MySQL and Windows

MySQL and Windows

Abstract

This is the MySQL Windows extract from the MySQL 5.0 Reference Manual.

Document generated on: 2009-06-02 (revision: 15165)

Chapter 1. Installing MySQL on Windows

A native Windows distribution of MySQL has been available since version 3.21 and represents a sizable percentage of the daily downloads of MySQL. This section describes the process for installing MySQL on Windows.

> **Note**
>
> If you are upgrading MySQL from an existing installation older than MySQL 4.1.5, you must first perform the procedure described in Section 1.14, “Upgrading MySQL on Windows”.

To run MySQL on Windows, you need the following:

- A Windows operating system such as 2000, XP, Vista, or Windows Server 2003. Only 32-bit and 64-bit versions of Windows 2000 and later are supported. Windows 95/98/ME and versions of Windows older than these are no longer supported.

  A Windows operating system permits you to run the MySQL server as a service. See Section 1.11, “Starting MySQL as a Windows Service”.

  Generally, you should install MySQL on Windows using an account that has administrator rights. Otherwise, you may encounter problems with certain operations such as editing the `PATH` environment variable or accessing the `Service Control Manager`.

- TCP/IP protocol support.

- Enough space on the hard drive to unpack, install, and create the databases in accordance with your requirements (generally a minimum of 200 megabytes is recommended.)

For a list of limitations within the Windows version of MySQL, see Windows Platform Limitations.

There may also be other requirements, depending on how you plan to use MySQL:

- If you plan to connect to the MySQL server via ODBC, you need a Connector/ODBC driver. See Connectors and APIs.

- If you need tables with a size larger than 4GB, install MySQL on an NTFS or newer file system. Don't forget to use `MAX_ROWS` and `AVG_ROW_LENGTH` when you create tables. See `CREATE TABLE` Syntax.

MySQL for Windows is available in several distribution formats:

- Binary distributions are available that contain a setup program that installs everything you need so that you can start the server immediately. Another binary distribution format contains an archive that you simply unpack in the installation location and then configure yourself. For details, see Section 1.1, “Choosing An Installation Package”.

- The source distribution contains all the code and support files for building the executables using the Visual Studio compiler system.

Generally speaking, you should use a binary distribution that includes an installer. It is simpler to use than the others, and you need no additional tools to get MySQL up and running. The installer for the Windows version of MySQL, combined with a GUI Configuration Wizard, automatically installs MySQL, creates an option file, starts the server, and secures the default user accounts.

> **Caution**
>
> Using virus scanning software such as Norton/Symantec Anti-Virus on directories containing MySQL data and temporary tables can cause issues, both in terms of the performance of MySQL and the virus-scanning software misidentifying the contents of the files as containing spam. This is because of the fingerprinting mechanism used by the virus scanning software, and the way in which MySQL rapidly updates different files, which may be identified as a potential security risk.
>
> After installing MySQL Server, it is recommended that you disable virus scanning on the main directory (`datadir`) being used to store your MySQL table data. There is usually a system built into the virus scanning software to allow certain directories to be specifically ignored during virus scanning.
>
> In addition, by default, MySQL creates temporary files in the standard Windows temporary directory. To prevent the temporary files also being scanned, you should configure a separate temporary directory for MySQL temporary files and add this to the virus scanning exclusion list. To do this, add a configuration option for the `tmpdir` parameter to

- your `my.ini` configuration file. For more information, see Section 1.7, “Creating an Option File”.

The following section describes how to install MySQL on Windows using a binary distribution. To use an installation package that does not include an installer, follow the procedure described in Section 1.5, “Installing MySQL from a Noinstall Zip Archive”. To install using a source distribution, see Chapter 4, *Installing MySQL from Source on Windows*.

MySQL distributions for Windows can be downloaded from http://dev.mysql.com/downloads/. See How to Get MySQL.

1.1. Choosing An Installation Package

For MySQL 5.0, there are three installation packages to choose from when installing MySQL on Windows:

- **The Essentials package.** This package has a file name similar to `mysql-essential-5.0.84-win32.msi` and contains the minimum set of files needed to install MySQL on Windows, including the Configuration Wizard. This package does not include optional components such as the embedded server and benchmark suite.
- **The Complete package.** This package has a file name similar to `mysql-5.0.84-win32.zip` and contains all files needed for a complete Windows installation, including the Configuration Wizard. This package includes optional components such as the embedded server and benchmark suite.
- **The no-install archive.** This package has a file name similar to `mysql-noinstall-5.0.84-win32.zip` and contains all the files found in the Complete install package, with the exception of the Configuration Wizard. This package does not include an automated installer, and must be manually installed and configured.

The Essentials package is recommended for most users. It is provided as an `.msi` file for use with the Windows Installer. The Complete and Noinstall distributions are packaged as Zip archives. To use them, you must have a tool that can unpack `.zip` files.

Your choice of install package affects the installation process you must follow. If you choose to install either the Essentials or Complete install packages, see Section 1.2, “Installing MySQL with the Automated Installer”. If you choose to install MySQL from the Noinstall archive, see Section 1.5, “Installing MySQL from a Noinstall Zip Archive”.

1.2. Installing MySQL with the Automated Installer

New MySQL users can use the MySQL Installation Wizard and MySQL Configuration Wizard to install MySQL on Windows. These are designed to install and configure MySQL in such a way that new users can immediately get started using MySQL.

The MySQL Installation Wizard and MySQL Configuration Wizard are available in the Essentials and Complete install packages. They are recommended for most standard MySQL installations. Exceptions include users who need to install multiple instances of MySQL on a single server host and advanced users who want complete control of server configuration.

1.3. Using the MySQL Installation Wizard

MySQL Installation Wizard is an installer for the MySQL server that uses the latest installer technologies for Microsoft Windows. The MySQL Installation Wizard, in combination with the MySQL Configuration Wizard, allows a user to install and configure a MySQL server that is ready for use immediately after installation.

The MySQL Installation Wizard is the standard installer for all MySQL server distributions, version 4.1.5 and higher. Users of previous versions of MySQL need to shut down and remove their existing MySQL installations manually before installing MySQL with the MySQL Installation Wizard. See Section 1.3.6, “Upgrading MySQL with the Installation Wizard”, for more information on upgrading from a previous version.

The Microsoft Windows Installer (MSI) is the standard for application installations on Windows 2000 and later versions. The MySQL Installation Wizard makes use of this technology to provide a smoother and more flexible installation process.

The Microsoft Windows Installer Engine was updated with the release of Windows XP; those using a previous version of Windows can reference this Microsoft Knowledge Base article for information on upgrading to the latest version of the Windows Installer Engine.

In addition, Microsoft has introduced the WiX (Windows Installer XML) toolkit, which is the first highly acknowledged Open Source project from Microsoft. We have switched to WiX because it is an Open Source project and it allows us to handle the complete Windows installation process in a flexible manner using scripts.

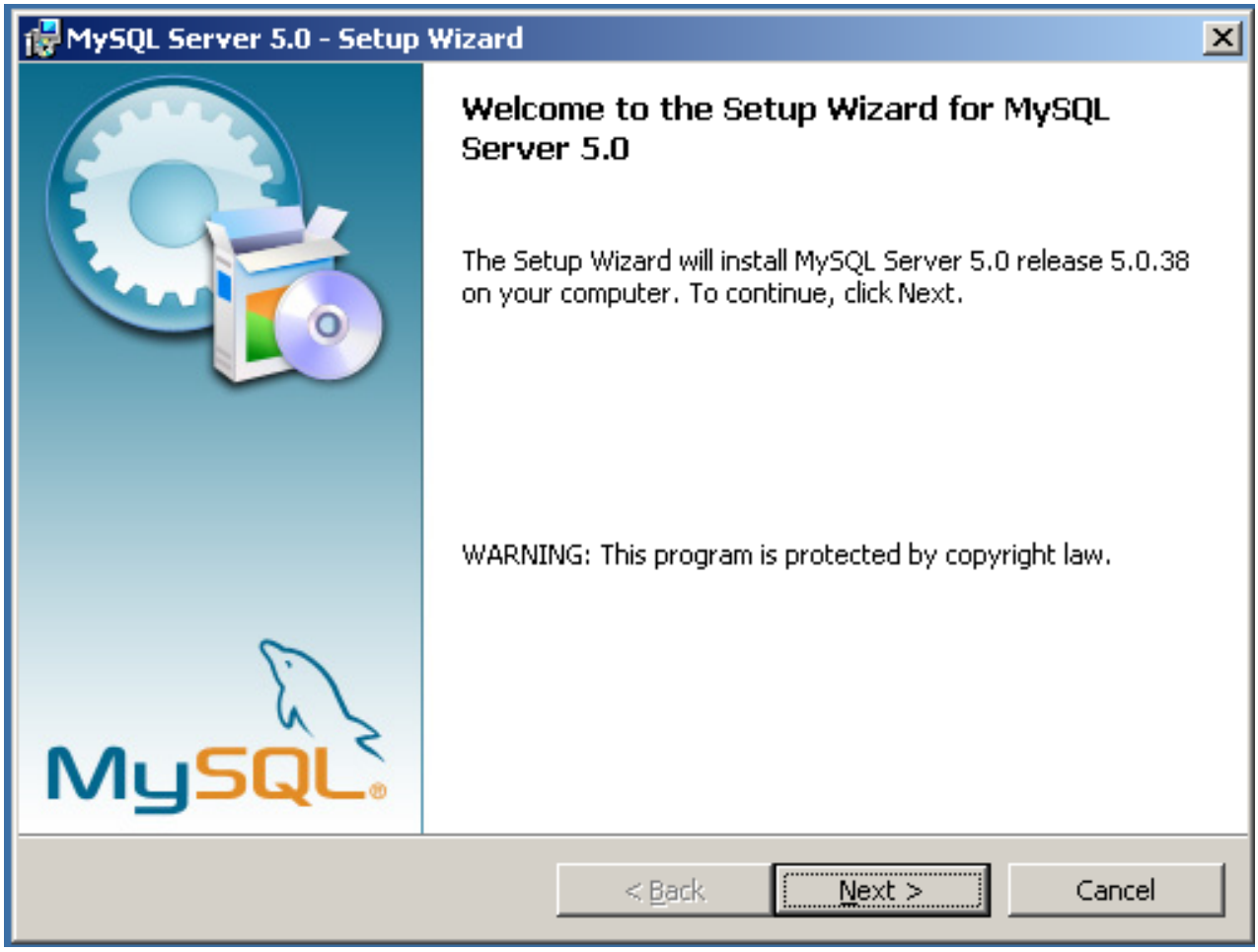
Improving the MySQL Installation Wizard depends on the support and feedback of users. If you find that the MySQL Installation Wizard is lacking some feature important to you, or if you discover a bug, please report it in our bugs database using the instructions given in How to Report Bugs or Problems.

1.3.1. Downloading and Starting the MySQL Installation Wizard

MySQL installation packages can be downloaded from http://dev.mysql.com/downloads/. If the package you download is contained within a Zip archive, you need to extract the archive first.

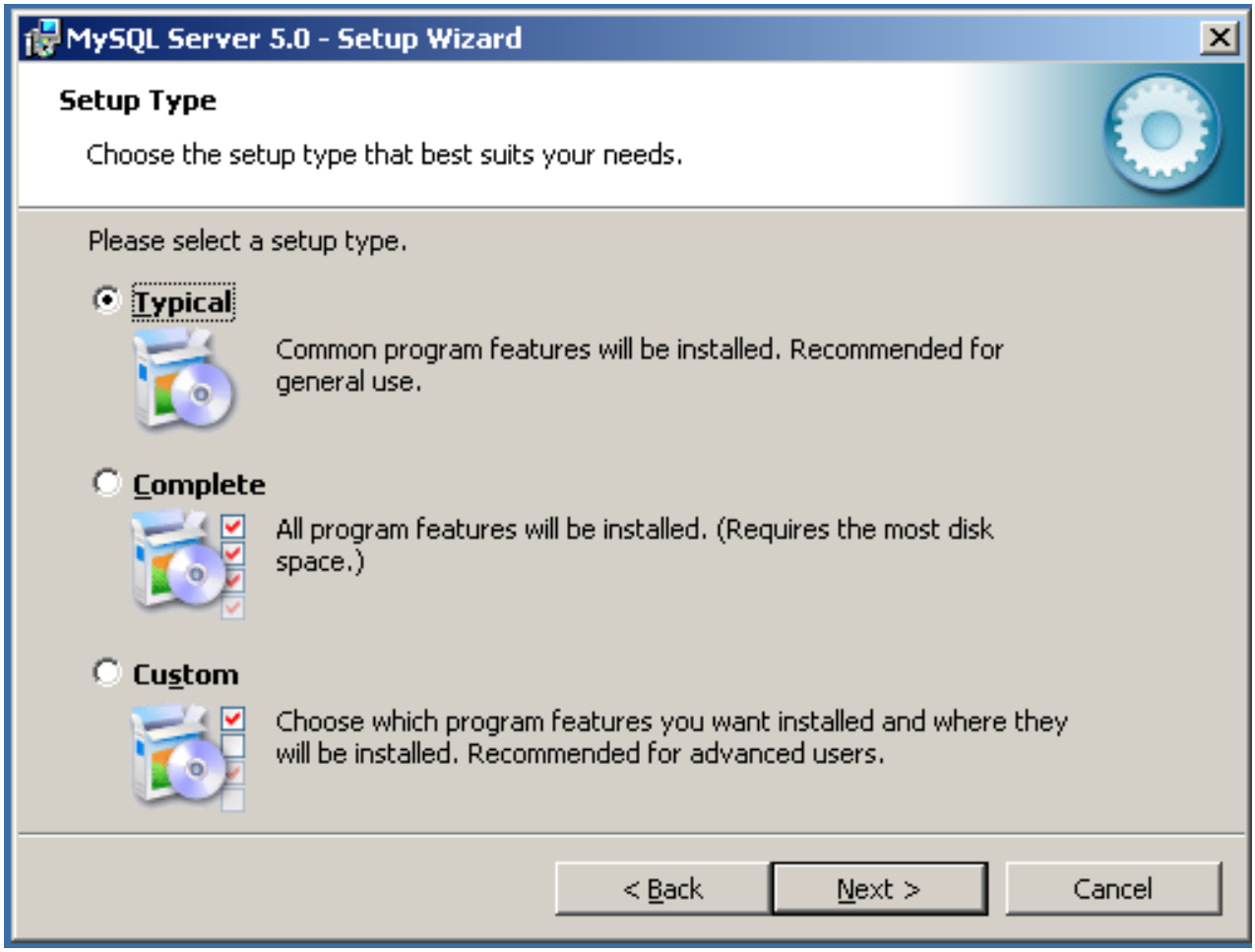
> **Note**
>
> If you are installing on Windows Vista it is best to open a network port for MySQL to use before beginning the installation. To do this, first ensure that you are logged in as an Administrator, then go to the `Control Panel` and double-click the `Windows Firewall` icon. Choose the `Allow a program through Windows Firewall` option and click the ADD PORT button. Enter `MySQL` into the **NAME** text box and `3306` (or other port of your choice) into the **PORT NUMBER** text box. Also ensure that the **TCP** protocol radio button is selected. If you wish, you can also limit access to the MySQL server by choosing the **CHANGE SCOPE** button. Confirm your choices by clicking the OK button. If you do not open a port prior to installation, you cannot configure the MySQL server immediately after installation. Additionally, when running the MySQL Installation Wizard on Windows Vista, ensure that you are logged in as a user with administrative rights.

