the Query Editor

Up to this point, we have created a database, table, view, and diagram without writing a line of code. Now let's start writing code. The tool we use for writing code is the Query Editor. Following are a few Query Editor features.

- The Query Designer makes it easy to design a query in the Query Editor. To do this, right-click inside the text area of a query pane and select **Design Query in Editor**. This opens a Query Designer dialog box that you can use to design the text that is selected in the Query Editor. The Query Designer is a graphical tool very similar to the View Designer. You can add and remove tables and views in the Diagram pane, graphically manipulate the criteria, preview (or edit) the Transact-SQL in the SQL pane, and view the results in the Results pane.
- You can use templates to speed up the development of Transact-SQL statements when you create SQL Server objects. Templates are files that include the basic structure of the Transact-SQL statements needed to create objects in a database.
- Query results are presented in either a grid or a free-form text window.
- Showplan information is available as a query option. A showplan illustrates the logical steps built into the execution plan of a Transact-SQL statement.

To open a new query window

1. Right-click a database and select **New Query**. Or, from the **File** menu select **New Query**.
2. From the main window, click the new query button.

3. The Query Editor window is much more than a free-text tool. If you right-click in the window, you will see a number of features that you can use. Right-click anywhere in the Query Editor and you can:
   - Cut, copy, and paste text.
   - Connect, disconnect, and change a connection.
   - Open a server in Object Explorer.
   - Execute SQL statements in the window.
   - Display an estimated execution plan.
   - Design a query in the editor.
   - Include the actual execution plan in the query.
   - Include client-side statistics in the query.
   - Output query results to a text, grid or file
   - View the Properties window.
   - Change query options.

Understanding basic query syntax

Some beginners find writing queries challenging. You do not want poor organization to make writing the queries harder.

In this section we look at how you can use the Query Editor to customize the query text presentation.

The order of columns in the query is not important. In fact, you can lay out the query any way you would like to in the editor. There are, however, a number of recommended ways to lay out text in batch statements.

In short statements, it does not make sense to use comments unless you are sharing the code snippet or asking a question. Comments provide information about what the code does. A person who is not familiar with the code can read the comments to figure out what happens when the code is executed. The query processor will complain about your comments if you do not follow the correct the correct syntax. You must use the "—" symbol. Comments are applied with double en dashes. The following code example illustrates one modality for writing queries.

```
/*comment out your code—
--when you write a query—always start the batch with a USE directive like—
USE AdventureWorks
--followed by GO
GO
-- Then move onto your statement(s)*/
```

You can see how this looks in Figure 8.

```
--comment out your code--
--when you write a query--always start the batch with a USE directive like--
USE AdventureWorksDW
--followed by GO--
GO
-- Then move onto your statement(s)--
SELECT
    FirstName as 'First'
    ,[MiddleName]as 'Middle'
    ,[LastName]as 'Last'
    ,[Title]
  -- I've placed each of my table columns in on a separate line for easy reading--

FROM [AdventureWorksDW].[dbo].[DimEmployee]as AWD
--I've set up my query editor text to have a yellow foreground for SQL Keywords.
WHERE AWD.MiddleName like 'B'
Order By LastName Asc
GO
```

Figure 8: Query formatting via the Query Editor

To learn more about writing Transact-SQL, see SQL Server 2005 Express Edition User Instances on MSDN.

Using Template Explorer

You can use a template to speed up the process of writing queries.

To open Template Explorer, from the **View** menu, select **Template Explorer**. Template Explorer contains many templates for creating database objects. Templates can help you to understand Transact-SQL. Each template contains all the code needed to create a particular database object. You can change the names of items in the template and you can add your own code to the template.

In each template, you will find code that illustrates the best way to write Transact-SQL. Many of the templates are complicated. If you do not know what to do when you start using a template, simply click the change parameters button on the main window as illustrated in Figure 9.

Figure 9: Change parameters for a template

When you click this button, a dialog box similar to the one in Figure 10 appears. Simply change the names and values as appropriate. When you click **OK**, the changed parameter appears back in the Query Editor window.

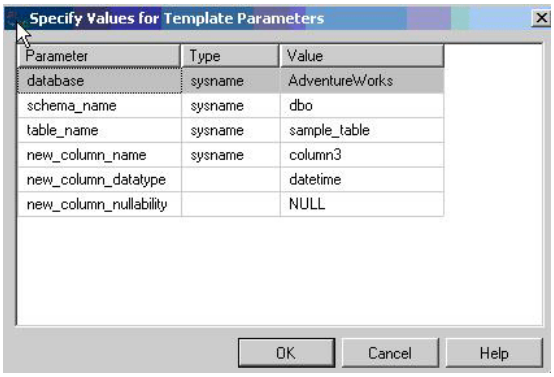

Figure 10: Specify Values for Template Parameters window

Now, you have a basic understanding of how to create the basic objects that make up a database. In the next sections, we look at managing the database and server by using SSMSE.

