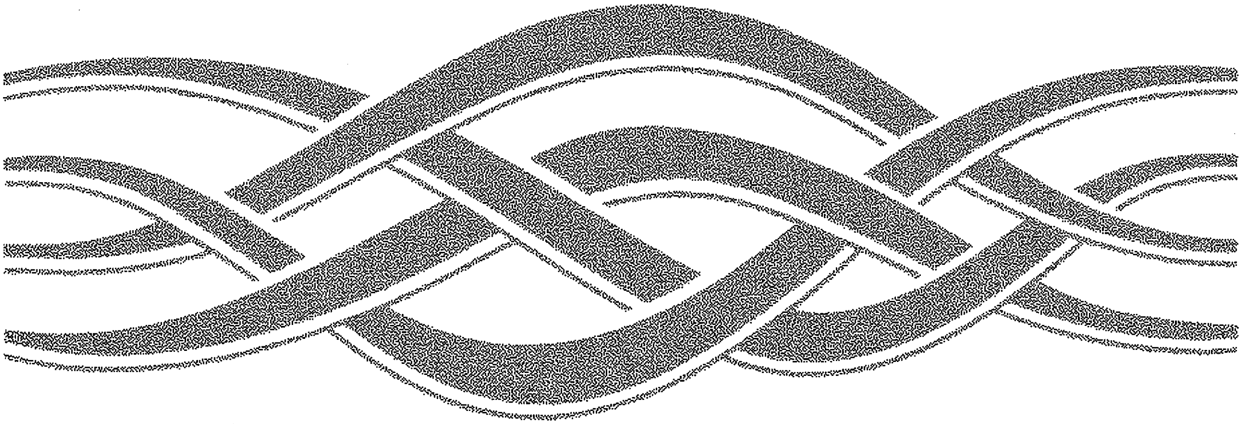
Setup



Microsoft®

SQL SERVER™

Setup

Microsoft® SQL Server™

Version 6.0

For the Microsoft® Windows NT™ Operating System

Microsoft Corporation

Document No. 63904
Printed in the United States of America

Contents

Before You Begin

Microsoft® SQL Server™ 6.0 extends the performance, reliability, and scalability of earlier releases of Microsoft SQL Server to meet the needs of customers who are deploying large-scale distributed information systems. SQL Server 6.0 provides powerful tools for enterprise-wide administration, data replication, parallel DBMS performance, and scalability to very large databases. SQL Server 6.0 also provides tight integration with OLE object technology.

SQL Server 6.0 continues to adhere to industry standards, with improved ANSI SQL support and language enhancements that include declarative referential integrity (DRI), and powerful server-based cursor support that far exceeds the ANSI standard.

About *SQL Server Setup*

Before you install Microsoft SQL Server 6.0, it is recommended that you review the following information about this book, document conventions, additional sources of information, and how to obtain technical support.

Microsoft SQL Server Setup provides instructions for installing the Microsoft SQL Server 6.0 and Microsoft® SQL Workstation 6.0 software. It explains installation options (such as character sets, sort orders, and network support) and how to upgrade from earlier versions of Microsoft SQL Server.

Note that this book does not provide configuration information. For details about how to configure an installed server or client, see the *Microsoft SQL Server Administrator's Companion.*

Document Conventions

This manual uses the following conventions to distinguish elements of text:

Convention	Purpose
UPPERCASE	Represents Transact-SQL® syntax, paths, and filenames.
SMALL CAPITALS	Represent key names, such as CTRL.
bold	Represents system procedures, utility programs, commands, and user-entered text.
italic	Represents database names, table names, and variables.
`monospace`	Represents examples, display text, and error messages.
[brackets]	Identify optional items in syntax. Type only the information within the brackets; do not type the brackets.
{braces}	Identify required items in syntax. Type only the information within the braces; do not type the braces.
| (vertical bar)	Means "or" and signifies that you can choose only one of the items within the brackets or braces.

Finding Further Information

In addition to the introductory information in this book, additional information about setup and details about using SQL Server 6.0 is available from the following sources.

Setup Online Help

To access online Help while using the **setup** program, choose the Help button that is part of most **setup** dialog boxes, or press F1.

Online Help for SQL Server Utilities

To access online Help from most of the graphical tools provided with SQL Server, choose the Help button or press F1.

To access online Help from **isql**, you can execute the **sp_helpsql** system stored procedure. The **sp_helpsql** system stored procedure provides syntax for Transact-SQL statements and system procedures, and information about other SQL Server topics. For more information about **sp_helpsql**, see the *Microsoft SQL Server Transact-SQL Reference*.

SQL Server Books Online

The **setup** program gives you the option of installing Microsoft SQL Server Books Online. If you choose this option, **setup** adds the Books Online icon to the Microsoft SQL Server 6.0 program group. By choosing this icon, you can access online versions of the SQL Server documentation.

Microsoft SQL Server System Administrator's Companion
Incorporates configuration, administration, and troubleshooting topics. Describes server and client configuration. Explains system administration tasks, such as managing devices and databases, backing up and recovering data, performing replication, scheduling tasks, managing security, and monitoring performance. Describes error conditions and gives troubleshooting procedures.

Microsoft SQL Server Database Developer's Companion
Provides an overview of Transact-SQL, which is an enhanced version of SQL (structured query language). Discusses managing data and database objects, such as tables, views, indexes, stored procedures, defaults, rules, constraints, and user-defined datatypes.

Microsoft SQL Server Transact-SQL Reference
Explains how to use Transact-SQL statements and other features to access databases, and provides syntax and descriptions for Transact-SQL statements, system procedures, utility programs, and system tables.

Microsoft SQL Server Setup
An online version of this book, which provides instructions for installing SQL Server 6.0 servers and clients.

Microsoft SQL Server Programming SQL Distributed Management Objects
Provides syntax and reference information for the SQL Distributed Management Objects, which are 32-bit OLE Automation objects for the Microsoft Windows® 95 and Windows NT™ operating systems. These objects, properties, methods, and collections are used to write scripts and programs that can administer multiple SQL Servers distributed across a network.

Microsoft SQL Server Programming DB-Library for C
Provides syntax and reference information for the DB-Library application programming interface (API), which is used to write C/C++ client programs for the Microsoft Windows®, Windows 95, Windows NT, and MS-DOS® operating systems.

Microsoft SQL Server Programming DB-Library for Visual Basic
Provides syntax and reference information for the DB-Library API, which is used to write Microsoft® Visual Basic® programming system–based client programs for the Windows operating system.

Microsoft SQL Server Programming Open Data Services
Provides syntax and reference information for the Microsoft Open Data Services API, which is used to write C/C++ server and gateway programs for the Windows NT operating system.

Obtaining Technical Support

Microsoft offers a variety of support options to help you get the most from Microsoft SQL Server 6.0. For information about these options, see the service and support card included in your SQL Server 6.0 package.

CHAPTER 1

Introduction

Microsoft SQL Server 6.0 (pronounced "Sequel Server") provides a powerful client/server platform for delivering critical business applications efficiently across multiple environments. Microsoft SQL Server 6.0 builds on the power, scalability, and manageability of the Microsoft Windows NT operating system by providing the reliability and advanced capabilities of high-end, client/server database management.

This documentation tells you how to install Microsoft SQL Server 6.0 server software and client software. It details the minimum hardware and software requirements for servers and clients, discusses the options and decisions you will encounter during an installation, takes you through installation, describes how to upgrade an existing installation, provides details about the character set and sort order choices you have during server installation, shows you how to start SQL Server, and tells you how to perform an unattended installation (an optional installation method that lets you install SQL Server 6.0 without having to respond to prompts for information).

SQL Server and SQL Workstation

Microsoft SQL Server 6.0 is a relational database management system that is capable—depending on your application and hardware environment—of supporting hundreds of simultaneous users. It is available in two package configurations: Microsoft SQL Server 6.0 and Microsoft SQL Workstation.

The Microsoft SQL Server 6.0 software is distributed on a compact disc (CD) that contains the **setup** program and server software for all supported hardware platforms. The CD also contains the Microsoft SQL Server client software and client **setup** programs.

Microsoft SQL Workstation 6.0 has the same features and functionality as Microsoft SQL Server, except that it is a *single-user* licensed product that supports a maximum of 15 simultaneous database connections and includes the Microsoft SQL Server 6.0 Programmer's Toolkit (PTK). SQL Workstation is intended to be primarily a development platform.

How to Use This Manual

This documentation is written for those who will install and upgrade Microsoft SQL Server 6.0 servers and clients. It assumes a working familiarity with database software and concepts, operating systems, network software and protocols, and microcomputer hardware.

Chapter 1, “Introduction,” provides a brief overview of the Microsoft SQL Server 6.0 product and this documentation.

Chapter 2, “Server Requirements,” lists the minimum requirements (hardware, software, file system, and so on) necessary for a SQL Server installation.

Chapter 3, “Server Options,” explains the options presented by the **setup** program during a server installation. It is important that you understand these choices before you begin installing the server components.

Chapter 4, “Server Installation,” explains how to install Microsoft SQL Server 6.0 server software components on computers running the Windows NT operating system. It also describes how to upgrade existing server installations or remove existing server installations.

Chapter 5, “Starting the Server,” tells you how to start, pause, continue, and stop SQL Server. It also tells you how to start and stop SQL Executive.

Chapter 6, “Client Installation,” presents client preinstallation requirements and tells you how to install Microsoft SQL Server client components on computers running the Windows NT, Windows, and MS-DOS operating systems. It also describes how to upgrade existing client installations.

Chapter 7, “Installation Troubleshooting,” provides information that you will find helpful if you are unable to install SQL Server successfully. (If you are unable to correct your problem after following the instructions in this chapter, or if you are unable to connect to SQL Server after it is installed, see the *Microsoft SQL Server Administrator’s Companion*, or contact your primary support provider.)

Appendix A, “Character Sets and Sort Orders,” details the character sets and sort orders you choose from during an installation. These determine the types of characters that SQL Server recognizes in your database data, the order in which SQL Server sorts data in response to queries, and the results that SQL Server returns for queries that involve character comparisons.

Appendix B, “Unattended Installation,” tells you how to perform an unattended installation, which allows you to install SQL Server 6.0 without having to respond to prompts for information from the **setup** program.

Installation Terminology

To help you understand the information presented in this documentation, we begin by defining some terms you will encounter while installing SQL Server. This is a brief list of definitions and is not intended to serve as a comprehensive SQL Server glossary. Rather, it is provided as a quick reference to help you understand some of the terms that are discussed in this documentation. You may find it useful to refer back to this list of definitions as you read through the other chapters of this documentation.

application log

The Windows NT application log is a special file maintained by Windows NT. It can be viewed only by using the Windows NT Event Viewer. When SQL Server is configured to use the Windows NT application log, each SQL Server session writes new events to that log. Unlike the SQL Server error log, a new application log is not created each time you start SQL Server.

auto-start options

By default, the **setup** program installs SQL Server and SQL Executive to run as manually started Windows NT services. Optionally, each can be configured to run as an automatically started service. For each service, this choice is called the auto-start option.

backup domain controller

BDC. In a Windows NT domain, a BDC is a server that receives a copy of the domain's security database from the primary domain controller and shares the user login authentication load.

Books Online

An online version of the SQL Server documentation set, SQL Server Books Online is an installation option presented by the **setup** program. If you choose this option, the SQL Server Books Online icon is added to the Microsoft SQL Server 6.0 program group. By choosing that icon, you can display SQL Server user documentation on your computer screen.

A particular benefit of Books Online is the full text search capability, which allows you to very rapidly search a book or a set of books for all instances of a word or phrase.

character set

The selected character set determines the types of characters that SQL Server recognizes in the databases. A character set is a set of 256 letters, numbers, and symbols specific to a country or a language. The printable characters of the first 128 values are the same for all character sets. The last 128 characters, sometimes referred to as extended characters, are different.

During installation it is critical that you select the appropriate character set. If you need to change it later, you will need to rebuild your databases and reload your data.

client

A front-end application that uses the services provided by a server. The computer that hosts the application is often referred to as the client computer.

SQL Server client software enables computers to connect to a SQL Server over the network.

database

A collection of information, data tables, and other objects that are organized and presented to serve a specific purpose, such as facilitation of searching, sorting, and recombination of data. Databases are stored on devices.

device

A file that databases are stored in. One database can be stored on several devices. SQL Server uses two types of devices: database devices, which store databases, and dump devices, which store backups of a database.

domain

In Windows NT, a domain is a collection of computers that are grouped for viewing and administrative purposes and that share a common security database. The security database is maintained on the primary domain controller, with copies on backup domain controllers.

error log

The SQL Server error log records information from SQL Server. You can view the error log by using SQL Enterprise Manager or any text editor. When SQL Server is configured to use its error log, each SQL Server session creates a new log, and SQL Server retains the last six logs created.

FAT file system

A method for managing disk storage. A FAT (file allocation table) file system is used by an operating system (for example, by the MS-DOS operating system) to keep track of the status of various segments of disk space used for file storage.

file system

The portion of an operating system that translates requests for file operations from an application program into low-level, sector-oriented tasks that can be understood by the drivers that control the disk drives.

SQL Server is usually installed on disk drives that are formatted for the NTFS or FAT file systems. It can be installed on a compressed NTFS volume, but at a performance penalty.

install

To set up the software to work with the computer.

An installation differs from an upgrade in that an installation provides a fresh version of the software that uses default (or user-set) configuration settings. Pre-existing user databases and data, security information, and configuration settings (if any) are not retained.

installation path

For SQL Server, the installation path is the drive and directory into which the SQL Server files will be copied. The default is C:\SQL60, although this can be set by the user who is installing.

After installation, this specific drive and directory are often referred to as the SQL Server root directory.

master **database**

The *master* database controls the user databases and the operation of SQL Server as a whole. It is automatically installed with SQL Server. It keeps track of such things as user accounts, remote user accounts, remote servers that this server can interact with, ongoing processes, the configurable environment variables, system error messages, the databases on SQL Server, the storage space allocated to each database, the tapes and disks available on the system, and the active locks.

MASTER device

The device that is automatically installed with SQL Server and is used to store the *master*, *model*, and *tempdb* system databases and transaction logs, and the *pubs* sample database and transaction log. The filename of the MASTER device is MASTER.DAT.

Net-Library

A library of functions for managing network connections and routing. Each Net-Library allows a computer to use a particular network protocol.

network adapter

An expansion card or other physical device used to connect a computer to a local area network.

NTFS

An advanced method for managing disk storage, designed for use specifically with the Windows NT operating system. It supports long filenames (but maintains shorter filenames for compatibility with computers running operating systems using the FAT file system). It also provides full security for access control, file system recovery, and extremely large storage media. It supports object oriented applications by treating all files as objects with user- and system-defined attributes.

primary domain controller

PDC. In a Windows NT domain, a PDC is a server that maintains the domain's security database and authenticates user logons. It also provides a copy of the domain's security database to backup domain controllers (BDCs), which share the user login authentication load.

***pubs* database**

A sample database provided with Microsoft SQL Server. The *pubs* database can be helpful when you are learning to use SQL Server 6.0. All Microsoft SQL Server 6.0 documentation uses the *pubs* sample database as the basis for examples.

remote server

A SQL Server that can be accessed through a user's local server. The **setup** program can install, upgrade, or configure remote servers.

script

A collection of statements used to create database objects. Transact SQL scripts are saved as files, usually ending with the SQL filename extension.

server

A computer on a local-area network (LAN) that controls access to resources such as files, printers, and communication devices.

When you use SQL Server, you can use only servers that are designated SQL Server servers (that have the SQL Server server software installed).

setup

SETUP.EXE is the program that you run to install, upgrade, or reconfigure SQL Server servers or clients. Different **setup** programs are supplied for each server hardware platform and—for Microsoft clients—each operating system.

setup initialization file

A text file, using regular Windows .INI file format, that stores configuration information and allows SQL Server 6.0 to be installed without a user having to be present to respond to prompts for information from the **setup** program.

sort order

A sort order is a set of rules that determines how SQL Server collates and presents data in response to database queries. The sort order determines the order in which data is presented in response to SQL Server statements involving GROUP BY, ORDER BY, and DISTINCT. The sort order also determines how certain queries are resolved, such as queries involving WHERE and DISTINCT.

During installation it is critical that you select the appropriate sort order. If you later need to change it, you will need to rebuild your databases and reload your data.

SQL Enterprise Manager

A graphical application that allows for easy enterprise-wide configuration and management of SQL Server and SQL Server objects. SQL Enterprise Manager provides a powerful scheduling engine, administrator alert capability, drag-and-drop control operations across multiple servers, and a built-in replication management interface. You can also use SQL Enterprise Manager to manage logins, permissions, and users; create scripts; manage devices and databases; back up databases and transaction logs; and manage tables, views, stored procedures, triggers, indexes, rules, defaults, and user-defined datatypes.

SQL Enterprise Manager is installed by the **setup** program as part of the server software on Windows NT–based computers, and as part of the client software on Windows NT– and Microsoft® Windows® 95–based computers. Because SQL Enterprise Manager is a 32-bit application, it cannot be installed on computers running 16-bit operating systems.

SQL Executive

A component of SQL Enterprise Manager, SQL Executive provides powerful scheduling capabilities that enable replication and additionally allow you to schedule other SQL Server events, including backups, DBCC operations, and other administrative maintenance tasks. SQL Executive replaces SQL Monitor, which was included in earlier releases of SQL Server.

SQL Server

Microsoft's a relational database management system that is capable—depending on your application and hardware environment—of supporting hundreds of simultaneous users.

standalone installation

An installation of SQL Server on a computer that is not connected to a network.

A standalone installation may be used to install SQL Server on a computer that will soon be but is not yet connected to a network (if for example, it does not yet have a network adapter card installed.) Or a standalone installation might be performed on a computer that is intended as a development system that will have no need for network connections.

transaction log

A reserved area of the database in which a log of all changes to the database is recorded. It is stored in the syslogs system table and is used by SQL Server during automatic recovery.

unattended installation

Unattended installation allows you to install SQL Server 6.0 without having to respond to prompts for information from the **setup** program. Instead, you create an initialization file, save the initialization file on a storage device accessible to the computer that will be set up, and start setup using some required options. During setup the configuration parameters are read from the initialization file.

upgrade

To change to a newer, usually more powerful and sophisticated version of the software. An upgrade differs from an installation in that an upgrade leaves in place existing user databases and data, security information, and many (but not necessarily all) configuration settings.

CHAPTER 2

Server Requirements

Before you install the server components, check that your system meets the minimum requirements detailed in this chapter.

Hardware Platforms

Microsoft SQL Server 6.0 can be installed on computers that are based on these microprocessor architectures:

- Alpha AXP™ microprocessors
- Intel® 32-bit x86-based microprocessors (such as the 80486 or the Pentium™)
- MIPS® microprocessors

For more information about supported hardware, see the Windows NT Hardware Compatibility List provided with your Windows NT software.

Hardware and Software Requirements

To install the Microsoft SQL Server 6.0 server software on a Windows NT–based computer, you need the following hardware and software:

You need	Your choices
Computer	Refer to the Windows NT Hardware Compatibility List for the processor architecture that corresponds to the SQL Server platform you are installing (that is, Intel, Alpha AXP, or MIPS). For example, if you are installing SQL Server for the Intel platform, you can choose any of the 32-bit Intel x86-based microprocessors, such as the 80486 or the Pentium.

