

# TABLE OF CONTENTS

## List of Tables

## Introduction

### 1. Installing SQL Server 2000

- 1.1 Minimum System Requirements
- 1.2 Installing SQL Server 2000
  - 1.2.1 Installing SQL Server 2000 from the CD-Rom
  - 1.2.2 Installing SQL Server 2000 on a Remote Computer
  - 1.2.3 Performing an Unattended Installation of SQL Server 2000
  - 1.2.4 System Databases Created during Installation
- 1.3 Installing Multiple Instances of SQL Server
  - 1.3.1 Version Switching
  - 1.3.2 Named Instances of SQL Server 2000
- 1.4 Upgrading to SQL Server 2000
  - 1.4.1 Upgrading from SQL Server 7.0
  - 1.4.2 Upgrading from SQL Server 6.5
  - 1.4.3 Upgrading from SQL Server 6.0

### 2. SQL Server 2000 Administration

- 2.1 Setting Up Additional SQL Features
  - 2.1.1 Setting Up SQL Mail and SQLAgentMail
  - 2.1.2 Setting Up Linked Servers
  - 2.1.3 Configuring SQL Server XML Support in IIS
- 2.2 Configuring the Server
  - 2.2.1 Configuring the Windows Server Operating System
  - 2.2.2 Configuring the SQL Server Service
- 2.3 Automating SQL Server 2000 Administration
  - 2.3.1 Operators
    - 2.3.1.1 Creating Operators
    - 2.3.1.2 Creating a Fail-Safe Operator
  - 2.3.2 Using Jobs
    - 2.3.2.1 Types of Job Steps
    - 2.3.2.2 Permissions and Ownership of Jobs
    - 2.3.2.3 Multiple Job Steps and Job Responses
    - 2.3.2.4 Scheduling Jobs
    - 2.3.2.5 Creating Jobs
    - 2.3.2.6 Multiserver Jobs

- 2.3.3 SQL Server 2000 Alerts
  - 2.3.3.1 SQL Server Event Alerts
  - 2.3.3.2 Performance Alerts
  - 2.3.3.3 Configuring Alerts
- 2.3.4 SQL Server 2000 Database Maintenance Plan

### **3. SQL Server Databases**

- 3.1 Datafiles
  - 3.1.1 Datafile Names
  - 3.1.2 Datafile Properties
- 3.2 Transaction Log Files
- 3.3 Filegroups
- 3.4 Tables
  - 3.4.1 Columns
  - 3.4.2 Constraints
    - 3.4.2.1 PRIMARY KEY Constraints
    - 3.4.2.2 UNIQUE Constraints
    - 3.4.2.3 FOREIGN KEY Constraints
    - 3.4.2.4 CHECK Constraints
    - 3.4.2.5 Defaults and DEFAULT Constraints
    - 3.4.2.6 Triggers
  - 3.4.3 System Tables
  - 3.4.4 Temporary Tables
  - 3.4.5 Table Views
  - 3.4.6 Internal Table Storage
- 3.5 Indexes
  - 3.5.1 Index Page Splits and Index Fragmentation
  - 3.5.2 Types of Indexes
    - 3.5.2.1 Clustered Indexes
    - 3.5.2.2 Nonclustered Indexes
  - 3.5.3 Creating Indexes
    - 3.5.3.1 Using Transact-SQL Statements to Create Indexes
    - 3.5.3.2 Using the Create Index Wizard to Create Indexes
  - 3.5.4 Special Indexes
    - 3.5.4.1 Indexed Computed Columns
    - 3.5.4.2 Indexed Views
  - 3.5.5 Full-Text Indexes And Full-text Catalogs
    - 3.5.5.1 Full-text Indexes
    - 3.5.5.2 Full-text Index Maintenance
  - 3.5.6 Updating Distribution Statistics
  - 3.5.7 Maintaining Indexes

### **4. Creating a SQL Server User Database**

- 4.1 Using SQL Server Enterprise Manager to Create a User Database
  - 4.1.1 Using SQL Server Enterprise Manager Directly
  - 4.1.2 Using the Create Database Wizard
  - 4.1.3 Using Transact-SQL to Create a User Database
- 4.2 Managing User Database Size
  - 4.2.1 Controlling Database and Datafile Size
  - 4.2.2 Controlling the Transaction Log File Size
- 4.3 Creating Additional Data and Transaction Log Files
- 4.4 Detaching and Attaching Databases
  - 4.4.1 Using SQL Server Enterprise Manager to Detach and Attach a Database
  - 4.4.2 Using Transact-SQL to Detach and Attach a Database
- 4.5 Setting Database Options
  - 4.5.1 Auto Options
  - 4.5.2 Cursor Options
  - 4.5.3 Recovery Options
  - 4.5.4 SQL Options
  - 4.5.5 State Options
  - 4.5.6 Termination Options
- 4.6 Altering a Database
- 4.7 Deleting a Database
  - 4.7.1 Using Enterprise Manager to Delete a Database
  - 4.7.2 Using Transact-SQL Statements to Delete a Database
- 4.8 Creating Tables
  - 4.8.1 Using Transact-SQL to Create a Table
  - 4.8.2 Using Enterprise Manager to Create a Table
  - 4.8.3 Naming Tables and Columns
  - 4.8.4 SQL Server 2000 Datatypes
    - 4.8.4.1 Numeric Datatypes
    - 4.8.4.2 Datetime Datatypes
    - 4.8.4.3 Character Datatypes
    - 4.8.4.4 Large Object (LOB) Datatypes
    - 4.8.4.5 Miscellaneous Datatypes
    - 4.8.4.6 Nullability and **NULL** Values
  - 4.8.5 Creating Tables in a Filegroup
  - 4.8.6 Using Identifier Columns
  - 4.8.7 Altering a Table
    - 4.8.7.1 Renaming an Existing Column in a Table
    - 4.8.7.2 Adding or Dropping Columns
    - 4.8.7.3 Adding or Dropping a Constraint