Basics of Database Administration

Whether you are a beginning developer or a budding database administrator, you need to have some understanding of database administration. Following are the basic functions of database administration.

- System management. Making sure the appropriate services and features are accessible and working. With SQL Server 2005, some features are turned off by default. Moreover, SQL Server Express is configured to not listen on the network via TCP/IP.
- Security management. Managing access to the data. This starts at the operating system and ends with the database. This paper focuses on security topics related to users and logins.
- Disaster preparedness. Recovering from user error and hardware and software failure. This paper covers backing up databases, understanding recovery models, and working with scripts to allow for the re-creation of database objects and data.

In the following sections, we look at how to handle these database administration tasks by using SQL Server Browser, SQL Configuration Manager, and the SQL Surface Area Configuration tool. We review system management tasks, security settings, and backup and recovery models.

System management

When SQL Server Express is installed using the default settings, it is configured to be as secure as possible. Microsoft calls this configuration secure by default. After SQL Server Express is installed, you might need to reconfigure parts of the server based on your expected usage. For example, by default SQL Server Express allows only local connections. If you need to connect to SQL Server Express through your network, configure SQL Server Express to allow remote connections. A number of tools are available for accomplishing various configuration tasks.

SQL Server Surface Area Configuration tool

The Surface Area Configuration tool provides a scriptable interface for setting global security states for SQL Server features and services. Because SQL Server 2005 follows the security principle of secure by default, many features are turned off by default. Moreover, unless you indicate otherwise, the services you choose to install do not start automatically after installation is complete. Use the Surface Area Configuration tool to configure SQL Server and to manage SQL Server security.

The Surface Area Configuration tool can be used to turn on and off services and features as well as to change configuration settings. It provides two basic sorting methods—View by Instance and View by Component. Use View by Component to quickly access and manage a certain grouping of features. View by Instance allows you to look over a specific instance of SQL Server and make changes.

If you plan to have applications connect to an instance of SQL Server Express over a network, you must enable remote connections via TCP/IP and turn on SQL Server Browser. We cover SQL Server Browser later in this paper in SQL Server Browser service; here we look at how to enable remote connections by using the Surface Area Configuration tool.

To navigate to the correct interfaces for working with services and remote connections, select **Surface Area for Services and Connections** from the start up screen in the Surface Area Configuration tool.

To display remote connections settings, click **Remote Connections** in the Features panel of the Surface Area Configuration tool as illustrated in Figure 11. Enable remote connections and TCP/IP. Figure 11 illustrates remote connections settings configured for remote connections via TCP/IP. Do not use Named Pipes unless you understand how to use this protocol.

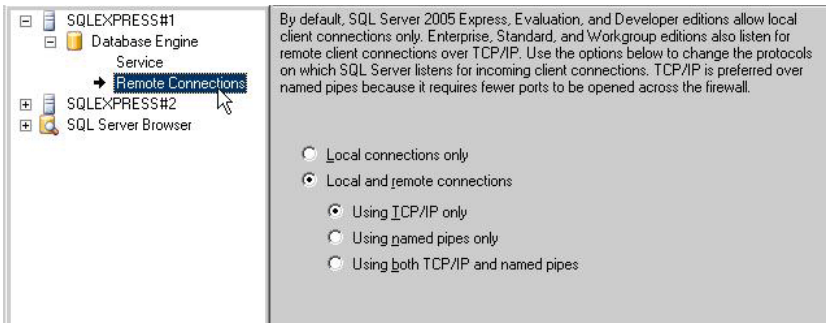

Figure 11: Remote permissions

To finalize the remote connection settings, click **Apply**.

Applications attempting to connect to your instance must be able to find the server on the network.

To resolve or listen for requests for SQL Server Express, the SQL Server Browser service must be enabled. The SQL Server Browser service provides name resolution for applications attempting to connect to SQL Server Express via a network. To start SQL Server Browser, click **SQL Server Browser** in the Surface Area Configuration tool. Set the service to On and then click **Apply**.

If you want the service to turn on automatically whenever the server is started, select **Automatic**. If you select **Manual**, whenever the operating system restarts you must open the Surface Area Configuration tool and start the SQL Server Browser service.

Warning: Turning on SQL Server Browser allows connections from the Internet to your computer. For security purposes, you might want to leave SQL Server Browser off on computers that are directly connected to the Internet. If there is only one SQL Server instance running on your computer, the first instance automatically binds to port 1433; that is the default port clients will try to connect to.

If more than one instance of SQL Server is running and you do not want to run SQL Server Browser, you can assign each SQL Server instance its own TCP/IP port and connect explicitly to that port.