You need	Your choices
Minimum memory	16 MB. Additional memory is recommended. If the server will be configured to participate in data replication as either a distribution server or a combined publication/distribution server, it must have at least 32 MB of memory, with at least 16 MB assigned to SQL Server.
Disk drive space	For a new installation, a minimum of 60 MB of available disk space for Intel-based computers, Alpha AXP–based computers, and MIPS-based computers. If you increase the size of the MASTER device from the default, additional space is required. If you install SQL Server Books Online to run from the hard disk, another 15 MB of available disk space is required; if you install SQL Server Books Online to run from the CD, only 1 MB of additional space is needed. For an upgrade from Microsoft SQL Server 4.2, approximately 50 MB of available hard disk space is required for all processor architectures.
Operating system	Microsoft SQL Server 6.0 software must be installed on a computer running the Microsoft Windows NT Server operating system, version 3.5 or later. Version 3.51 is recommended. Microsoft SQL Workstation 6.0 software can be installed on computers running the Windows NT Server or Windows NT Workstation operating systems, version 3.5 or later. Version 3.51 is recommended. SQL Server 32-bit client software can be installed on computers running the Windows NT Server, Microsoft Windows NT Workstation, or Microsoft Windows 95 operating systems. For more information see Chapter 6, "Client Installation."
Network software	SQL Server runs on the Windows NT platform using the built-in network software. Unless Banyan® VINES® or DEC™ PATHWORKS™ will be used, additional network software is not required. Full support for Novell® NetWare® clients is provided via NWLink.
Network adapter	Any of the network adapters supported by Windows NT. (If you're running a standalone SQL Server, you do not need a network adapter.) For information about supported network adapters, see your documentation for Windows NT.

Note Microsoft SQL Server 6.0 is not recommended for installation on a Primary Domain Controller (PDC) or a Backup Domain Controller (BDC), because those computers perform the resource-intensive tasks of maintaining and replicating the network accounts database and performing network login authentications.

File Systems

Server performance is not affected by the file system used (FAT or NTFS). Your choice of file system should be decided by factors other than performance.

- The FAT file system allows dual booting with computers running MS-DOS.
- The NTFS file system has security and recovery advantages. If you do not need to dual boot Windows NT with MS-DOS, NTFS is recommended.

  SQL Server can be installed on a compressed NTFS volume, but at a performance penalty.

For information about choosing the appropriate file system, see your documentation for Windows NT.

Server Names

The server name used by SQL Server is taken from the computer's name. Before installing Microsoft SQL Server on a server computer, make sure that the server's computer name is a valid SQL identifier.

The first character must be a letter or an underscore (_). Characters following the first character can include letters, numbers, or the symbols #, $, or _. No embedded spaces are allowed.

If other characters are used in the computer name, rename the computer before installing SQL Server. For instructions on renaming computers, see your documentation for Windows NT. Also, if the renamed computer is a member of a Windows NT Server domain, contact your network administrator to have the new computer name added to the domain.

SMS Compatibility

Microsoft SQL Server 6.0 is compatible with Microsoft® Systems Management Server (SMS) version 1.0A. However, it is not compatible with version 1.0. During an installation or an upgrade, if SMS version 1.0 is detected, the SQL Server **setup** program will display a warning message.

CHAPTER 3

Server Options

This chapter explains the options presented by the **setup** program during a server installation. It is important that you understand these choices before you begin installing the server components.

Name, Company, and Product ID

During installation the setup program will prompt you to supply a name, company, and product ID.

Name
Is the name of a user who is responsible for using or administering this server. This is a required entry.

Company
Is the name of your company or other organization. This is an optional entry.

Product ID
Is an identifier provided on the software's CD box. Although this is an optional entry, it is strongly recommended that you enter the product ID, because it is used for product support purposes.

Remote Installation

You can use the SQL Server **setup** program to install the software on a remote computer. When you do so, you are prompted for the following information:

Remote Server Name

The computer name for the remote computer (for example, REMOTE).

SQL Installation Drive

The remote logical drive on which to install the SQL Server software. The default is C.

For example, to install SQL Server on the default share for the D drive (D$) on a remote computer, you would specify remote logical drive D for the SQL Installation Drive.

Windows NT Installation Drive

The remote logical drive on which the Windows NT root is already installed. This is where the **setup** program copies SQL Server client dynamic-link libraries (DLLs) and DLLs required only during installation. The default is the current setting for the SQL Installation Drive.

For example, if the Windows NT root for a remote computer is on the default share for its D drive (D$), you would specify remote logical drive D for the Windows NT Installation Drive.

Master Database Installation Drive

The remote logical drive on which to install the SQL Server *master* database. The default is the current setting for the SQL Installation Drive.

For example, to install the *master* database on the default share for the D drive (D$) on a remote computer, you would specify remote logical drive D for the Master Database Installation Drive.

Remote Logical Drive

The remote logical drive for the currently selected option (the SQL Server Installation Drive, the Windows NT Installation Drive, or the Master Database Installation Drive).

The default remote logical drive for the SQL Server Installation Drive is C. The default for the other two options is the current setting of the SQL Server Installation Drive.

On the local computer, the **setup** program assigns as many as three unused drive letters to correspond to the logical drives on the remote computer. This is *not* the same drive letter as the remote logical drive (which corresponds to the actual drive on the remote computer on which the files are to be installed). For example, if you're installing SQL Server on the D drive of a remote computer named \\REMOTE from a local computer named \\LOCAL, you would enter REMOTE as the Remote Server Name and D as the SQL Installation Drive.

On the local computer, the **setup** program assigns a unique logical drive letter (for example, X) for the drive on the remote computer (in this example, \\REMOTE\D$). You cannot change the logical drive letter assigned by the **setup** program. During the installation process, when prompting for a path, the **setup** program adds this unique logical drive letter to the default. In this example, when prompting for the SQL Server root directory, the **setup** program would use X:\SQL60 to correspond to \\REMOTE\D$\SQL60.

After you've installed the software on the remote computer, you can use SQL Enterprise Manager or SQL Service Manager to start and stop the remote SQL Server. For more information, see Chapter 5, "Running SQL Server and Running SQL Executive."

Root Directory

The default SQL Server root directory is C:\SQL60, although during installation the **setup** program allows you to change this setting. SQL Server directory names must conform to the FAT 8.3 naming convention, even if the directory is installed on an NTFS volume. Do not use spaces in the directory name.

The following directories are created under the SQL Server root directory during installation:

Directory	Contents
BIN	MS-DOS– and Windows-based client executable files and Windows dynamic-link library (DLL) files
BINN	Windows NT–based client and server executable files and online Help files
CHARSETS	Character set and sort order files
DATA	Database devices
DLL	Dynamic-link library files
INSTALL	Installation scripts and output files
LOG	Error log files
REPLDATA	Used by replication as the working directory for the distribution database
SAMPLE	Sample files
SYMBOLS	Files and libraries that can be used to generate stack dumps and traces on computers running Windows NT 3.51 or later.

MASTER Device Options

When you install SQL Server the **setup** program creates the MASTER database device, which is used to store the *master*, *model*, and *tempdb* system databases and transaction logs and, optionally, the *pubs* sample database and transaction log. During installation, the **setup** program prompts you to provide settings for these MASTER database device options:

Drive
The drive on which the MASTER database device will be installed. The default is C.

Directory
The path and filename for the MASTER database device. The default is \SQL60\DATA\MASTER.DAT.

The directory and filenames must conform to the FAT 8.3 naming convention, even if installed on an NTFS volume.

MASTER Device Size (MB)

The space to be allocated to the MASTER database device. The default (and minimum) size is 25 MB. If you have sufficient disk space, consider making the MASTER database device even larger. This will save you time later, as your need for additional space grows.

Note that if you later determine you need additional device space, you can use DISK RESIZE to increase the size of the MASTER database device. For information about DISK RESIZE, see the *Microsoft SQL Server Transact-SQL Reference*.

Character Set

A character set is a set of 256 uppercase and lowercase letters, numbers, and symbols. The printable characters of the first 128 values are the same for all character set choices. The last 128 characters, sometimes referred to as *extended characters*, differ from set to set. Because there are more than 256 characters used by the languages of the world, each character set has a somewhat different "repertoire" of extended characters.

The selected character set determines the types of characters that SQL Server recognizes in your database data. You should use the same character set for both the client and the server, or your results can be unpredictable. (You must also use corresponding character sets when bulk loading data.) However, if your databases use only the first 128 characters of a character set (if, for example, you don't use characters with diacritical marks), it makes no difference which character set you use because the first 128 characters are the same for all SQL Server character set choices.

Important It is critical that you select the correct character set when you first install SQL Server. If you need to change character sets after you've installed SQL Server, you must rebuild your databases and reload your data.

SQL Server offers three character sets:

ISO 8859-1 (Latin 1 or ANSI)

The default character set. It is compatible with the ANSI characters used by the Microsoft Windows and Windows NT operating systems.

Code page 850 (Multilingual)
Includes all characters for most languages of European, North American, and South American countries. Use this character set when you have MS-DOS–based client applications that use extended characters.

Code page 437 (U.S. English)
Although the most commonly used character set in the United States, includes many graphics characters not usually stored in databases. Unless you have an overriding reason to select this character set, choose code page 850, which provides more compatibility with languages other than United States English.

Note Additional character sets and sort orders are being developed for international users and may be included in those listed by the **setup** program.

For additional information about SQL Server character set choices, see Appendix A, "Character Sets and Sort Orders."

Sort Order

A sort order is a set of rules that determines how SQL Server collates and presents data in response to database queries. The sort order determines the order in which data is presented in response to SQL Server statements involving GROUP BY, ORDER BY, and DISTINCT. It also determines how certain queries are resolved, such as those involving WHERE and DISTINCT.

Important It is critical that you select the correct sort order when you first install SQL Server. If you need to change sort orders after you've installed SQL Server, you must rebuild your databases and reload your data.

SQL Server offers several sort orders:

- Dictionary order, case-insensitive (the default)
- Binary order
- Dictionary order, case-sensitive
- Dictionary order, case-insensitive, uppercase preference
- Dictionary order, case-insensitive, accent-insensitive
- Strict compatibility with version 1.*x* case-insensitive databases

- Alternate dictionary order, case-sensitive
- Alternate dictionary order, case-insensitive
- Alternate dictionary order, case-insensitive, uppercase preference
- Alternate dictionary order, case-insensitive, accent-insensitive
- Scandinavian dictionary order, case-sensitive
- Scandinavian dictionary order, case-insensitive, uppercase preference

The version 1.*x*-compatible sort order applies only to code page 850.

The following table shows how the sort order affects comparisons and sorting:

Sort order	Comparison examples	Sorting example
Dictionary order, case-insensitive	A = a, Ä = ä, Å = å, a ≠ à ≠ á ≠ â ≠ ä ≠ å, A ≠ Ä ≠ Å	A, a, à, á, â, ä, Ä, Å, å (See Note 2)
Binary	See Note 1	See Note 1
Dictionary order, case-sensitive	A ≠ a, Ä ≠ ä, Å ≠ å, a ≠ à ≠ á ≠ â ≠ ä ≠ å, A ≠ Ä ≠ Å	A, a, à, á, â, Ä, ä, Å, å
Dictionary order case-insensitive, uppercase preference	A = a, Ä = ä, Å = å, a ≠ à ≠ á ≠ â ≠ ä ≠ å, A ≠ Ä ≠ Å	A, a, à, á, â, Ä, ä, Å, å
Dictionary order case-insensitive, accent-insensitive	A = a = à = á = â = Ä = ä = Å = å	A, a, à, á, â, Ä, ä, Å, å
Strict compatibility with version 1.*x* case-insensitive databases	See Note 3	See Note 3
Custom	See Note 4	See Note 4

1 Comparisons and sorting for this sort order are based on the numeric values of the characters in the installed character set.

2 This is just one example of how the characters might be sorted. Because this sort order does not have uppercase preference, it is faster than the corresponding sort order with uppercase preference, but you cannot predict whether an uppercase character will come before or after its corresponding lowercase character.

3 For the first 128 characters, comparisons and sorting for this sort order are the same as the case-insensitive dictionary sort order. For the last 128 characters, comparisons and sorting are the same as the Binary sort order.

4 Comparisons and sorting for a custom sort order depend on the sort order.

The four alternate sort orders and the two Scandinavian sort orders provide results similar to their counterparts. For details about how these sort orders differ from their counterparts, see Appendix A, "Character Sets and Sort Orders."

When upgrading a version 4.2*x* SQL Server, **setup** installs the new program files but does not change the installed sort order.

For additional information about SQL Server sort order choices, see Appendix A, "Character Sets and Sort Orders."

Network Support

Microsoft SQL Server uses network libraries, known as Net-Libraries, to pass network packets back and forth between clients and a SQL Server. The Net-Libraries, implemented as dynamic-link libraries (DLLs), perform the network operations required to communicate using specific interprocess communication (IPC) mechanism protocols.

SQL Server includes a set of server Net-Libraries that enable simultaneous connections from clients running named pipes or other interprocess communication (IPC) mechanisms. Named pipes are installed by default (and it is recommended that named pipes not be removed, even if other Net-Libraries are chosen). During installation, the **setup** program gives you the option of setting up one or more of these other Net-Libraries on the server:

Multi-Protocol

The Multi-Protocol Net-Library takes advantage of the powerful remote procedure call (RPC) facility of Windows NT. Unlike other Net-Libraries, Multi-Protocol Net-Library need only be selected as an option from the Select Network Protocols dialog box; it does not require the entry of any configuration parameters. Multi-Protocol Net-Library:

- Communicates over most IPC mechanisms supported by Windows NT. (Note that for SQL Server 6.0, only TCP/IP Windows Sockets, NWLink IPX/SPX, and Named Pipes are considered tested and supported.)
- Allows the use of integrated security over all protocols that RPC supports (including Windows-based Novell clients using SPX or IPXODI).
- Supports encryption for both user password authentication and data.
- Offers performance comparable to native IPC Net-Libraries for most applications.

NWLink IPX/SPX

This Net-Library allows SQL Server to communicate using the NWLink IPX/SPX protocol.

If you set up SQL Server to listen on NWLink IPX/SPX, the **setup** program prompts you for the Novell Bindery service name in which to register SQL Server on the Novell network. The default service name is the computer name of the server computer. The Net-Library allows Novell SPX clients to connect.

TCP/IP Sockets

This Net-Library allows SQL Server to communicate using standard Windows Sockets as the IPC method across the TCP/IP protocol.

If you set up SQL Server to listen on TCP/IP, the **setup** program prompts you for the TCP/IP port number SQL Server will listen on when accepting connections from TCP/IP sockets clients. The default is 1433, the official Internet Assigned Number Authority (IANA) socket number for Microsoft SQL Server.

Banyan VINES

SQL Server supports Banyan VINES Sequenced Packet Protocol (SPP) as the IPC method across the Banyan VINES IP network protocol. Banyan VINES support for Windows NT–based clients and servers is available only for SQL Server on the Intel platform; it is not currently available on the MIPS or Alpha AXP platforms.

If you set up SQL Server to listen on Banyan VINES, the **setup** program prompts you for a StreetTalk™ PC-based service name. This has the form *servicename@group@org*, where *servicename* is the StreetTalk PC-based service name used by SQL Server, *group* is the group, and *org* is the organization. The PC-based service name used by SQL Server must first be created using the MSERVICE program included with your VINES software. Also, to start SQL Server you must be logged in with administrative privileges.

Note that before you can configure SQL Server to listen on Banyan VINES, you must install the Banyan VINES software (available from Banyan).

AppleTalk ADSP

For Intel-, MIPS-, and Alpha AXP–based computers, server-side AppleTalk® (ADSP) Net-Libraries are provided for use with Microsoft SQL Server. The server-side AppleTalk Net-Library allows Apple® Macintosh®–based clients to connect to SQL Server via native AppleTalk (as opposed to TCP/IP).

If you set up SQL Server to listen on AppleTalk, the **setup** program prompts you for the AppleTalk service object name. The AppleTalk service object name is assigned by your system administrator, who might want to use the machine name of the server for consistency. For example:

ACCOUNTING1

Note that it is not necessary to enter the AppleTalk zone, because the local zone will be used when registering the service.

DECnet

For Intel-, MIPS-, and Alpha AXP–based computers, server-side DECnet™ sockets Net-Libraries provide connectivity with PATHWORKS networks by allowing clients running on VMS™ to connect to Microsoft SQL Server via DECnet sockets.

If you set up SQL Server to listen on DECnet sockets, the **setup** program prompts you for the DECnet object ID that SQL Server will listen on when accepting connections from clients under this Net-Library.

The value of the DECnet object ID is set by your System Administrator; it should not conflict with other services on the system listening on DECnet sockets. The object ID can be either numeric or alphanumeric. To specify a numeric ID, preface it with a pound (#) sign. For example:

#150

To specify an alphanumeric value, type the ID of your choice. For example:

SQL1

For detailed information about object IDs, see your VMS documentation.

A server can listen on multiple Net-Libraries at one time. In addition to listening on the other server Net-Libraries you select, by default SQL Server always listens on named pipes as well.

By default, SQL Server listens on the standard pipe, \\.**\pipe\sql\query**, for named pipe connections. You can change the pipe name. You can also drop named pipe support and set SQL Server to listen only on other Net-Libraries.

You cannot change Net-Library configurations during an upgrade. The existing Net-Library configurations are carried over and can be modified after upgrade using SQL Setup from the Microsoft SQL Server 6.0 program group.

Auto-Start Options

By default, the **setup** program installs SQL Server and SQL Executive to run as manually started Windows NT services. However, during installation **setup** gives you the option to install each as an automatically started service.

SQL Server Books Online

During installation the **setup** program gives you the option of installing the SQL Server Books Online. This option adds the SQL Server Books Online icon to the Microsoft SQL Server 6.0 program group. By choosing this icon, you can view SQL Server documentation online.

You can install SQL Server Books Online to run from disk or the CD. You can also choose not to install SQL Server Books Online.

Installing SQL Server Books Online to run from the CD requires only about 1 MB of available disk space. Installing SQL Server Books Online to run from the disk requires about 15 MB of available disk space and provides faster performance.

SQL Executive User Account

Usually, SQL Server and SQL Executive are started and run as Windows NT services. This integration with the Windows NT Service Control Manager allows SQL Server and SQL Executive to be started and stopped as services, either locally or remotely.

Windows NT services run in the security context of a user account that they are assigned. By default, the system assigns the computer's LocalSystem account to services. However, services can have a specifically created Windows NT user account assigned to them.

Before you install SQL Server, you should create a Windows NT user account that can be assigned to the SQL Executive service. This account will enable the service to accomplish connectivity-related tasks, such as performing replication or running scheduled tasks that involve other servers.

During SQL Server installation, the **setup** program prompts you for the domain, username, and password of a Windows NT user account, which it then assigns to SQL Executive as its user account. (By default, the **setup** program always assigns SQL Server the LocalSystem account; you can change the account after installation by using the Services application in the Control Panel.) For more information, see Chapter 4, "Server Installation."

Licensing Mode

Microsoft SQL Server 6.0 (a Microsoft Back Office server product) requires a Server License, plus a Client Access License for each client computer that accesses the server. You must acquire these licenses prior to using the product. A Server License is provided with the server software. Client Access Licenses are purchased separately.

Client Access Licenses can be used in one of two licensing modes. This choice of modes offers you the flexibility to choose the option that best meets your requirements:

- *Per Seat.* Each computer or workstation that will be accessing or otherwise using any SQL Server—using either Microsoft-provided client software or third-party software—requires a Client Access License.
- *Per Server.* You specify a maximum number of simultaneous user connections for the server. An equal number of Client Access Licenses are required for that server.

During installation of a server, you are asked to choose between these two licensing modes. Before the **setup** program can proceed, you must make your selection and then accept the terms of the license agreement.

If you are unsure which licensing mode to choose, select the Per Server option. After installation of a server, you are permitted to convert from Per Server mode to Per Seat mode at no additional cost. This is a one-time, one-way conversion option. It is not necessary to notify Microsoft if you elect to make this change. To change licensing mode to Per Seat after installation, use the Licensing application in the Control Panel. For more information, see the online Help for the Licensing application.

You are not legally permitted to change the licensing mode from Per Seat to Per Server.