- 4.8.7.4 Using CHECK Constraint and FOREIGN KEY Constraint
- 4.8.7.5 Altering the Datatype or Nullability of a Column
- 4.8.7.6 Enabling or Disabling a Trigger
- 4.8.8 Deleting Tables from a SQL Server Database

- 4.9 Database Backup and Restoration
  - 4.9.1 The Backup Process
  - 4.9.2 Backups Types
  - 4.9.3 Recovery Models or the Recovery Option
  - 4.9.4 The Database Recovery Process
  - 4.9.5 Manual Recovery Options

## 5. Populating the Database

- 5.1 Data Transformation Services (DTS)
  - 5.1.1 Storing DTS Packages
  - 5.1.2 Using DTS Package Execution Utilities
  - 5.1.3 Using DTS Package Logs and Execution Files
- 5.2 Using Bcp
- 5.3 The **BULK INSERT** Statement

## 6. Configuring SQL Server 2000 Security

- 6.1 SQL Server 2000 Login Authentication
  - 6.1.1 Windows Authentication
  - 6.1.2 Mixed Mode and SQL Server Authentication
  - 6.1.3 Changing Authentication Modes
- 6.2 Client Net-Libraries Authentication Protocols
- 6.3 Creating SQL Server 2000 Logins
- 6.4 SQL Server 2000 Permissions
  - 6.4.1 Permission Inheritance
  - 6.4.2 Permission Conflicts
  - 6.4.3 Statement Permissions
  - 6.4.4 Database Object Permissions
- 6.5 SQL Server 2000 User Accounts and SQL Roles
  - 6.5.1 Server Roles and Server-Wide SQL Server 2000 Permissions
  - 6.5.2 Fixed Database Roles and Database-Wide Permissions
  - 6.5.3 User-Defined Database Role
  - 6.5.4 Application Roles
- 6.6 Monitoring SQL Server 2000 Database Access

## **7. Monitoring SQL Server 2000 Performance**

7.1 Identifying Performance Bottlenecks

7.2 Determine Trends

7.3 Monitoring Tools

7.3.1 System Monitor

7.3.2 SQL Profiler

7.3.3 Task Manager

7.3.4 SQL Query Analyzer

7.3.5 Transact-SQL

7.3.5.1 System Stored Procedures

7.3.5.2 Database Console Command (DBCC)

7.3.5.3 Built-in Functions

7.3.6 SQL Server Enterprise Manager

## **8. Practice Labs**

8.1 Installing SQL Server 2000.

8.2 Creating a New User Database

8.3 Creating a New Table

8.4 Creating Logins

8.5 Creating Operators.

8.6 Creating a Scheduled Job

8.7 Creating Alerts

8.7.1 Creating Event Alerts

8.7.2 Creating Performance Alerts

8.8 Creating a Database Maintenance Plan

## LIST OF TABLES

TABLE 1.1	Minimum System Requirements for SQL Server 2000 Enterprise Edition
TABLE 3.1	Some Common SQL Server 2000 Data Types
TABLE 3.2	Information contained in the page header
TABLE 3.3	The seven <b>SET</b> options for Indexed Views and Indexed Computed Columns
TABLE 4.1	Common Default Database Properties for a SQL Server 2000 Database
TABLE 4.2	The SQL Server 2000 Reserved Keywords
TABLE 4.3	SQL Server Integer Datatypes
TABLE 4.4	SQL Server 2000 Datetime Datatypes
TABLE 4.5	Restrictions on Certain Column Types
TABLE 4.6	SQL Server 2000 backup options
TABLE 5.1	Types of DTS Transformations
TABLE 5.2	Commonly Used Parameters for Bcp
TABLE 6.1	The Security Capabilities of Windows Authentication and SQL Server Authentication
TABLE 6.2	SQL Server 2000 Transact-SQL Statement Permissions
TABLE 6.3	Types of Object Permissions Associated with Database Objects
TABLE 6.4	SQL Server 2000 Server Roles and associated Permissions.
TABLE 6.5	SQL Server 2000 Database Roles and associated Permissions
TABLE 6.6	System Stored Procedures Used to Return Access Information
TABLE 7.1	Memory Object Counters
TABLE 7.2	I/O Object Counters
TABLE 7.3	Processor Object Counters
TABLE 7.4	SQL Server 2000 Performance Objects
TABLE 7.5	SQL Event Categories Monitored with SQL Profiler
TABLE 7.6	SQL Profiler Preconfigured Trace Templates
TABLE 7.7	<b>DBCC</b> Statements Used to Monitor SQL Server 2000
TABLE 7.8	Commonly Used Transact-SQL Global Counters
TABLE 7.9	Information Provided in the Process Info Container

# Installing, Configuring and Administering Microsoft SQL Server 2000 Enterprise Edition

**Exam Code: 070-228**

## Certifications:

<b>Microsoft Certified (MCP)</b>	
<b>Microsoft Certified Database Administrator (MCDBA)</b>	<b>Core</b>
<b>Microsoft Certified Systems Administrator (MCSA)</b>	<b>Elective</b>
<b>Microsoft Certified Systems Engineer (MCSE)</b>	<b>Elective</b>