SQL Server Configuration Manager

SQL Server Configuration Manager provides detailed control over the services and network protocols used by SQL Server 2005. SQL Server Configuration Manager is an

MMC snap-in application. It consolidates the SQL Server 2000 Network Service utility and the Services utility into one application. In SQL Server Configuration Manager, you can explicitly change ports and IP addresses, create and destroy aliases for servers, enable and disable protocols, and more. To manage services, you can use SQL Server Configuration Manager to start and stop services, change logon credentials, and peer into Windows registry settings for the services.

Table 1 compares the features of SQL Server Configuration Manager and the Surface Area Configuration tool.

Feature	SQL Server Surface Area Configuration tool	SQL Server Configuration Manager
Start and stop services	YES	YES
Turn on and off features	YES	NO
Configure network clients	NO	YES
Configure native clients	NO	YES
Manage remote connections	YES	YES
Create network aliases	NO	YES
Set up aliases	NO	YES
Change TCP/IP addresses	NO	YES

Table 1 Comparison of SQL Configuration Manager to Surface Area tool

SQL Server Browser service

When an instance of SQL Server starts, if TCP/IP protocol is enabled, the server is assigned a TCP/IP port. If Named Pipes protocol is enabled, SQL Server listens on a specific named pipe. This port, or pipe, is used by that specific instance to exchange data with client applications. During installation, TCP port 1433 and pipe \sql\query are assigned to the default instance. The server administrator can use SQL Server Configuration Manager to change these. Since only one instance of SQL Server can use a port or pipe, different port numbers and pipe names are assigned for named instances. By design, when named instances are configured to use dynamic ports, an available port is assigned when SQL Server starts. You can assign a specific port to a SQL Server instance if you want to.

At start time, SQL Server Browser starts and claims UDP port 1434. SQL Server Browser reads the registry, identifies all SQL Server instances on the computer, and notes which ports and named pipes they use. When a server has two or more network cards, SQL Server Browser returns the first enabled port it encounters for SQL Server. SQL Server 2005 and SQL Server Browser support IPv6 and IPv4. When SQL Server 2005 clients request SQL Server resources, the client network library uses port 1434 to send a UDP message to the server. SQL Server Browser responds with the TCP/IP port or named pipe of the requested instance. The network library on the client application then completes the connection by sending a request to the server by using the port or named pipe of the desired instance.

Warning: For security purposes, you may want to leave SQL Server Browser off on computers that are directly connected to the Internet.

User access to databases

This section covers basic security concepts as they relate to SQL Server Express databases. In SQL Server 2005, there are two authentication mechanisms—SQL Server Authentication and Windows Authentication.

Logins are server principals that can connect to a particular server. Logins use either Windows Authentication or SQL Server Authentication to identify themselves to the server. Windows Authentication uses the credentials of the Microsoft Windows® account (or the Windows group that the Windows account belongs to) that is being run. SQL Server Authentication uses a user name and password provided by SQL Server outside the Windows security infrastructure. Windows Authentication is easier for database users to use because it is automatic; no special login name or password is required. Windows Authentication is more secure because it relies on Windows to maintain security information rather than on the SQL Server local security database.

Microsoft recommends using Windows Authentication, although there are times when you will want to use SQL Server Authentication.

Users are the security principal that a login maps to for a particular database. This mapping allows a particular login to have different privileges in different databases. A login could be mapped as the database owner (dbo) of one database (with unlimited privileges in that database) and as a user who has only read access to a few tables in another database.

In general, the work flow for database administrators is to first create a login, and then create users for that login in the databases to which the login should have access.

Let's further our understanding of the terms with some simple definitions.

Windows Authentication login: An entity who is a member of either a domain or local machine. This is a Windows account that can log on to a computer. Windows Authentication logins can be created for both Windows accounts and Windows groups. The Windows Authentication login account is mapped to a SQL Server login.

SQL login: An entity that has the ability to log on to a server instance and which lives within a SQL Server instance. This login can play both server-level roles and database-level roles.

SQL user: A database user with roles and privileges regulated at the database level. This user plays only database–level roles.

Role-based security is the recommended way to provide access to databases. In role-based security, you assign object privileges to roles. You then add and remove users from those roles. This makes it very simple to administer security because you do not have to manage permissions for each user on each object individually. Role-based security is an efficient way to handle database access when the number of tables, views, and other objects in a database crosses into the hundreds or thousands.

For more information:

To learn more about managing security, please see the following articles on MSDN.

To learn how to create a user, see CREATE LOGIN (Transact-SQL)

To learn how to create a login, see CREATE USER (Transact-SQL).

To learn how to alter a login using Transact-SQL , see ALTER LOGIN (Transact-SQL).