To review the licensing agreement, see the *Client License Access Agreement* or the online Help for the Licensing application in the Control Panel.

Per Seat Mode

Under the Per Seat licensing mode, each client computer requires a Client Access License to connect to any Microsoft SQL Server 6.0 computer for data access purposes. Once a client computer is licensed, it may be used to access any Microsoft SQL Server computer at no additional charge.

If you select the Per Seat licensing mode, any number of licensed computers can be used to connect at the same time to any Microsoft SQL Server. You must purchase a separate Client Access License for each computer, even if you use client data access software from Microsoft, from a third-party vendor, or from any other client software supported by Windows NT Server.

The Per Seat option is often the most economical one for networks in which clients tend to connect to more than one server.

To review the Per Seat licensing agreement, see the *Client License Access Agreement* or the online Help for the Licensing application in the Control Panel.

Per Server Mode

With Per Server licensing, each Client Access License is assigned to a particular database server and allows one client to connect to that server for data access. You must have at least as many Client Access Licenses dedicated to that server as the maximum number of client computers that are connected to that server at one time.

If you select the Per Server option, during server installation you specify the number of Client Access Licenses (corresponding to the maximum number of concurrent client connections) that you have purchased for that server.

Note When a server reaches the maximum number of concurrent connections, Microsoft SQL Server 6.0 displays an error message and will not allow another computer to connect until at least one currently connected client disconnects.

This mode of licensing is available for Microsoft SQL Server 4.21a or later. The Per Server option is often the most economical one for networks in which clients tend to connect to only one occasional-use or special-purpose server.

To review the Per Server licensing agreement, see the *Client License Access Agreement* or the online Help for the Licensing application in the Control Panel.

Node-based Client Access

Node-based client access permits all data access connections originating from a single client (one unique workstation and/or computer) to be counted against a single Client Access License. Both client licensing modes provide support for node-based client access.

Beginning with version 6.0, Microsoft SQL Server client access software supports this model. This capability is available for both ODBC and DB-Library and allows Microsoft SQL Server 6.0 to automatically detect multiple data access connections originating from the same networked client. When this occurs, those connections are then counted against a single Client Access License. Application coding changes are not required to take advantage of this new node-based client access capability.

When counting Client Access Licenses, SQL Server takes advantage of this new data access client capability whenever possible. However, there are two client configurations in which clients will be counted on a per individual connection basis:

- **Previous versions of client access software**

  Because these clients do not support node-based data access, SQL Server cannot detect multiple connections originating from the same node. Each connection counts as one client.
- **Non-Microsoft client access software**

  Data access software from other vendors may not support node-based data access. If it does not, each connection counts as one client.

Increasing Client Access Licenses After Server Installation

Client Access Licenses may be added at any time.

If a SQL Server is licensed using the Per Server mode, after you purchase the additional licenses they must be applied to that server. Use the Licensing application in the Control Panel of the server to increase the number of users to equal the total number of Client Access Licenses purchased for that server.

For more information, see the online Help for the Licensing application in the Control Panel.

CHAPTER 4

Server Installation

This chapter explains how to install Microsoft SQL Server 6.0 server software components on computers running the Windows NT operating system. It also describes how to upgrade existing server installations.

For information on configuring an installed server, see the *Microsoft SQL Server Administrator's Companion.*

Creating a User Account for SQL Executive

During SQL Server installation, the **setup** program asks you to provide the domain, username, and password of a Windows NT user account, which it then assigns to SQL Executive. (As an option, you can select the server's LocalSystem account, but using the LocalSystem account prevents the use of several features, such as replication and running scheduled tasks involving other servers.)

Before installing SQL Server, use the User Manager application (included as part of Windows NT Server) to create a user account that you will assign to SQL Executive. This account will usually be a domain user account (although it can also be a user account on the local server). This account should be reserved for use by SQL Executive (users should not use it to log on).

The SQL Executive user account must belong to the Administrators local group on the SQL Server computer. In addition, it is a good idea to select the Password Never Expires option, be granted the Log On As A Service right on the SQL Server computer, and have all logon hours allowed (although this is only necessary for domain accounts, since this option does not apply to accounts on the local server).

Several SQL Servers can share the same user account. In fact, when setting up replication it is a good idea for a publication server and all its subscription servers to share the same account.

During SQL Server installation, if you do not know (or have not yet created) this Windows NT user account, you can complete the installation without specifying a user account. In this case, the **setup** program will install SQL Executive using the computer's LocalSystem account. Later, you can use the Services application in the Control Panel (or the Services command in Server Manager) to assign an appropriate user account to SQL Executive. Note, however, that if SQL Executive is assigned the LocalSystem account, it runs in the security context of that account and can be prevented from accomplishing many connectivity-related tasks.

When upgrading from SQL Server version 4.2*x* to SQL Server 6.0, you will be prompted for user account information in the same manner as for a new SQL Server 6.0 installation.

For information about creating Windows NT user accounts, granting advanced user rights, setting password expiration, and managing group memberships, see your documentation for Windows NT or the User Manager online Help.

Note After you have assigned a domain user account to SQL Executive, if you start SQL Executive and the server cannot for any reason access a domain controller, the logon attempt will fail and SQL Executive will not start. You can start it by running the Services application from the Control Panel (or by using the Services command from Server Manager) and assigning the LocalSystem account to SQL Executive. However, remember that if SQL Executive is assigned the LocalSystem account, it runs in the security context of that account and can be prevented from accomplishing many connectivity-related tasks.

It is important that the password for the Windows NT user account not be changed or allowed to expire. If SQL Executive will not run after having been assigned a Windows NT user account, the password may have been changed or there may not be a domain controller available to validate the logon. If the password has changed, use User Manager to set a new password for that account. Use the Services application in the Control Panel to reenter the new password for the SQL Executive service.

Installing SQL Server

After you've reviewed the information in Chapter 3, "Server Options," and after you've created a user account for SQL Server, you're ready to install Microsoft SQL Server.

You can install the software directly from the Microsoft SQL Server 6.0 CD, or you can copy the software onto a network drive and install from there. If you plan to install from a network drive, simply copy the entire contents of the CD into a shared network directory.

Installing the Server Software

Before you start the **setup** program, you must log on to the system using a username that has administrative privileges. (This is required only when installing server components; you can install the client components using any valid username.)

The SQL Server 6.0 CD contains three separate SETUP directories, one for Intel-based computers (the \I386 directory), one for MIPS-based computers (the \MIPS directory), and one for Alpha AXP–based computers (the \ALPHA directory). You must install SQL Server from the directory containing the software compatible with your hardware platform. You cannot use a different processor architecture's **setup** program to install SQL Server.

▶ **To install Microsoft SQL Server 6.0**

1. From the directory containing the software compatible with your hardware platform's processor architecture, run SETUP.EXE and complete each dialog box that appears.
2. If you are installing the software on a remote computer, when the Options dialog box appears, choose the Remote button, complete the Remote Setup dialog box, and then choose the OK button.

   If you are installing the software on the local computer, skip this step.
3. From the Options dialog box, select the Install SQL Server and Utilities option, and then choose the Continue button.
4. Follow the on-screen instructions. You can choose the Help button or press F1 to access the **setup** program's online Help.

   For information about each option presented during installation, see Chapter 3, "Server Options."

Note When you perform a new installation on a remote computer, the **setup** program configures the remote SQL Server as if the computer has 16 MB of memory. If this default is not correct, use SQL Enterprise Manager or the **sp_configure** system procedure on the remote SQL Server to change the setting of the **memory** configuration option. For information about setting the **memory** configuration option, see the *Microsoft SQL Server Transact-SQL Reference* or the *Microsoft SQL Server Administrator's Companion.*

During installation, selecting the default character set and sort order results in a rapid installation (in most cases, 5 to 10 minutes or less). Choosing an alternate character set or sort order can result in a somewhat longer installation time (but still less than 30 minutes).

After the server is installed, you may need to perform several configuration tasks, such as setting server configuration parameters, setting security parameters, and registering the server and adding it to a server group. For details see "Configuring SQL Server," later in this chapter.

Installing Alongside SQL Server 4.2

SQL Server 6.0 and SQL Server 4.2 can be installed alongside each other on the same server. Both can be run simultaneously. SQL Server 6.0 will be treated as a new installation and will not incorporate any configuration settings from the existing SQL Server 4.2 installation.

In a side-by-side installation, each SQL Server version must be installed in its own directory tree. In many cases, SQL Server 4.2 will have been installed under the \SQL root directory (the old default) and SQL Server 6.0 will be installed under the \SQL60 root directory (the new default).

Since both SQL Server 4.2 and SQL Server 6.0 use the same default named pipe, either SQL Server 6.0 or SQL Server 4.2 must be configured to use an alternate named pipe. For SQL Server 4.2, an alternate named pipe can be set by editing the Windows NT Registry. For SQL Server 6.0, an alternate named pipe can be set using the **setup** program's Change Network Support option. (It is not necessary to use the Registry Editor.) Note that if you change the named pipe a server listens on, clients that will connect to the server using named pipes must also be set up to connect to that server using the new pipe name.

If you will be installing any other Net-Libraries, make sure they are configured using names and parameters that do not duplicate those used by the SQL Server 4.2 installation.

▶ **To install SQL Server 6.0 alongside an existing SQL Server 4.2 installation**

1. From the directory containing the software compatible with your hardware platform's processor architecture, run SETUP.EXE and complete each dialog box that appears.
2. If you are installing the software on a remote computer, when the Options dialog box appears, choose the Remote button, complete the Remote Setup dialog box, and then choose the OK button.

   If you are installing the software on the local computer, skip this step.

3. From the Options dialog box, select the Install SQL Server and Utilities option, and then choose the Continue button.

   Do not select the Upgrade SQL Server option.

4. Follow the on-screen instructions until the installation is complete.

5. After SQL Server 6.0 is installed and the computer has been rebooted, change the default named pipe for one of the SQL Server installations on that computer.

   The default named pipe that a server listens on is \\.**\pipe\sql\query**.

   - For SQL Server 4.2, an alternate named pipe can be set by editing the Windows NT Registry. Change the ListenOn value for the key
     HKEY_LOCAL_MACHINE\
     SOFTWARE\
     Microsoft\
     SQLServer\
     Server

   - For SQL Server 6.0, an alternate named pipe can be set using the **setup** program's Change Network Support option. (It is not necessary to use the Registry Editor.)

   For example, you might change the version 4.2 server to use \\.**\pipe\sql42\query**, but leave the version 6.0 server with the default, \\.**\pipe\sql\query**.

6. As necessary, set up clients to use the alternate pipe name.

   Because you have changed the named pipe that one of the SQL Servers listens on, clients that will connect to that server using named pipes must be configured to connect using the new pipe name. For Windows NT–, Windows 95–, and Windows-based computers, you can set the alternate pipe name by using the Advanced window of the SQL Server Client Configuration Utility. Use different server names for the different pipes that the client connects to.

   For example, you might set up clients to use *terrier* for **\\terrier\pipe\sql\query** and *terrier42* for **\\terrier\pipe\sql42\query**.

Note Utilities such as SQL Performance Monitor will connect to the server that is using the default named pipe.

Installing a Standalone Server

The procedure for installing SQL Server software on a standalone computer (one that does not have a network adapter) is nearly the same as the standard installation procedure, except all network and remote **setup** options are unavailable.

Important If the **setup** program is unable to start the network, it assumes that you're installing on a standalone computer. If you're installing SQL Server on a computer that has a network adapter and the **setup** program is unable to start the network, exit the **setup** program, correct your network problem, and then restart the **setup** program.

Using Online Help During Installation

If you need instructions or more information while using the **setup** program, online Help is available. Simply choose the Help button that is part of most **setup** dialog boxes, or press F1.

The Help topic presented will usually be specific to the current dialog box. If you want other Help information, choose the Contents button in the Help window, and Help Contents will appear. The Help Contents is divided into these categories:

- **Introduction**, which lists topics providing overview information
- **How To**, which lists topics providing procedural information
- **More About**, which lists topics providing reference information

Upgrading SQL Server

You can upgrade a Microsoft SQL Server 4.2*x* server on a Windows NT–based computer to SQL Server 6.0 by running the **setup** program and choosing the upgrade option. You can upgrade a Microsoft SQL Server 4.2 for OS/2® operating system server to Microsoft SQL Server 6.0 by first upgrading the operating system to Windows NT Server 3.5 or later (version 3.51 is recommended) and then running the SQL Server **setup** program to upgrade the SQL Server files. Upgrades from SQL Server 1.*x* are no longer supported.

Because of improvements to the system (such as new system tables and datatypes), an upgrade to SQL Server 6.0 is a one-way operation and cannot be reversed. Once a version 4.2*x* database has been upgraded to version 6.0, it cannot be used again with SQL Server 4.2*x*.

Before upgrading a Microsoft SQL Server 4.2*x* server, you should run the CHKUPG.EXE utility, which checks that database status is acceptable, that all necessary comments exist in the *syscomments* system table, and that there are no keyword conflicts.

Most applications written to work with SQL Server 4.2*x* should work unmodified with SQL Server 6.0. However, because new keywords have been added that may conflict with existing applications, this will not be true in all cases. For a list of new keywords, see "Keywords," later in this chapter. Corrections to the behavior of subqueries may also affect applications that depended on the old behavior.

Note For information about language, statement, query, and other compatibility issues between SQL Server 6.0 and SQL Server 4.2*x* databases, see the compatibility information provided in the *Microsoft SQL Server Transact-SQL Reference.*

The **setup** program cannot upgrade from one processor version to another (for example, from Intel- to MIPS-based computers). Also, it is not possible to dump a database in one processor architecture and load it in another. Even if the operation appears to succeed, serious problems will subsequently result. For example, you cannot dump a database from an Alpha AXP–based computer and load it into an Intel-based computer. Nor should you copy MASTER.DAT from an Intel-based computer to a MIPS- or Alpha AXP–based computer and run **setup** to upgrade it. However, you can use SQL Transfer Manager to simply transfer data from one architecture to another.

Note SQL Server 6.0 detects and prevents attempts to load a dump from a different SQL Server 6.0 processor architecture, but it cannot prevent an attempt to load a dump from a different SQL Server 4.2*x* processor architecture, because that information is not recorded by a SQL Server 4.2*x*–level database dump.

Before Upgrading

Before you begin an upgrade:

- Back up all databases (including the *master* database) before upgrading the server.
- If any databases are read only, run **sp_dboption** and set the **read only** option to **FALSE**. For information on **sp_dboption**, see the *Microsoft SQL Server Transact-SQL Reference.*
- Run the CHKUPG.EXE utility, which checks that database status is acceptable, that all necessary comments exist in the *syscomments* system table, and that there are no keyword conflicts. CHKUPG.EXE creates a report file. Review the report and take any action the report indicates is necessary to prepare the existing installation for an upgrade.

  For more information, see "Running the CHKUPG.EXE Utility," later in this chapter.
- Check that there is adequate physical disk space for the upgrade.

  At least 50 MB of free disk space is required to upgrade from SQL Server 4.2*x* to SQL Server 6.0.

  The *master* database must have at least 7 MB free, but if that much database space is not available, the **setup** program automatically alters the *master* database to increase the space allocated to it. If necessary, **setup** also increases the size of the MASTER device (as long as adequate disk space is available).
- Check that no applications that could be using any of the SQL Server DLLs are running on the server you are going to upgrade. (These applications include SQL Server, SQL Monitor, and SQL Server client applications such as **isql**, SQL Administrator, and SQL Object Manager.) Also check that Microsoft® Word for Windows® and the Microsoft® Visual C++® development system are not running, and that no applications are using the Microsoft Visual C++ run-time library (MSVCRT20.DLL).

  In general, it is a good idea to make sure no applications are running on the server you are upgrading.
- Log on to the system under a user account that has administrative privileges.

Running the CHKUPG.EXE Utility

Before you begin an upgrade you should run the CHKUPG.EXE utility, which checks that database status is acceptable, that all necessary comments exist in the *syscomments* system table, and that there are no keyword conflicts. CHKUPG.EXE writes this information to an output file. Review this report and take any action the report indicates is necessary to prepare the existing installation for an upgrade.

▶ **To check a SQL Server 4.2*x* installation prior to upgrading**

- From the directory containing the software compatible with your hardware platform's processor architecture (Intel, Alpha AXP, or MIPS), run the CHKUPG.EXE utility. The utility has this syntax:

  chkupg [**/Usa**] [**/P***password*] [**/S***servername*] /**o***filename*

  where

  password
  Is the system administrator's password. If no *password* is provided, NULL is assumed.

  servername
  Is the name of the server being upgraded. If no *servername* is provided, the local server is assumed.

  filename
  Is the name of the output file. It must be a fully qualified filename, and include the drive and directory on the server where the output file will be placed.

For example:

```
chkupg /Usa /P /S /oc:\sql\install\myreport.out
```

After running CHKUPG.EXE, read its output file using any text editor.

If the CHKUPG.EXE output file indicates that the database is read only, run **sp_dboption** and set the **read only** option to **FALSE**. For information on **sp_dboption**, see the *Microsoft SQL Server Transact-SQL Reference*.

If the CHKUPG.EXE output file indicates that some text in the *syscomments* system table is missing or altered, drop and re-create the related objects, and then run CHKUPG.EXE again to verify that the necessary text exists in *syscomments*. (You cannot upgrade all objects from SQL Server 4.2*x* systems to SQL Server 6.0 if you have deleted data from the *text* column in the *syscomments* table. The **setup** program uses the comments in *syscomments* to upgrade database objects.)

CHKUPG.EXE reports both on keywords implemented in SQL Server 6.0 and on words that are reserved for future use but are not yet implemented. Reserved words are compatible with SQL Server 6.0 but can be expected to cause future conflicts. Reserved words are indicated in the output file by this notation:

```
[SQL-92 keyword]
```

For more information, see "Keywords" and "Reserved Words," later in this chapter.

If the CHKUPG.EXE output file indicates conflicts with SQL Server 6.0 keywords, change the conflicting database or database object name and alter the applications that access those objects so that they use the new names. Keyword conflicts can be resolved either before or after the server is upgraded. Keyword conflicts will not cause an upgrade to fail.

CHKUPG.EXE only checks and reports on the current SQL Server 4.2*x* system. It does not correct incompatibilities, change read-only databases, add comments to *syscomments*, or take any other corrective action.

When you run the **setup** program and perform an upgrade from SQL Server 4.2*x* to SQL Server 6.0, one of the first things **setup** does is automatically run the CHKUPG.EXE program and inform you if a problem exists. Although **setup** runs CHKUPG.EXE automatically, it is still recommended that you first run CHKUPG.EXE separately.

Keywords

Following are lists of SQL Server keywords and reserved words. These are the SQL Server keywords. Keywords new for SQL Server 6.0 are shown in **bold** type.