## About This Study Guide

This Study Guide is based on the current pool of exam questions for the 070-228 ' Installing, Configuring, and Administering Microsoft SQL Server 2000 Enterprise Edition exam. Where the SQL Books OnLine (BOL) was used for reference purposes, the updated version was used. As such it accurately provides all the information required to pass the Microsoft 070-228 exam and is organized around the specific skills that are tested in that exam. Thus, the information contained in this Study Guide is specific to the 070-228 exam and does not represent a complete reference work on the subject of Installing, Configuring, and Administering Microsoft SQL Server 2000 Enterprise Edition. Although this StudyGuide does not deal with the different editions of SQL Server 2000, it does include the information required to answer questions related to the installation, configuration, and administration of Microsoft SQL Server 7.0 and the installation and administration of Windows 2000 Professional and Windows 2000 Server that may be asked during the exam. Topics covered in this Study Guide includes Installing and Configuring SQL Server 2000; Upgrading to SQL Server 2000; Creating and Managing a Linked Server; Configuring SQL Mail and SQLAgentMail; Troubleshoot Failed Installations; Creating SQL Server 2000 Databases; Configuring Database Options for Performance; Attaching and Detaching Databases; Creating and Altering Databases; Working with Filegroups and Transaction Logs; Expanding and Shrinking a Database; Creating and Manage Database Objects; Managing, Monitoring, and Troubleshooting SQL Server 2000 Databases; Managing Database Fragmentation; Modifying the Database Schema; Performing Disaster Recovery Operations; Extracting and Transforming Data with SQL Server 2000; Importing and Export Data; Developing and Manage DTS Packages; Configuring, Maintaining, and Troubleshoot Replication Services; Managing OLE DB Providers; Managing and Monitoring SQL Server 2000 Security; Configuring Mixed Security Modes or Windows Authentication; Creating and Managing Database Users; Creating and Managing Security Roles; and Managing, Monitoring, and Troubleshooting SQL Server 2000.

## Intended Audience

This Study Guide is targeted specifically at people who wish to take the Microsoft MCSE or MCSA elective exam or MCDBA core exam 070-228 ' Installing, Configuring, and Administering Microsoft SQL Server 2000 Enterprise Edition. This information in this Study Guide is specific to the exam. It is not a complete reference work. Although our Study Guides are aimed at new comers to the world of IT, the concepts dealt with in this Study Guide are complex and require an understanding of material provided for the MCSA / MCSE / MCDBA exam: 070-228 ' Installing, Configuring, and Administering Microsoft SQL Server 2000

Enterprise Edition. Knowledge of the Microsoft MCSE exam 070-210 ' Installing, Configuring, and Administering Microsoft Windows 2000 Professional and the Microsoft MCSE exam 070-215 ' Installing, Configuring, and Administering Microsoft Windows 2000 Server would also be advantageous.

### How To Use This Study Guide

To benefit from this Study Guide we recommend that you:

- Study each chapter carefully until you fully understand the information. This will require regular and disciplined work. Where possible, attempt to implement the information in a lab setup.
- Perform all labs that are included in this Study Guide to gain practical experience, referring back to the text so that you understand the information better. Remember, it is easier to understand how tasks are performed by practicing those tasks rather than trying to memorize each step.
- Be sure that you have studied and understand the entire Study Guide before you take the exam.

**Note:** Remember to pay special attention to these note boxes as they contain important additional information that is specific to the exam.

Good luck!

## 1. Installing SQL Server 2000

### 1.1 Minimum System ReZuirements

Before you install SQL Server 2000 Enterprise Edition on a computer, you must first determine if the computer meets the minimum system requirements for SQL Server 2000 Enterprise Edition. Table 1.1 lists the minimum system requirements for SQL Server 2000 Enterprise Editions. However, in most cases the actual required hardware requirement will exceed the minimum hardware requirements. These minimum hardware requirements thus form a baseline for determining system requirements. In addition, the minimum memory requirement for SQL Server 2000 varies depending upon the Windows operating system on which you wish to install SQL Server 2000 Enterprise Edition.

TABLE 1.1: *Minimum System Requirements for SQL Server 2000 Enterprise Edition*

Hardware	Minimum ReZuirements
CPU	Intel Pentium 166 MHz or similar
Memory	64 Mb (128 Mb recommended)
Operating System	Windows NT 4.0 Server (Service Pack 5)
Hard Disk Space	110 Mb (250 Mb for typical installation)
Monitor	VGA or higher resolution (800 600 resolution required for SQL Server graphical tools)
Pointing Device	Mouse
CD-Rom Drive	Required for installation.

### 1.2 Installing SQL Server 2000

You can install SQL Server 2000 Enterprise Edition on the local computer from the SQL Server 2000 installation disk or on a remote computer over the network. In addition you can also perform an unattended installation of SQL Server 2000 Enterprise Edition.

#### 1.2.1 Installing SQL Server 2000 Enterprise Edition from the CD-Rom

The actual installation of SQL Server 2000 Enterprise Edition on a local computer is performed from within the Operating System and is straight forward. The SQL Server 2000 installation program starts automatically when you insert the SQL Server 2000 Enterprise Edition installation disk in to the CD-Rom drive or you can start the installation by running *autorun.exe* in the root of the SQL Server 2000 Enterprise Edition installation disk. The longest part of the installation is building and configuring SQL Server's master database, which stores configuration information, information about the other databases, and many system stored procedures.