For more information on database security considerations, see Security Considerations for Databases and Database Applications. This page also contains links to important topics.

Before you continue:

Before continuing, please read the following MSDN articles:

Permissions

Permissions Hierarchy

Principals

Securables

Creating SQL users

You can create a new SQL user by using either a dialog box or Transact-SQL. We use the dialog box method in this paper.

To begin, navigate to the database to receive the new user. Right-click the security folder and select **New**. A submenu appears as shown in Figure 12.

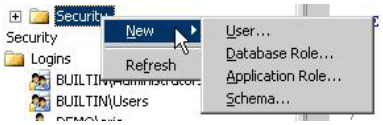

Figure 12: Security dialog box options

To open the Database User – New dialog box, select **User**.

The **General** tab on this dialog box provides text boxes for a user name and login. The login is used to authenticate the user. You can also select schemas and database roles for the SQL user on the **General** tab. The **Securables** tab is the same as the SQL Login dialog box. The **Extended Properties** tab does not provide any functionality for the SQL Server Express version of SQL Server Management Studio.

Note that there is no status dialog box. To change the status of a user, you open the Login dialog box. This is done by opening the security folder, navigating to the user, and right-clicking **Properties**.

Creating SQL logins

The Login-New dialog box is in the server-level security folder. You can find this by expanding the server node folder in Object Explorer. Right-click the security folder to open the Login-New dialog box. The Login-New dialog box provides basic authentication information. You can choose Windows Authentication or SQL Server Authentication. Windows Authentication requires the Windows user to exist on a local machine or domain. You can choose SQL Server Authentication for SQL Server logins that do not exist as domain users (see Figure 13).

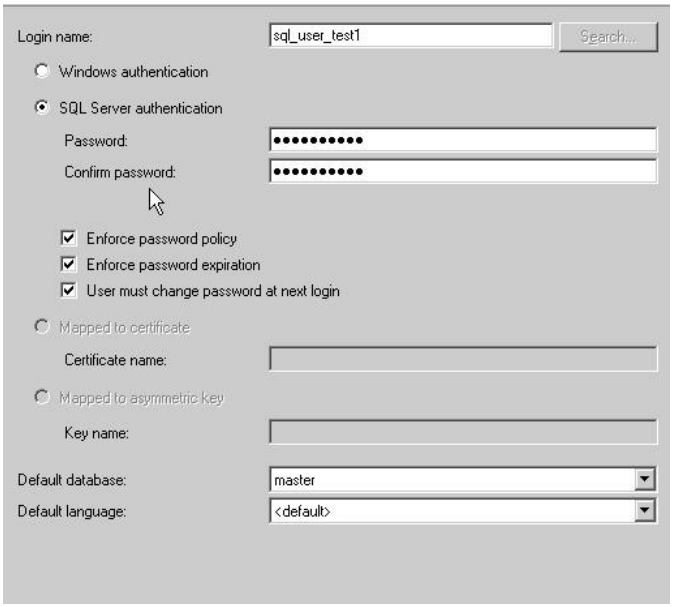

Figure 13: Login-New dialog box

After you supply a user name, you apply specific settings that user. Let's take a look some of the important settings in this dialog box.

Server Roles tab

Assign server-wide roles to a particular login. These are powerful roles that effect access to important administrative tasks.

User Mappings tab

Set up user mapping. User mappings provide the relationship between a login and corresponding SQL user. Select the system databases only if the login needs access to the system database. If you are using a schema, select the schema. Lastly, select the database roles. Typically a login will have the **db_datareader** and **db_datawriter** fixed database roles.

Securables tab

Select privileges for logins. For more information, see Securables on MSDN.

Status tab

Turn on/off the login and SQL user. Following are the states you can adjust for a given user and login.

Grant: Gives a user a privilege, such as to execute a query on the database. In general, Grant means that the login can connect to the server.

Deny: Revokes a privilege. A login is specifically not allowed to connect to the server even if the user is a member of a Windows group that would otherwise have access.

Enable: Login can connect to the server

Disable: Login cannot connect to the server

This section covered creating users and logins, mapping them to each other, and granting or denying them access to the server and database. This is really just the tip of the security iceberg, and I recommend spending the time needed to understand this area.

For more information:

To learn all about roles and what they do, see the following articles on MSDN.

Server-Level Roles

Database-Level Roles

Disaster Preparedness

The goal of disaster preparedness is to bring the database back online after a failure. There are many different types of conditions that force a database offline. This section discusses how to be ready if the working version of your database crashes or goes offline. Specifically this section covers:

- Backing up a database
- Restoring a database
- Using scripts to recreate your database
- Using scripts to re-input data into your database

Preparing for a disaster differs depending on whether the database is hosted locally or by a Web hosting or other service company.