ADD
ALL
ALTER
AND
ANY
AS
ASC
AVG
BEGIN
BETWEEN
BREAK
BROWSE
BULK
BY
CASE
CHECK
CHECKPOINT
CLOSE
CLUSTERED
COALESCE
COMMIT
COMMITTED
COMPUTE
CONFIRM
CONSTRAINT
CONTINUE
CONTROLROW
CONVERT
COUNT
CREATE
CURRENT
CURRENT_DATE
CURRENT_TIME
CURRENT_TIMESTAMP
CURRENT_USER
CURSOR
DATABASE
DBCC
DEALLOCATE
DECLARE
DEFAULT
DELETE
DESC
DISK
DISTINCT
DOUBLE
DROP
DUMMY
DUMP
ELSE
END
ERRLVL
ERROREXIT
EXCEPT
EXEC
EXECUTE
EXISTS
EXIT
FETCH
FILLFACTOR
FLOPPY
FOR
FOREIGN
FROM
GOTO
GRANT
GROUP
HAVING
HOLDLOCK
IDENTITY
IDENTITY_INSERT
IDENTITYCOL
IF
IN
INDEX
INSENSITIVE
INSERT
INTERSECT
INTO
IS
ISOLATION
KEY
KILL
LEVEL
LIKE
LINENO
LOAD
MAX
MIN
MIRROREXIT
NOCHECK
NONCLUSTERED
NOT
NULL
NULLIF
OF

- OFF
- OFFSETS
- ON
- ONCE
- **ONLY**
- **OPEN**
- **OPTION**
- OR
- ORDER
- OVER
- PERM
- PERMANENT
- **PIPE**
- PLAN
- **PRECISION**
- PREPARE
- **PRIMARY**
- PRINT
- PROC
- PROCEDURE
- PROCESSEXIT
- PUBLIC
- RAISERROR
- **READ**
- RECONFIGURE
- **REFERENCES**
- REPEATABLE
- **REPLICATION**
- RETURN
- REVOKE
- ROLLBACK
- ROWCOUNT
- RULE
- SAVE
- **SCROLL**
- SELECT
- **SERIALIZABLE**
- **SESSION_USER**
- SET
- SETUSER
- SHUTDOWN
- **SOME**
- STATISTICS
- **SUM**
- **SYSTEM_USER**
- TABLE
- TAPE
- TEMP
- TEMPORARY
- TEXTSIZE
- **THEN**
- TO
- TRAN
- TRANSACTION
- TRIGGER
- TRUNCATE
- TSEQUAL
- **UNCOMMITTED**
- UNION
- UNIQUE
- UPDATE
- **UPDATETEXT**
- USE
- **USER**
- VALUES
- **VARYING**
- VIEW
- WAITFOR
- **WHEN**
- WHERE
- WHILE
- WITH
- WRITETEXT

Reserved Words

These are SQL Server reserved words. Reserved words are compatible with SQL Server 6.0 but can be expected to cause conflicts in the future.

ABSOLUTE
ACTION
ALLOCATE
ARE
ASSERTION
AT
AUTHORIZATION
BOTH
CASCADE
CASCADED
CAST
CATALOG
CHAR_LENGTH
CHARACTER
CHARACTER_LENGTH
COLLATE
COLLATION
COLUMN
CONNECT
CONNECTION
CONSTRAINTS
CORRESPONDING
DATE
DAY
DEFERRABLE
DEFERRED
DESCRIBE
DESCRIPTOR
DIAGNOSTICS
DISCONNECT
DOMAIN
END_EXEC
ESCAPE
EXCEPTION
EXPIREDATE
EXTERNAL
EXTRACT
FALSE
FILE
FIRST
FULL
GET
GLOBAL
HOUR
IMMEDIATE
INITIALLY
INNER
INPUT
INTERVAL
JOIN
LAST
LEADING
LEFT
LOCAL
MATCH
MINUTE
MONTH
NAMES
NATIONAL
NATURAL
NCHAR
NEXT
NO
OCTET_LENGTH
OUTER
OUTPUT
OVERLAPS
PAD
PARTIAL
POSITION
PRESERVE
PRIOR
PRIVILEGES
RELATIVE
RESTRICT
RETAINDAYS
RIGHT
ROWS
SCHEMA
SECOND
SESSION
SIZE
SPACE
SQLSTATE
TIME
TIMESTAMP
TIMEZONE_HOUR
TIMEZONE_MINUTE
TRAILING
TRANSLATE

TRANSLATION
TRUE
UNKNOWN
USAGE
USING
VALUE
VOLUME
WORK
WRITE
YEAR
ZONE

For more information about keywords and reserved words, see the *Microsoft SQL Server Transact-SQL Reference.*

Upgrading from SQL Server 4.2*x*

You can upgrade Microsoft SQL Server 4.2*x* on a Windows NT–based computer to SQL Server 6.0 by using the following instructions.

For information on upgrading from SQL Server 4.2 for OS/2, see "Upgrading from SQL Server for OS/2," later in this chapter.

▶ **To upgrade Microsoft SQL Server 4.2*x* to SQL Server 6.0**

1. Complete the upgrade prerequisites described earlier in "Before Upgrading" and "Running the CHKUPG.EXE Utility."
2. From the directory containing the software compatible with your hardware platform's processor architecture (Intel, Alpha AXP, or MIPS), run SETUP.EXE and follow the on-screen instructions.
3. If you are installing the software on a remote computer, when the Options dialog box appears, choose the Remote button, complete the Remote Setup dialog box, and then choose the OK button.

   If you are installing the software on the local computer, skip this step.
4. From the Options dialog box, select the Upgrade SQL Server option, and then choose the Continue button.
5. Follow the on-screen instructions. You can choose the Help button or press F1 to access the **setup** program's online Help.

   For information about each option presented during installation, see Chapter 3, "Server Options."

Regardless of which networks you've configured SQL Server to listen on, the **setup** program always uses named pipes when connecting to the server during an upgrade. (Named pipes work between Windows NT–based computers no matter which network transport they are running.)

Important If the DB-Library software on the computer from which the **setup** program was started is configured to use an IPC mechanism other than named pipes, the **setup** program temporarily changes the default network to named pipes. If in the rare case that **setup** terminates unexpectedly without restoring the original default network value, you might need to reset this value yourself by using the SQL Server Client Configuration Utility. In addition, if you've been starting SQL Server on an alternate pipe or have disabled named pipes altogether, you must temporarily reconfigure SQL Server to listen on the standard pipe (\\.**\pipe\sql\query**) before beginning the upgrade.

After completing the upgrade and rebooting the server computer, check your Windows NT search path (**path**) to be sure that the \SQL\BINN directory precedes any \SQL\BIN or \SQL\BINP directories used for earlier versions of SQL Server. If the server computer has multiple versions of SQL Server components, checking that \SQL\BINN appears first in the path ensures that the current Windows NT–based version is the one that will run.

▶ **To check your Windows NT search path**

- From an operating-system prompt, type:

  path

After the server is upgraded, you may need to perform several configuration tasks, such as setting server configuration parameters, setting security parameters, and registering the server and adding it to a server group. For details, see "Configuring SQL Server," later in this chapter.

Upgrading from an Earlier SQL Server 6.0 Release

If you are upgrading from an earlier SQL Server 6.0 installation, follow this procedure. From the directory containing the software compatible with your hardware platform's processor architecture, simply run the **setup** program and choose the Upgrade SQL Server and Utilities option. It is not necessary to run the CHKUPG.EXE utility.

Upgrading from SQL Server for OS/2

You can upgrade a Microsoft SQL Server 4.2 for OS/2 server to Microsoft SQL Server 6.0 by first upgrading the operating system to Windows NT Server and then running the SQL Server **setup** program to upgrade the SQL Server files.

▶ **To upgrade a Microsoft SQL Server for OS/2 system to SQL Server 6.0**

1. Back up all your databases (including the *master* database).
2. Upgrade the operating system.

   If you will be installing SQL Server 6.0, upgrade from OS/2 to Windows NT Server version 3.5 or later. Version 3.51 is recommended. If you will be installing SQL Workstation 6.0, upgrade to Windows NT Workstation version 3.5 or later. Version 3.51 is recommended.

   For operating-system upgrade information, see your documentation for Windows NT.
3. From the i386 directory containing the software compatible with the Intel processor architecture, run the SQL Server 6.0 **setup** program and follow the on-screen instructions.
4. When the Options dialog box appears, select the Upgrade SQL Server option, and then choose the Continue button.
5. Follow the on-screen instructions. You can choose the Help button or press F1 to access the **setup** program's online Help.

Installing or Upgrading SQL Workstation

Microsoft SQL Workstation 6.0 has the same features and functionality as Microsoft SQL Server 6.0, except that it is a *single-user* licensed product that supports a maximum of 15 simultaneous database connections. SQL Workstation is primarily intended to be used as a development platform.

To install, upgrade, test, configure, or remove SQL Workstation, follow the same instructions provided in this chapter for SQL Server.

Testing an Installation

When a server installation or upgrade is complete, you can test the installation by connecting to it locally by running the **isql** utility from the server computer.

▶ To test the installation

1. Reboot the server computer, and then start SQL Server by typing the following from an operating-system prompt:

   ```
   net start mssqlserver
   ```

2. Connect to SQL Server by typing:

   ```
   isql /Usa /P
   ```

   If **isql** connects, you'll see the following **isql** prompt:

   ```
   1>
   ```

   If **isql** cannot connect, you'll get a DB-Library error. (For help in identifying and correcting the problem, see Chapter 7, "Installation Troubleshooting.")

3. Enter a simple query by typing:

   ```
   select @@servername
   go
   ```

   The **isql** utility returns the server name, as shown in this example:

   ```
   1> select @@servername
   2> go
   ------------------------------
   WOLFHOUND

   (1 row affected)
   1>
   ```

4. Verify that you have checked a version 6.0 server by typing:

   ```
   select @@version
   go
   ```

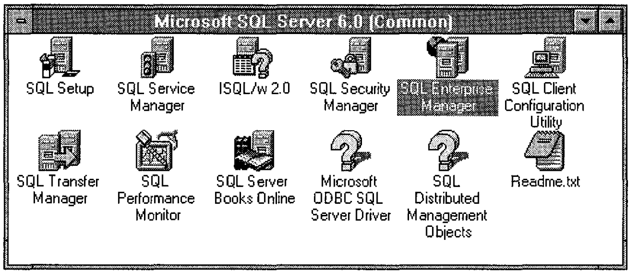
   The **isql** utility returns the version information.

5. Quit the **isql** utility by typing:

   ```
   exit
   ```

Configuring SQL Server

After the server is upgraded, you may need to perform several configuration tasks, such as setting server configuration parameters, setting security parameters, and registering the server and adding it to a server group. For some post-upgrade configuration tasks you use SQL Enterprise Manager; for other tasks, you use SQL Setup. These applications can be started from the Microsoft SQL Server 6.0 program group, shown here:



You can use SQL Enterprise Manager to configure many server options. For a newly installed server, you would typically use SQL Enterprise Manager to:

- Register the server and add it to a server group
- Set security options
- Set server options
- Set configuration options

Of course, SQL Enterprise Manager has additional substantial capabilities. For information on using SQL Enterprise Manager, see the SQL Enterprise Manager online Help or the *Microsoft SQL Server Administrator's Companion.*

You can use SQL Setup not only to install new software and upgrade existing software, but also to configure the server after installation and install specific components. After a server is installed, you can use SQL Setup to:

- Change network support
- Add a language
- Rebuild the *master* database and, optionally, change the character set and sort order

- Set security options
- Set server options
- Remove SQL Server

Important SQL Setup updates the Windows NT Registry with all the information required to run SQL Server. If you need to update any of the information written to the Registry, reconfigure the installation options by running SQL Setup again or by running SQL Enterprise Manager, rather than using the Windows NT Registry editor (**regedt32**). Using SQL Enterprise Manager or SQL Setup protects you from inadvertently changing entries that should not be changed and ensures that the integrity of the Registry is maintained.

For more information on configuring a server after installation or upgrade, see the *Microsoft SQL Server Administrator's Companion.*

Removing SQL Server

When you remove a SQL Server installation, the **setup** program deletes all SQL Server entries from the Windows NT Registry. It also gives you the option to automatically delete the SQL Server directory tree from the root down. (If you choose this option after **setup** has removed SQL Server, you must manually delete the SQL Server root directory and a few remaining files. This procedure is necessary because **setup** cannot delete itself.)

▶ **To remove a SQL Server installation**

1. From the Microsoft SQL Server 6.0 program group, choose the SQL Setup icon.
2. Follow the on-screen instructions until the Options dialog box appears.
3. If you're removing the software from a remote computer, choose the Remote button, complete the Remote Setup dialog box, and then choose the OK button.
4. From the Options dialog box, select the Remove SQL Server option.
5. Choose the Continue button.

   The Remove SQL Server dialog box appears.
6. To remove SQL Server files, select the Remove Files From check box.
7. Choose the Resume button.

CHAPTER 5

Starting the Server

This chapter tells you how to start, pause, continue, and stop SQL Server and how to start and stop SQL Executive.

Running SQL Server

The topics in this section describe how to run SQL Server as a service; configure SQL Server for automatic startup; start, stop, pause, and continue SQL Server; and run SQL Server with or without a network.

Running SQL Server as a Service

Because Microsoft SQL Server is integrated with the Windows NT Service Control Manager, it can be started and stopped as a service, either locally or remotely. It can be manually started, stopped, paused, and continued using SQL Enterprise Manager, SQL Service Manager, the Services application in the Control Panel, or the NET START command.

Optionally, SQL Server can be started automatically as a service when Windows NT starts. You can specify this option during installation by using the **setup** program. After installation, you can specify this option by using SQL Enterprise Manager.

When you start SQL Server as a Windows NT service, you can log off from the Windows NT network without shutting down SQL Server.

Note that SQL Server can also be started as an application, from a command prompt. For more information, see “Other Startup Procedures,” later in this chapter.

Configuring SQL Server for Automatic Startup

▶ **After installation, to configure SQL Server as an automatically started service**

1. From the Microsoft SQL Server 6.0 program group, start SQL Enterprise Manager.
2. From SQL Enterprise Manager's Server Manager window, open a server group and select a server.
3. From the Server menu, choose Configurations.

   The Server Configuration dialog box appears.
4. From the Server Configuration dialog box, choose the Server Options tab.
5. Select the Auto Start Server At Boot Time Check box.
6. Choose the OK button.

To cancel automatic startup of SQL Server, perform the same procedure but clear the Auto Start Server At Boot Time Check box.

You can also use the Services application in the Windows NT Control Panel to configure SQL Server as an automatically started service. For information, see your documentation for Windows NT.

Starting SQL Server Manually

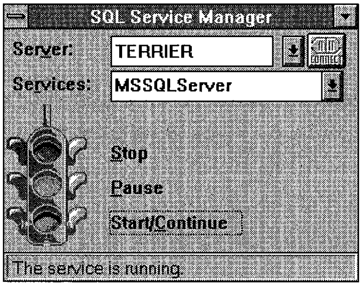
You can start SQL Server 6.0 in several ways—by using SQL Service Manager, SQL Enterprise Manager, or the Services application in the Control Panel. From an operating-system prompt, you can start it as a service or as an executable file. You can also start it in a minimal configuration (useful if the setting of a configuration value—for example, over-committing memory—has prevented the server from starting).

Starting SQL Server Using SQL Service Manager

▶ **To start SQL Server using SQL Service Manager**

1. From the Microsoft SQL Server 6.0 program group, start SQL Service Manager.

   The SQL Service manager window appears:

   Server: TERRIER

   Services: MSSQLServer

   The service is running.

2. If the service is a remote service, select or enter the remote server name in the Server box, and then choose the Connect button.

   This connects you to the remote server and fills the Services box with the names of the SQL Server services that are registered on the remote computer. Note that the Server box appears only if the Remote Control option on the System menu (opened by choosing the "-" box in the upper left-hand corner of the window) is turned on.

3. In the Services box, select MSSQLServer.

   The current state of the service is indicated by the light on the traffic signal and by the message in the status bar.

4. Choose the green Start/Continue light.

5. If a confirmation box appears, choose the Yes button.

   SQL Server starts as a service under the Windows NT Service Control Manager using the default startup options written to the Windows NT Registry by the **setup** program.

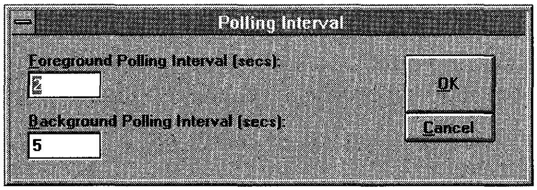
You can also stop, pause and continue SQL Server by using the same procedure used to start the service. To stop the service, select the red light, to pause the service, select the yellow light, to continue a paused service, select the green light.

The SQL Service Manager System menu (opened by choosing the "-" box in the upper left-hand corner of the window) includes these options:

Action Verify
Indicates whether you want the SQL Service Manager to present a confirmation box before stopping, pausing, starting, or continuing a service, including dependent services such as SQL Executive. (For details about the dependencies between SQL Server and SQL Executive, see "Running SQL Executive," later in this chapter.)

Remote Control
Indicates whether you want the SQL Service Manager window to include the Server box for controlling services on remote servers.

Polling Interval
Displays the Polling Interval dialog box, which is used to set how often the state of the service (SQL Server or SQL Executive) is checked.



Foreground Polling Interval (secs): 2

Background Polling Interval (secs): 5

The Polling Interval dialog box contains two options:

The Foreground Polling Interval (secs.)
Sets the polling interval, in seconds, for the SQL Service Manager when it is running in the foreground (not minimized). The default is 2.

Background Polling Interval (secs.)
Sets the polling interval, in seconds, for the SQL Service Manager when it is running in the background (minimized). The default is 5.

Starting SQL Server Using SQL Enterprise Manager

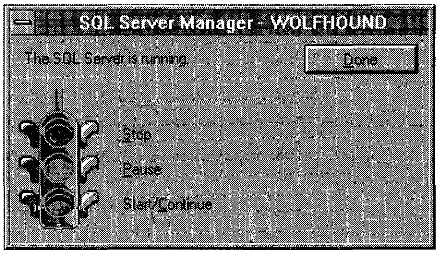
▶ **To start SQL Server using SQL Enterprise Manager**

1. From the Microsoft SQL Server 6.0 program group, start SQL Enterprise Manager.
2. From SQL Enterprise Manager's Server Manager window, open a server group and select a server.

3. From the toolbar, choose the Services button (the stoplight).

   The SQL Service Manager window appears.

4. From the Services window of SQL Server Manager, choose the green Start/Continue light.



The SQL Server is running

   Status text in the SQL Server Manager dialog box informs you when SQL Server has started.

You can also stop, pause and continue SQL Server by using the same procedure used to start the service. To stop the service, select the red light, to pause the service, select the yellow light, to continue a paused service, select the green light.

You can also start the server using a drop-down menu available from the Server Manager window.

▶ To start SQL Server using Server Manager's drop-down menu

1. From the Microsoft SQL Server 6.0 program group, start SQL Enterprise Manager.
2. From SQL Enterprise Manager's Server Manager window, open a server group and select a server.
3. Using the right mouse button, click the server name. Then, from the drop-down menu that appears, choose Start.

   The server starts.

You can also stop, pause, and continue SQL Server by using the drop-down menu.

Other Startup Procedures

You can also start SQL Server by:

- Using the Services application in the Control Panel. For information, see your documentation for Windows NT.
- Typing the following from an operating-system prompt:

  net start mssqlserver
- Running the SQL Server application, SQLSERVR.EXE, from an operating-system command prompt using the options described under "Startup Options," later in this chapter.

Note that the first three methods start SQL Server as a service under the Windows NT Service Control Manager, but the third method does not. When SQL Server is started independent of the Service Control Manager, all system messages print in the window used to start SQL Server, and you cannot pause, stop, or continue SQL Server as a service using SQL Enterprise Manager, SQL Service Manager, the Control Panel Services application, or the **net** commands.

When you start SQL Server from the command line, any command-line options that you enter take precedence over the default command-line options written to the Windows NT Registry by the **setup** program. In addition, you must shut down SQL Server before logging off from the Windows NT network.

The following example shows how to start SQL Server from the command line using C:\SQL\DATA\MASTER.DAT as the MASTER database device path and C:\SQL\LOG\ERRORLOG as the error log path:

```
sqlservr -c -dc:\sql\data\master.dat -ec:\sql\log\errorlog
```

For more information on the options used in this example, see "Startup Options," later in this chapter.

Startup Options

Default SQL Server startup options are stored in the Windows NT Registry. These startup options are used as defaults whenever SQL Server is started, whether it is started as a service or from the command line. Use SQL Enterprise Manager to change the default startup options. (For details, see the *Microsoft SQL Server Administrator's Companion.*)

You can start SQL Server using the following options (which will not affect the default settings in the Registry):

-c
Shortens startup time by starting SQL Server independent of the Windows NT Service Control Manager.