The process for starting the wizard depends on the contents of the installation package you download. If there is a `setup.exe` file present, double-click it to start the installation process. If there is an `.msi` file present, double-click it to start the installation process.



1.3.2. Choosing an Installation Type

There are three installation types available: **Typical**, **Complete**, and **Custom**.



The **Typical** installation type installs the MySQL server, the `mysql` command-line client, and the command-line utilities. The command-line clients and utilities include `mysqldump`, `myisamchk`, and several other tools to help you manage the MySQL server.

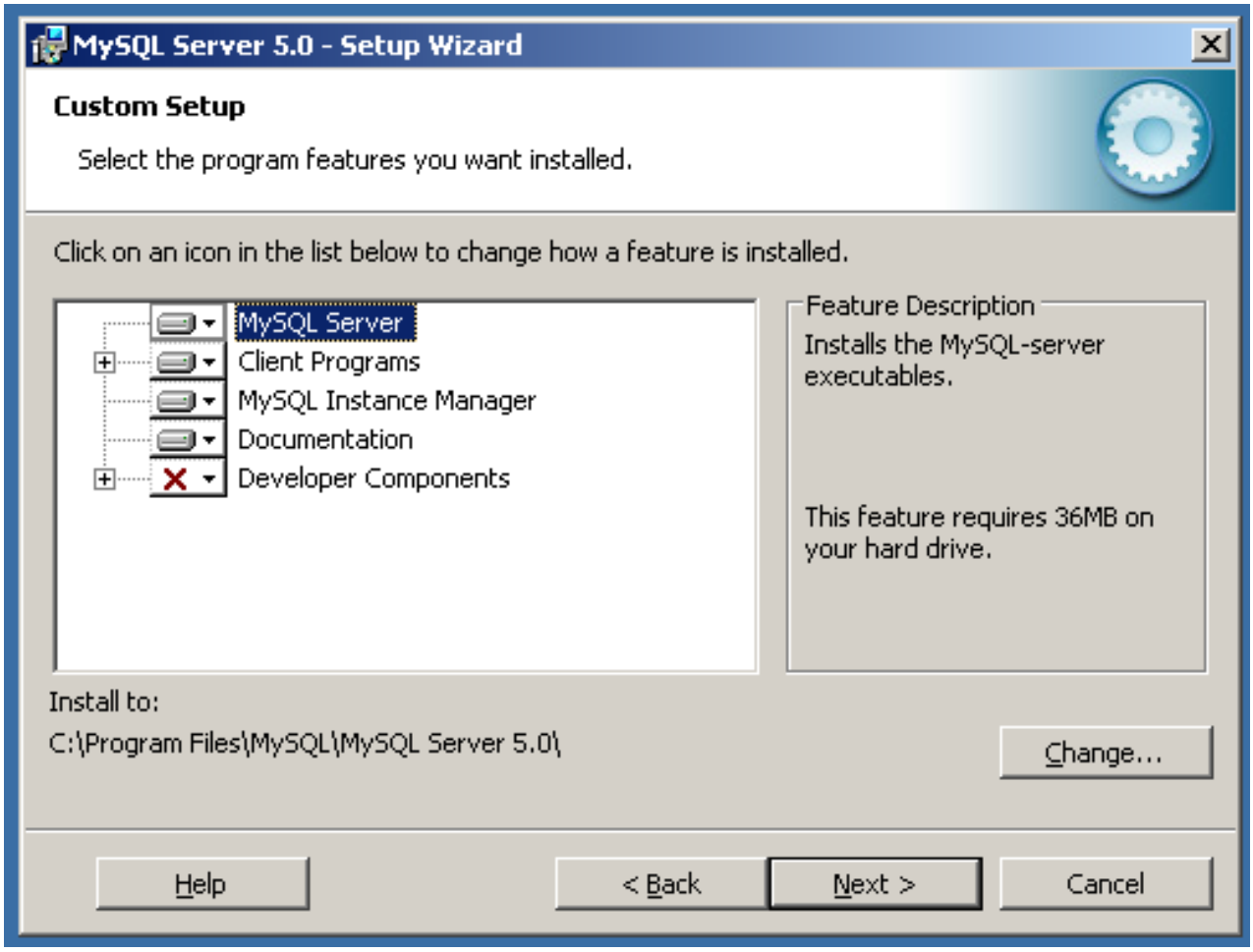
The **Complete** installation type installs all components included in the installation package. The full installation package includes components such as the embedded server library, the benchmark suite, support scripts, and documentation.

The **Custom** installation type gives you complete control over which packages you wish to install and the installation path that is used. See Section 1.3.3, "The Custom Installation Dialog", for more information on performing a custom install.

If you choose the **Typical** or **Complete** installation types and click the NEXT button, you advance to the confirmation screen to verify your choices and begin the installation. If you choose the **Custom** installation type and click the NEXT button, you advance to the custom installation dialog, described in Section 1.3.3, "The Custom Installation Dialog".

1.3.3. The Custom Installation Dialog

If you wish to change the installation path or the specific components that are installed by the MySQL Installation Wizard, choose the **Custom** installation type.



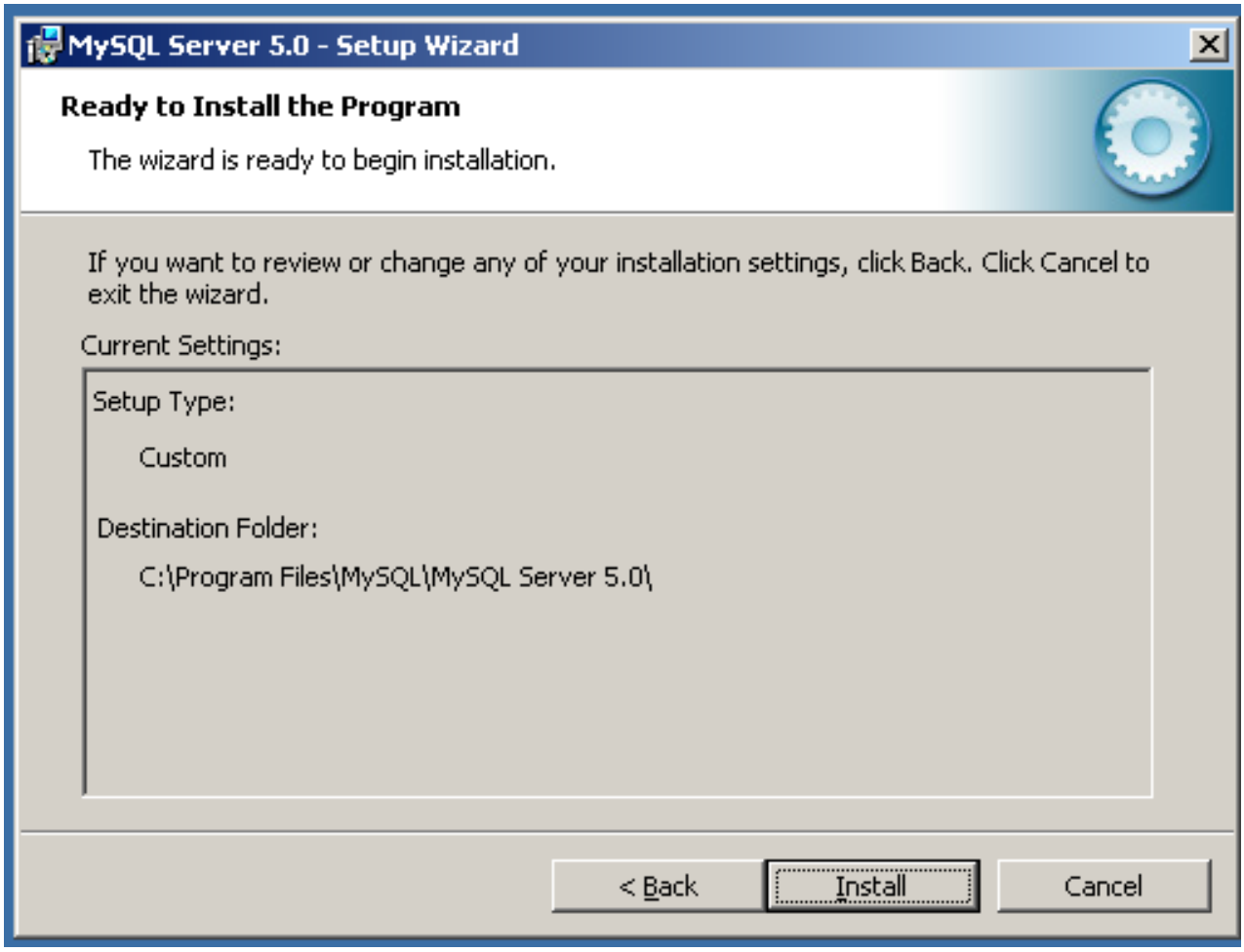
A tree view on the left side of the custom install dialog lists all available components. Components that are not installed have a red X icon; components that are installed have a gray icon. To change whether a component is installed, click on that component's icon and choose a new option from the drop-down list that appears.

You can change the default installation path by clicking the CHANGE... button to the right of the displayed installation path.

After choosing your installation components and installation path, click the NEXT button to advance to the confirmation dialog.

1.3.4. The Confirmation Dialog

Once you choose an installation type and optionally choose your installation components, you advance to the confirmation dialog. Your installation type and installation path are displayed for you to review.



To install MySQL if you are satisfied with your settings, click the INSTALL button. To change your settings, click the BACK button. To exit the MySQL Installation Wizard without installing MySQL, click the CANCEL button.

After installation is complete, you have the option of registering with the MySQL web site. Registration gives you access to post in the MySQL forums at forums.mysql.com, along with the ability to report bugs at bugs.mysql.com and to subscribe to our newsletter. The final screen of the installer provides a summary of the installation and gives you the option to launch the MySQL Configuration Wizard, which you can use to create a configuration file, install the MySQL service, and configure security settings.

1.3.5. Changes Made by MySQL Installation Wizard

Once you click the INSTALL button, the MySQL Installation Wizard begins the installation process and makes certain changes to your system which are described in the sections that follow.

Changes to the Registry

The MySQL Installation Wizard creates one Windows registry key in a typical install situation, located in `HKEY_LOCAL_MACHINE\SOFTWARE\MySQL AB`.

The MySQL Installation Wizard creates a key named after the major version of the server that is being installed, such as `MySQL Server 5.0`. It contains two string values, `Location` and `Version`. The `Location` string contains the path to the installation directory. In a default installation it contains `C:\Program Files\MySQL\MySQL Server 5.0\`. The `Version` string contains the release number. For example, for an installation of MySQL Server 5.0.84, the key contains a value of `5.0.84`.

These registry keys are used to help external tools identify the installed location of the MySQL server, preventing a complete scan of the hard-disk to determine the installation path of the MySQL server. The registry keys are not required to run the server, and if you install MySQL using the `noinstall` Zip archive, the registry keys are not created.

Changes to the Start Menu

The MySQL Installation Wizard creates a new entry in the Windows START menu under a common MySQL menu heading named after the major version of MySQL that you have installed. For example, if you install MySQL 5.0, the MySQL Installation Wizard creates a MySQL Server 5.0 section in the START menu.

The following entries are created within the new START menu section:

- MySQL Command Line Client: This is a shortcut to the `mysql` command-line client and is configured to connect as the `root` user. The shortcut prompts for a `root` user password when you connect.
- MySQL Server Instance Config Wizard: This is a shortcut to the MySQL Configuration Wizard. Use this shortcut to configure a newly installed server, or to reconfigure an existing server.
- MySQL Documentation: This is a link to the MySQL server documentation that is stored locally in the MySQL server installation directory. This option is not available when the MySQL server is installed using the Essentials installation package.

Changes to the File System

The MySQL Installation Wizard by default installs the MySQL 5.0 server to `C:\Program Files\MySQL\MySQL Server 5.0`, where `Program Files` is the default location for applications in your system, and `5.0` is the major version of your MySQL server. This is the recommended location for the MySQL server, replacing the former default location `C:\mysql`.

By default, all MySQL applications are stored in a common directory at `C:\Program Files\MySQL`, where `Program Files` is the default location for applications in your Windows installation. A typical MySQL installation on a developer machine might look like this:

```
C:\Program Files\MySQL\MySQL Server 5.0
C:\Program Files\MySQL\MySQL Administrator 1.0
C:\Program Files\MySQL\MySQL Query Browser 1.0
```

This approach makes it easier to manage and maintain all MySQL applications installed on a particular system.

1.3.6. Upgrading MySQL with the Installation Wizard

The MySQL Installation Wizard can perform server upgrades automatically using the upgrade capabilities of MSI. That means you do not need to remove a previous installation manually before installing a new release. The installer automatically shuts down and removes the previous MySQL service before installing the new version.