Local databases

The most common way to be prepared for any disaster is to have a backup copy of the database. We use backups to keep copies of the data. We make a complete set of scripts to re-create the database from scratch in case the back up fails or the server needs to be re-created. The following sections cover the mechanics of backing up and creating the copies of the databases that we need to bring the database back online.

Hosted databases

Hosted databases present a unique set of challenges and fair number of unknowns. When you select a hosting provider, they will probably market themselves as providing high availability. In reality, you have no real way of knowing if they are backing up YOUR database on a regular basis. I recommend that you inquire directly about getting script versions of your database. This can usually be done via their trouble ticket system.

Backing up and restoring local databases

There are three ways to back up a local database:

- Use the Backup dialog box.
- Use a backup template.
- Detach the database and copy the files from the data folder to another location.

The following sections look at the Database Backup dialog box.

To open the Database Backup dialog box, right-click a database in Object Explorer, point at **Tasks** and select **Backup**. Figure 14 shows the basic settings for a database back up.

There are two tabs on the Database Backup dialog box. First, we cover the **General** tab.

All database on the server are listed in the drop-down list. Select the database.

For the back up type there are two choices. If this is the first back up of this database, do a FULL back up. This creates a complete backup of all data. The second choice,

differential, creates an incremental backup of the data. This is useful for extremely large databases, where the backup files are sourced across many drives.

You will notice there is a destination window, with a couple of items in it. By default, SQL Server uses a backup folder in the default SQL Server install location as the destination. You can also use backup media, which acts as a repository for backups.

Let's take a look at how we use backup media.

The **Options** tab, shown in Figure 14, provides a mechanism for managing the backup files. For SQL Server Express, because the database size is limited, I recommend using the **Append to the existing backup set** option. If you want to have just one backup set, and do not care about keeping previous backups, the **Overwrite all existing backup sets** option is appropriate. Under the **Reliability** option group you should select both reliability features (verify and checksum) because there is nothing worse than having a restore fail on a bad backup.

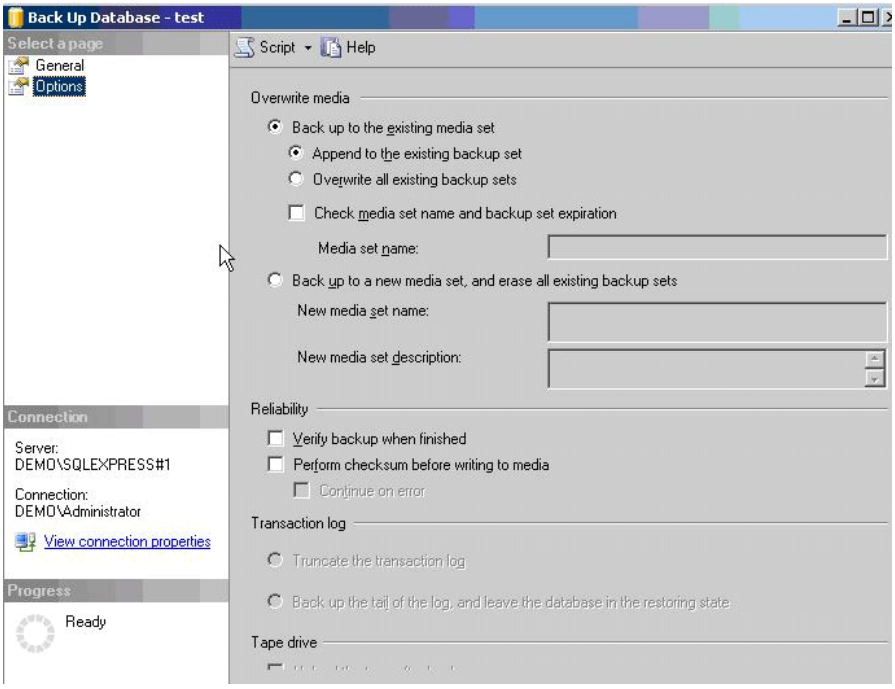

Figure 14: Options tab on the Back Up Database dialog box

Creating a backup device

A backup device is a predefined storage location for backups. For many organizations it consists of a removal tape drive, that automatically backs up data. Another method is to use a backup device. An approach that I like is to use an external drive or another internal drive as a backup device. Having a backup disk system that is separate from the server provides an easy means for transporting the database from one server or facility to the next.

To create a backup device, expand the Server Objects folder in Object Explorer and right-click the Backup Devices folder. Select **New Backup Device**, and provide a name and file location for the device. Now when you create backups, you can simply use the backup devise and SQL Server manages the backup files. Without a backup devise, you

must manually manage the backups. This method is well suited for situations where maintaining a history of backups is important.