-d*master_device_path*
Indicates the fully qualified path for the MASTER database device (typically, C:\SQL60\DATA\MASTER.DAT). If you do not provide this option, the existing Registry parameters are used.

-e*error_log_path*
Indicates the fully qualified path for the error log file (typically, C:\SQL60\LOG\ERRORLOG). If you do not provide this option, the existing Registry parameters are used.

-f
Starts SQL Server in minimal configuration. Useful if the setting of a configuration value (for example, over-committing memory) has prevented the server from starting. For more information, see "Starting SQL Server with Minimal Configuration," later in this chapter.

-m
Starts SQL Server in single-user mode. When you start SQL Server in single-user mode, only a single user can connect, and the CHECKPOINT mechanism, which guarantees that completed transactions are regularly written from the disk cache to the database device, is not started. (Typically, this option is used if you experience problems with system databases that need to be repaired.)

-n
Does not use the Windows NT event log to log SQL Server events. If you start SQL Server with the **-n** option, it is advisable to also use the **-e** option, or SQL Server events will not be logged.

-p*precision_level*
Specifies the maximum level of precision to be supported by *decimal* and *numeric* datatypes. By default, SQL Server has a maximum precision of 28. The acceptable values for *precision_level* are from 1 to 38; however, if no *precision_level* is supplied with the **-p** parameter, a maximum precision of 38 will be allowed for *decimal* and *numeric* datatypes.

-r*master_mirror*

Indicates the fully qualified path for the device used to mirror the MASTER database device. Use this option when starting SQL Server if the MASTER database device is damaged. (SQL Server always attempts to use the primary device specified by the **-d** option before it uses the mirror device specified by the **-r** option.)

-s*registry_key*

Starts SQL Server using an alternate set of startup parameters stored in the Registry under the key *registry_key*. This option, which can be run only from the command line, lets you select from multiple previously defined startup configurations. You could, for example, create an alternate Registry key to start SQL Server in single-user mode. (This applies only to the SERVER subkey.)

/T*trace#*

Indicates that SQL Server should be started with a specified trace flag (*trace#*) in effect. Trace flags are used to start the server with non-standard behavior. For a list of available trace flags (*trace#*), see the *Microsoft SQL Server Transact-SQL Reference.*

Important When specifying a trace flag, use **/T** to pass the trace flag number. A lowercase t (**/t**) is accepted by SQL Server, but it sets other internal trace flags that are needed only by SQL Server support engineers.

-x

Disables the keeping of CPU time and cache-hit ratio statistics. Allows maximum performance.

Starting SQL Server with Minimal Configuration

The **-f** startup option enables you to start the server with minimal configuration. The **-f** option is useful if the setting of a configuration value (for example, overcommitting memory) has prevented the server from starting.

Note Use the minimal configuration option only for correcting configuration problems (when the server will not start because a configuration value is set to an unacceptable value). It should not be used as the standard way of starting the server.

When you start SQL Server using the minimal configuration option:

- Temporary changes in memory usage are affected. User connections, open databases, locks, open objects, language information in cache, and asynchronous I/O are all set to the minimum values shown by **sp_configure**. The procedure cache is set to 50 percent, with minimal total procedure and buffer cache.
- SQL Server starts in single-user mode (the equivalent of the **-m** option). Only a single user can connect, and the CHECKPOINT mechanism (which guarantees that completed transactions are regularly written from the disk cache to the database device) is not started.
- Remote access is disabled.
- Read-ahead is disabled.
- *Tempdb_in_ram* (an **sp_configure** parameter) is set to 2 MB.
- Autoexec procedures are not run.

Once the server has been started with the minimal configuration option, you will likely want to do one or more of the following:

- Change **sp_configure** values and reconfigure
- Create a new disk device with DISK INIT
- Make a device a default device using **sp_diskdefault**

▶ To start SQL Server with minimal configuration from a command prompt

- Type:

  sqlservr -f

▶ To start SQL Server with minimal configuration and independent of the Windows NT Service Control Manager

- From a command prompt, type:

  sqlservr -c -f

▶ To start the server with minimal configuration, using the Services application in the Control Panel

1. Open the Control Panel and start the Services application.
2. From the list of services, select SQL Server.
3. In the Startup Parameters box, type **-f**.
4. Choose the Start button.

Pausing SQL Server

When you pause SQL Server, users who have connections to SQL Server can finish their tasks, but new connections are not allowed. You can pause the SQL Server service only if you started it as a service. For example, you might pause SQL Server a few minutes before shutting it down.

You can also pause the SQL Server service by using SQL Enterprise Manager, SQL Service Manager, the Services application in the Windows NT Control Panel, or by typing the following from an operating-system prompt:

```
net pause mssqlserver
```

Continuing SQL Server

You can continue the SQL Server service (resume accepting connections after pausing) by using SQL Enterprise Manager, SQL Service Manager, the Services application in the Windows NT Control Panel, or by typing the following from an operating-system prompt:

```
net continue mssqlserver
```

Stopping SQL Server

You can stop SQL Server locally from the server or remotely from a client or another server. If you've started SQL Server as a service, you can stop it by using SQL Enterprise Manager, SQL Service Manager, the Services application in the Control Panel, or the **net stop mssqlserver** command. If you've started SQL Server from the command line, you can stop SQL Server from a client (either locally or remotely) by using the SHUTDOWN command. You can stop SQL Server from a command prompt window by pressing CTRL + C.

When you use any of these procedures to stop SQL Server, SQL Server shuts down in an orderly fashion by performing the following steps:

- Disabling logins (except for system administrators)
- Performing a CHECKPOINT in every database

In addition, if you stop SQL Server from a client, SQL Server waits for any currently executing Transact-SQL statements or stored procedures to finish. To bring the system to an immediate halt, you must issue the statement SHUTDOWN WITH NOWAIT from a client.

When you plan to shut down SQL Server, it's a good idea to first pause the SQL Server service and then broadcast a message that SQL Server will be stopped. Give users enough time to finish their current tasks, and then shut down SQL Server as described in the preceding section.

▶ To stop SQL Server using SQL Service Manager

1. From the Microsoft SQL Server 6.0 program group, start SQL Service Manager.

   The SQL Service manager Window appears.
2. To stop a remote SQL Server, in the Server box, select or enter the name of the remote server, and then choose the Connect button.
3. From the Service box, select MSSQLServer.
4. Choose the yellow Pause light to pause the service.
5. Send a message informing connected users that the server will soon be shutting down. After an appropriate interval, proceed to step 6.
6. Choose the red Stop light.

▶ To stop SQL Server using SQL Enterprise Manager

1. From the Microsoft SQL Server 6.0 program group, start SQL Enterprise Manager.

   The SQL Enterprise Manager Window appears.
2. From SQL Enterprise Manager's Server Manager window, open a server group and select a server.
3. From the toolbar, select the Services icon (the stoplight).
4. Choose the yellow Pause light to pause the service.
5. Send a message informing connected users that the server will soon be shutting down. After an appropriate interval, proceed to step 6.
6. Choose the red Stop light.
7. Choose the Done button.

▶ To stop SQL Server from operating-system prompt

- Type:

  net stop mssqlserver

▶ To stop SQL Server from an isql client

- Type:

  SHUTDOWN [WITH NOWAIT]

  Issuing SHUTDOWN waits for currently executing Transact-SQL statements or stored procedures to finish before shutting down. Issuing SHUTDOWN WITH NOWAIT brings the system to an immediate halt.

▶ To broadcast a shutdown message

- From an operating-system prompt, type:

  net send /users "*message*"

  For example:

  ```
  net send /users "SQL Server is going down in 20 minutes.
  Please disconnect."
  ```

  The **/users** option causes the specified message to be sent to all users connected to the server. For descriptions of other **net send** options, see your documentation for Windows NT.

You can also stop SQL Server locally from the server by using the Services application in the Control Panel. For more information, see your documentation for Windows NT.

Running SQL Server on a Network

For SQL Server to communicate over the network, the Server (network) service must be running. By default, Windows NT starts its built-in Server service automatically whenever the Windows NT starts. If necessary, you can find out whether the Server service has started by typing the following command at an operating-system prompt:

net start

If the Server service has been started, it will be listed in the **net start** output along with the other running services, as shown here:

```
C:\> net start
These Windows NT services are started:

   ClipBook Server
   Computer Browser
   EventLog
   Messenger
   Network DDE
   Network DDE DSDM
   Server
   Workstation

The command completed successfully.
C:\>
```

If the Server service has not been started, start it by typing the following command at an operating-system prompt:

net start server

You'll receive the following message to indicate that the service has been started:

```
The Server service was started successfully.
```

You can also use the Services application in the Control Panel to check service status and to start and stop services. For details, see your documentation for Windows NT.

Running SQL Server Without a Network

When running SQL Server without a network, you follow the same procedures as if SQL Server were running with a network, except that you don't need to start the built-in Server service. Because SQL Enterprise Manager, SQL Service Manager, and the **net start** and **net stop** commands are always functional (even without a network), the procedures for starting and stopping SQL Server are identical for network or standalone operation.

When connecting to a standalone SQL Server from a local client such as **isql**, you bypass the network and connect directly to the SQL Server using a local pipe. The only difference between a local pipe and a network pipe is whether or not the network is involved—both establish a connection with SQL Server using the standard pipe (**\pipe\sql\query**) unless otherwise directed.

When you connect to a local SQL Server without specifying a server name, you're using a local pipe. When you connect to a local SQL Server and explicitly specify a server name, you're using either a network pipe or another network IPC mechanism such as IPX/SPX (if you have configured SQL Server to use multiple networks). Because a standalone SQL Server does not support network pipes, you must omit the unnecessary *server_name* argument when connecting to the SQL Server from a client. For example, to connect to a standalone SQL Server from **isql**, type:

```
isql /Usa /P
```

Running SQL Executive

SQL Executive is a Windows NT service that resides on the server computer. SQL Executive provides the SQL Server 6.0 scheduling engine, offers robust and varied task scheduling and alerting abilities, and is capable of handling large client/server environments. SQL Executive administration is integrated within SQL Enterprise Manager. Scheduling information is maintained in the *msdb* database (which is created automatically by **setup**). For information on using and administering SQL Executive, see the *Microsoft SQL Server Administrator's Companion.*

SQL Executive can be manually started and stopped in the same manner as SQL Server, by using SQL Service Manager or the Services application in the Control Panel. Use the same procedures provided earlier in this chapter for SQL Server, except those that specify the SQL Executive service rather than the SQL Server (MSSQLServer) service. SQL Executive cannot be paused or continued.

Optionally, SQL Executive can be started automatically when Windows NT starts. You can specify this option during installation using the **setup** program, or after installation using SQL Enterprise Manager or the Services option in the Control Panel. For SQL Enterprise Manager, use the same procedures described earlier in "Configuring SQL Server for Automatic Startup," but select the Auto Start Executive At Boot Time option.

During installation, SQL Executive should be configured to log on as a service using a special Windows NT user account created for that purpose. For information about this user account, see Chapter 4, "Server Installation."

If SQL Executive detects that the MSSQLServer service has stopped unexpectedly, it will automatically attempt up to three times to restart MSSQLServer. This automated restart behavior can be adjusted by modifying values in the Windows NT Registry.

RestartSQLServer specifies whether SQL Executive will attempt to restart the MSSQLServer service once it has been detected as having stopped unexpectedly. Setting this value to 0 prevents automated restart attempts.

```
MSSQLServer\
    SQLExecutive\
        RestartSQLServer
```

Type: REG_DWORD
Value: 0 (false) or 1 (true).
The default is 1.

RestartSQLServerInterval specifies, in minutes, the delay between attempts to restart the MSSQLServer service. Changing this value changes the interval between restart attempts.

```
MSSQLServer\
    SQLExecutive\
        RestartSQLServerInterval
```

Type: REG_DWORD
Value: Between 1 and 60 (inclusive).
The default is 5.

EventLogPollingPeriod specifies, in seconds, the interval between alert engine polls of the Windows NT applications log (as the alert engine checks for logged SQL Server events).

```
MSSQLServer\
    SQLExecutive\
        EventLogPollingPeriod
```

Type: REG_DWORD
Value: Between 10 and 3600 (inclusive).
The default is 20.

CHAPTER 6

Client Installation

This chapter discusses preinstallation requirements and provides installation instructions for Microsoft SQL Server client components on computers running the Windows NT, Windows 95, Windows, Windows for Workgroups, or MS-DOS operating systems. It also describes how to upgrade existing client installations.

For information on configuring an installed client, see the *Microsoft SQL Server Administrator's Companion.*

Client Platforms

The SQL Server **setup** program can install 32-bit client software (including SQL Enterprise Manager) on computers running these operating systems:

- Microsoft Windows NT Server version 3.5 or later. (Version 3.51 is recommended.)
- Microsoft Windows NT Workstation version 3.5 or later. (Version 3.51 is recommended.)
- Microsoft Windows 95.

Client software is also provided for computers running these 16-bit operating systems:

- Microsoft Windows (throughout this book, Windows refers to Windows- and Windows for Workgroups-based clients).
- MS-DOS.

Client Hardware and Software Requirements

To install the 32-bit SQL Server client utilities and libraries, the computer must be equipped with the following hardware and software:

You need	Your choices
Computer	Refer to the Windows NT Hardware Compatibility List for the processor architecture that corresponds to the SQL Server platform you are installing (that is, Intel, Alpha AXP, or MIPS). For example, if you are installing SQL Server for the Intel platform, your choices are any of the 32-bit Intel x86-based microprocessors, such as the 80486 or the Pentium.
Minimum memory	16 MB.
Disk drive space	A minimum of 7 MB of free disk space. If you install SQL Server Books Online to run from the hard disk, another 15 MB of available disk space is required; if you install SQL Server Books Online to run from the CD, only 1 MB of additional space is needed.
Operating system	Microsoft Windows NT Server version 3.5 or later, Microsoft Windows NT Workstation version 3.5 or later, or Microsoft Windows 95. (For Windows NT Server and Windows NT Workstation, version 3.51 is recommended.)
Network software	SQL Server client software runs on the Windows NT platform using the built-in network software. Unless Banyan VINES or DEC PATHWORKS will be used, additional network software is not required.
Network adapter	Any of the network adapters supported by Windows NT. If you're not using network operations (if, for example, you're connecting locally to a standalone SQL Server), a network adapter is not required. For information about supported network adapters, see your documentation for Windows NT.

The 16-bit SQL Server client utilities and libraries can be installed on computers running the Windows or MS-DOS operating systems.

Note The CD provided with Windows NT Server provides the latest version of Microsoft Windows for Workgroups. Use this version on computers that will run the Windows for Workgroups operating system and will have the SQL Server client software installed.

The 16-bit SQL Server client utilities can also be installed on computers running the Windows NT operating system (although for Windows NT–based computers, the 32-bit client software is recommended):

- 16-bit Windows-based SQL Server client software runs on Windows NT under Windows on Windows (WOW).
- MS-DOS–based SQL Server client software runs on Windows NT under the virtual DOS machine (VDM).

Client Installation Options

This section provides information on the choices and options you will encounter during client installation to help you to make decisions about these options prior to actually performing the installation.

Client Options for 32-bit Computers

When installing client utilities on a Windows NT– or Windows 95–based client computer, you must make decisions about:

- The installation path
- Client utilities
- SQL Server Books Online

Installation Path

For Windows NT– and Windows 95–based clients, the default SQL Server client installation root directory is C:\SQL60, although you can specify another directory. These are the directories created under the root directory:

Directory	Contents
C:\SQL60\BINN	Windows NT–based client executable files
C:\SQL60\DLL	Dynamic-link library files
C:\SQL60\INSTALL	SQL Server Books Online files

The SQL Server **setup** program places copies of the DB-Library and Net-Library DLLs in the \WINDOWS\SYSTEM32 directory.

Client Utilities

Microsoft SQL Server includes several client utilities, which are used for specific database and network tasks. Although the client utilities are always installed on the server during a new installation, they can also be installed on a 32-bit workstation independent of the server software. When you install the client software separately from the server software, the **setup** program always installs the client support libraries, such as DB-Library. Some of the utilities, however, are optional.

These are the client utility installation choices:

BCP
Installs the bulk copy utility (**bcp**), used to copy data to or from an operating-system file.

ISQL/w
Installs both versions of the **isql** utility (ISQL/w and **isql**), used to enter Transact-SQL commands and procedures.

SQL Enterprise Manager
Installs SQL Enterprise Manager, used to perform server and enterprise administration tasks.

SQL Security Manager
Installs SQL Security Manager, used to manage user accounts for SQL Server that have security integration with Windows NT.

Configuration Diagnostics
Installs the following configuration utilities:

> **SQL Client Configuration Utility**
> Used to determine which version of DB-Library is installed on a client and to set up SQL Server connection information on a client.
>
> **makepipe/readpipe**
> Used to test if network named pipes are working.

For details about the client utilities, see the *Microsoft SQL Server Administrator's Companion* and *Microsoft SQL Server Transact-SQL Reference*.

SQL Server Books Online

During 32-bit client software installation, the **setup** program gives you the option of installing SQL Server Books Online. This option adds the SQL Server Books Online icon to the Microsoft SQL Server 6.0 program group. By choosing this icon, you can view SQL Server documentation online.

You can install SQL Server Books Online to run from disk (which uses more disk space but increases performance) or to run from CD (which reduces disk space used). You can also choose not to install SQL Server Books Online.

Client Options for 16-bit Computers

Client software is provided for Windows- and MS-DOS–based computers.

Options for Windows-based Clients

For Windows- and Windows for Workgroups–based clients, the default SQL Server root directory is C:\SQL60, although during installation you can specify a different directory. The only directory created under the root is C:\SQL60\BIN, which contains Windows-based client executable files and Windows dynamic-link library (DLL) files. A named pipe Net-Library is placed in the \WINDOWS\SYSTEM directory.

During installation, this **setup** program creates the SQL Server Tools program group and installs ISQL/w and the SQL Client Configuration Utility.

During installation, choose one of the following as the default Net-Library:

- Named Pipes
- Novell SPX
- Banyan VINES
- TCP/IP Sockets
- Multi-Protocol

After installation, you can use the SQL Client Configuration Utility to configure the client to use other Net-Libraries. For information on these Net-Libraries, see the *Microsoft SQL Server Administrator's Companion.*

During 16-bit client software installation on Windows-based clients, the **setup** program gives you the option of installing SQL Server Books Online on the client's hard disk. This option adds the SQL Server Books Online icon to the Microsoft SQL Server Tools program group. By choosing this icon, you can view SQL Server documentation online.

Options for MS-DOS–based Clients

You create the root directories for MS-DOS–based clients. The preferred SQL Server root directory is C:\SQL60, although you can specify a different directory. Under the SQL Server root directory, create the \BIN subdirectory. In this directory, place the MS-DOS–based client executable files.

The MS-DOS–based client software provides these client utilities:

- **bcp**
- **isql**

Three TSRs can be used to communicate with the underlying network layer:

- Named Pipes (DBNMPIPE.EXE)
- Novell SPX (DBMSSPX.EXE)
- Banyan VINES (DBMSVINE.EXE)

For information on these Net-Libraries, see the *Microsoft SQL Server Administrator's Companion.*

Installing Client Software

You can install the SQL Server client software directly from the Microsoft SQL Server 6.0 CD, or you can copy the software onto a network drive and install from there. If you plan to install from a network drive, simply copy the entire contents of the CD into a shared network directory.