#### 1.2.2 Installing SQL Server 2000 Enterprise Edition on a Remote Computer

Installing SQL Server 2000 Enterprise Edition on a remote computer is similar to installing SQL Server 2000 Enterprise Edition on a local computer. This can be done by selecting the remote installation option after running the SQL Server 2000 Enterprise Edition installation program. Once you choose the remote installation option, you will be prompted for the information pertaining to the remote installation. This information includes the computer name of the remote compute on which SQL Server 2000 is to be installed and the installation path relative to the remote computer. This information is stored in a setup initialization

file called *setup.iss*. Once all the required information is collected, the SQL Server 2000 Enterprise Edition installation program is started on the remote computer, and the local installation program closes. The SQL Server 2000 Enterprise Edition remote program (*remsetup.exe*) then copies the required files to the *\admin\$* share directory and runs an unattended installation on the remote computer using the options specified in the *setup.iss* file.

**Note:** To install SQL Server 2000 Enterprise Edition remotely, you must run the SQL Server 2000 Enterprise Edition installation program using a user account that has administrative privileges for the remote computer.

### 1.2.3 Performing an Unattended Installation of SQL Server 2000

SQL Server 2000 Enterprise Edition allows you to perform an unattended installation of SQL Server 2000. The process of performing an unattended installation involves running a batch file, which calls a setup initialization (*.iss*) file, from the command-prompt. The *.iss* file contains all setup entries for the options you want to configure for your SQL Server 2000 Enterprise Edition installation. The command-prompt syntax for performing an unattended setup is:

```
Start /Wait D:\X86\Setup\SetupsZ1.exe k=SMS -s -m -SMS -f1 "C:\Setup.iss"
```

There are a number of methods you can use to create a setup initialization file for an unattended installation of SQL Server 2000 Enterprise Edition.

- The SQL Server 2000 installation program provides an option to record an unattended *.iss* file. With this option you proceed through the interactive installation program and select the installation options you want. These options are then recorded in an *.iss* file that is stored in the *\Winnt* folder. You can then use this *.iss* file as is, or you can modify the file by using a text editor.
- You can use one of the three *.iss* files provided in the root of the SQL Server 2000 Enterprise Edition installation disc as is or you can modify them using a text editor. The three *.iss* files on the SQL Server 2000 Enterprise Edition installation disk are:
  - *sqlins.iss* which can be used to perform a typical installation of SQL Server 2000 and can be called by running *sqlins.bat* from the command-prompt.
  - *sqlcli.iss* which can be used to perform a typical installation of SQL Server 2000 and can be called by running *sqlcli.bat* from the command-prompt.
  - *sqlcst.iss* which can be used to perform a typical installation of SQL Server 2000 and can be called by running *sqlcst.bat* from the command-prompt.
- You can also modify the *setup.iss* file that is automatically created when install SQL Server 2000. This file is stored in the *\Winnt* directory and contains the options you chose when you installed SQL Server 2000. You can use a text editor to modify this file but you must add the **[SdFinish=0]** section before you can successfully use it.

### 1.2.4 System Databases Created during Installation

When you install SQL Server 2000, four system databases are automatically created these are the master database, the tempdb database, the model database and the msdb database.

- the **master database** contains all the system level information, the SQL Server 2000 initialization and configuration information, including all login accounts and the location of the primary files for all user databases;
- the **tempdb database** holds all the temporary tables and temporary stored procedures. A new copy of the tempdb database is created every time SQL Server 2000 is started;
- the **model database** serves as a template that is used in the creation of all user databases and in the recreation of the tempdb database every time SQL Server 2000 is started. You can alter the model database to include user-defined data types, tables, and other database objects that you want to be included in all subsequent databases that you create;
- the **msdb database** holds the tables that SQL Server Agent uses for scheduling jobs and alerts and for recording operators. This database also holds tables used for replication.

#### Operators

Operators are the database users that are assigned responsibility for jobs and alerts on the SQL Server 2000 database

### 1.3 Installing Multiple Instances of SQL Server

SQL Server 2000 is designed to simultaneously support multiple versions and/or installations of SQL Server on the same computer. This can be accomplished either through the use of version switching or through the use of multiple instances of SQL Server. These methods allow you to run SQL Server 6.5, SQL Server 7.0, and SQL Server 2000 on the same computer, although only two versions may be running at any given time, one of which must be SQL Server 2000.

When you install SQL Server 2000, you have the option to install the new version of SQL Server 2000 as the default instance or as a named instance. You can install one default instance of SQL Server, but you can install many named instances of SQL Server 2000. By using named instances you can install multiple additional versions or instances of SQL Server alongside the existing default instance. This is important if you want to install multiple versions of SQL Server as an installation of SQL Server 6.5 and SQL Server 7.0 can only exist as a default instance. Only SQL Server 2000 can be installed as a named instance.

When you install SQL Server 2000, the SQL Server 2000 installation program will automatically detect whether a default instance of SQL Server is already installed on the computer. If a default instance is not detected, the installation program will allow you to choose to install a default or a named instance of SQL Server 2000. If a default instance of SQL Server 2000 is already installed on the computer, the SQL Server 2000 installation program will give you several options depending on what version of SQL Server is already installed as the default instance:

- If the default instance is a SQL Server 2000 installation, you can install a named instance of SQL Server 2000.
- If the default instance is a SQL Server 7.0 installation, you can upgrade the default instance to SQL Server 2000 or you can install a named instance of SQL Server 2000.
- If the default instance is a SQL Server 6.5 installation, you can install SQL Server 2000 as the default instance or as a named instance. If you install it as the default instance, you can use the version switch (*switch.exe*) utility to switch between SQL Server 6.5 and SQL Server 2000. You must however install Service pack 5 for SQL Server 6.5 before you install an instance of SQL Server 2000 on the same computer.