Automatic upgrades are available only when upgrading between installations that have the same major and minor version numbers. For example, you can upgrade automatically from MySQL 4.1.5 to MySQL 4.1.6, but not from MySQL 4.1 to MySQL 5.0.

See Section 1.14, “Upgrading MySQL on Windows”.

1.4. MySQL Server Instance Configuration Wizard

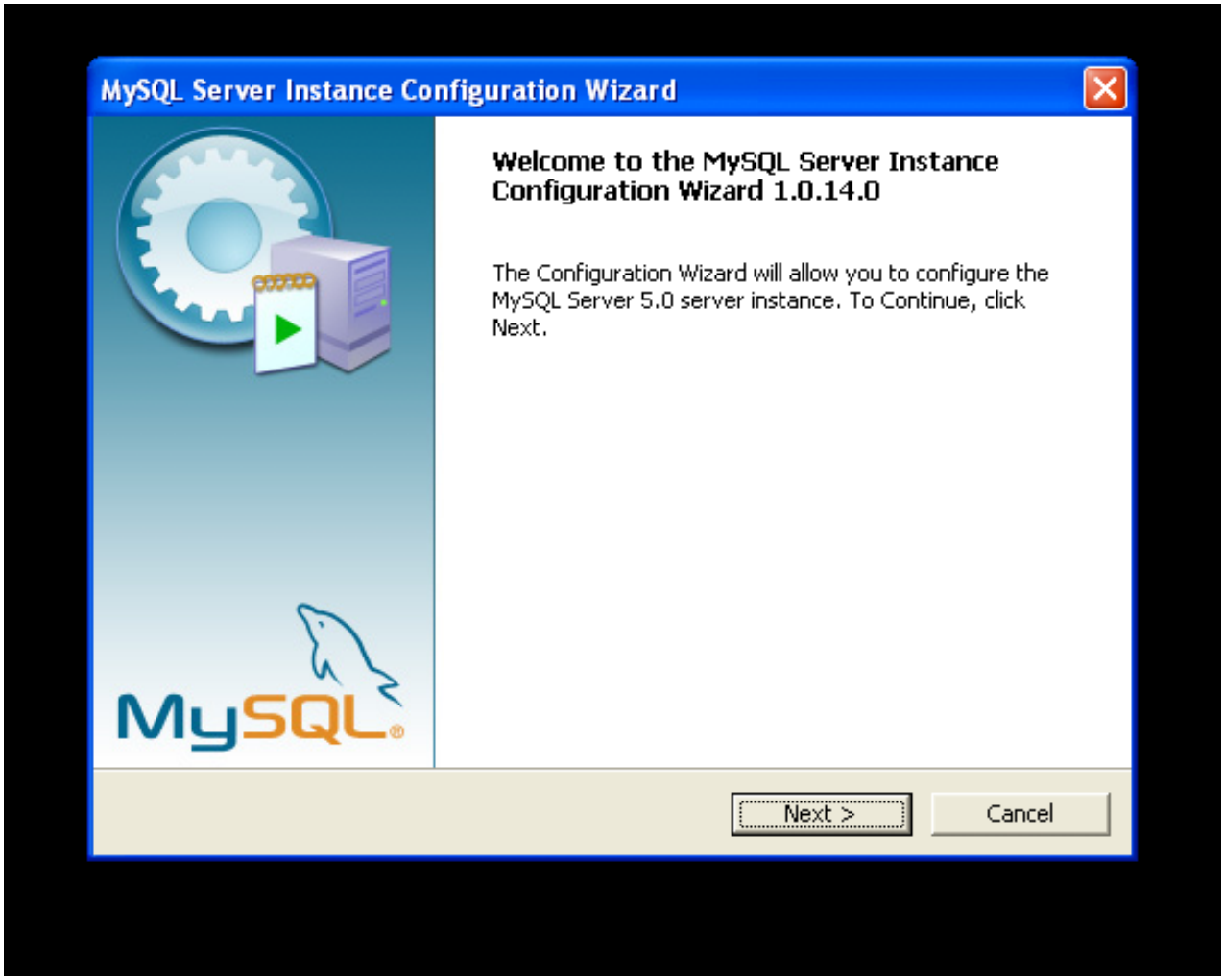
The MySQL Server Instance Configuration Wizard helps automate the process of configuring your server. It creates a custom MySQL configuration file (`my.ini` or `my.cnf`) by asking you a series of questions and then applying your responses to a template to generate the configuration file that is tuned to your installation.

The MySQL Server Instance Configuration Wizard is included with the MySQL 5.0 server. The MySQL Server Instance Configuration Wizard is only available for Windows.

1.4.1. Starting the MySQL Server Instance Configuration Wizard

The MySQL Server Instance Configuration Wizard is normally started as part of the installation process. You should only need to run the MySQL Server Instance Configuration Wizard again when you need to change the configuration parameters of your server.

If you chose not to open a port prior to installing MySQL on Windows Vista, you can choose to use the MySQL Server Configuration Wizard after installation. However, you must open a port in the Windows Firewall. To do this see the instructions given in Section 1.3.1, “Downloading and Starting the MySQL Installation Wizard”. Rather than opening a port, you also have the option of adding MySQL as a program that bypasses the Windows Firewall. One or the other option is sufficient — you need not do both. Additionally, when running the MySQL Server Configuration Wizard on Windows Vista ensure that you are logged in as a user with administrative rights.



You can launch the MySQL Configuration Wizard by clicking the MySQL Server Instance Config Wizard entry in the MySQL section of the Windows START menu.

Alternatively, you can navigate to the `bin` directory of your MySQL installation and launch the `MySQLInstanceConfig.exe` file directly.

The MySQL Server Instance Configuration Wizard places the `my.ini` file in the installation directory for the MySQL server. This helps associate configuration files with particular server instances.

To ensure that the MySQL server knows where to look for the `my.ini` file, an argument similar to this is passed to the MySQL server as part of the service installation:

```
--defaults-file="C:\Program Files\MySQL\MySQL Server 5.0\my.ini"
```

Here, *`C:\Program Files\MySQL\MySQL Server 5.0`* is replaced with the installation path to the MySQL Server. The `--defaults-file` option instructs the MySQL server to read the specified file for configuration options when it starts.

Apart from making changes to the `my.ini` file by running the MySQL Server Instance Configuration Wizard again, you can modify it by opening it with a text editor and making any necessary changes. You can also modify the server configuration with the MySQL Administrator utility. For more information about server configuration, see Server Command Options.

MySQL clients and utilities such as the `mysql` and `mysqldump` command-line clients are not able to locate the `my.ini` file located in the server installation directory. To configure the client and utility applications, create a new `my.ini` file in the Windows installation directory (for example, `C:\WINDOWS`).

Under Windows Server 2003, Windows Server 2000 and Windows XP, MySQL Server Instance Configuration Wizard will configure MySQL to work as a Windows service. To start and stop MySQL you use the `Services` application that is supplied as part of the Windows Administrator Tools.

1.4.2. Choosing a Maintenance Option

If the MySQL Server Instance Configuration Wizard detects an existing configuration file, you have the option of either reconfiguring your existing server, or removing the server instance by deleting the configuration file and stopping and removing the MySQL service.

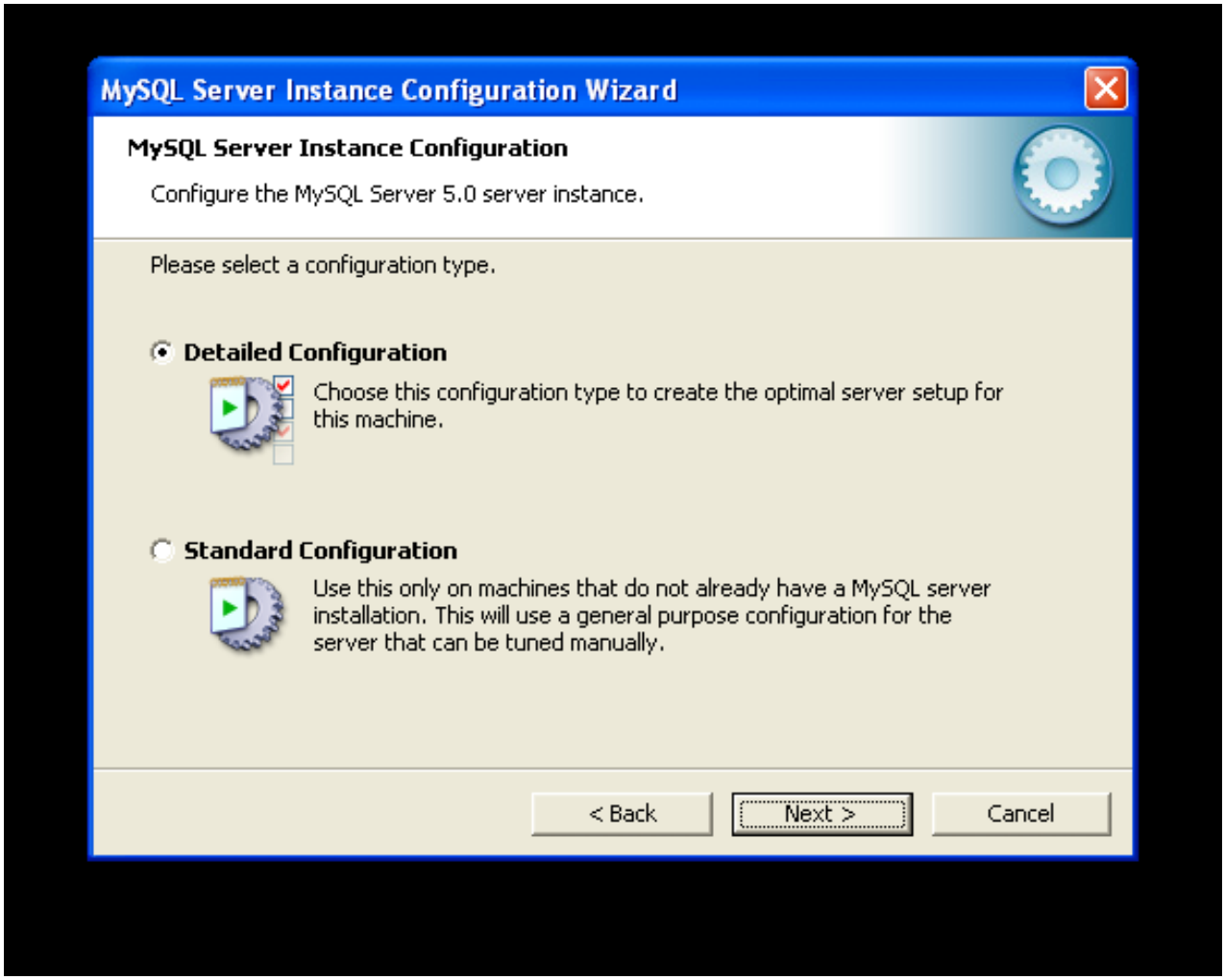
To reconfigure an existing server, choose the Re-configure Instance option and click the NEXT button. Any existing configuration file is not overwritten, but renamed (within the same directory) using a timestamp (Windows) or sequential number (Linux). To remove the existing server instance, choose the Remove Instance option and click the NEXT button.

If you choose the Remove Instance option, you advance to a confirmation window. Click the EXECUTE button. The MySQL Server Configuration Wizard stops and removes the MySQL service, and then deletes the configuration file. The server installation and its `data` folder are not removed.

If you choose the Re-configure Instance option, you advance to the CONFIGURATION TYPE dialog where you can choose the type of installation that you wish to configure.

1.4.3. Choosing a Configuration Type

When you start the MySQL Server Instance Configuration Wizard for a new MySQL installation, or choose the Re-configure Instance option for an existing installation, you advance to the CONFIGURATION TYPE dialog.



There are two configuration types available: Detailed Configuration and Standard Configuration. The Standard Configuration option is intended for new users who want to get started with MySQL quickly without having to make many decisions about server configuration. The Detailed Configuration option is intended for advanced users who want more fine-grained control over server configuration.

If you are new to MySQL and need a server configured as a single-user developer machine, the Standard Configuration should suit your needs. Choosing the Standard Configuration option causes the MySQL Configuration Wizard to set all configuration options automatically with the exception of Service Options and Security Options.

The Standard Configuration sets options that may be incompatible with systems where there are existing MySQL installations. If you have an existing MySQL installation on your system in addition to the installation you wish to configure, the Detailed Config-

uration option is recommended.

To complete the Standard Configuration, please refer to the sections on Service Options and Security Options in Section 1.4.10, “The Service Options Dialog”, and Section 1.4.11, “The Security Options Dialog”, respectively.

1.4.4. The Server Type Dialog

There are three different server types available to choose from. The server type that you choose affects the decisions that the MySQL Server Instance Configuration Wizard makes with regard to memory, disk, and processor usage.