▶ **To install the Client software**

- For 32-bit client computers, run the same **setup** program used to install the SQL Server server components. The **setup** program is included on the SQL Server CD in the directory (\ALPHA, \I386, or \MIPS) containing the software compatible with your hardware platform's processor architecture.
- For Windows- and Windows for Workgroups–based clients, run the **setup** program located in the \CLIENTS\WIN16 directory.
- For MS-DOS–based clients, copy the files from the \CLIENTS\MSDOS directory into the \SQL60\BIN directory on the client.

Installing Client Software on 32-bit Computers

The following procedure explains how to install Microsoft SQL Server client software on computers running the Windows NT or Windows 95 operating systems. You can run the **setup** program from a local drive or from a network drive. Remember that while running **setup**, you can choose the Help button or press F1 to access the **setup** program's online Help.

Note You do not need to install SQL Server client utilities on a computer that already has the SQL Server server components installed on it, since the client utilities are automatically installed with the server components.

▶ To install client software on a Windows NT–based computer

1. From the directory containing the software compatible with your hardware platform's processor architecture, run SETUP.EXE and complete each dialog box that appears.
2. If you are installing the software on a remote computer, when the Options dialog box appears, choose the Remote button, complete the Remote Setup dialog box, and then choose the OK button.

   If you are installing the software on the local computer, skip this step.
3. From the Options dialog box, select the Install Utilities Only option, and then choose the Continue button.

   The Install Client Utilities dialog box appears.
4. Complete the Install Utilities dialog box.
   - In the Drive box, select the drive on which to install the client utilities.
   - In the Directory box, type the name of the root directory in which to install the client utilities. The default is \SQL60.
   - The Utilities To Be Installed list displays all the available utilities, with a check box for each. By default, all utilities are selected for installation. For each utility that you do not want to install, clear the check box.
5. When you have completed the Install Utilities dialog box, choose the Continue button.

   The SQL Server Books Online dialog box appears.
6. Complete the SQL Server Books Online dialog box, and choose the Continue button.
7. When the installation is complete, the Completed dialog box appears. Choose the Reboot button.

   After rebooting the computer, you can access the SQL Server graphical client utilities from the Microsoft SQL Server 6.0 program group.

When you install client utilities on a Windows 95–based computer, the AUTOEXEC.BAT file is updated with the path of the SQL Server root directory. The old AUTOEXEC.BAT file is saved as AUTOEXEC.*nnn*, where *nnn* is a number from 001 to 099.

▶ To install client software on a Windows 95–based computer

1. From the i386 directory, run SETUP.EXE and complete each dialog box that appears.
2. When the Install or Remove Client Utilities dialog box appears, select the Install Utilities Only option and then choose the Continue button.

   The Install Client Utilities dialog box appears.
3. Complete the Install Utilities dialog box and choose the Continue button.

   The SQL Server Books Online dialog box appears.
4. Complete the SQL Server Books Online dialog box and choose the Continue button.

When the installation is complete, the Completed dialog box appears.

▶ To upgrade an existing client installation on a 32-bit computer

- Make sure that no SQL Server client components are running on the computer, then perform the same steps as for a new installation.

▶ To remove the 32-bit client utilities from a Windows 95–based computer

- Run the **setup** program and choose the Remove Client Utilities option.

You can also install the MS-DOS– or 16-bit Windows-based client software on computers running the Windows NT or Windows 95 operating systems:

- To install the MS-DOS–based client software on a computer running Windows NT or Windows 95, run SETUP.EXE from the \CLIENTS\MSDOS directory of the drive containing the SQL Server 6.0 installation software. For information, see “Installing MS-DOS–based Clients,” later in this chapter.
- To install 16-bit Windows-based client software on a computer running Windows NT or Windows 95, run SETUP.EXE from the \CLIENTS\WIN16 directory of the drive containing the SQL Server 6.0 installation software. For information, see “Installing Windows-based Clients,” later in this chapter.

You can also use the **setup** program to remove the client utilities from a Windows 95–based computer.

Installing Client Software on 16-bit Computers

Client software can be installed on Windows- and MS-DOS–based computers.

Installing Windows-based Clients

You can run the **setup** program for Windows-based clients from a local drive or from a network drive. Remember that while running **setup**, you can choose the Help button or press F1 to access the **setup** program's online Help.

▶ **To install client software on a Windows-based computer**

1. From the \CLIENTS\WIN16 directory of the drive containing the SQL Server 6.0 installation software, run SETUP.EXE.
2. Follow the on-screen instructions. You will be asked to:
   - Specify the drive and directory where the software will be installed. The default is C:\SQL60.
   - Choose whether to install the optional tools (ISQL/w and the SQL Client Configuration Utility). The default is to install all optional tools.
   - Specify the default Net-Library. This can be Named Pipes, Novell SPX, Banyan VINES, TCP/IP Sockets, or Multi-Protocol.
3. When the installation is finished, a completion dialog box appears. Choose the OK button to exit the **setup** program.

To upgrade an existing client installation on a Windows- or Windows for Workgroups–based computer, first make sure that no SQL Server client components are running on the computer and then perform the same steps as for a new installation.

Installing MS-DOS–based Clients

No setup program is provided for MS-DOS–based clients. To install the software, copy the files from the \CLIENTS\MSDOS directory on the SQL Server CD into a \SQL60\BIN directory you create on the client.

It is recommended that you add the \SQL60\BIN directory you create to the computer's path. You can also add an entry to the AUTOEXEC.BAT file to start the SQL Server client software each time the client starts.

To upgrade an existing client installation on a MS-DOS–based computer, first make sure that no SQL Server client components are running on the computer, and then perform the same steps as for a new installation. To be sure a TSR is not loaded, type the following from an operating-system prompt:

enddblib

Installing ODBC Drivers

If you will use the Microsoft® Access® database management system, the Microsoft Visual Basic programming system, or another ODBC application to connect to SQL Server from a client, you should install the latest ODBC drivers on that client. ODBC drivers can be installed on client workstations running the Windows NT, Windows 95, Windows for Workgroups, and Windows operating systems.

For computers running Windows NT, the ODBC driver is installed automatically when the SQL Server **setup** program installs the server or client software.

For computers running Windows or Windows for Workgroups, you must install the ODBC drivers after the SQL Server client software is installed. An ODBC **setup** program is provided on the SQL Server CD in the \CLIENTS\WIN16\ODBC directory. This **setup** program installs the latest ODBC SQL Server driver, the ODBC Driver Manager application (ODBC) in the Control Panel, and the associated ODBC Help file.

▶ **To install the ODBC driver on a Windows- or Windows for Workgroups–based computer**

1. From the \CLIENTS\WIN16\ODBC directory on the SQL Server 6.0 CD, run SETUP.EXE.
2. Follow the on-screen instructions. For more information during the installation, choose the Help button.

If you are upgrading an existing installation, the new ODBC driver will replace the existing driver. If you want to keep the existing driver, make a backup copy of SQLSRVR.DLL and DRVSSRVR.HLP, found in the \WINDOWS\SYSTEM directory.

For more information about installing, configuring, and using the ODBC driver, see the online Help for the ODBC **setup** program and the online Help for the ODBC application in the Control Panel.

Configuring an Installed Client

Once the SQL Server client software is installed, optional client configuration parameters can be set by the SQL Server Client Configuration Utility on client computers running the Windows NT, Windows 95, Windows, and Windows for Workgroups operating systems.

No client configuration utility is provided for MS-DOS–based clients, because MS-DOS–based clients use a terminate-and-stay-resident (TSR) program, and only one Net-Library TSR can be loaded at a time.

For information on setting optional client configuration parameters, see the *Microsoft SQL Server Administrator's Companion.*

CHAPTER 7

Installation Troubleshooting

This chapter provides information that you will find helpful if you are unable to install SQL Server for Windows NT successfully.

If you are unable to correct your problem after following the instructions in this chapter, or if you are unable to connect to SQL Server after it is installed, refer to the *Microsoft SQL Server Administrator's Companion,* or contact your primary support provider.

Installation Failures

The most common cause of SQL Server for Windows NT installation failures is insufficient disk space. If your SQL Server installation has failed, the first thing you should do is check your available disk space.

If you don't seem to have a disk space problem, check the SQL Server error log or the Windows NT event log, or check the installation output files for any information that might help you isolate the problem. For information, see "Viewing Logs," later in this chapter.

In some cases, disk space problems are caused by the Windows NT virtual memory page file (PAGEFILE.SYS) growing during installation. In these cases, the message written to the log will indicate that there is a memory allocation error. For SQL Server, the PAGEFILE.SYS file must be at least 12 MB greater than the amount of memory installed on the computer. For example, for a 16-MB system, the minimum PAGEFILE.SYS size is 28 MB. The more memory that is installed on the computer, the less space you need for your PAGEFILE.SYS file.

If you are running into installation problems caused by PAGEFILE.SYS growing during the installation, you can correct the problem by:

- Freeing some space by deleting unnecessary files from your hard disk.
- Using the System application in the Windows NT Control Panel to ensure that the settings for your PAGEFILE.SYS allow it to grow to the recommended size.

For more information about checking and configuring the size of PAGEFILE.SYS, see your documentation for Windows NT.

Installation Output Files

During the installation process, the **setup** program creates a number of output files in the \SQL60\INSTALL directory that can be useful for troubleshooting installation problems. The output files have the filename extension .OUT. If the **setup** program fails to complete, examine the .OUT files for information about what caused the failure. You can examine the .OUT file by using any text editor.

Each .OUT file represents an installation script. You can search for .OUT files by typing this command from the \SQL60\INSTALL directory:

```
dir *.out /od
```

The last file returned by this command will often be the .OUT file for the script that failed. In that .OUT file, the last line will often reference the error that caused the installation to fail.

Viewing Logs

Microsoft SQL Server logs events in two locations:

- *The SQL Server error log,* which can be viewed by using SQL Enterprise Manager or any text editor. When SQL Server is configured to use its error log, each SQL Server session creates a new log, and SQL Server retains the last six logs created.

- *The Windows NT application log*, which is a special file that can be viewed only by using the Windows NT Event Viewer. When SQL Server is configured to use the Windows NT application log, each SQL Server session writes new events to that log. (Unlike the SQL Server error log, a new application log is not created each time you start SQL Server.)

Both logs record similar information; however, the SQL Server error log contains only SQL Server messages, and the Windows NT event log contains messages for all activities on the computer. Both logs automatically time stamp all events recorded.

By default, **setup** configures the server to use both logs. You can use SQL Enterprise manager to control whether SQL Server uses either or both of the logs. For more information, see the SQL Enterprise Manager online Help or the *Microsoft SQL Server Administrator's Companion.*

Note SQL Executive also writes events to the Windows NT application log and to a Task History that can be displayed using SQL Enterprise Manager. For more information on viewing the task history, see the SQL Enterprise Manager online Help or the *Microsoft SQL Server Administrator's Companion.*

Viewing the SQL Server Error Log

You can also view the SQL Server error log by using any text editor. By default, the error log is \SQL60\LOG\ERRORLOG.

When SQL Server is configured to use its error log, each SQL Server session creates a new log, and SQL Server retains the last six logs created. The current log has no filename extension. The old logs are distinguished by their filename extension (.1, .2, .3, and so on), with the lowest number being the most recent log.

If the server was successfully installed, you can also view the error log by using SQL Enterprise Manager.

▶ To view the SQL Server error log using SQL Enterprise Manager

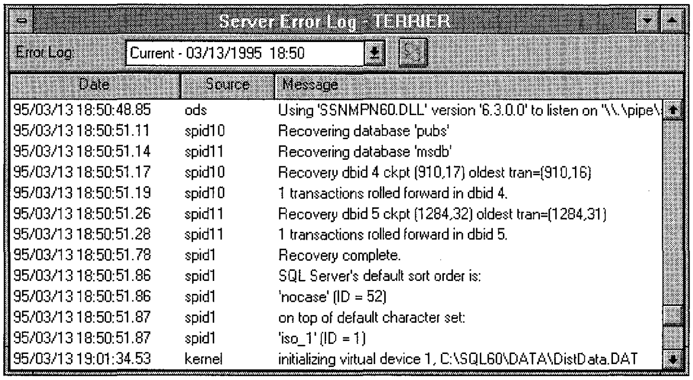
1. From the Microsoft SQL Server 6.0 program group, start SQL Enterprise Manager.

   The SQL Enterprise Manager window appears.

2. From SQL Enterprise Manager's Server Manager window, open a server group and select a server.

3. From the Server menu, choose Error Log.

   The SQL Server error log appears.



```
95/03/13 18:50:48.85   ods      Using 'SSNMPN60.DLL' version '6.3.0.0' to listen on '\\.\pipe\
95/03/13 18:50:51.11   spid10   Recovering database 'pubs'
95/03/13 18:50:51.14   spid11   Recovering database 'msdb'
95/03/13 18:50:51.17   spid10   Recovery dbid 4 ckpt (910,17) oldest tran=(910,16)
95/03/13 18:50:51.19   spid10   1 transactions rolled forward in dbid 4.
95/03/13 18:50:51.26   spid11   Recovery dbid 5 ckpt (1284,32) oldest tran=(1284,31)
95/03/13 18:50:51.28   spid11   1 transactions rolled forward in dbid 5.
95/03/13 18:50:51.78   spid1    Recovery complete.
95/03/13 18:50:51.86   spid1    SQL Server's default sort order is:
95/03/13 18:50:51.86   spid1    'nocase' (ID = 52)
95/03/13 18:50:51.87   spid1    on top of default character set:
95/03/13 18:50:51.87   spid1    'iso_1' (ID = 1)
95/03/13 19:01:34.53   kernel   initializing virtual device 1, C:\SQL60\DATA\DistData.DAT
```

When SQL Server is configured to use its error log, each SQL Server session creates a new log, and SQL Server retains the last six logs created. The current log has no filename extension. SQL Server maintains the last 6 error logs created. The old logs are distinguished by their filename extension (.1, .2, .3, and so on), with the lowest number being the most recent log.

Viewing the Windows NT Application Log

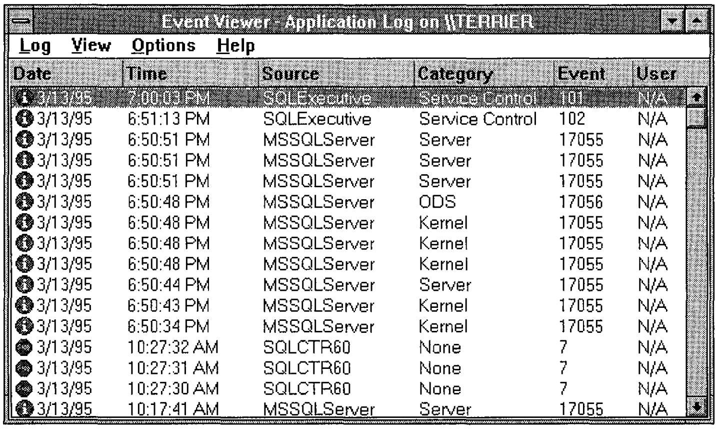
When SQL Server is configured to use the Windows NT application log, each SQL Server session writes new events to that log. The Windows NT application log is a special file that can be viewed and managed only by using Windows NT Event Viewer. Unlike the SQL Server error log, a new application log is not created each time you start SQL Server.

▶ To view the Windows NT application log

Event Viewer

1. From the Administrative Tools program group, start the Event Viewer application.

   The Event Viewer window appears.



Date	Time	Source	Category	Event	User
3/13/95	7:00:03 PM	SQLExecutive	Service Control	101	N/A
3/13/95	6:51:13 PM	SQLExecutive	Service Control	102	N/A
3/13/95	6:50:51 PM	MSSQLServer	Server	17055	N/A
3/13/95	6:50:51 PM	MSSQLServer	Server	17055	N/A
3/13/95	6:50:51 PM	MSSQLServer	Server	17055	N/A
3/13/95	6:50:48 PM	MSSQLServer	ODS	17056	N/A
3/13/95	6:50:48 PM	MSSQLServer	Kernel	17055	N/A
3/13/95	6:50:48 PM	MSSQLServer	Kernel	17055	N/A
3/13/95	6:50:48 PM	MSSQLServer	Kernel	17055	N/A
3/13/95	6:50:44 PM	MSSQLServer	Server	17055	N/A
3/13/95	6:50:43 PM	MSSQLServer	Kernel	17055	N/A
3/13/95	6:50:34 PM	MSSQLServer	Kernel	17055	N/A
3/13/95	10:27:32 AM	SQLCTR60	None	7	N/A
3/13/95	10:27:31 AM	SQLCTR60	None	7	N/A
3/13/95	10:27:30 AM	SQLCTR60	None	7	N/A
3/13/95	10:17:41 AM	MSSQLServer	Server	17055	N/A

   SQL Server events are identified by the entry **MSSQLServer** in the Source column. SQL Executive events are identified by the entry **SQLExecutive**

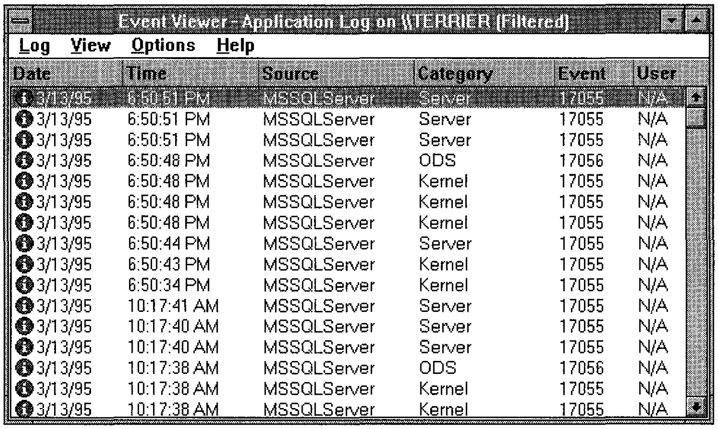
2. To view the log of a remote computer, from the Log menu choose Select Computer, and then complete the dialog box that appears.

   If you will be viewing the log of the local computer, skip this step.

3. If the Application log is not displayed, from the Log menu choose Application.

   Three Windows NT logs—Application, System, and Security—can be displayed by Event Viewer. The name of the displayed log is shown in the title bar of the dialog box. SQL Server and SQL Executive events are written to the Application log.

4. Optionally, to display only SQL Server events, choose Filter Events from the View menu, select **MSSQLServer** from the Source box, and then choose the OK button.

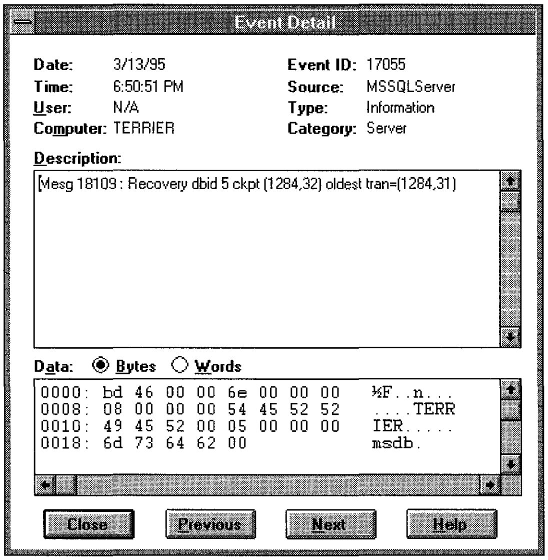


Date	Time	Source	Category	Event	User
3/13/95	6:50:51 PM	MSSQLServer	Server	17055	N/A
3/13/95	6:50:51 PM	MSSQLServer	Server	17055	N/A
3/13/95	6:50:51 PM	MSSQLServer	Server	17055	N/A
3/13/95	6:50:48 PM	MSSQLServer	ODS	17056	N/A
3/13/95	6:50:48 PM	MSSQLServer	Kernel	17055	N/A
3/13/95	6:50:48 PM	MSSQLServer	Kernel	17055	N/A
3/13/95	6:50:48 PM	MSSQLServer	Kernel	17055	N/A
3/13/95	6:50:44 PM	MSSQLServer	Server	17055	N/A
3/13/95	6:50:43 PM	MSSQLServer	Kernel	17055	N/A
3/13/95	6:50:34 PM	MSSQLServer	Kernel	17055	N/A
3/13/95	10:17:41 AM	MSSQLServer	Server	17055	N/A
3/13/95	10:17:40 AM	MSSQLServer	Server	17055	N/A
3/13/95	10:17:40 AM	MSSQLServer	Server	17055	N/A
3/13/95	10:17:38 AM	MSSQLServer	ODS	17056	N/A
3/13/95	10:17:38 AM	MSSQLServer	Kernel	17055	N/A
3/13/95	10:17:38 AM	MSSQLServer	Kernel	17055	N/A

To view only SQL Executive events, select **SQLExecutive** instead.