### 1.3.1 Version Switching

With version switching you can install SQL Server 7.0 or SQL Server 2000 as the **default instance** on a computer on which SQL Server 6.5 is already installed. You can then use the *vs* utility to switch between SQL Server 6.5 and either the default instance of SQL Server 7.0 or the default instance of SQL Server 2000. This allows you to control which version of SQL Server is to run as the default instance of SQL Server at any given point in time. It however does not allow multiple instances or versions to run simultaneously.

**Note:** You cannot version-switch between SQL Server 7.0 and SQL Server 2000. Version switching is available only between **SQL Server 6.5** and the default instance of either SQL Server 2000 or SQL Server 7.0.

### 1.3.2 Named Instances of SQL Server 2000

By using a named instance of SQL Server 2000 you can install and run SQL Server 2000 as a named instance on a computer on which either SQL Server 6.5 or SQL Server 7.0 is installed without having to upgrade the existing installation. This allows you to retain and run your existing version of SQL Server while also running SQL Server 2000 on the same computer. You can run any number of named instances of SQL Server 2000 concurrently on the same computer. A named instance can also run at the same time as a default instance of SQL Server. It can therefore run concurrently with an existing installation of SQL Server 6.5 or SQL Server 7.0, both of which can only exist as a default instance. The instance name cannot exceed 16 characters. By installing SQL Server 2000 as a named instance on a computer on which SQL Server 7.0 or SQL Server 6.5 is already installed, you can maintain the default instance of SQL Server 7.0 or SQL 6.5 on that computer. However, when you install SQL Server 2000 as a named instance on a computer on which SQL Server 7.0 is already installed, all your SQL Server 7.0 client tools will be upgraded to SQL Server 2000 client tools for all instances.

Named instances of SQL Server 2000 can then be accessed by specifying the network name of the computer and the name of the named instance of SQL Server 2000 in the format `<computer_name>\<instance_name>`. However, most applications must use SQL Server 2000 client components to connect to a named instance although the SQL Server version 7.0 Client Network Utility can be used to configure a server alias name. This will allow applications to use SQL Server 7.0 client components to connect to a named instance of SQL Server 2000.

## 1.4 Upgrading to SQL Server 2000

When you insert the Server 2000 Enterprise Edition Installation disk into the CD-Rom, the SQL Server 2000 installation program will automatically detect if a version of SQL Server is already installed on your computer and will provide you with the appropriate upgrade options.

### 1.4.1 Upgrading from SQL Server 7.0

If you already have SQL Server 7.0 installed on your computer, the SQL Server 2000 installation will give you the option to install a separate named instance of SQL Server 2000 or to upgrade to SQL Server 2000. If you choose to upgrade option, the installation program rebuilds all the system stored procedures to ensure that the most recent versions are installed. The database files for each database are also modified to conform to the structure of SQL Server 2000 database files. Once the upgrade is complete, all your SQL Server 7 databases will be available on your SQL Server 2000 server.

You can also choose to upgrade an individual database from SQL Server 7.0 to SQL Server 2000 by installing a separate named instance of SQL Server 2000 alongside your SQL Server 7.0 server. You can then load database backups from your SQL Server 7.0 server into your SQL Server 2000 server; you can use the `sp_attach_db` system stored procedure or the Copy Database Wizard to connect database files from a SQL Server 7.0 database to your SQL Server 2000. You however cannot move a database from SQL Server 2000 back to SQL Server 7.0. In other words, you cannot load a SQL Server 2000 backup into a SQL Server 7.0 database, and you cannot use the `sp_attach_db` system stored procedure with SQL Server 7.0 to connect to files from a SQL Server 2000 database.

**Note:** You must use the `EXEC` keyword when using a stored procedure, such as `sp_attach_db`, in a Transact-SQL statement.

### 1.4.2 Upgrading from SQL Server 6.5

If the Server 2000 Enterprise Edition installation program detects SQL Server 6.5 installed on your computer, it gives you the option of installing a new SQL Server 2000 server as a default instance or to set up a version switch with your SQL Server 6.5 server. In either case, your SQL Server 6.5 is not upgraded or altered and the new instance of SQL Server 2000 is installed. Because of structural differences between SQL Server 6.5 and SQL Server 7.0, your SQL Server 6.5 databases, not SQL Server 6.5 itself, must be completely rebuilt or upgraded using the SQL Server Upgrade Wizard before it can be used in SQL Server 2000. You can run SQL Server Upgrade Wizard while the installation of the separate named instance SQL Server 2000 is taking place, or you can run it at a later stage.