Automating backup

SQL Server Express Edition does not contain functionality to back up databases on a regular schedule or automatically. Also, data and log files cannot be copied while they are in use by SQL Server, so ordinary file backup does not work on database files that are not closed. You can, however, use the operating system to automate backup. The Windows operating system has a program called Task Scheduler. You can use it to set up a regularly scheduled task to back up a database.

You will find a good article on SQLDBATips.com with examples of how to automate backup of the database. Here is the link to the article:

< http://www.sqldbatips.com/showarticle.asp?ID=27>

The basic requirements for backup automation are:

- Create a Transact-SQL script.
- Use SQLCMD to execute the script by passing the script to SQLCMD.
- Create a scheduled task in Windows. To open the Task Scheduler, open the **Start** menu and select **Programs**, then select **Accessories**, **System Tools**, **Scheduled Tasks**.

Another strategy for automating backups is to use the Backup dialog box in SQL Server Express Edition to set backup options, then script the backup command to a file rather than execute it directly in the dialog box. Users can then use SQLCMD to execute the script in a Task Scheduler task.

Alternatively, if you are knowledgeable about shadow copy, you can back up the SQL Server directory on the file system and accomplish the same thing. The major difference between copying the files and running the backup is related to transactional replay.

Restoring databases

To restore a database, open the **Tasks** menu, select **Restore**, then select **database**. A Restore dialog box appears. There are two panes in this dialog box—the General pane and the Options pane. In the General pane, you supply the destination and the source for the restore operation.

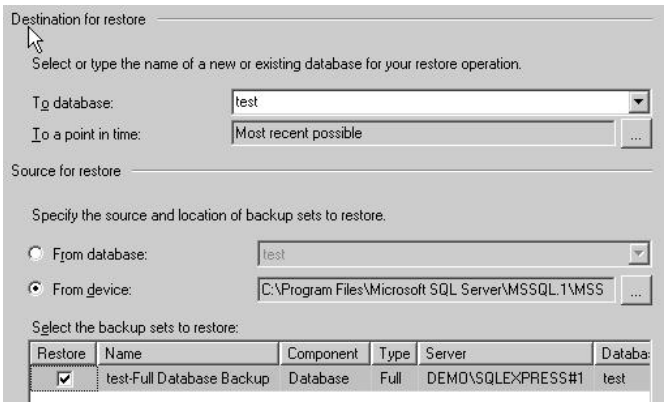

Figure 15: Database Restore dialog box

On the **Database Restore Options** tab, you can set options for:

- Overwriting the existing database.
- Preserving replication settings.
- Prompting before storing each backup (applies only to multi-backup restore operations).
- Restricting access to the database after restoration.

There are three restoration states that affect the use of the database upon completion of the restore operation. These states are as follows:

- **Restore with Recovery**. The database is ready to use (the default setting).
- **Restore with No Recovery**. The database is not ready.
- **Restore with Stand-by**. The database is in read-only condition.

Typically, you leave all the defaults as selected.

Advanced Database Administration

In the following sections, we take a look at a few important features that every database administrator (DBA) should understand. We look at how to access database settings by using server properties, database catalog views, and dynamic management views (DMVs). I'll introduce you to the dedicated administrator connection, which is your special connection for handling a server which has gone astray and needs to be brought into a steady state. We'll also look at the Activity Monitor and some advanced features of SQL Server Express.

Server properties

Often users want to configure SQL Server Express to use a particular amount of resources on the computer. To do this, in Object Explorer, right-click a server to open the Server Properties window. The most commonly changed setting is the **Server Memory Options**. To customize the amount of memory used by SQL Server Express, click **Memory page** and you can customize the minimum memory used or set a lower maximum memory.

System databases

SQL Server has four system databases. These system databases provide a unique infrastructure for SQL Server. These databases are important and should be included in any disaster recovery plan. The **master** database in particular needs to be backed up before each change to the database infrastructure.

Master. Records all the system-level information for an instance of SQL Server.

Model. Used by SQL Server Agent for scheduling alerts and jobs.

MSDB. Used as the template for all databases created on the instance of SQL Server. Modifications made to the **model** database, such as database size, collation, recovery model, and other database options.

Temp. A workspace for holding temporary objects or intermediate result sets.

Resource database. A read-only database that contains system objects that are included with SQL Server 2005. System objects are physically persisted in

the **Resource** database, but they logically appear in the sys schema of every database. This database is hidden.

For more information:

For more information about the system databases, see System Databases on MSDN.

To learn more about backing up and restoring system databases, see Backing Up and Restoring System Databases.

Shrinking the database and files

You can increase performance by removing extra space allocations on the disk system. There are two types of shrinking. *Database shrinking* reduces the padded space for the database. *File shrinking* reduces the amount of disk space allocated.

To shrink a database or file, point to a database in the Databases folder in Object Explorer and right-click **Tasks**. Select **Shrink**. Then choose either **database** or **files**.