5. To view more information about an event, double-click the event.

   The Event Detail dialog box appears.



Date: 3/13/95
Time: 6:50:51 PM
User: N/A
Computer: TERRIER
Event ID: 17055
Source: MSSQLServer
Type: Information
Category: Server

Description:
Mesg 18109 : Recovery dbid 5 ckpt (1284,32) oldest tran=(1284,31)

Data: ● Bytes ○ Words

```
0000: bd 46 00 00 6e 00 00 00   ½F..n...
0008: 08 00 00 00 54 45 52 52   ....TERR
0010: 49 45 52 00 05 00 00 00   IER.....
0018: 6d 73 64 62 00            msdb.
```

For more information about using Event Viewer, managing the application log, and understanding the information it presents, see the online Help for Event Viewer or your documentation for Windows NT.

APPENDIX A

Character Sets and Sort Orders

The character set and sort order choices you make during installation determine the types of characters that SQL Server recognizes in your database data, the order in which SQL Server sorts data in response to queries, and the results that SQL Server returns for queries that involve character comparisons.

Carefully consider the options before installing character sets and sort orders. Changing these options after installation requires you to rebuild your databases and reload your data.

Note Additional character sets and sort orders are being developed for international users and may be included in those listed by the **setup** program.

Character Sets

The character set and sort order choices you make during installation determine the types of characters that SQL Server recognizes in your database data. For information about character sets, see the topics that follow.

Note During an installation, selecting the default character set and sort order results in a rapid installation (in most cases, 5 to 10 minutes or less). Choosing an alternate character set or sort order can result in a somewhat longer installation time (but still less than 30 minutes).

About Character Sets

Following are the SQL Server character set choices. The default is ISO 8859-1 (Latin 1 or ANSI).

- ISO 8859-1 (Latin 1 or ANSI)
- Code page 850 (Multilingual)
- Code page 437 (U.S. English)

You must use the same character set on both the client and the server, or your results can be unpredictable. However, if your databases use only the first 128 characters of a character set (for example, if you don't use characters with diacritical marks), it makes no difference which character set you use because the first 128 characters are the same for all the character set choices.

The SQL Server character set choices are illustrated in the following figures and are described in the following sections. For additional information about character sets, refer to your documentation for Windows NT.

0		32		64	@	96	`	128	Ç	160	á	192	└	224	Ó
1	☺	33	!	65	A	97	a	129	ü	161	í	193	┴	225	ß
2	☻	34	"	66	B	98	b	130	é	162	ó	194	┬	226	Ô
3	♥	35	#	67	C	99	c	131	â	163	ú	195	├	227	Ò
4	♦	36	$	68	D	100	d	132	ä	164	ñ	196	─	228	õ
5	♣	37	%	69	E	101	e	133	à	165	Ñ	197	┼	229	Õ
6	♠	38	&	70	F	102	f	134	å	166	ª	198	ã	230	µ
7	•	39	'	71	G	103	g	135	ç	167	º	199	Ã	231	þ
8	◘	40	(	72	H	104	h	136	ê	168	¿	200	╚	232	Þ
9	○	41	)	73	I	105	i	137	ë	169	®	201	╔	233	Ú
10	◙	42	*	74	J	106	j	138	è	170	¬	202	╩	234	Û
11	♂	43	+	75	K	107	k	139	ï	171	½	203	╦	235	Ù
12	♀	44	,	76	L	108	l	140	î	172	¼	204	╠	236	ý
13	♪	45	-	77	M	109	m	141	ì	173	¡	205	═	237	Ý
14	♫	46	.	78	N	110	n	142	Ä	174	«	206	╬	238	¯
15	☼	47	/	79	O	111	o	143	Å	175	»	207	¤	239	´
16	►	48	0	80	P	112	p	144	É	176	░	208	ð	240	-
17	◄	49	1	81	Q	113	q	145	æ	177	▒	209	Ð	241	±
18	↕	50	2	82	R	114	r	146	Æ	178	▓	210	Ê	242	‗
19	‼	51	3	83	S	115	s	147	ô	179	│	211	Ë	243	¾
20	¶	52	4	84	T	116	t	148	ö	180	┤	212	È	244	¶
21	§	53	5	85	U	117	u	149	ò	181	Á	213	ı	245	§
22	▬	54	6	86	V	118	v	150	û	182	Â	214	Í	246	÷
23	↨	55	7	87	W	119	w	151	ù	183	À	215	Î	247	¸
24	↑	56	8	88	X	120	x	152	ÿ	184	©	216	Ï	248	°
25	↓	57	9	89	Y	121	y	153	Ö	185	╣	217	┘	249	¨
26	→	58	:	90	Z	122	z	154	Ü	186	║	218	┌	250	·
27	←	59	;	91	[	123	{	155	ø	187	╗	219	█	251	¹
28	∟	60	<	92	\	124	¦	156	£	188	╝	220	▄	252	³
29	↔	61	=	93	]	125	}	157	Ø	189	¢	221	¦	253	²
30	▲	62	>	94	^	126	~	158	×	190	¥	222	Ì	254	■
31	▼	63	?	95	_	127	⌂	159	ƒ	191	┐	223	▀	255	

Code page 850 (Multilingual)

0		32		64	@	96	`	128	Ç	160	á	192	└	224	α
1	☺	33	!	65	A	97	a	129	ü	161	í	193	┴	225	ß
2	☻	34	"	66	B	98	b	130	é	162	ó	194	┬	226	Γ
3	♥	35	#	67	C	99	c	131	â	163	ú	195	├	227	π
4	♦	36	$	68	D	100	d	132	ä	164	ñ	196	─	228	Σ
5	♣	37	%	69	E	101	e	133	à	165	Ñ	197	┼	229	σ
6	♠	38	&	70	F	102	f	134	å	166	ª	198	╞	230	µ
7	•	39	'	71	G	103	g	135	ç	167	º	199	╟	231	τ
8	◘	40	(	72	H	104	h	136	ê	168	¿	200	╚	232	Φ
9	○	41	)	73	I	105	i	137	ë	169	⌐	201	╔	233	Θ
10	◙	42	*	74	J	106	j	138	è	170	¬	202	╩	234	Ω
11	♂	43	+	75	K	107	k	139	ï	171	½	203	╦	235	δ
12	♀	44	,	76	L	108	l	140	î	172	¼	204	╠	236	∞
13	♪	45	-	77	M	109	m	141	ì	173	¡	205	═	237	φ
14	♫	46	.	78	N	110	n	142	Ä	174	«	206	╬	238	ε
15	☼	47	/	79	O	111	o	143	Å	175	»	207	╧	239	∩
16	►	48	0	80	P	112	p	144	É	176	░	208	╨	240	≡
17	◄	49	1	81	Q	113	q	145	æ	177	▒	209	╤	241	±
18	↕	50	2	82	R	114	r	146	Æ	178	▓	210	╥	242	≥
19	‼	51	3	83	S	115	s	147	ô	179	│	211	╙	243	≤
20	¶	52	4	84	T	116	t	148	ö	180	┤	212	╘	244	⌠
21	§	53	5	85	U	117	u	149	ò	181	╡	213	╒	245	⌡
22	▬	54	6	86	V	118	v	150	û	182	╢	214	╓	246	÷
23	↨	55	7	87	W	119	w	151	ù	183	╖	215	╫	247	≈
24	↑	56	8	88	X	120	x	152	ÿ	184	╕	216	╪	248	°
25	↓	57	9	89	Y	121	y	153	Ö	185	╣	217	┘	249	∙
26	→	58	:	90	Z	122	z	154	Ü	186	║	218	┌	250	·
27	←	59	;	91	[	123	{	155	¢	187	╗	219	█	251	√
28	∟	60	<	92	\	124	¦	156	£	188	╝	220	▄	252	ⁿ
29	↔	61	=	93	]	125	}	157	¥	189	╜	221	▌	253	²
30	▲	62	>	94	^	126	~	158	₧	190	╛	222	▐	254	■
31	▼	63	?	95	_	127	⌂	159	ƒ	191	┐	223	▀	255	

Code page 437 (U.S. English)

0	■	32		64	@	96	`	128	■	160		192	À	224	à
1	■	33	!	65	A	97	a	129	■	161	¡	193	Á	225	á
2	■	34	"	66	B	98	b	130	■	162	¢	194	Â	226	â
3	■	35	#	67	C	99	c	131	■	163	£	195	Ã	227	ã
4	■	36	$	68	D	100	d	132	■	164	¤	196	Ä	228	ä
5	■	37	%	69	E	101	e	133	■	165	¥	197	Å	229	å
6	■	38	&	70	F	102	f	134	■	166	¦	198	Æ	230	æ
7	■	39	'	71	G	103	g	135	■	167	§	199	Ç	231	ç
8	■	40	(	72	H	104	h	136	■	168	¨	200	È	232	è
9	■	41	)	73	I	105	i	137	■	169	©	201	É	233	é
10	■	42	*	74	J	106	j	138	■	170	ª	202	Ê	234	ê
11	■	43	+	75	K	107	k	139	■	171	«	203	Ë	235	ë
12	■	44	,	76	L	108	l	140	■	172	¬	204	Ì	236	ì
13	■	45	-	77	M	109	m	141	■	173	-	205	Í	237	í
14	■	46	.	78	N	110	n	142	■	174	®	206	Î	238	î
15	■	47	/	79	O	111	o	143	■	175	¯	207	Ï	239	ï
16	■	48	0	80	P	112	p	144	■	176	°	208	Ð	240	ð
17	■	49	1	81	Q	113	q	145	‘	177	±	209	Ñ	241	ñ
18	■	50	2	82	R	114	r	146	’	178	²	210	Ò	242	ò
19	■	51	3	83	S	115	s	147	■	179	³	211	Ó	243	ó
20	■	52	4	84	T	116	t	148	■	180	´	212	Ô	244	ô
21	■	53	5	85	U	117	u	149	■	181	µ	213	Õ	245	õ
22	■	54	6	86	V	118	v	150	■	182	¶	214	Ö	246	ö
23	■	55	7	87	W	119	w	151	■	183	·	215	×	247	÷
24	■	56	8	88	X	120	x	152	■	184	¸	216	Ø	248	ø
25	■	57	9	89	Y	121	y	153	■	185	¹	217	Ù	249	ù
26	■	58	:	90	Z	122	z	154	■	186	º	218	Ú	250	ú
27	■	59	;	91	[	123	{	155	■	187	»	219	Û	251	û
28	■	60	<	92	\	124	|	156	■	188	¼	220	Ü	252	ü
29	■	61	=	93	]	125	}	157	■	189	½	221	Ý	253	ý
30	■	62	>	94	^	126	~	158	■	190	¾	222	Þ	254	þ
31	■	63	?	95	_	127	■	159	■	191	¿	223	ß	255	ÿ

■ Indicates that this character is not printable.

ISO 8859-1 (Latin 1 or ANSI)

ISO 8859-1

This is the default character set. ISO 8859-1 is also known as the Latin 1 or ANSI character set. It is compatible with the ANSI characters used by the Windows NT and Windows operating systems.

The ISO 8859-1 character set might be appropriate if you intend to use Windows NT– and Windows-based clients exclusively, or if you need to maintain exact compatibility with a SQL Server environment for UNIX or VMS.

Code Page 850

Code page 850 is a multilingual character set that includes all the characters used by most of the languages of European, North American, and South American countries.

Code Page 437

Code page 437 is the most commonly used character set in the United States. If you have character-based applications that depend on extended characters for graphics, use this character set. Otherwise, use code page ISO 8859-1, which provides more compatibility with languages other than United States English.

You can find out if you need to use code page 437 for specific applications by changing your Windows NT configuration to use ISO 8859-1 and then running your application. If the application looks the way you expect it to, use ISO 8859-1. If the application looks unusual, use code page 437. For information about setting the code page in your Windows NT configuration, see your documentation for Windows NT.

Setting Alternate Code Pages

The code page or pages a client uses are determined by system settings. The procedure for changing these system settings depends on the client operating system.

▶ **To set alternate code pages under the Windows NT operating system**

- Use the International application in the Windows NT Control Panel as described in your documentation for Windows NT.

▶ **To set alternate code pages under the Windows or MS-DOS operating systems**

1. Add or change entries in the CONFIG.SYS file, as indicated in the following example:

   ```
   COUNTRY=001,850,C:\DOS\COUNTRY.SYS
   DEVICE=C:\DOS\DISPLAY.SYS CON=(EGA,850,2)
   ```

   (These entries prepare the console device for code page 850.)

2. Add or change entries in the AUTOEXEC.BAT file, as indicated in the following example:

   ```
   NLSFUNC
   MODE CON CP PREP=((850,437)C:\DOS\EGA.CPI)
   CHCP 850
   ```

   (These entries set code pages 850 and 437 as alternate code pages, and set code page 850 as the default.)

Sort Orders

The sort order chosen for a server determines the order in which SQL Server sorts data in response to queries, as well as the results that SQL Server returns for queries that involve character comparisons.

For information, choose one of these topics:

About Sort Orders

SQL Server has these sort orders. The default is Dictionary order, case-insensitive.

- Dictionary order, case-insensitive
- Binary order
- Dictionary order, case-sensitive
- Dictionary order, case-insensitive, uppercase preference
- Dictionary order, case-insensitive, accent-insensitive
- Strict compatibility with version 1.*x* case-insensitive databases
- Alternate dictionary order, case-sensitive

- Alternate dictionary order, case-insensitive
- Alternate dictionary order, case-insensitive, uppercase preference
- Alternate dictionary order, case-insensitive, accent-insensitive
- Scandinavian dictionary order, case-sensitive
- Scandinavian dictionary order, case-insensitive, uppercase preference
- Custom

The 1.*x*-compatible sort order applies only to code page 850.

Binary order is the fastest sort order, but strings sorted using Binary order are not necessarily listed in dictionary order. The dictionary sort orders are generally listed in the same order as in a standard dictionary, but each dictionary sort order has a different set of rules for sorting uppercase characters, lowercase characters, and characters with diacritical marks.

The following sections describe each of the sort order choices and give examples of how similar characters compare and sort for each. For performance comparisons, see "Performance Comparisons," later in this appendix.

Dictionary Order, Case-insensitive

This is the default sort order. With this sort order, characters are compared according to the following rules:

- Uppercase and lowercase characters are treated as equivalent.
- Characters with diacritical marks are treated as different from all other characters (except their corresponding uppercase or lowercase characters).

For example:

```
A = a
Ä = ä
Å = å
a ≠ à ≠ á ≠ â ≠ ä ≠ å
A ≠ Ä ≠ Å
Jose = jose
Jose ≠ Josè
```

With this sort order, characters are compared according to the following rules:

- Uppercase characters are not given sorting preference over their corresponding lowercase characters (they can appear before or after them).
- Uppercase characters with diacritical marks are not given sorting preference over their corresponding lowercase characters (they can appear before or after them).
- Characters with diacritical marks are sorted after characters without diacritical marks.
- Characters with diacritical marks are sorted in the order they appear in the character set.

For example:

```
A, a, à, á, â, Ä, ä, Å, å
a, A, à, á, â, Ä, ä, Å, å
A, a, à, á, â, ä, Ä, Å, å
```

Binary Order

Binary order is the simplest and fastest sort order. The collating sequence for this sort order is based on the numeric value (0 through 255) of the characters in the installed character set. Using the binary sort order, *ZYXWV* sorts before *abcde* because the character *Z* precedes the character *a* in all character sets. Sorting for extended characters (such as characters with diacritical marks) depends on the installed character set.

If you want your data to sort in a more conventional order, choose one of the dictionary sort orders. Remember, however, that dictionary sort orders come with a tradeoff in performance.

Dictionary Order, Case-sensitive

With this sort order, characters are compared according to the following rules:

- Uppercase and lowercase characters are treated as not equivalent.
- Characters with diacritical marks are treated as different from all other characters (including their corresponding uppercase or lowercase characters).

For example:

```
A ≠ a
Ä ≠ ä
Å ≠ å
a ≠ à ≠ á ≠ â ≠ ä ≠ å
A ≠ Ä ≠ Å
Jose ≠ jose
Jose ≠ José
```

With this sort order, characters are compared according to the following rules:

- Uppercase characters are sorted before their corresponding lowercase characters.
- Uppercase characters with diacritical marks are sorted before their corresponding lowercase characters.
- Characters with diacritical marks are sorted after characters without diacritical marks.
- Characters with diacritical marks are sorted in the order they appear in the character set.

For example:

```
A, a, à, á, â, Ä, ä, Å, å
```

Dictionary Order, Case-insensitive, Uppercase Preference

With this sort order, characters with diacritical marks are treated as different from all other characters (except their corresponding uppercase or lowercase characters). For example:

```
A = a
Ä = ä
Å = å
a ≠ à ≠ á ≠ â ≠ ä ≠ å
A ≠ Ä ≠ Å
Jose = jose
Jose ≠ José
```

With this sort order, characters are compared according to the following rules:

- Uppercase characters are sorted before their corresponding lowercase characters.
- Uppercase characters with diacritical marks are sorted before their corresponding lowercase characters.
- Characters with diacritical marks are sorted after characters without diacritical marks.
- Characters with diacritical marks are sorted in the order they appear in the character set

For example:

```
A, a, à, á, â, Ä, ä, Å, å
```

Dictionary Order, Case-insensitive, Accent-insensitive

With this sort order, characters are compared according to the following rules:

- Uppercase and lowercase characters are treated as equivalent.
- Characters with diacritical marks are treated as equal to their counterparts without diacritical marks.

For example:

```
A = a = à = á = â = Ä = ä = Å = å
Jose = jose = José = josé
```

With this sort order, lowercase characters are paired with their corresponding uppercase characters (for example, *a* and *A*) and are sorted before any corresponding uppercase character with a diacritical mark (for example, *Ä*).

For example:

```
A, a, à, á, â, Ä, ä, Å, å
```

Additional Sort Orders for Code Page 850

In addition to the sort orders described in the preceding sections, SQL Server includes additional sort orders for code page 850:

- Strict compatibility with version 1.x case-insensitive databases
- Alternate dictionary order, case-sensitive
- Alternate dictionary order, case-insensitive
- Alternate dictionary order, case-insensitive, uppercase preference
- Alternate dictionary order, case-insensitive, accent-insensitive
- Scandinavian dictionary order, case-sensitive
- Scandinavian dictionary order, case-insensitive, uppercase preference

Version 1.*x*–compatible Sort Order

This sort order is provided for strict compatibility with SQL Server version 1.*x* case-insensitive databases. It uses the dictionary sort order with case insensitivity and no uppercase preference for the first 128 characters of the character set and the Binary sort order for the remaining 128 characters. Using this sort order choice, *A* and *a* are sorted and grouped together, but *à* and *Ä* are sorted as unique characters according to their binary values in the installed character set.

This option offers no advantage over other sort orders other than strict compatibility with earlier versions of SQL Server. Unless you have case-insensitive databases set up for earlier versions of SQL Server that use extended characters, and you require the upgraded version to have exactly the same, less functional sort order semantics, you should choose another sort order.