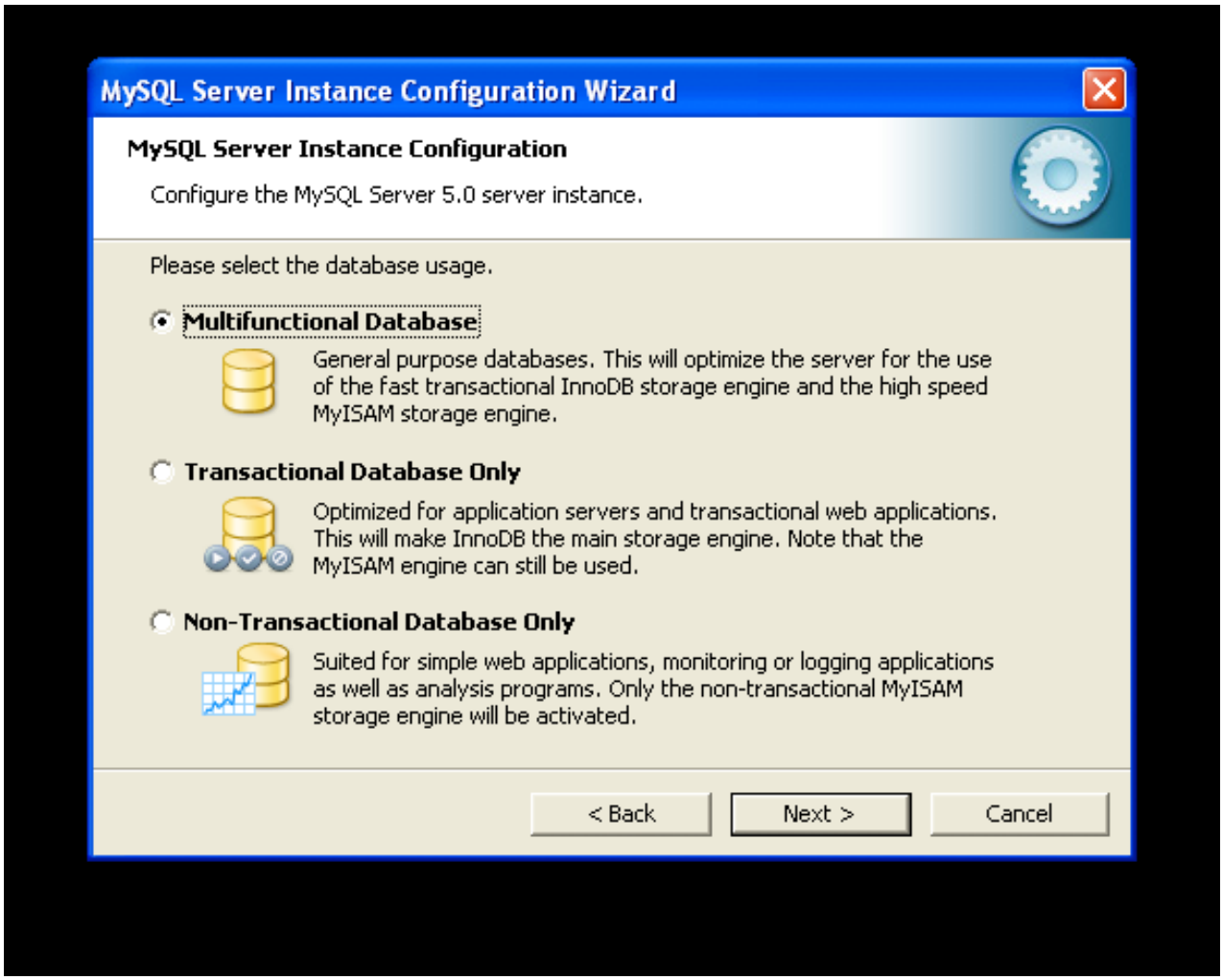
- Developer Machine: Choose this option for a typical desktop workstation where MySQL is intended only for personal use. It is assumed that many other desktop applications are running. The MySQL server is configured to use minimal system resources.
- Server Machine: Choose this option for a server machine where the MySQL server is running alongside other server applications such as FTP, email, and Web servers. The MySQL server is configured to use a moderate portion of the system resources.
- Dedicated MySQL Server Machine: Choose this option for a server machine that is intended to run only the MySQL server. It is assumed that no other applications are running. The MySQL server is configured to use all available system resources.

> **Note**
>
> By selecting one of the preconfigured configurations, the values and settings of various options in your `my.cnf` or `my.ini` will be altered accordingly. The default values and options as described in the reference manual may therefore be different to the options and values that were created during the execution of the configuration wizard.

1.4.5. The Database Usage Dialog

The DATABASE USAGE dialog allows you to indicate the storage engines that you expect to use when creating MySQL tables. The

option you choose determines whether the InnoDB storage engine is available and what percentage of the server resources are available to InnoDB.



- Multifunctional Database: This option enables both the InnoDB and MyISAM storage engines and divides resources evenly between the two. This option is recommended for users who use both storage engines on a regular basis.
- Transactional Database Only: This option enables both the InnoDB and MyISAM storage engines, but dedicates most server resources to the InnoDB storage engine. This option is recommended for users who use InnoDB almost exclusively and make only minimal use of MyISAM.
- Non-Transactional Database Only: This option disables the InnoDB storage engine completely and dedicates all server resources to the MyISAM storage engine. This option is recommended for users who do not use InnoDB.

The Configuration Wizard uses a template to generate the server configuration file. The DATABASE USAGE dialog sets one of the following option strings:

```
Multifunctional Database:        MIXED
Transactional Database Only:     INNODB
Non-Transactional Database Only: MYISAM
```

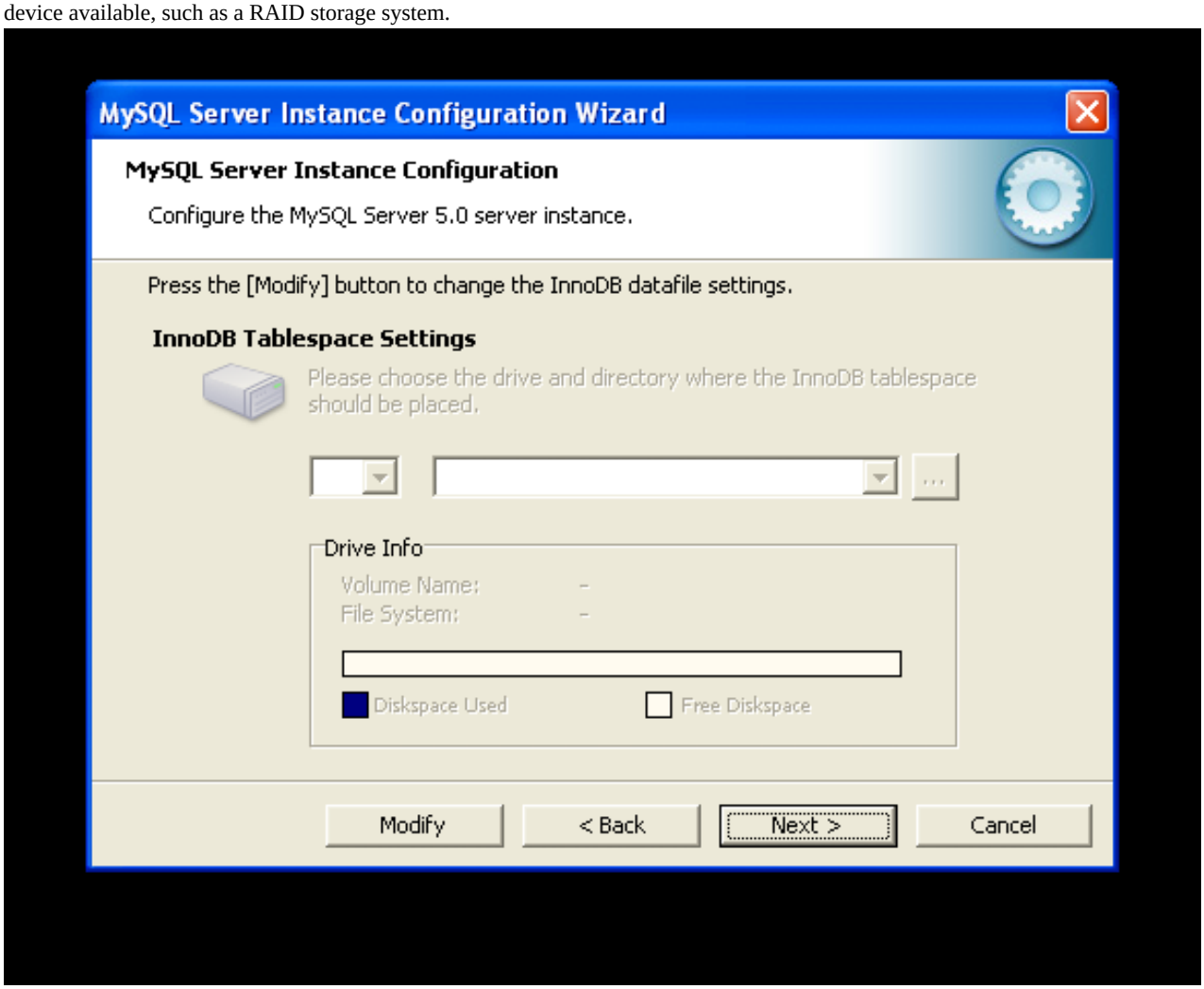
When these options are processed through the default template (my-template.ini) the result is:

```
Multifunctional Database:
default-storage-engine=InnoDB
_myisam_pct=50
Transactional Database Only:
default-storage-engine=InnoDB
_myisam_pct=5
Non-Transactional Database Only:
default-storage-engine=MyISAM
_myisam_pct=100
skip-innodb
```

The `_myisam_pct` value is used to calculate the percentage of resources dedicated to `MyISAM`. The remaining resources are allocated to `InnoDB`.

1.4.6. The InnoDB Tablespace Dialog

Some users may want to locate the `InnoDB` tablespace files in a different location than the MySQL server data directory. Placing the tablespace files in a separate location can be desirable if your system has a higher capacity or higher performance storage device available, such as a RAID storage system.

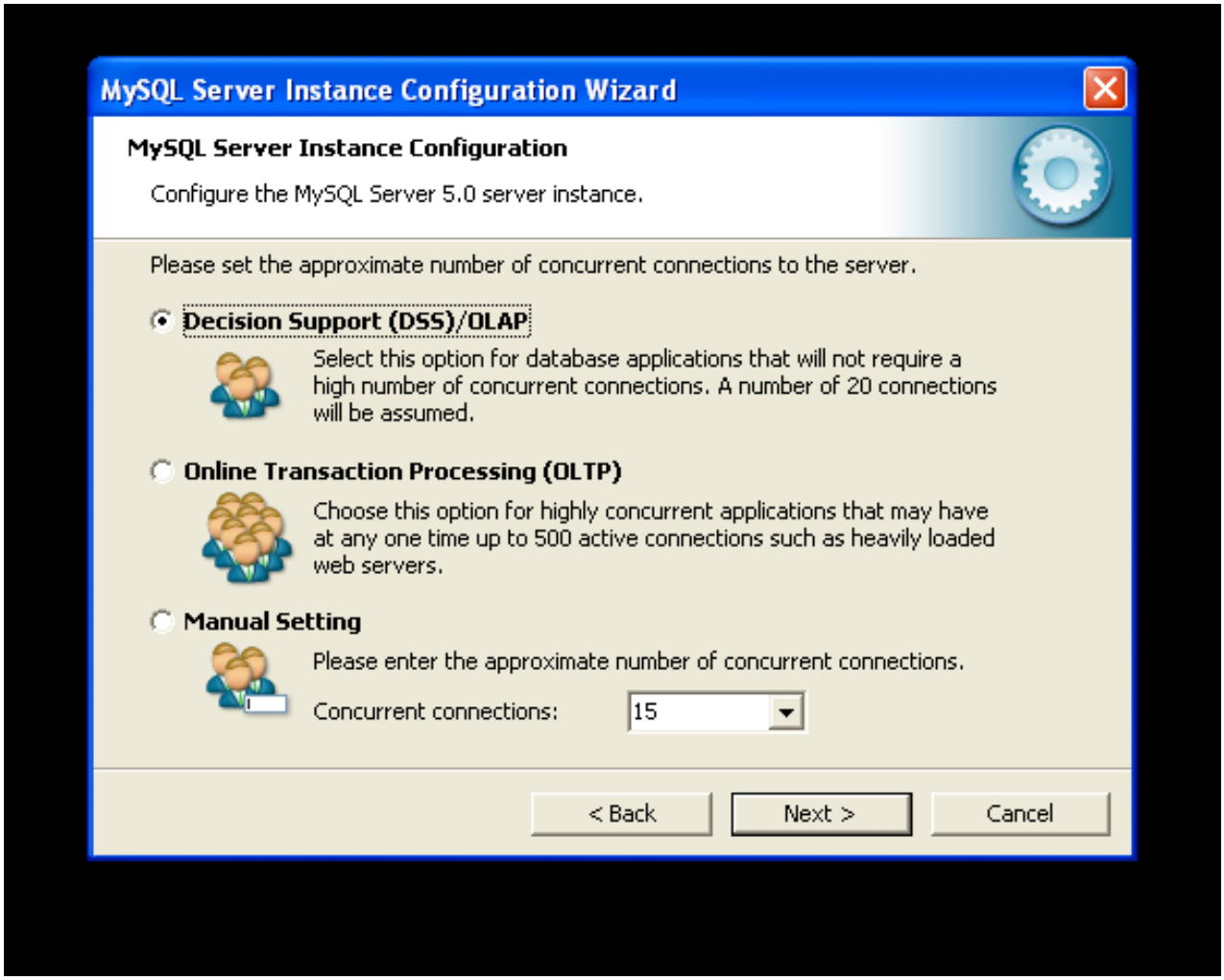


To change the default location for the `InnoDB` tablespace files, choose a new drive from the drop-down list of drive letters and choose a new path from the drop-down list of paths. To create a custom path, click the ... button.

If you are modifying the configuration of an existing server, you must click the MODIFY button before you change the path. In this situation you must move the existing tablespace files to the new location manually before starting the server.