Either the Shrink Database or the Shrink File dialog box appears. The Shrink Database dialog box is simple to understand. The Shrink File dialog box, shown in Figure 16, is rather more complex.

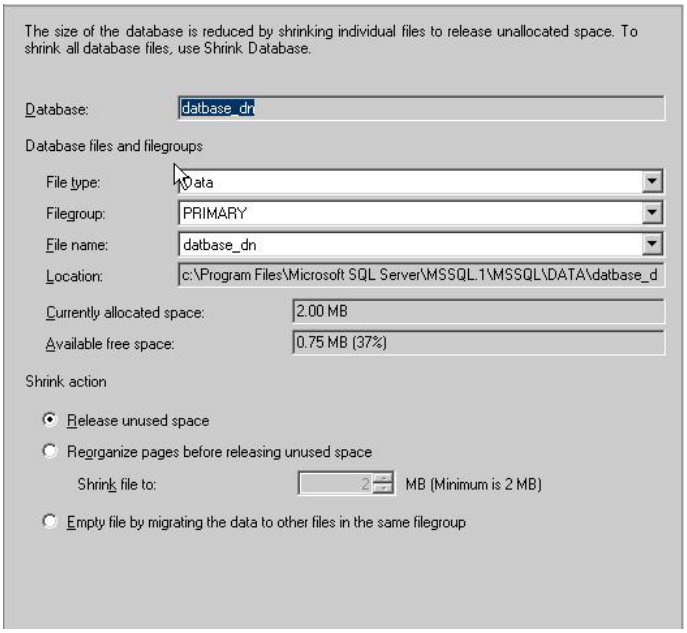

Figure 16: Shrink Files dialog box

As you can see in Figure 16, there is a significant amount of free space in my database—37%. SQL Server holds space on the drives for more data storage. If we do not need to hold disk space, shrinking the files frees up space.

The most important setting in this dialog box is the **Shrink action** option group. Of the three choices, **Release unused space** is the most simple. This option should be used most of the time. The **Reorganize pages before releasing unused space** option tells SQL Server to reorganize the data layout to meet a certain size. You specify the size in the **Shrink file to** setting. You cannot shrink the database to smaller then the size that it is physically, so if your current database size is 3 MB, then the minimum is 3 MB. The last option, **Empty file by migrating to other files in the same filegroup** provides a means for off-loading the data.

Attaching and detaching databases

The ability to attach and detach a database is useful for moving a database from one server to another. With the database detached, you can copy the MDF to another location and execute a database attach. Attaching is different than database restore or back up. It provides a simple means for moving the database from one computer to another.

To transport a database by using backup and restore functionality, you would have to create an empty version of the database on the destination, then restore the backup into the empty database. A backup across a network might fail. Although it would seem natural to use a backup operation, a detach/copy/attach process is in fact speedier. In Figure 17, we see how to accomplish a detach.

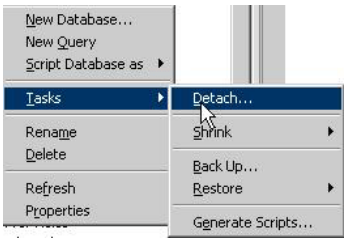

Figure 17: Detaching from the Tasks window

Database catalog views

To better protect SQL Server databases, the system catalogs have been locked down. They cannot be changed. Also, the system catalogs are not universally visible. SQL Server provides a set of SQL views that provide information about the system catalog. These views are read-only and are designed to show the metadata.

To query a catalog view, look up the appropriate system function and execute the query with SELECT statement filters. The naming convention for the catalog views is user friendly. You can run queries against the **master** database and all user databases.

The following code is a simple example of querying the database-level view for principals.

```
Select type_desc,name,default_schema_name from sys.database_principals
where type_desc like 'sql_user'
```

For more information:

To learn more about database catalog views, see Catalog Views (Transact-SQL) on MSDN.

Dynamic management views

SQL Server 2005 provides more than 80 new dynamic management views (DMVs). DMVs are a completely new topology for troubleshooting SQL Server database issues. They are divided into groups from the server level down to the database level. Special views are provided for checking on .NET assemblies, security, and others. DMVs include not only current data, but also aggregated historical data.

SQL Server 2005 turns on a default trace, which provides a means for finding out what happened when an error occurs. You can think of the default trace as a flight recorder. DMVs use the default trace. This is where DMVs are really interesting: administrators can review the default trace after an unplanned incident and see what happened. Moreover, when you first log on to SQL Server, the summary reports in the main window are created from the historical data.

DMVs are actually database views. They can be found in the Views folder under each user's Database System database folder. (The prefix for the views is *dm_*.) DMVs are organized into five general categories. The views are categorized by the environmental factors they report on.