Alternate Sort Orders

The Alternate sort orders are similar to their counterparts, but differ as follows:

- For the case-sensitive option, all uppercase variants of a character are sorted before any equivalent lowercase variants. For example:

  ```
  A, À, Á, Ä, a, à, á, ä
  ```

  This is in contrast to the standard case-sensitive option in which the uppercase and lowercase pairings are maintained for each distinct character. For example, the characters in the previous example are sorted as follows using the standard dictionary order with case sensitivity:

  ```
  A, a, À, à, Á, á, Ä, ä
  ```

- *Ø* and *ø* sort at the end of the character set.
- *Ñ* is sorted distinctly from *N*.

Scandinavian Sort Orders

The Scandinavian sort orders differ from the other sort orders because the extended characters (*Ø*, *Å*, *Ä*, and *Ö*) are sorted after *Z*.

Sort Order IDs

The following table shows the sort order ID for each SQL Server sort order. The filename is the name of the corresponding sort order file installed in one of three character set directories under \SQL60\CHARSETS, and the filename extension indicates the corresponding character set.

ID	Name	Filename
30	Binary order	CP437BIN.437
31	Dictionary order, case-sensitive	DICTION.437
32	Dictionary order, case-insensitive	NOCASE.437
33	Dictionary order, case-insensitive, uppercase preference	NOCASEPR.437
34	Dictionary order, case-insensitive, accent-insensitive	NOACCENT.437
40	Binary order	CP850BIN.850
41	Dictionary order, case-sensitive	DICTION.859
42	Dictionary order, case-insensitive	NOCASE.850
43	Dictionary order, case-insensitive, uppercase preference	NOCASEPR.850
44	Dictionary order, case-insensitive, accent-insensitive	NOACCENT.850
49	Strict compatibility with Version 1.*x* case-insensitive databases	NOCASE34.850
50	Binary order	ISO_1BIN.ISO
51	Dictionary order, case-sensitive	DICTION.ISO
52	Dictionary order, case-insensitive	NOCASE.ISO
53	Dictionary order, case-insensitive, uppercase preference	NOCASEPR.ISO
54	Dictionary order, case-insensitive, accent-insensitive	NOACCENT.ISO

ID	Name	Filename
55	Alternate dictionary order, case-sensitive	ALTDICT.850
56	Alternate dictionary order, case-insensitive, uppercase preference	ALTNOCSP.850
57	Alternate dictionary order, case-insensitive, accent-insensitive	ALTNOACC.850
58	Scandinavian dictionary order, case-insensitive, uppercase preference	SCANNOCP.850
59	Scandinavian dictionary order, case-sensitive	SCANDICT.850
61	Alternate dictionary order, case-insensitive	ALTNOCS.850

Note Additional character sets and sort orders are being developed for international users and may be included in those listed by the **setup** program.

Performance Comparisons

Binary order is the fastest sort order option.

The following table shows the approximate decrease in sort order performance, relative to Binary order, for each sort order. The operation evaluated involved string comparisons on character fields.

Sort order	Percent slower
Dictionary order, case-sensitive	20%
Dictionary order, case-insensitive	20%
Dictionary order, case-insensitive, uppercase preference	20%
Dictionary order, case-insensitive, accent-insensitive	35%

The percentages in the table are approximate. The overall performance difference between sort orders varies significantly depending on the types of operations and the type of data used. Sorting operations that depend on string comparisons on character fields (for example GROUP BY and LIKE statements) take more time to process than other sorting operations and have the greatest effect on sorting speed.

With certain queries, if there is a nonunique clustered index on a character value, sort order options that define uppercase preference require the optimizer to perform a sort. Using a sort order with no uppercase preference, such as Dictionary order, case-insensitive (no uppercase preference), greatly improves performance because no sorting is necessary. Choosing Dictionary order, case-insensitive (no uppercase preference), causes an uppercase *A* and a lowercase *a* to be treated the same in an ORDER BY statement.

Custom

SQL Server can install custom sort orders from definition files as they become available. A sort order definition file can have any legal filename, but the convention is to use the filename extension .SRT.

▶ **To install a custom definition file**

1. If SQL Server is not yet installed, start the **setup** program and install SQL Server. Or after the server is installed, from the Microsoft SQL Server 6.0 program group start the **setup** program and choose the option to rebuild the MASTER database.
2. Follow the on-screen instructions until the Installation Options dialog box (for a new installation) or the Rebuild Options dialog box (for a MASTER database rebuild) appears.
3. Select the Order button.
4. From the list in the Sort Orders dialog box, scroll to the bottom and select Custom; then choose OK.
5. Continue with **setup**, following the on-screen instructions until the SQL Server Path dialog box appears.
6. If necessary, insert the disk containing the definition file into the disk drive.
7. In the Sort Filename box, enter the full path to the definition file. Include the drive letter, directory, and filename.
8. In the Sort Order Number box, enter the sort order ID.

   If a sort order ID is specified in documentation provided with the definition file, use that number. If no sort order ID is specified, use any number not currently used for a sort order. For a list of sort order IDs for the sort orders provided with SQL Server, see "Sort Order IDs," in this appendix.
9. Choose the Continue button and continue with **setup**, following the on-screen instructions until the installation or rebuild is complete.

The Effect on Passwords of Changing Sort Orders

Because of the introduction of one-way encrypted passwords with SQL Server 6.0, changing the sort order has an impact on the use of passwords.

When a login ID is added or a password is changed, the password encryption algorithm takes the supplied password and generates a numeric representation of it, which is stored in the *syslogins* table. (If you look in *master..syslogins.password*, what looks like garbled text is actually a textual representation of the binary encrypted password).

- If the password is generated on a case-sensitive server, the original password text, as typed, is supplied to the password algorithm. When the password is subsequently checked, the user must supply the same characters in the appropriate case for authentication.
- If the password is generated on a case-insensitive server, the password is converted to uppercase before generating and checking the encrypted form. When the password is subsequently checked, the user can supply the characters in upper- or lowercase.

When changing sort orders, if a user transfers *syslogins* out of the server with one sort order and into a new server with another sort order, the encrypted passwords will be moved as generated. This has the following effects:

- If moving from case-insensitive to case-sensitive servers, users will need to type passwords in uppercase to be authenticated.
- If moving from case-sensitive to case-insensitive servers, users will not be able to log in unless the password generated in the case-sensitive server was entirely in uppercase.

When changing sort orders, take the appropriate steps to ensure that users will be able to log in to the server when the sort order conversion is complete. If the system administrator (SA) is not able to log in because of the effects described, you will need to restore the original sort order to correct the problem.

For information about changing character sets and sort orders and for information about login IDs and passwords, see the *Microsoft SQL Server Administrator's Companion.*

APPENDIX B

Unattended Installation

In most cases you will run install SQL Server interactively, starting SETUP.EXE (as described in Chapter 4, "Server Installation") and determining the settings for each installation option by responding to the **setup** program's on-screen instructions. This can be referred to as an *attended installation.*

The **setup** program also allows you to install SQL Server without having to respond to prompts from the **setup** program. Instead, you create an initialization file that contains the settings for each installation option, and you save the initialization file on a storage device accessible to the computer(s) that will be set up. You can create as many initialization files as needed to support the servers to be installed. When appropriate, you start SETUP.EXE with command-line options that cause it to run in unattended mode, specifying the initialization file to be used to configure the new installation. This is referred to as *unattended installation.*

Unattended installation is convenient if you will be installing several SQL Servers with identical configurations. It is also convenient if you would like to use SQL Executive to schedule an installation.

Unattended setup supports installation for the local server only. It does not permit remote server installation.

Creating a Setup Initialization File

The **setup** initialization file contains the information that will be used to set configuration parameters when SQL Server is installed. Instead of responding to prompts for information from the **setup** program during installation, you enter configuration parameters ahead of time, in the configuration file, and during installation **setup** obtains the necessary settings from there.

▶ To create a setup initialization file

1. Using any text editor, create a new text file.
2. Enter configuration information following the format described in "Format of the Setup Initialization File" and "A Sample Setup Initialization File" later in this appendix.
3. Save the initialization file on a storage device accessible to the computers that will be set up. Save as a text file, using any filename ending in the .INI filename extension.

Optionally, you can create custom scripts that will run during installation. All custom scripts must be placed in one directory on the server being installed (or in one directory accessible to the server). The path to the scripts and the name of the scripts must be entered in the **setup** initialization file. For more information, see "[Scripts]," later in this appendix.

All the entries in the initialization file are strings and can also be created using the Windows function **WritePrivateProfileString**. For example:

```
// Assume lpszKeyValue is user's name string from user interface
LSTR lpszKeyValue

// Needs to be written to [License]FullName in SETUP.INI
WritePrivateProfileString("License","FullName",
   lpszKeyValue,"SETUP.INI")
```

For information about using **WritePrivateProfileString**, see your Windows programming documentation.

Important The **setup** program does not perform error checking of the values entered in the initialization file. It is up to the user to ensure that the .INI file contains valid values.

Performing an Unattended Installation

To perform an unattended installation, run SETUP.EXE from the command line using the following syntax:

setup /t IniFilePath = *filename*

where *filename* is a fully qualified filename, including the path. The path can be a local path or a redirected path to a network drive, but it cannot be a UNC path. Note that IniFilePath is case-sensitive and must be typed with the capitalization exactly as shown. There must be a space on each side of the equal sign (=).

For example, if you have created an initialization file named MYPC.INI and have saved it on drive E: in the \SETINI directory, from the directory containing the **setup** program for the server's processor architecture (that is, Intel, MIPS, or Alpha AXP), you would type:

```
setup /t IniFilePath = e:\setini\mypc.ini
```

After installation delete the setup initialization file, since it contains the password for the SQL Executive service logon account.

Format of the Setup Initialization File

The initialization file used by **setup** is a text file using regular Windows .INI file format. Following are tables listing the sections, entries, descriptions, and (where appropriate) values that can be included in the file. For examples, see "A Sample Setup Initialization File," later in this appendix.

[License]

These values correspond to those user types in the Enter Name and Organization dialog box and the licensing dialog boxes of the **setup** program.

FullName and *Mode* are required. If Mode is set to 0 (Per Server), *PerServerUsers* is required. *Orgname* and *ProductID* are optional, although an entry for *ProductID* is strongly recommended.

Entry	Description	Value
FullName	User or owner	A name
Orgname	Organization	A name
ProductID	Product identification number (supplied on the SQL Server CD box)	An ID
Mode	Licensing mode	0 Per Server 1 Per Seat
PerServerUsers	Number of users in Per Server mode	A number

[SQLPath]

These values correspond to those user types in the SQL Server Installation Path dialog box of the **setup** program. An entry is required for each value.

Entry	Description	Value
SQLPath	The directory where SQL Server will be installed	The SQL Server root directory. For example: \SQL60
LogicalSQLDrive	The drive where SQL Server will be installed	The drive. For example: E:

[MasterPath]

These values correspond to those user types in the MASTER Device Creation dialog box of the **setup** program. An entry is required for each value.

Entry	Description	Value
MasterSize	MASTER device size, in MB *	20 minimum. More is recommended. For example: 25
MasterDBFileName	MASTER device filename	A file name ending in the .DAT filename extension. For example: MASTER.DAT
MasterDBPath	The directory where the MASTER device will be placed	The path. For example: \SQL60\DATA
LogicalDBDrive	The drive where the MASTER device will be created	A drive letter. For example: C:

* *MasterSize* plus 9 MB must be less than the available disk space on the server being installed.

[NewOptions]

These values correspond to the check boxes at the bottom of the Installation Options dialog box of the **setup** program. An entry is required for each value.

Entry	Description	Value
AutoServerService	Indicates whether SQL Server will be automatically started with the system	CHECKED | NOTCHECKED
AutoExecutiveService	Indicates whether SQL Executive will be automatically started with the system	CHECKED | NOTCHECKED
BooksOnline	Installation selection for SQL Server Books Online	1 Copy to hard disk 2 Run from CD 3 Do not install

[Charset]

This value correspond to the one selected in the Select Character Set dialog box of the **setup** program. A CharSet entry is required.

Entry	Description	Value

CharSet	The characters set installed for the server	cpISO = ANSI (ISO 8859-1)
		cp 850 = 850 Multilingual
		cp437 = 437 U.S. English

For more information about character sets, see Appendix A, "Character Sets and Sort Orders."

[SortOrder]

These values correspond to those selected in the Select Sort Order dialog box of the **setup** program. An entry is required for each value.

The valid *SortFileName* entries are dependent on the *CharSet* value entered.

Entry	Description	Value
SortFileName	Sort filename	See the following table of sort file values.
SortConfigValue	Sort order value in *sysconfig*	See the following table of sort configuration values.

These are the valid sort orders for each character set.

CharSet entry	Valid SortFileName	Sort order
cpISO	iso_1bin.iso	Binary
	diction.iso	Dictionary order, case-sensitive
	nocase.iso	Dictionary order, case-insensitive
	nocasepr.iso	Dictionary order, case-insensitive, uppercase preference
	noaccent.iso	Dictionary order, case-insensitive, accent-insensitive
cp850	cp850bin.850	Binary
	diction.850	Dictionary order, case-sensitive
	nocase.850	Dictionary order, case-insensitive
	nocasepr.850	Dictionary order, case-insensitive, uppercase preference
	noaccent.850	Dictionary order, case-insensitive, accent-insensitive
	nocase34.850	Strict compatibility with Version 1.x case-sensitive databases
	altdic.850	Alternate dictionary order, case-sensitive
	altnocs850	Alternate dictionary order, case-insensitive
	altnoacc.850	Alternate dictionary order, case-insensitive, accent-insensitive
	altnocsp.850	Alternate dictionary order, case-insensitive, uppercase preference
	scannocp.850	Scandinavian dictionary order, case-insensitive, uppercase preference
	scandict.850	Scandinavian dictionary order, case-sensitive
cp437	cp437bin.437	Binary
	diction.437	Dictionary order, case-sensitive
	noaccent.437	Dictionary order, case-insensitive, accent-insensitive
	nocase.437	Dictionary order, case-insensitive
	nocasepr.437	Dictionary order, case-insensitive, uppercase preference

These are the valid entries for *SortConfigValue*.

For this SortFileName	Use this SortConfigValue entry
iso_1bin.iso	50
diction.iso	51
nocase.iso	52
nocasepr.iso	53
noaccent.iso	54
cp850bin.850	40
diction.850	41
nocase.850	42
nocasepr.850	43
noaccent.850	44
nocase34.850	49
altdic.850	55
altnoacc.850	57
altnocsp.850	56
scannocp.850	58
scandict.850	59
altnocs.850	61
cp437bin.437	30
diction.437	31
noaccent.437	34
nocase.437	32
nocasepr.437	33

For more information about sort orders, see Appendix A, "Character Sets and Sort Orders."

[Network]

These values correspond to the Net-Libraries that a user selects in the Select Network Protocols dialog box of the **setup** program and to the values a user types in the individual dialog boxes that subsequently appear for each selected Net Library.

A Net-Library is specified by entering the Net-Library name in the *NetLibList*. The *NetLibList* is entered in the format: {"*NetLib1*","*NetLib2*", ...}. Each Net-Library name is enclosed in double quotation marks and separated by commas. No spaces are allowed after a comma. For example:

```
NetLibList={"SSNMPN60","SSMSRP60","SSMSSP60","SSMSSO60"}
```

The named pipes Net-Library must always be entered, because during installation it is used by **setup**. If needed, named pipe support can be removed after installation. One or more other Net-Libraries can also be entered. For each Net-Library entered, the corresponding parameter value must be provided. For example, if you enter SSMSRP60 in the *NetLibList*, you must also provide a *MultiProtEncrypt* entry.

Entry	**Description**	**Value**
NetLibList	List of Net-Libraries to load[1]	One or more Net-Library names[1] Entered as: {"*NetLib1*","*NetLib2*", ...}
ServerNMPipe	The named pipe name [2,3]	
MultiProtEncrypt	Sets server-enforced encryption[2,4]	CHECKED | NOTCHECKED
SPXServiceName	SPX service name[2,5]	
TCPIPSocketNumber	TCP/IP socket number[2,6]	
ADSPObjectName	ADSP object name[2,7]	
VinesStreetName	Banyan VINES StreetTalk name[2]	
DecNetObjectID	DECnet object ID[2,8]	

[1] See the table that follows for Net-Library values.

[2] For information about valid parameters to enter for each Net-Library, see the *Microsoft SQL Server Administrator's Companion*.

[3] The default named pipe name is **\\.\pipe\sql\query**.

[4] The default Multi-Protocol encryption setting is NOTCHECKED.

[5] The default SPX service name is the computer name.

[6] The default TCP/IP socket number is 1433.

[7] The default ADSP object name is the computer name.

[8] The default DECnet object ID is 150.

These are the valid *NetLibList* entries:

NetLibList Entry	Is Net-Library for	Associated Value
SSNMPN60	Named Pipes	ServerNMPipe
SSMSRP60	Multi-Protocol	MultiProtEncrypt
SSMSSP60	NWLink IPX/SPX	ServiceName
SSMSS060	TCP/IP Sockets	TCPIPSocketNumber
SSMSAD60	AppleTalk ADSP	ADSPObjectName
SSMSVI60	Banyan VINES	StreetName
SSMSDE60	DECnet Sockets	DecNetObjectID

[LogonAccount]

These values correspond to those displayed in the SQL Executive Log On Account dialog box.

If *LocalSystem* is set to CHECKED, entries for *UserName* and *UserPassword* are not necessary. If *LocalSystem* is set to NOTCHECKED, entries for *UserName* and *UserPassword* are required.

Entry	Description	Value
LocalSystem	Indicates whether SQLExecutive runs under a Windows NT user account or the server's LocalSystem account.	CHECKED | NOTCHECKED
UserName	The username for the Windows NT user account.	The username. For example: user80.
UserPassword	The password for the user account.	The password. For example: XYZ45MNO.

[Scripts]

These entries specify a path to and the name of custom scripts that will be executed when the **setup** program runs. The path must be a directory on the local server being installed or a directory accessible to that server. These are optional entries, but if a *CustomScriptList* is entered, a *CustScPath* entry is required.

Entry	Description	Value
CustScPath	A path to a directory where customization scripts can be found.	A drive letter and path. For example: C:\TMP
CustomScriptList	A list of custom scripts to execute. The scripts must exist in the directory specified by *CustScPath.*	One or more script names, entered as: {"*customscript1*", "*customscript2*", ...} Each script name is enclosed in double quotation marks and separated by commas. No spaces are allowed after a comma.

A Sample Setup Initialization File

```
[License]
FullName=Royce Aydel
OrgName=CLXD5 Corporation
ProductID=AB1234567
Mode=0
PerServerUsers=50

[SQLPath]
SQLPath=\SQL60
LogicalSQLDrive=D:

[MasterPath]
MasterSize=30
LogicalDBDrive=D:
MasterDBPath=\SQL60\DATA
MasterDBFileName=MASTER.DAT

[NewOptions]
AutoServerService=CHECKED
AutoExecutiveService=CHECKED
BooksOnline=1
```

```
[CharSet]
CharSet=cpISO

[SortOrder]
SortFileName=nocase.iso
SortConfigValue=52

[Network]
NetLibList={"SSNMPN60","SSMSRP60","SSMSSP60","SSMSSO60"}
ServerNMPipe=\\.\pipe\sql\query
MultiProtEncrypt=NOTCHECKED
SPXServiceName=ROYCEA3
TCPIPSocketNumber=1433

[LogonAccount]
LocalSystem=NOTCHECKED
Username=ExecService
Password=ABC1234

[Scripts]
CustomScPath=C:\TEMP
CustScriptList="script1.sql","script2.sql","script3.sql"
```

Index

* 6 3 9 0 4 *

Recyclable Paper

Microsoft®