1.4.7. The Concurrent Connections Dialog

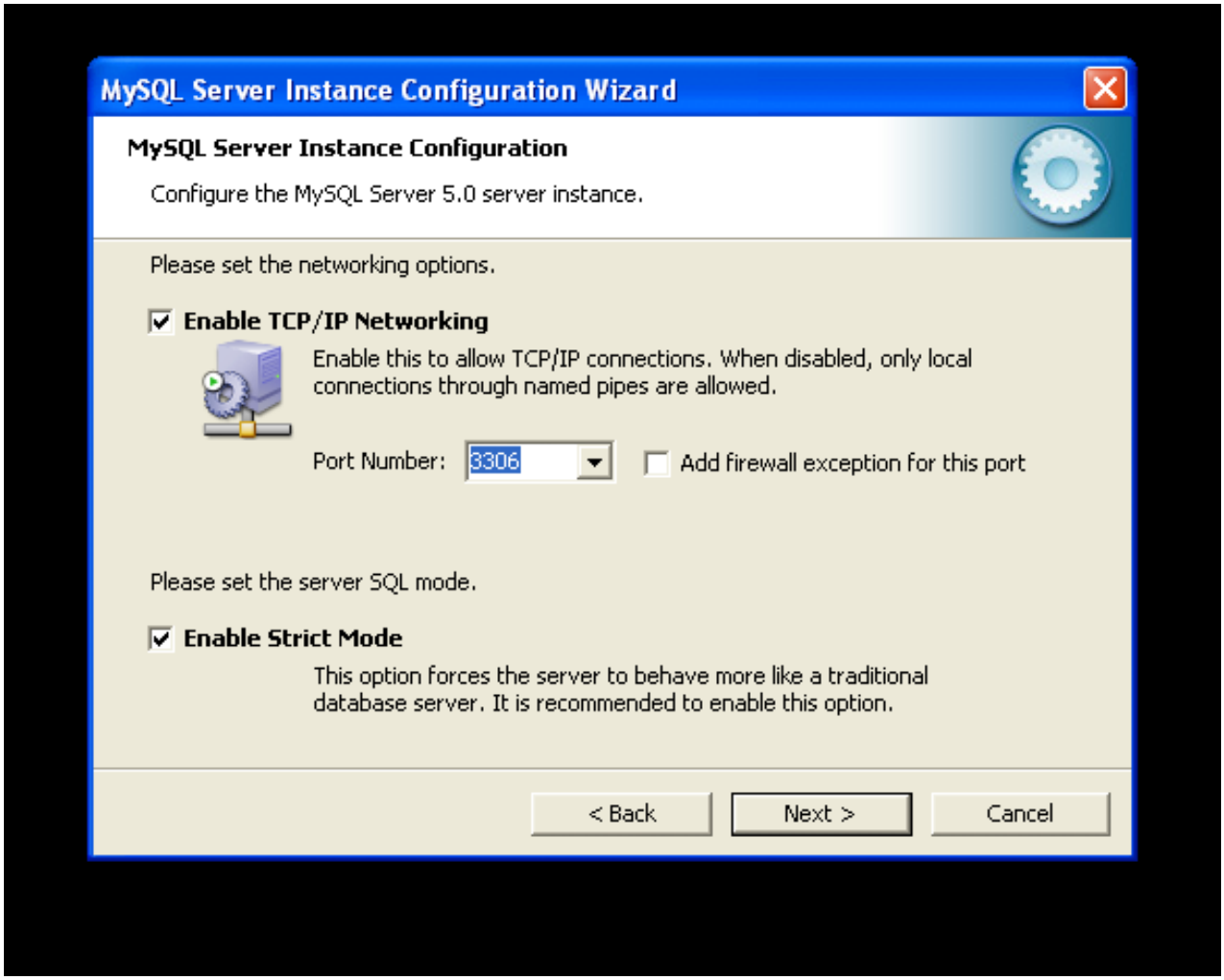
To prevent the server from running out of resources, it is important to limit the number of concurrent connections to the MySQL server that can be established. The CONCURRENT CONNECTIONS dialog allows you to choose the expected usage of your server, and sets the limit for concurrent connections accordingly. It is also possible to set the concurrent connection limit manually.



- Decision Support (DSS)/OLAP: Choose this option if your server does not require a large number of concurrent connections. The maximum number of connections is set at 100, with an average of 20 concurrent connections assumed.
- Online Transaction Processing (OLTP): Choose this option if your server requires a large number of concurrent connections. The maximum number of connections is set at 500.
- Manual Setting: Choose this option to set the maximum number of concurrent connections to the server manually. Choose the number of concurrent connections from the drop-down box provided, or enter the maximum number of connections into the drop-down box if the number you desire is not listed.

1.4.8. The Networking and Strict Mode Options Dialog

Use the NETWORKING OPTIONS dialog to enable or disable TCP/IP networking and to configure the port number that is used to connect to the MySQL server.



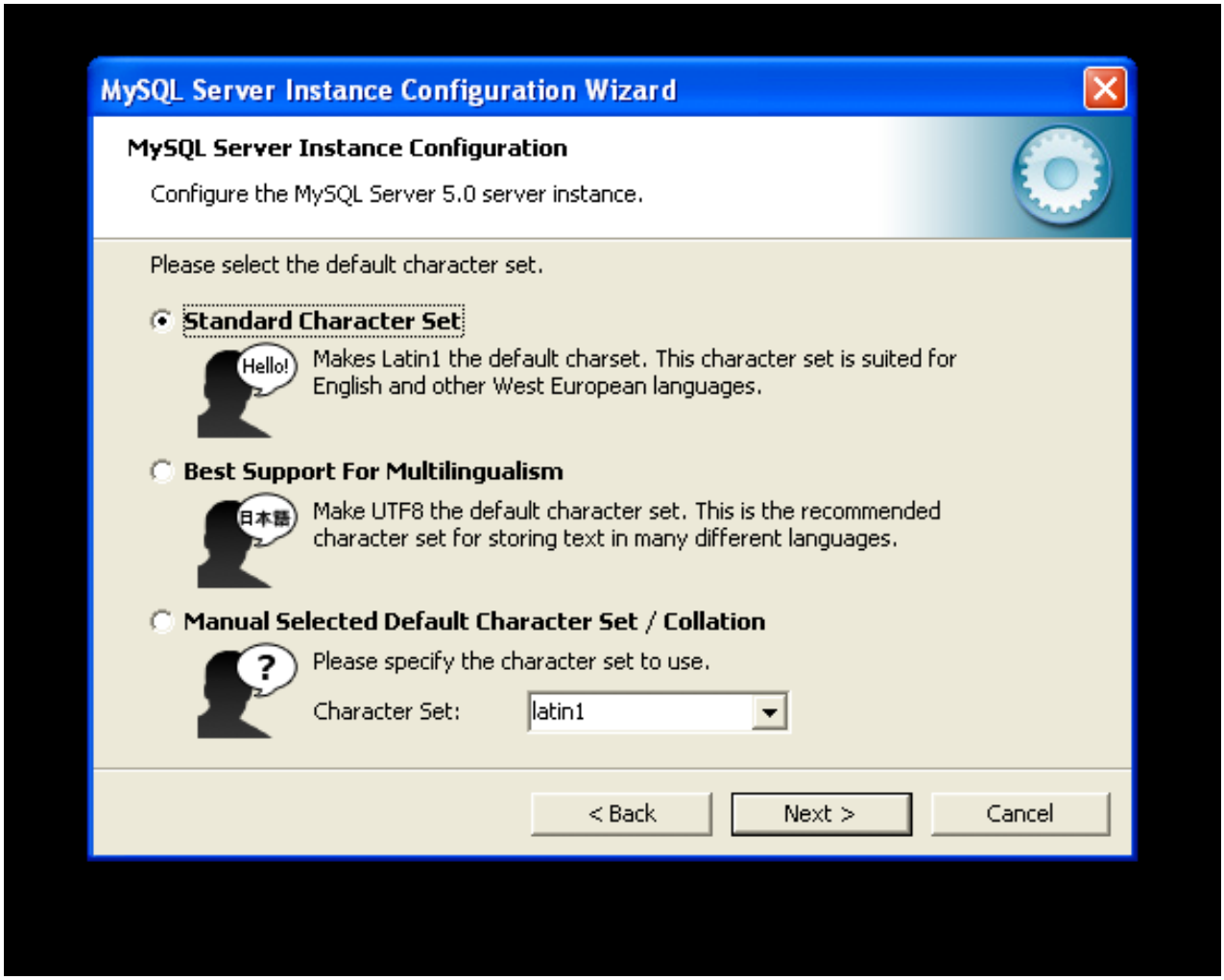
TCP/IP networking is enabled by default. To disable TCP/IP networking, uncheck the box next to the Enable TCP/IP Networking option.

Port 3306 is used by default. To change the port used to access MySQL, choose a new port number from the drop-down box or type a new port number directly into the drop-down box. If the port number you choose is in use, you are prompted to confirm your choice of port number.

Set the SERVER SQL MODE to either enable or disable strict mode. Enabling strict mode (default) makes MySQL behave more like other database management systems. *If you run applications that rely on MySQL's old "forgiving" behavior, make sure to either adapt those applications or to disable strict mode.* For more information about strict mode, see Server SQL Modes.

1.4.9. The Character Set Dialog

The MySQL server supports multiple character sets and it is possible to set a default server character set that is applied to all tables, columns, and databases unless overridden. Use the CHARACTER SET dialog to change the default character set of the MySQL server.



- Standard Character Set: Choose this option if you want to use `latin1` as the default server character set. `latin1` is used for English and many Western European languages.
- Best Support For Multilingualism: Choose this option if you want to use `utf8` as the default server character set. This is a Unicode character set that can store characters from many different languages.
- Manual Selected Default Character Set / Collation: Choose this option if you want to pick the server's default character set manually. Choose the desired character set from the provided drop-down list.

1.4.10. The Service Options Dialog

On Windows platforms, the MySQL server can be installed as a Windows service. When installed this way, the MySQL server can be started automatically during system startup, and even restarted automatically by Windows in the event of a service failure.

The MySQL Server Instance Configuration Wizard installs the MySQL server as a service by default, using the service name `MySQL`. If you do not wish to install the service, uncheck the box next to the Install As Windows Service option. You can change the service name by picking a new service name from the drop-down box provided or by entering a new service name into the drop-down box.

> **Note**
>
> Service names can include any legal character except forward (`/`) or backward (`\`) slashes, and must be less than 256 characters long.

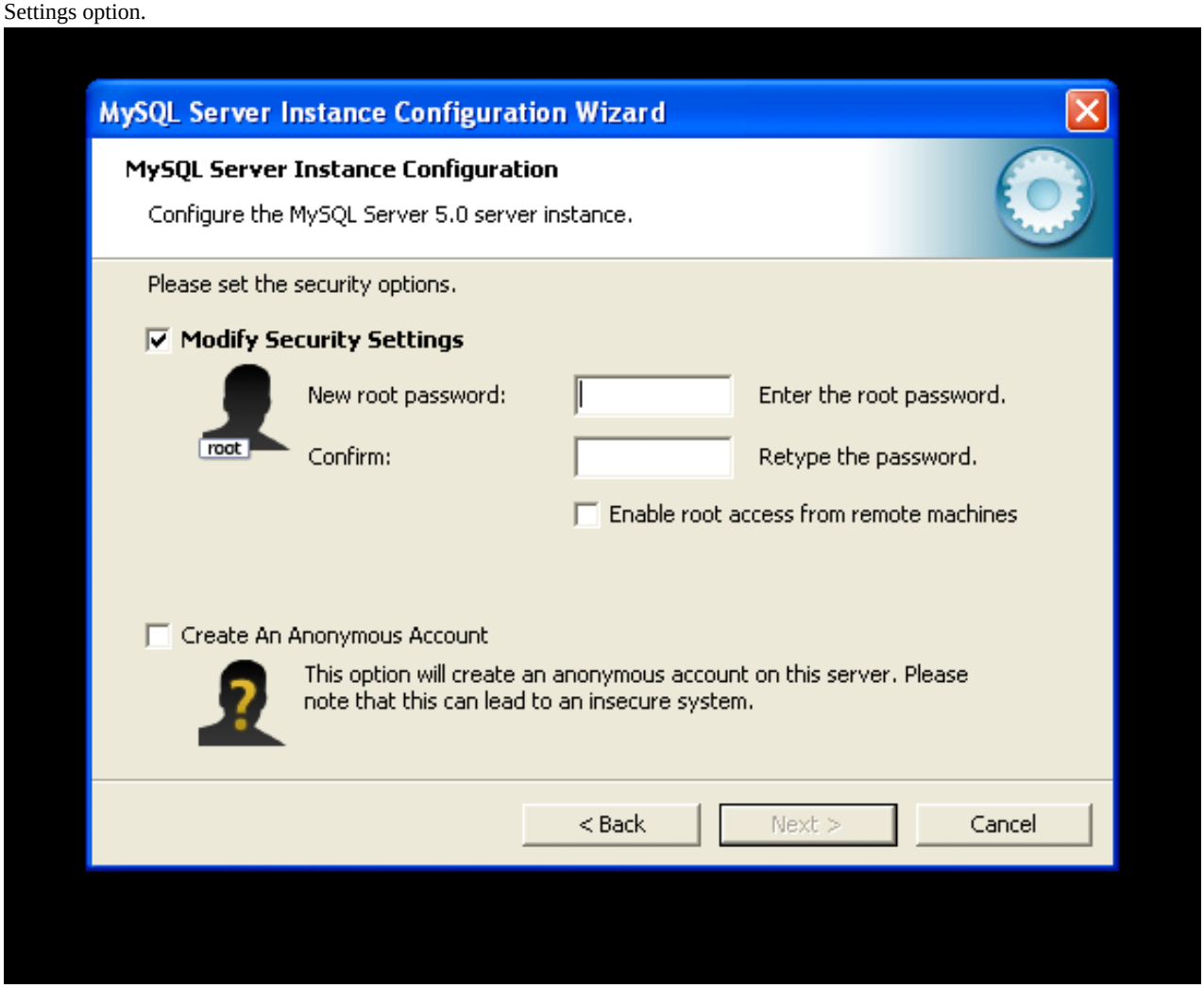
> **Warning**
>
> If you are installing multiple versions of MySQL onto the same machine, you *must* choose a different service name for each version that you install. If you do not choose a different service for each installed version then the service manager information will be inconsistent and this will cause problems when you try to uninstall a previous version.

> If you have already installed multiple versions using the same service name, you must manually edit the contents of the `HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services` parameters within the Windows registry to update the association of the service name with the correct server version.
>
> Typically, when installing multiple versions you create a service name based on the version information. For example, you might install MySQL 5.x as `mysql5`, or specific versions such as MySQL 5.0.56 as `mysql50056`.

To install the MySQL server as a service but not have it started automatically at startup, uncheck the box next to the Launch the MySQL Server Automatically option.

1.4.11. The Security Options Dialog

It is strongly recommended that you set a `root` password for your MySQL server, and the MySQL Server Instance Configuration Wizard requires by default that you do so. If you do not wish to set a `root` password, uncheck the box next to the Modify Security Settings option.