You can also opt to install SQL Server 2000 as a default instance. However, SQL Server 6.5 can only exist as a default instance of SQL Server and you can only have one default instance running on your computer at a time. Also, if you install a default instance of SQL Server 2000 on a computer running an instance of SQL Server 6.5, the default instance of SQL Server 2000 becomes the accessible instance of SQL Server, and the SQL Server 2000 program group replaces the instance of SQL Server 6.5 program group on the Start Menu. You can however run either the default instance of SQL Server 2000 or the instance of SQL Server 6.5 by using the SQL Server-Verswitch entry on the Start Menu to switch between the default instance of SQL Server 2000 and the instance of SQL Server 6.5. When you switch from SQL Server 2000 to SQL Server 6.5, the instance of SQL Server 2000 becomes inactive, and the SQL Server 6.5 program group replaces the SQL Server 2000 program group on the Start menu. When you switch from SQL Server 6.5 to SQL Server 2000, the process is reversed.

If you install one or more named instances of SQL Server 2000 on a computer running SQL Server 6.5 and there is no default instance of SQL Server 2000, the instance of SQL Server 6.5 remains active as the default instance. Both the SQL Server 2000 and SQL Server 6.5 program groups appear on the Start menu. You should use the SQL Server 6.5 tools to manage the default instance of SQL Server 6.5, and the SQL Server 2000 tools to manage the named instances of SQL Server 2000.

If you install both named and default instances of SQL Server 2000 on a computer running SQL Server 6.5, you can run the named instances of SQL Server 2000 at any time, but must version-switch between the default instance of SQL Server 2000 and the default instance of SQL Server 6.5. The SQL Server 2000 program group always appears on the Start menu. The SQL Server 6.5 program group appears on the Start menu whenever you have version switched to make SQL Server 6.5 the active default instance. The SQL

Server 6.5 program group does not appear when you have version switched to make SQL Server 2000 the active default instance.

### **1.4.3 Upgrading from SQL Server 6.0**

There is no direct upgrade path when you want to upgrade SQL Server 6.0 to SQL Server 2000. Instead you must either upgrade SQL Server 6.0 to SQL Server 7.0 by using the Server Upgrade Wizard provided with SQL Server 7.0 and then you can run the SQL Server 2000 Enterprise Edition installation program to upgrade from SQL Server 7.0 to SQL Server 2000. Alternatively, you can convert the SQL Server 6.0 data to SQL Server 6.5, and then convert the SQL Server 6.5 data to SQL Server 2000.

## 2. SQL Server 2000 Administration

### 2.1 Setting Up Additional SQL Features

You can configure SQL Server 2000 to send and receive e-mail, notify pagers, connect with linked servers for distributed queries, and integrate with IIS for XML support. However, you must set up these features before you can take advantage of them.

#### 2.1.1 Setting Up SQL Mail and SQLAgentMail

SQL Server 2000 can connect with **Microsoft Exchange Server**, **Microsoft Windows Mail**, or a Post Office Protocol 3 (**POP3**) server to send and receive messages using either **SQL Mail** or **SQLAgentMail**. Both SQL Mail and SQLAgentMail require a **MAPI client application** such as Microsoft Outlook on the local SQL Server 2000 computer and a MAPI messaging profile that requires the use of the **same** domain user account that is used to start the instance of SQL Server. You can create the MAPI messaging profile by using either the MAPI client or the Mail program in Control Panel. If different domain user accounts are used by each service, you must set up a messaging profile for each domain user account. You must log on to Windows 2000 as the domain user to configure the messaging profile for that domain user. Furthermore, if the domain user account is not a local administrator, you might need to give the account permission to log on interactively before you can create a messaging profile for this user account. The messaging profile contains the connection information used by the MAPI client to connect to the Microsoft Exchange Server, Microsoft Windows Mail, and/or a POP3 server.

Once the MAPI messaging profile has been configured and tested, you can use **SQL Server Enterprise Manager** to set up SQL Mail and/or SQLAgentMail. You can also use the `p_startmail` system stored procedure to set up SQL Mail.

To set up SQL Mail using SQL Server 2000 Enterprise Manager:

- Click on the **START** button.
- Point on **PROGRAMS**.
- Point on **MICROSOFT SQL SERVER**.
- Click on **ENTERPRISE MANAGER**.
- Expand the **MICROSOFT SQL SERVERS** container.
- Expand the **SQL SERVER GROUP** container.
- Expand the **instance of SQL Server** for which you want to configure SQL Mail.
- Expand the **SUPPORT SERVICES** container.
- Right-click **SQL MAIL**.
- Click **PROPERTIES** to display the **SQL Mail Configuration** dialog box.
- Select the **Messaging Profile Name** from the **Profile Name** drop-down list.
- Click the **TEST** button to verify that a **mail session** can be established for the Messaging Profile.
- Click **OK**.

To set up SQLAgentMail using SQL Server 2000 Enterprise Manager:

- Click on the **START** button.
- Point on **PROGRAMS**.

- Point on **MICROSOFT SQL SERVER**.
- Click on **ENTERPRISE MANAGER**.
- Expand the **MICROSOFT SQL SERVERS** container.
- Expand the **SQL SERVER GROUP** container.
- Expand the **instance of SQL Server** for which you want to configure SQL Mail.
- Expand the **MANAGEMENT** container.
- Right-click **SQL SERVER AGENT**.
- Click **PROPERTIES** to display the **General** tab on the **SQL Server Agent Properties** dialog box.
- Select the **Messaging Profile** from the **Mail Profile** drop-down list in the **Mail Session** group box.
- Click the **TEST** button to verify that a mail session can be established.
- Click **OK**.

- **SQL Mail** is the mail service of the **SQL Server service** and uses the `sp_sendmail` extended stored procedure to send e-mail from Transact-SQL batches, scripts, stored procedures, and triggers. SQL Mail establishes the mail session if required. The content of a SQL Mail e-mail message can be:
  - a result set from a query;
  - a message string;
  - a Transact-SQL statement or batch for execution; or
  - a page for an electronic pager.