- DMVs with the name *dm_exec_** provide information about execution of user modules and connections.
- DMVs that use the naming convention *dm_os_** report on memory, locks, and execution scheduling.
- DMVs that use the naming convention *dm_trans_** provide insight into transactions and isolation.
- DMVs that use the naming convention *dm_io_** provide information for monitoring disk I/O.

A DBA would be remiss not to understand DMVs. When your server acts up and freezes, you can combine DMVs with another new feature: the dedicated administrator connection (DAC). With the DAC and DMVs, you can find the offending process or job and kill it without restarting the server.

For more information:

To learn more about DMVs, see Dynamic Management Views and Functions on MSDN.

Dedicated administrator connection

The dedicated administrator connection is a special connection that can be use to log on to SQL Server when all other connections fail. Only one dedicated administrator connection can be used on a server instance. Do not use the DAC for general connections to the database.

You can access the DAC only by using SQLCMD in SQL Server Express.

To use the dedicated administrator connection

1. Use SQLCMD to connect to the database. The SQLCMD –A command provides the command-line connection.
2. To turn on the DAC, you must enable trace flag 7806. To do this, run the follow Transact-SQL commands from the Query Editor window or issue the same query by using SQLCMD.

```
USE Master
DBCC TRACEON (7806)
GO
```

If you have a SQL Server Express server that is not responding

1. To use SQLCMD to log on to the server, run the following command:

```
C:\ SQLCMD –Sadmin:<instancename> – D master
```

2. With the connection established, execute the following dynamic management view to find the current sessions:

```
select * from sys.dm_exec_sessions
```

You might run a more refined version of the previous query to find out which sessions are taking a long time or are stuck. The following code is an example.

```
Select session_id, total_elapsed_time,memory_usage,status
from sys.dm_exec_sessions
```

Once you have determined the offending process, you can end it. This is done by using the Kill command in Transact-SQL.

```
Kill (process id)
```

For example:

```
Kill(54)
```

For more information:

To learn more about the Kill command, see KILL (Transact-SQL) on MSDN.

Activity Monitor

The Current Activity window in SQL Server 2005 Management Studio Express Edition graphically displays information about:

- Current user connections and locks.
- Process number, status, locks, and commands that active users are running.
- Objects that are locked and the kinds of locks that are present.

If you are the system administrator for the database, you can view additional information about a selected process or terminate a selected process. The Current Activity window is limited at the database level.

Linked servers

You can link together two database servers that are physically separate. You might do this, for example, to share data or to perform an action on one server based on an action of another server.

Linking servers provides a mechanism for working across database servers in a secure manner. Use the Linked Server dialog box to set up user and login mappings between two servers, and to connect to a data source outside of SQL Server. For example, you can link to a Microsoft Access database.

One of the main uses for linked servers is to import data into SQL Server Express. Once a linked server configuration has been created, users can issue make table queries to copy the data into their SQL Server database.

For more information:

For an overview of linked servers, see Linking Servers on MSDN.

Replication

SQL Server Express Edition has limited replication capabilities. A SQL Server Express database can only be a subscriber to a SQL Server Workgroup, Standard, or Enterprise Edition publication. You can use the Replication Subscription dialog box to set up the relationship. Because support for SQL Server replication is not installed by default, you must run the setup program and select **Replication** in the Advanced Options dialog box.

For more information:

For an overview of replication, see Implementing Replication on MSDN.

For information on replicating data with SQL Server Express, see Replicating Data to SQL Server Express.

Conclusion

This paper covered the basics of SQL Server 2005 Management Studio Express Edition. We covered basic object creation and database administrative tasks. We also learned about security and disaster preparedness. With this knowledge, you are ready to manage SQL Server Express.

For more information:

http://www.microsoft.com/technet/prodtechnol/sql/default.mspx

Did this paper help you? Please give us your feedback. On a scale of 1 (poor) to 5 (excellent), how would you rate this paper?

Resources

Overview of SQL Server Express:

http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dnsse/html/sseoverview.asp

Security in SQL Server Express:

http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dnsse/html/ssesecurity.asp?frame=true

Embedding SQL Server Express into Custom Applications:

http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dnsse/html/EmSQLExCustApp.asp?frame=true

SQL Server Express User Instances:

http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dnsse/html/sqlexpuserinst.asp?frame=true

Web Development with Visual Web Developer 2005 Express Edition and SQL Server 2005 Express Edition, Part 1

http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dnsse/html/VWD_SSE.asp?frame=true

MSDN Forum for SQL Server Management Studio:
http://forums.microsoft.com/MSDN/ShowForum.aspx?ForumID=84&SiteID=1

SQL Server Express Team blog:
http://blogs.msdn.com/sqlexpress/default.aspx