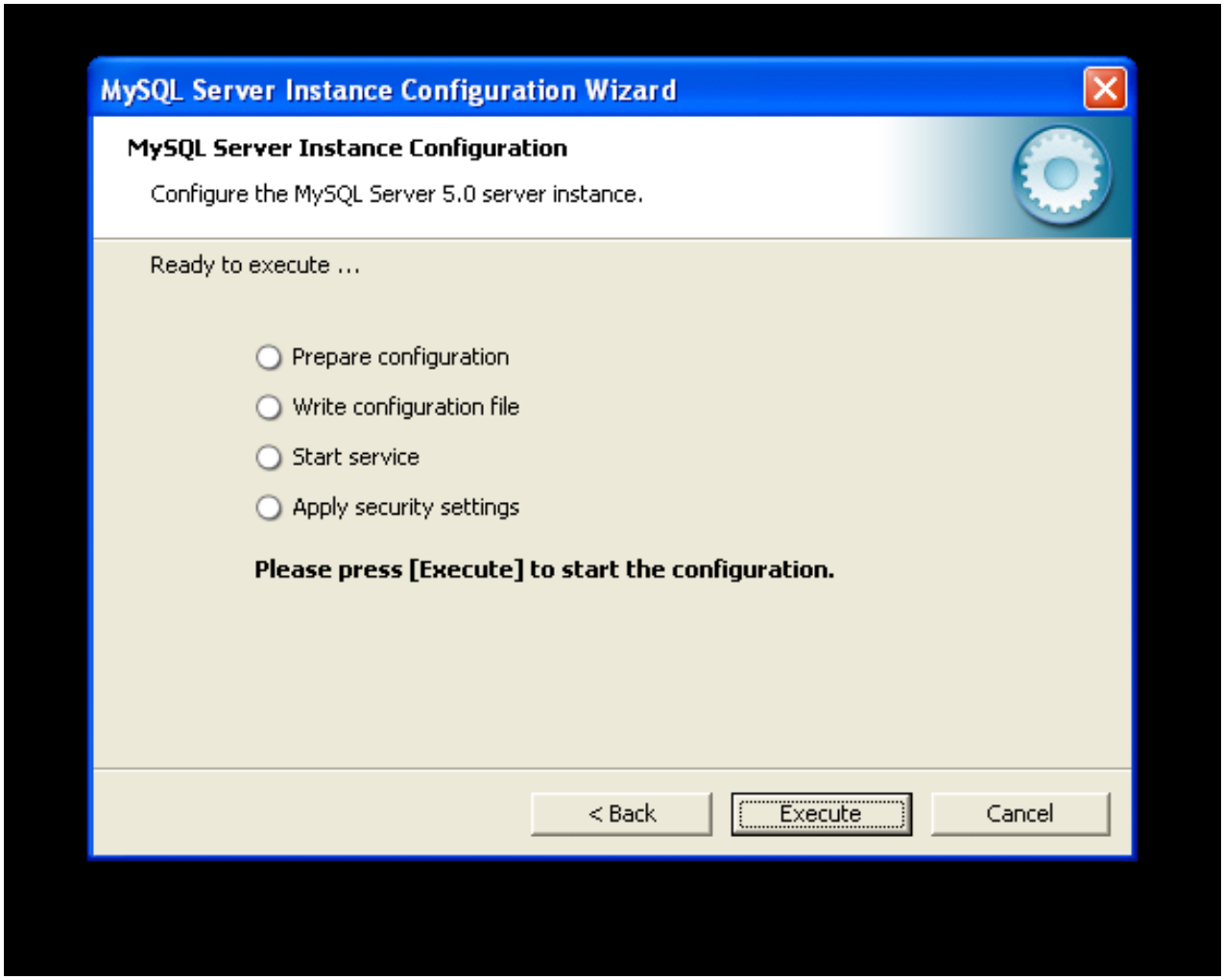
To set the `root` password, enter the desired password into both the New root password and Confirm boxes. If you are reconfiguring an existing server, you need to enter the existing `root` password into the Current root password box.

To prevent `root` logins from across the network, check the box next to the Root may only connect from localhost option. This increases the security of your `root` account.

To create an anonymous user account, check the box next to the Create An Anonymous Account option. Creating an anonymous account can decrease server security and cause login and permission difficulties. For this reason, it is not recommended.

1.4.12. The Confirmation Dialog

The final dialog in the MySQL Server Instance Configuration Wizard is the CONFIRMATION DIALOG. To start the configuration process, click the EXECUTE button. To return to a previous dialog, click the BACK button. To exit the MySQL Server Instance Configuration Wizard without configuring the server, click the CANCEL button.



After you click the EXECUTE button, the MySQL Server Instance Configuration Wizard performs a series of tasks and displays the progress onscreen as the tasks are performed.

The MySQL Server Instance Configuration Wizard first determines configuration file options based on your choices using a template prepared by MySQL developers and engineers. This template is named `my-template.ini` and is located in your server installation directory.

The MySQL Configuration Wizard then writes these options to the corresponding configuration file.

If you chose to create a service for the MySQL server, the MySQL Server Instance Configuration Wizard creates and starts the service. If you are reconfiguring an existing service, the MySQL Server Instance Configuration Wizard restarts the service to apply your configuration changes.

If you chose to set a `root` password, the MySQL Configuration Wizard connects to the server, sets your new `root` password, and applies any other security settings you may have selected.

After the MySQL Server Instance Configuration Wizard has completed its tasks, it displays a summary. Click the FINISH button to exit the MySQL Server Configuration Wizard.

1.5. Installing MySQL from a Noinstall Zip Archive

Users who are installing from the Noinstall package can use the instructions in this section to manually install MySQL. The process for installing MySQL from a Zip archive is as follows:

1. Extract the archive to the desired install directory
2. Create an option file
3. Choose a MySQL server type

4. Start the MySQL server
5. Secure the default user accounts

This process is described in the sections that follow.

1.6. Extracting the Install Archive

To install MySQL manually, do the following:

1. If you are upgrading from a previous version please refer to Section 1.14, “Upgrading MySQL on Windows”, before beginning the upgrade process.
2. Make sure that you are logged in as a user with administrator privileges.
3. Choose an installation location. Traditionally, the MySQL server is installed in `C:\mysql`. The MySQL Installation Wizard installs MySQL under `C:\Program Files\MySQL`. If you do not install MySQL at `C:\mysql`, you must specify the path to the install directory during startup or in an option file. See Section 1.7, “Creating an Option File”.
4. Extract the install archive to the chosen installation location using your preferred Zip archive tool. Some tools may extract the archive to a folder within your chosen installation location. If this occurs, you can move the contents of the subfolder into the chosen installation location.

1.7. Creating an Option File

If you need to specify startup options when you run the server, you can indicate them on the command line or place them in an option file. For options that are used every time the server starts, you may find it most convenient to use an option file to specify your MySQL configuration. This is particularly true under the following circumstances:

- The installation or data directory locations are different from the default locations (`C:\Program Files\MySQL\MySQL Server 5.0` and `C:\Program Files\MySQL\MySQL Server 5.0\data`).
- You need to tune the server settings.

When the MySQL server starts on Windows, it looks for options in two files: the `my.ini` file in the Windows directory, and the `C:\my.cnf` file. The Windows directory typically is named something like `C:\WINDOWS`. You can determine its exact location from the value of the `WINDIR` environment variable using the following command:

```
shell> echo %WINDIR%
```

MySQL looks for options first in the `my.ini` file, and then in the `my.cnf` file. However, to avoid confusion, it is best if you use only one file. If your PC uses a boot loader where `C:` is not the boot drive, your only option is to use the `my.ini` file. Whichever option file you use, it must be a plain text file.

You can also make use of the example option files included with your MySQL distribution; see Preconfigured Option Files.

An option file can be created and modified with any text editor, such as Notepad. For example, if MySQL is installed in `E:\mysql` and the data directory is in `E:\mydata\data`, you can create an option file containing a `[mysqld]` section to specify values for the `basedir` and `datadir` options:

```
[mysqld]
# set basedir to your installation path
basedir=E:/mysql
# set datadir to the location of your data directory
datadir=E:/mydata/data
```

Note that Windows path names are specified in option files using (forward) slashes rather than backslashes. If you do use backslashes, you must double them:

```
[mysqld]
# set basedir to your installation path
basedir=E:\\mysql
# set datadir to the location of your data directory
datadir=E:\\mydata\\data
```

On Windows, the MySQL installer places the data directory directly under the directory where you install MySQL. If you would

like to use a data directory in a different location, you should copy the entire contents of the `data` directory to the new location. For example, if MySQL is installed in `C:\Program Files\MySQL\MySQL Server 5.0`, the data directory is by default in `C:\Program Files\MySQL\MySQL Server 5.0\data`. If you want to use `E:\mydata` as the data directory instead, you must do two things:

1. Move the entire `data` directory and all of its contents from `C:\Program Files\MySQL\MySQL Server 5.0\data` to `E:\mydata`.
2. Use a `--datadir` option to specify the new data directory location each time you start the server.

1.8. Selecting a MySQL Server Type

The following table shows the available servers for Windows in MySQL 5.0.

Binary	Description
`mysqld-nt`	Optimized binary with named-pipe support
`mysqld`	Optimized binary without named-pipe support
`mysqld-debug`	Like `mysqld-nt`, but compiled with full debugging and automatic memory allocation checking

All of the preceding binaries are optimized for modern Intel processors, but should work on any Intel i386-class or higher processor.

Each of the servers in a distribution support the same set of storage engines. The `SHOW ENGINES` statement displays which engines a given server supports.

All Windows MySQL 5.0 servers have support for symbolic linking of database directories.

MySQL supports TCP/IP on all Windows platforms. MySQL servers on Windows support named pipes as indicated in the following list. However, the default is to use TCP/IP regardless of platform. (Named pipes are slower than TCP/IP in many Windows configurations.)

Use of named pipes is subject to these conditions:

- Named pipes are enabled only if you start the server with the `--enable-named-pipe` option. It is necessary to use this option explicitly because some users have experienced problems with shutting down the MySQL server when named pipes were used.
- Named-pipe connections are allowed only by the `mysqld-nt` and `mysqld-debug` servers.

> **Note**
>
> Most of the examples in this manual use `mysqld` as the server name. If you choose to use a different server, such as `mysqld-nt`, make the appropriate substitutions in the commands that are shown in the examples.

1.9. Starting the Server for the First Time

This section gives a general overview of starting the MySQL server. The following sections provide more specific information for starting the MySQL server from the command line or as a Windows service.

The information here applies primarily if you installed MySQL using the `Noinstall` version, or if you wish to configure and test MySQL manually rather than with the GUI tools.

The examples in these sections assume that MySQL is installed under the default location of `C:\Program Files\MySQL\MySQL Server 5.0`. Adjust the path names shown in the examples if you have MySQL installed in a different location.

Clients have two options. They can use TCP/IP, or they can use a named pipe if the server supports named-pipe connections.

MySQL for Windows also supports shared-memory connections if the server is started with the `--shared-memory` option. Clients can connect through shared memory by using the `--protocol=MEMORY` option.

For information about which server binary to run, see Section 1.8, “Selecting a MySQL Server Type”.

Testing is best done from a command prompt in a console window (or "DOS window"). In this way you can have the server display status messages in the window where they are easy to see. If something is wrong with your configuration, these messages make it easier for you to identify and fix any problems.

To start the server, enter this command:

```
shell> "C:\Program Files\MySQL\MySQL Server 5.0\bin\mysqld" --console
```

For a server that includes `InnoDB` support, you should see the messages similar to those following as it starts (the path names and sizes may differ):

```
InnoDB: The first specified datafile c:\ibdata\ibdata1 did not exist:
InnoDB: a new database to be created!
InnoDB: Setting file c:\ibdata\ibdata1 size to 209715200
InnoDB: Database physically writes the file full: wait...
InnoDB: Log file c:\iblogs\ib_logfile0 did not exist: new to be created
InnoDB: Setting log file c:\iblogs\ib_logfile0 size to 31457280
InnoDB: Log file c:\iblogs\ib_logfile1 did not exist: new to be created
InnoDB: Setting log file c:\iblogs\ib_logfile1 size to 31457280
InnoDB: Log file c:\iblogs\ib_logfile2 did not exist: new to be created
InnoDB: Setting log file c:\iblogs\ib_logfile2 size to 31457280
InnoDB: Doublewrite buffer not found: creating new
InnoDB: Doublewrite buffer created
InnoDB: creating foreign key constraint system tables
InnoDB: foreign key constraint system tables created
011024 10:58:25  InnoDB: Started
```

When the server finishes its startup sequence, you should see something like this, which indicates that the server is ready to service client connections:

```
mysqld: ready for connections
Version: '5.0.84'  socket: ''  port: 3306
```

The server continues to write to the console any further diagnostic output it produces. You can open a new console window in which to run client programs.

If you omit the `--console` option, the server writes diagnostic output to the error log in the data directory (`C:\Program Files\MySQL\MySQL Server 5.0\data` by default). The error log is the file with the `.err` extension.

> **Note**
>
> The accounts that are listed in the MySQL grant tables initially have no passwords. After starting the server, you should set up passwords for them using the instructions in Post-Installation Setup and Testing.

1.10. Starting MySQL from the Windows Command Line

The MySQL server can be started manually from the command line. This can be done on any version of Windows.

To start the `mysqld` server from the command line, you should start a console window (or "DOS window") and enter this command:

```
shell> "C:\Program Files\MySQL\MySQL Server 5.0\bin\mysqld"
```

The path to `mysqld` may vary depending on the install location of MySQL on your system.

You can stop the MySQL server by executing this command:

```
shell> "C:\Program Files\MySQL\MySQL Server 5.0\bin\mysqladmin" -u root shutdown
```

> **Note**
>
> If the MySQL `root` user account has a password, you need to invoke `mysqladmin` with the `-p` option and supply the password when prompted.

This command invokes the MySQL administrative utility `mysqladmin` to connect to the server and tell it to shut down. The command connects as the MySQL `root` user, which is the default administrative account in the MySQL grant system. Note that users in the MySQL grant system are wholly independent from any login users under Windows.

If `mysqld` doesn't start, check the error log to see whether the server wrote any messages there to indicate the cause of the problem. The error log is located in the `C:\Program Files\MySQL\MySQL Server 5.0\data` directory. It is the file with a suffix of `.err`. You can also try to start the server as `mysqld --console`; in this case, you may get some useful information on the screen that may help solve the problem.