The SQL Server service makes use of the `sp_processmail` system stored procedure, or the `sp_findnextmsg`, `sp_readmail`, and `sp_deletemail` extended stored procedures to process e-mails sent to the domain user account used by the SQL Server service. This e-mail typically contains a Transact-SQL statement or batch for execution, with the result set being returned by reply e-mail, including an optional cc: list.

- **SQLAgentMail** is the mail service of the **SQL Server Agent service**. The SQL Server Agent service starts a mail session upon startup if a mail session is configured in SQL Server Enterprise Manager and sends e-mail and electronic pager notifications to designated users in response to the triggering of an alert or the success or failure of a job.

### 2.1.2 Setting Up Linked Servers

SQL Server 2000 can connect to linked servers. The primary use of a linked server configuration is the execution of distributed queries, joining information from multiple databases on multiple servers. You can use the `sp_addlinkedserver` and `sp_addlinkedsrvlogin` system stored procedures, or SQL Server Enterprise Manager to set up a linked server configuration to any OLE DB data source that can be another database; a text file; a spreadsheet; or the results of full-text content searches.

To set up Linked Servers using SQL Server 2000 Enterprise Manager:

- Click on the **START** button.
- Point on **PROGRAMS**.
- Point on **MICROSOFT SQL SERVER**.

- Click on **ENTERPRISE MANAGER**.
- Expand the **MICROSOFT SQL SERVERS** container.
- Expand the **SQL SERVER GROUP** container.
- Expand the **instance of SQL Server** for which you want to configure Linked Servers.
- Expand the **SECURITY** container.
- Right-click **LINKED SERVERS**.
- Click **NEW LINKED SERVERS** to display **General** tab of the **Linked Server Properties – Linked Server** dialog box.
- Enter the name of the linked server in the **Linked Server** text box or click the **SQL SERVER** radio button in the **Server Type** group box to create a link to a **named instance** of SQL Server and enter the **network and instance name of the SQL Server instance** of the linked server in the **Linked Server** text box.
- Select the **OLE DB provider** from the **Provider Name** drop-down list.
- You click on the **PROVIDER OPTIONS** button to configure the **Linked Server Options** for the selected OLE DB provider.
- Click on the **SECURITY** tab to map and configure local logins to remote logins.
- Click on the **SERVER OPTIONS** tab to configure advanced connection parameters for a linked server.
- Click **OK**.

To use the `sp_addlinkedserver` and `sp_addlinkedsrvlogin` system stored procedures to set up a linked server configuration:

```
EXEC sp_addlinkedserver @server = '<linkedserver_name>' ,
    @provider = '<OLE_DB_provider_name>' ,
    @datasrc = '<data_source>'
GO
EXEC sp_addlinkedsrvlogin '<linkedserver_name>' ,
    'FALSE' ,
    '<login_name>' ,
    '<login_password>'
```

### 2.1.3 Configuring SQL Server XML Support in IIS

You can configure a SQL Server 2000 instance as an XML-enabled database server by configuring an IIS virtual directory linked to SQL Server 2000 support. This enables SQL Server 2000 to provide for:

- **HTTP access;**
- **XML-Data schemas and XPath Queries;**
- The **retrieval and writing** of XML data; and
- The ability to set XML documents as command text and to return result sets as a **stream**.

The **IIS Virtual Directory Management For SQL Server** utility can be used to create a virtual directory within IIS and can be selected by clicking **Configure SQL XML Support In IIS** in the **Microsoft SQL Server** program group.

To create a virtual directory within IIS:

- Click on the **START** button.
- Point on **PROGRAMS**.
- Point on **MICROSOFT SQL SERVER**.
- Click on **CONFIGURE SQL XML SUPPORT IN IIS**.
- Expand the Instance of SQL Server for which you want to configure XML support.
- Right-click **DEFAULT WEB SITE**.
- Point to **NEW**.
- Click **VIRTUAL DIRECTORY** to display the **General** tab of the **New Virtual Directory Properties** dialog box.
- In the **Virtual Directory Name** group box specify a **user-friendly name** for the virtual directory.
- In the **Local Path** group box, specify a path on the local computer to the files that will be made accessible through this virtual directory.
- Click on the **SECURITY** tab
- Specify the **authentication method** users will use to obtain access to SQL Server 2000.
- Click on the **DATA SOURCE** tab
- In the **SQL Server** group box specify the instance of SQL Server 2000 whose data is being published through this virtual directory.
- Click on the **SETTINGS** tab
- Specify the **type of access** to the SQL Server 2000 instance you want to provide by selecting the appropriate check boxes.
- Click **OK** to complete the configuration.

In addition you can click on the **NEW** button in the **Virtual Names** tab of the **New Virtual Directory Properties** dialog box to specify any virtual names you want to create and you can click on the **ADVANCED** tab to specify a different location for the **SZisapi.dll** in the **ISAPI Location** group box, provide additional user settings in the **Additional User Settings** group box, or disable caching in the **Caching Options** group box.

## 2.2 Configuring the Server

### 2.2.1 Configuring the Windows Server Operating System

There are three system settings in the Windows server operating system that affect SQL Server 2000 that you can configure.