The last option is to start `mysqld` with the `--standalone` and `--debug` options. In this case, `mysqld` writes a log file `C:\mysqld.trace` that should contain the reason why `mysqld` doesn't start. See MySQL Internals: Porting.

Use `mysqld --verbose --help` to display all the options that `mysqld` supports.

1.11. Starting MySQL as a Windows Service

On Windows, the recommended way to run MySQL is to install it as a Windows service, whereby MySQL starts and stops automatically when Windows starts and stops. A MySQL server installed as a service can also be controlled from the command line using `NET` commands, or with the graphical `Services` utility. Generally, to install MySQL as a Windows service you should be logged in using an account that has administrator rights.

The `Services` utility (the Windows `Service Control Manager`) can be found in the Windows Control Panel (under Administrative Tools on Windows 2000, XP, Vista, and Server 2003). To avoid conflicts, it is advisable to close the `Services` utility while performing server installation or removal operations from the command line.

Before installing MySQL as a Windows service, you should first stop the current server if it is running by using the following command:

```
shell> "C:\Program Files\MySQL\MySQL Server 5.0\bin\mysqladmin"
          -u root shutdown
```

> **Note**
>
> If the MySQL `root` user account has a password, you need to invoke `mysqladmin` with the `-p` option and supply the password when prompted.

This command invokes the MySQL administrative utility `mysqladmin` to connect to the server and tell it to shut down. The command connects as the MySQL `root` user, which is the default administrative account in the MySQL grant system. Note that users in the MySQL grant system are wholly independent from any login users under Windows.

Install the server as a service using this command:

```
shell> "C:\Program Files\MySQL\MySQL Server 5.0\bin\mysqld" --install
```

The service-installation command does not start the server. Instructions for that are given later in this section.

To make it easier to invoke MySQL programs, you can add the path name of the MySQL `bin` directory to your Windows system `PATH` environment variable:

- On the Windows desktop, right-click on the My Computer icon, and select Properties.
- Next select the Advanced tab from the SYSTEM PROPERTIES menu that appears, and click the ENVIRONMENT VARIABLES button.
- Under **SYSTEM VARIABLES**, select Path, and then click the EDIT button. The EDIT SYSTEM VARIABLE dialogue should appear.
- Place your cursor at the end of the text appearing in the space marked **VARIABLE VALUE**. (Use the **End** key to ensure that your cursor is positioned at the very end of the text in this space.) Then enter the complete path name of your MySQL `bin` directory (for example, `C:\Program Files\MySQL\MySQL Server 5.0\bin`), Note that there should be a semicolon separating this path from any values present in this field. Dismiss this dialogue, and each dialogue in turn, by clicking OK until all of the dialogues that were opened have been dismissed. You should now be able to invoke any MySQL executable program by typing its name at the DOS prompt from any directory on the system, without having to supply the path. This includes the servers, the `mysql` client, and all MySQL command-line utilities such as `mysqladmin` and `mysqldump`.

  You should not add the MySQL `bin` directory to your Windows `PATH` if you are running multiple MySQL servers on the same machine.

> **Warning**
>
> You must exercise great care when editing your system `PATH` by hand; accidental deletion or modification of any portion of the existing `PATH` value can leave you with a malfunctioning or even unusable system.

The following additional arguments can be used in MySQL 5.0 when installing the service:

- You can specify a service name immediately following the `--install` option. The default service name is `MySQL`.
- If a service name is given, it can be followed by a single option. By convention, this should be

file_name to specify the name of an option file from which the server should read options when it starts.

The use of a single option other than `--defaults-file` is possible but discouraged. `--defaults-file` is more flexible because it enables you to specify multiple startup options for the server by placing them in the named option file. Also, in MySQL 5.0, use of an option different from `--defaults-file` is not supported until 5.0.3.

- As of MySQL 5.0.1, you can also specify a `--local-service` option following the service name. This causes the server to run using the `LocalService` Windows account that has limited system privileges. This account is available only for Windows XP or newer. If both `--defaults-file` and `--local-service` are given following the service name, they can be in any order.

For a MySQL server that is installed as a Windows service, the following rules determine the service name and option files that the server uses:

- If the service-installation command specifies no service name or the default service name (`MySQL`) following the `--install` option, the server uses the a service name of `MySQL` and reads options from the `[mysqld]` group in the standard option files.
- If the service-installation command specifies a service name other than `MySQL` following the `--install` option, the server uses that service name. It reads options from the `[mysqld]` group and the group that has the same name as the service in the standard option files. This allows you to use the `[mysqld]` group for options that should be used by all MySQL services, and an option group with the service name for use by the server installed with that service name.
- If the service-installation command specifies a `--defaults-file` option after the service name, the server reads options only from the `[mysqld]` group of the named file and ignores the standard option files.

As a more complex example, consider the following command:

```
shell> "C:\Program Files\MySQL\MySQL Server 5.0\bin\mysqld"
         --install MySQL --defaults-file=C:\my-opts.cnf
```

Here, the default service name (`MySQL`) is given after the `--install` option. If no `--defaults-file` option had been given, this command would have the effect of causing the server to read the `[mysqld]` group from the standard option files. However, because the `--defaults-file` option is present, the server reads options from the `[mysqld]` option group, and only from the named file.

You can also specify options as Start parameters in the Windows `Services` utility before you start the MySQL service.

Once a MySQL server has been installed as a service, Windows starts the service automatically whenever Windows starts. The service also can be started immediately from the `Services` utility, or by using a `NET START MySQL` command. The `NET` command is not case sensitive.

When run as a service, `mysqld` has no access to a console window, so no messages can be seen there. If `mysqld` does not start, check the error log to see whether the server wrote any messages there to indicate the cause of the problem. The error log is located in the MySQL data directory (for example, `C:\Program Files\MySQL\MySQL Server 5.0\data`). It is the file with a suffix of `.err`.

When a MySQL server has been installed as a service, and the service is running, Windows stops the service automatically when Windows shuts down. The server also can be stopped manually by using the `Services` utility, the `NET STOP MySQL` command, or the `mysqladmin shutdown` command.

You also have the choice of installing the server as a manual service if you do not wish for the service to be started automatically during the boot process. To do this, use the `--install-manual` option rather than the `--install` option:

```
shell> "C:\Program Files\MySQL\MySQL Server 5.0\bin\mysqld" --install-manual
```

To remove a server that is installed as a service, first stop it if it is running by executing `NET STOP MySQL`. Then use the `--remove` option to remove it:

```
shell> "C:\Program Files\MySQL\MySQL Server 5.0\bin\mysqld" --remove
```

If `mysqld` is not running as a service, you can start it from the command line. For instructions, see Section 1.10, “Starting MySQL from the Windows Command Line”.

Please see Section 1.13, “Troubleshooting a MySQL Installation Under Windows”, if you encounter difficulties during installation.

1.12. Testing The MySQL Installation

You can test whether the MySQL server is working by executing any of the following commands:

```
shell> "C:\Program Files\MySQL\MySQL Server 5.0\bin\mysqlshow"
shell> "C:\Program Files\MySQL\MySQL Server 5.0\bin\mysqlshow" -u root mysql
shell> "C:\Program Files\MySQL\MySQL Server 5.0\bin\mysqladmin" version status proc
shell> "C:\Program Files\MySQL\MySQL Server 5.0\bin\mysql" test
```

If `mysqld` is slow to respond to TCP/IP connections from client programs, there is probably a problem with your DNS. In this case, start `mysqld` with the `--skip-name-resolve` option and use only `localhost` and IP numbers in the `Host` column of the MySQL grant tables.

You can force a MySQL client to use a named-pipe connection rather than TCP/IP by specifying the `--pipe` or `--protocol=PIPE` option, or by specifying `.` (period) as the host name. Use the `--socket` option to specify the name of the pipe if you do not want to use the default pipe name.

Note that if you have set a password for the `root` account, deleted the anonymous account, or created a new user account, then you must use the appropriate `-u` and `-p` options with the commands shown above in order to connect with the MySQL Server. See Connecting to the MySQL Server.

For more information about `mysqlshow`, see `mysqlshow`.

1.13. Troubleshooting a MySQL Installation Under Windows

When installing and running MySQL for the first time, you may encounter certain errors that prevent the MySQL server from starting. The purpose of this section is to help you diagnose and correct some of these errors.

Your first resource when troubleshooting server issues is the error log. The MySQL server uses the error log to record information relevant to the error that prevents the server from starting. The error log is located in the data directory specified in your `my.ini` file. The default data directory location is `C:\Program Files\MySQL\MySQL Server 5.0\data`. See The Error Log.

Another source of information regarding possible errors is the console messages displayed when the MySQL service is starting. Use the `NET START MySQL` command from the command line after installing `mysqld` as a service to see any error messages regarding the starting of the MySQL server as a service. See Section 1.11, “Starting MySQL as a Windows Service”.

The following examples show other common error messages you may encounter when installing MySQL and starting the server for the first time:

- If the MySQL server cannot find the `mysql` privileges database or other critical files, you may see these messages:

  ```
  System error 1067 has occurred.
  Fatal error: Can't open privilege tables: Table 'mysql.host' doesn't exist
  ```

  These messages often occur when the MySQL base or data directories are installed in different locations than the default locations (`C:\Program Files\MySQL\MySQL Server 5.0` and `C:\Program Files\MySQL\MySQL Server 5.0\data`, respectively).

  This situation may occur when MySQL is upgraded and installed to a new location, but the configuration file is not updated to reflect the new location. In addition, there may be old and new configuration files that conflict. Be sure to delete or rename any old configuration files when upgrading MySQL.

  If you have installed MySQL to a directory other than `C:\Program Files\MySQL\MySQL Server 5.0`, you need to ensure that the MySQL server is aware of this through the use of a configuration (`my.ini`) file. The `my.ini` file needs to be located in your Windows directory, typically `C:\WINDOWS`. You can determine its exact location from the value of the `WINDIR` environment variable by issuing the following command from the command prompt:

  ```
  shell> echo %WINDIR%
  ```

  An option file can be created and modified with any text editor, such as Notepad. For example, if MySQL is installed in `E:\mysql` and the data directory is `D:\MySQLdata`, you can create the option file and set up a `[mysqld]` section to specify values for the `basedir` and `datadir` options:

  ```
  [mysqld]
  # set basedir to your installation path
  basedir=E:/mysql
  # set datadir to the location of your data directory
  datadir=D:/MySQLdata
  ```

  Note that Windows path names are specified in option files using (forward) slashes rather than backslashes. If you do use backslashes, you must double them:

```
[mysqld]
# set basedir to your installation path
basedir=C:\\Program Files\\MySQL\\MySQL Server 5.0
# set datadir to the location of your data directory
datadir=D:\\MySQLdata
```

> **MySQL Enterprise**
> For expert advice on the start-up options appropriate to your circumstances, subscribe to the MySQL Enterprise Monitor. For more information, see http://www.mysql.com/products/enterprise/advisors.html.

If you change the `datadir` value in your MySQL configuration file, you must move the contents of the existing MySQL data directory before restarting the MySQL server.

See Section 1.7, “Creating an Option File”.

- If you reinstall or upgrade MySQL without first stopping and removing the existing MySQL service and install MySQL using the MySQL Configuration Wizard, you may see this error:

```
Error: Cannot create Windows service for MySql. Error: 0
```

This occurs when the Configuration Wizard tries to install the service and finds an existing service with the same name.

One solution to this problem is to choose a service name other than `mysql` when using the configuration wizard. This allows the new service to be installed correctly, but leaves the outdated service in place. Although this is harmless, it is best to remove old services that are no longer in use.

To permanently remove the old `mysql` service, execute the following command as a user with administrative privileges, on the command-line:

```
shell> sc delete mysql
[SC] DeleteService SUCCESS
```

If the `sc` utility is not available for your version of Windows, download the `delsrv` utility from http://www.microsoft.com/windows2000/techinfo/reskit/tools/existing/delsrv-o.asp and use the `delsrv mysql` syntax.

1.14. Upgrading MySQL on Windows

This section lists some of the steps you should take when upgrading MySQL on Windows.

1. Review Upgrading MySQL, for additional information on upgrading MySQL that is not specific to Windows.
2. You should always back up your current MySQL installation before performing an upgrade. See Database Backups.
3. Download the latest Windows distribution of MySQL from http://dev.mysql.com/downloads/.
4. Before upgrading MySQL, you must stop the server. If the server is installed as a service, stop the service with the following command from the command prompt:

   ```
   shell> NET STOP MySQL
   ```

   If you are not running the MySQL server as a service, use the following command to stop it:

   ```
   shell> "C:\Program Files\MySQL\MySQL Server 5.0\bin\mysqladmin" -u root shutdown
   ```

   > **Note**
   >
   > If the MySQL `root` user account has a password, you need to invoke `mysqladmin` with the `-p` option and supply the password when prompted.

5. When upgrading to MySQL 5.0 from a version previous to 4.1.5, or when upgrading from a version of MySQL installed from a Zip archive to a version of MySQL installed with the MySQL Installation Wizard, you must manually remove the previous installation and MySQL service (if the server is installed as a service).

   To remove the MySQL service, use the following command:

   ```
   shell> C:\mysql\bin\mysqld --remove
   ```

If you do not remove the existing service, the MySQL Installation Wizard may fail to properly install the new MySQL service.

6. If you are using the MySQL Installation Wizard, start the wizard as described in Section 1.3, "Using the MySQL Installation Wizard".

7. If you are installing MySQL from a Zip archive, extract the archive. You may either overwrite your existing MySQL installation (usually located at `C:\mysql`), or install it into a different directory, such as `C:\mysql5`. Overwriting the existing installation is recommended.

8. If you were running MySQL as a Windows service and you had to remove the service earlier in this procedure, reinstall the service. (See Section 1.11, "Starting MySQL as a Windows Service".)

9. Restart the server. For example, use `NET START MySQL` if you run MySQL as a service, or invoke `mysqld` directly otherwise.

10. If you encounter errors, see Section 1.13, "Troubleshooting a MySQL Installation Under Windows".

1.15. MySQL on Windows Compared to MySQL on Unix

MySQL for Windows has proven itself to be very stable. The Windows version of MySQL has the same features as the corresponding Unix version, with the following exceptions:

- **Limited number of ports**

  Windows systems have about 4,000 ports available for client connections, and after a connection on a port closes, it takes two to four minutes before the port can be reused. In situations where clients connect to and disconnect from the server at a high rate, it is possible for all available ports to be used up before closed ports become available again. If this happens, the MySQL server appears to be unresponsive even though it is running. Note that ports may be used by other applications running on the machine as well, in which case the number of ports available to MySQL is lower.