- During the installation of SQL Server 2000 on any Microsoft Windows 2000 or Microsoft Windows NT 4.0 server edition, the SQL Server Setup program automatically configures the operating system to **maximize throughput for network applications**. This setting optimizes server memory for distributed applications that perform their own memory caching. Changing this default setting is not recommended. Furthermore, the Full-Text Search feature in SQL Server 2000 requires this setting.
- In addition, the SQL Server 2000 Setup program automatically configures the operating system to run **background and foreground services with equal amounts of processor resources**. This setting is optimal for background tasks. If you are connecting to your SQL Server 2000 instance from a local client, you can improve the responsiveness of the local client and all other foreground applications by optimizing performance for applications.
- SQL Server 2000 is designed to **minimize hard disk paging**. However, Windows 2000 and Windows NT 4.0 virtual memory size and configuration can affect SQL Server 2000 performance. You should, in

general, set the **virtual memory size to 1.5 times the amount of physical memory** on the computer. In addition, if you are using the **Full-Text Search feature**, the virtual memory size should be set to **three times** the amount of physical memory for optimal performance. You can also place paging files on **multiple disks** to improve performance. You should, however, avoid placing paging files on disks containing **data** or **transaction log files**.

### 2.2.2 Configuring the SQL Server Service

Although the SQL Server service is self-tuning and self-regulating and most SQL Server 2000 installations perform optimally using the default settings, manual setting of certain SQL Server service parameters can improve performance. You can manually control the behavior of the SQL Server service in relation to:

- connections;
- databases;
- memory;
- the processor(s); and
- the server.

You can use SQL Server Enterprise Manager or the **sp\_configure** system stored procedure to view or change most SQL Server 2000 configuration settings. Executing the **sp\_configure** system stored procedure with no parameters displays the current settings for all configuration options. Some options are not visible unless you enable Show Advanced Options. A setting of zero for an option indicates that the SQL Server service is managing the option dynamically. After changing a setting with **sp\_configure**, you must either issue the **RECONFIGURE** statement or restart the SQL Server service for the change to take effect. When changing a setting using SQL Server Enterprise Manager, it will prompt you if you must restart the SQL Server service in order for the setting to take effect.

In terms of memory, SQL Server 2000 dynamically allocates memory within the buffer cache to optimize performance. It bases the amount of memory used on the SQL Server 2000 load and competing memory requirements from other server applications. If all available physical memory is already committed to a server application, it takes processor cycles to reallocate memory between server applications. To ensure that physical memory is immediately available for all server applications running on the Windows computer, you can set a minimum and a maximum server memory value. This will cause SQL Server 2000 to allocate and reallocate memory between these minimum and maximum values. However, on a dedicated SQL Server 2000 computer, you can set the minimum and maximum value to the same high value, i.e. the maximum value, to improve performance. In this event, SQL Server 2000 will be allocated memory as required and will retain the allocated memory ' in other words it will not reallocated the memory to other server applications.

If you are running the Full-Text Search feature, you might need to set a maximum memory value so that SQL Server 2000 can reserve sufficient memory for the Microsoft Search service to run optimally. The amount required for the Microsoft Search service depends on the size of tables that contain full-text indexes and the level of full-text query activity.

## 2.3 Automating SQL Server 2000 Administration

SQL Server 2000 allows you to automate a variety of administrative tasks.

### 2.3.1 Operators

The first step in automating administrative tasks is to define **operators**, which are **computers**, **users** or **message groups** that are configured to receive notifications from **SQL Server Agent** using **e-mail**, **pager**, or **NET SEND**; and are to be notified of the success, failure, or completion of an automated task, or on the occurrence of specified events or conditions.

- SQL Server Agent can notify an operator using **e-mail** provided that **SQLAgentMail** has been configured. **SQLAgentMail** requires that the **SQL Server Agent service** use a domain user account. This domain user account must have a **MAPI messaging profile** on the computer on which SQL Server Agent is running.
- SQL Server Agent can also notify an operator using a **pager**. Pager notification is implemented using **e-mail** and **third-party paging software**. Because pager notification relies on e-mail, **SQLAgentMail** must be configured in order to enable pager notification.
- SQL Server Agent can also notify an operator via network pop-up using **NET SEND**, which is available from the **Windows 2000** and **Windows NT 4.0** operating systems and uses the **Windows Messenger service**, which must be running on the recipient computer as well as the sending computer. Messages can be sent to users, computers, or messaging names on the network. A messaging name is an alias that a computer will accept messages for and can be created using the **NET NAME** command-prompt utility.

You can also configure SQL Server Agent to notify a **fail-safe operator** in response to an alert if the designated operator cannot be paged or the SQL Server Agent cannot access system tables in the msdb database.

#### 2.3.1.1 Creating Operators

You can create operators using either **SQL Server Enterprise Manager** or **Transact-SQL** system stored procedures.

To create an operator using **SQL Server Enterprise Manager**:

- Click on the **START** button.
- Point on **PROGRAMS**.
- Point on **MICROSOFT SQL SERVER**.
- Click on **ENTERPRISE MANAGER**.
- Expand the **MICROSOFT SQL SERVERS** container.
- Expand the **SQL SERVER GROUP** container.
- Expand the **instance of SQL Server** for which you want to configure an operator.
- Expand the **MANAGEMENT** container.
- Expand the **SQL SERVER AGENT** container.
- Right-click the **OPERATORS** container.
- Click **NEW OPERATOR** to display the **General** tab of the **New Operator Properties** dialog box.
- Enter a **unique name** for the operator in the **Name** text box.
- Enter the **address information** for all or any of the three notification types.
- Click on the **TEST** button to test the notification configuration.