  For more information about this problem, see http://support.microsoft.com/default.aspx?scid=kb;en-us;196271.

- **Concurrent reads**

  MySQL depends on the `pread()` and `pwrite()` system calls to be able to mix `INSERT` and `SELECT`. Currently, we use mutexes to emulate `pread()` and `pwrite()`. We intend to replace the file level interface with a virtual interface in the future so that we can use the `readfile()`/`writefile()` interface to get more speed. The current implementation limits the number of open files that MySQL 5.0 can use to 2,048, which means that you cannot run as many concurrent threads on Windows as on Unix.

- **Blocking read**

  MySQL uses a blocking read for each connection. That has the following implications if named-pipe connections are enabled:

  - A connection is not disconnected automatically after eight hours, as happens with the Unix version of MySQL.
  - If a connection hangs, it is not possible to break it without killing MySQL.
  - `mysqladmin kill` does not work on a sleeping connection.
  - `mysqladmin shutdown` cannot abort as long as there are sleeping connections.

  We plan to fix this problem in the future.

- **`ALTER TABLE`**

  While you are executing an `ALTER TABLE` statement, the table is locked from being used by other threads. This has to do with the fact that on Windows, you can't delete a file that is in use by another thread. In the future, we may find some way to work around this problem.

- **`DROP TABLE`**

  `DROP TABLE` on a table that is in use by a `MERGE` table does not work on Windows because the `MERGE` handler does the table mapping hidden from the upper layer of MySQL. Because Windows does not allow dropping files that are open, you first must flush all `MERGE` tables (with `FLUSH TABLES`) or drop the `MERGE` table before dropping the table.

- **`DATA DIRECTORY` and `INDEX DIRECTORY`**

The `DATA DIRECTORY` and `INDEX DIRECTORY` options for `CREATE TABLE` are ignored on Windows, because Windows doesn't support symbolic links. These options also are ignored on systems that have a non-functional `realpath()` call.

- **`DROP DATABASE`**

  You cannot drop a database that is in use by some thread.

- **Case-insensitive names**

  File names are not case sensitive on Windows, so MySQL database and table names are also not case sensitive on Windows. The only restriction is that database and table names must be specified using the same case throughout a given statement. See Identifier Case Sensitivity.

- **Directory and file names**

  On Windows, MySQL Server supports only directory and file names that are compatible with the current ANSI code pages. For example, the following Japanese directory name will not work in the Western locale (code page 1252):

  ```
  datadir="C:/ç»´å#ºç#¾ç§#å#³äº#ä¸-æ##ç»´å#ºç#¾ç§#"
  ```

  The same limitation applies to directory and file names referred to in SQL statements, such as the data file path name in `LOAD DATA INFILE`.

- **The "`\`" path name separator character**

  Path name components in Windows are separated by the "`\`" character, which is also the escape character in MySQL. If you are using `LOAD DATA INFILE` or `SELECT ... INTO OUTFILE`, use Unix-style file names with "`/`" characters:

  ```
  mysql> LOAD DATA INFILE 'C:/tmp/skr.txt' INTO TABLE skr;
  mysql> SELECT * INTO OUTFILE 'C:/tmp/skr.txt' FROM skr;
  ```

  Alternatively, you must double the "`\`" character:

  ```
  mysql> LOAD DATA INFILE 'C:\\tmp\\skr.txt' INTO TABLE skr;
  mysql> SELECT * INTO OUTFILE 'C:\\tmp\\skr.txt' FROM skr;
  ```

- **Problems with pipes**

  Pipes do not work reliably from the Windows command-line prompt. If the pipe includes the character `^Z` / `CHAR(24)`, Windows thinks that it has encountered end-of-file and aborts the program.

  This is mainly a problem when you try to apply a binary log as follows:

  ```
  shell> mysqlbinlog binary_log_file | mysql --user=root
  ```

  If you have a problem applying the log and suspect that it is because of a `^Z` / `CHAR(24)` character, you can use the following workaround:

  ```
  shell> mysqlbinlog binary_log_file --result-file=/tmp/bin.sql
  shell> mysql --user=root --execute "source /tmp/bin.sql"
  ```

  The latter command also can be used to reliably read in any SQL file that may contain binary data.

- **`Access denied for user` error**

  If MySQL cannot resolve your host name properly, you may get the following error when you attempt to run a MySQL client program to connect to a server running on the same machine:

  ```
  Access denied for user 'some_user'@'unknown'
  to database 'mysql'
  ```

  To fix this problem, you should create a file named `\windows\hosts` containing the following information:

  ```
  127.0.0.1       localhost
  ```

Chapter 2. Connection to MySQL Server Failing on Windows

When you're running a MySQL server on Windows with many TCP/IP connections to it, and you're experiencing that quite often your clients get a `Can't connect to MySQL server` error, the reason might be that Windows doesn't allow for enough ephemeral (short-lived) ports to serve those connections.

By default, Windows allows 5000 ephemeral (short-lived) TCP ports to the user. After any port is closed it will remain in a `TIME_WAIT` status for 120 seconds. This status allows the connection to be reused at a much lower cost than reinitializing a brand new connection. However, the port will not be available again until this time expires.

With a small stack of available TCP ports (5000) and a high number of TCP ports being open and closed over a short period of time along with the `TIME_WAIT` status you have a good chance for running out of ports. There are two ways to address this problem:

- Reduce the number of TCP ports consumed quickly by investigating connection pooling or persistent connections where possible
- Tune some settings in the Windows registry (see below)

IMPORTANT: The following procedure involves modifying the Windows registry. Before you modify the registry, make sure to back it up and make sure that you understand how to restore the registry if a problem occurs. For information about how to back up, restore, and edit the registry, view the following article in the Microsoft Knowledge Base: http://support.microsoft.com/kb/256986/EN-US/.

1. Start Registry Editor (`Regedt32.exe`).
2. Locate the following key in the registry:

   ```
   HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\Tcpip\Parameters
   ```

3. On the `Edit` menu, click `Add Value`, and then add the following registry value:

   ```
   Value Name: MaxUserPort
   Data Type: REG_DWORD
   Value: 65534
   ```

   This sets the number of ephemeral ports available to any user. The valid range is between 5000 and 65534 (decimal). The default value is 0x1388 (5000 decimal).

4. On the `Edit` menu, click `Add Value`, and then add the following registry value:

   ```
   Value Name: TcpTimedWaitDelay
   Data Type: REG_DWORD
   Value: 30
   ```

   This sets the number of seconds to hold a TCP port connection in `TIME_WAIT` state before closing. The valid range is between 0 (zero) and 300 (decimal). The default value is 0x78 (120 decimal).

5. Quit Registry Editor.
6. Reboot the machine.

Note: Undoing the above should be as simple as deleting the registry entries you've created.

Chapter 3. Resetting the Root Password on Windows Systems

Use the following procedure for resetting the password for any MySQL `root` accounts on Windows:

1. Log on to your system as Administrator.
2. Stop the MySQL server if it is running. For a server that is running as a Windows service, go to the Services manager:

   ```
   Start Menu -> Control Panel -> Administrative Tools -> Services
   ```

   Then find the MySQL service in the list, and stop it.

   If your server is not running as a service, you may need to use the Task Manager to force it to stop.
3. Create a text file and place the following statements in it. Replace the password with the password that you want to use.

   ```
   UPDATE mysql.user SET Password=PASSWORD('MyNewPass') WHERE User='root';
   FLUSH PRIVILEGES;
   ```

   The `UPDATE` and `FLUSH` statements each must be written on a single line. The `UPDATE` statement resets the password for all existing `root` accounts, and the `FLUSH` statement tells the server to reload the grant tables into memory.
4. Save the file. For this example, the file will be named `C:\mysql-init.txt`.
5. Open a console window to get to the command prompt:

   ```
   Start Menu -> Run -> cmd
   ```

6. Start the MySQL server with the special `--init-file` option:

   ```
   C:\> C:\mysql\bin\mysqld-nt --init-file=C:\mysql-init.txt
   ```

   If you installed MySQL to a location other than `C:\mysql`, adjust the command accordingly.

   The server executes the contents of the file named by the `--init-file` option at startup, changing each `root` account password.

   You can also add the `--console` option to the command if you want server output to appear in the console window rather than in a log file.

   If you installed MySQL using the MySQL Installation Wizard, you may need to specify a `--defaults-file` option:

   ```
   C:\> "C:\Program Files\MySQL\MySQL Server 5.0\bin\mysqld-nt.exe"
           --defaults-file="C:\Program Files\MySQL\MySQL Server 5.0\my.ini"
           --init-file=C:\mysql-init.txt
   ```

   The appropriate `--defaults-file` setting can be found using the Services Manager:

   ```
   Start Menu -> Control Panel -> Administrative Tools -> Services
   ```

   Find the MySQL service in the list, right-click on it, and choose the `Properties` option. The `Path to executable` field contains the `--defaults-file` setting.
7. After the server has started successfully, delete `C:\mysql-init.txt`.
8. Stop the MySQL server, then restart it in normal mode again. If you run the server as a service, start it from the Windows Services window. If you start the server manually, use whatever command you normally use.

You should now be able to connect to MySQL as `root` using the new password.

Chapter 4. Installing MySQL from Source on Windows

These instructions describe how to build binaries from source for MySQL 5.0 on Windows. Instructions are provided for building binaries from a standard source distribution or from the Bazaar tree that contains the latest development source.

> **Note**
>
> The instructions here are strictly for users who want to test MySQL on Microsoft Windows from the latest source distribution or from the Bazaar tree. For production use, we do not advise using a MySQL server built by yourself from source. Normally, it is best to use precompiled binary distributions of MySQL that are built specifically for optimal performance on Windows by Sun Microsystems, Inc. Instructions for installing binary distributions are available in Chapter 1, *Installing MySQL on Windows*.

To build MySQL on Windows from source, you must satisfy the following system, compiler, and resource requirements:

- Windows 2000, Windows XP, or newer version.

  Windows Vista is supported when using Visual Studio 2005 provided you have installed the following updates:

  - Microsoft Visual Studio 2005 Professional Edition - ENU Service Pack 1 (KB926601)
  - Security Update for Microsoft Visual Studio 2005 Professional Edition - ENU (KB937061)
  - Update for Microsoft Visual Studio 2005 Professional Edition - ENU (KB932232)

- To build from the standard source distribution, you will need CMake, which can be downloaded from http://www.cmake.org. After installing, modify your path to include the `cmake` binary.
- Microsoft Visual C++ 2005 Express Edition, Visual Studio .Net 2003 (7.1), or Visual Studio 2005 (8.0) compiler system.
- If you are using Visual C++ 2005 Express Edition, you must also install an appropriate Platform SDK. More information and links to downloads for various Windows platforms is available from http://www.microsoft.com/downloads/details.aspx?familyid=0baf2b35-c656-4969-ace8-e4c0c0716adb.
- If you are compiling from a Bazaar tree or making changes to the parser, you need `bison` for Windows, which can be downloaded from http://gnuwin32.sourceforge.net/packages/bison.htm. Download the package labeled “Complete package, excluding sources”. After installing the package, modify your path to include the `bison` binary and ensure that this binary is accessible from Visual Studio.
- Cygwin might be necessary if you want to run the test script or package the compiled binaries and support files into a Zip archive. (Cygwin is needed only to test or package the distribution, not to build it.) Cygwin is available from http://cygwin.com.
- 3GB to 5GB of disk space.

The exact system requirements can be found here: http://msdn.microsoft.com/vstudio/Previous/2003/sysreqs/default.aspx and http://msdn.microsoft.com/vstudio/products/sysreqs/default.aspx

There are three solutions available for building from the source code on Windows:

- Build from the standard MySQL source distribution. For this you will need CMake and Visual C++ Express Edition or Visual Studio. Using this method you can select the storage engines that are included in your build. To use this method, see Section 4.1, “Building MySQL from the Standard Source Distribution”.
- Build from the MySQL Windows source distribution. The Windows source distribution includes ready-made Visual Studio solution files that enable support for all storage engines (except `NDB`). To build using using method you only need Visual C++ Express Edition or Visual Studio. To use this method, see Section 4.2, “Building MySQL from a Windows Source Distribution”.
- Build directly from the Bazaar source repository. For this you will need CMake, Visual C++ Express Edition or Visual Studio, and `bison`. For this method you need to create the distribution on a Unix system and then copy the generated files to your Windows build environment. To use this method, see Section 4.5, “Creating a Windows Source Package from the Bazaar Repository”.

If you find something not working as expected, or you have suggestions about ways to improve the current build process on Windows, please send a message to the `win32` mailing list. See MySQL Mailing Lists.

4.1. Building MySQL from the Standard Source Distribution

You can build MySQL on Windows by using a combination of `cmake` and Microsoft Visual Studio .NET 2003 (7.1), Microsoft Visual Studio 2005 (8.0) or Microsoft Visual C++ 2005 Express Edition. You must have the appropriate Microsoft Platform SDK installed.

> **Note**
>
> To compile from the source code using CMake you must use the standard source distribution (for example, `mysql-5.0.45.tar.gz`). You build from the same distribution as used to build MySQL on Unix, Linux and other platforms. Do *not* use the Windows Source distributions as they do not contain the necessary configuration script and other files.

Follow this procedure to build MySQL:

1. If you are installing from a packaged source distribution, create a work directory (for example, `C:\workdir`), and unpack the source distribution there using `WinZip` or another Windows tool that can read `.zip` files. This directory is the work directory in the following instructions.
2. If you are installing from a Bazaar tree, the root directory of that tree is the work directory in the following instructions.
3. Using a command shell, navigate to the work directory and run the following command:

   ```
   C:\workdir>win\configure.js options
   ```

   If you have associated the `.js` file extension with an application such as a text editor, then you may need to use the following command to force `configure.js` to be executed as a script:

   ```
   C:\workdir>cscript win\configure.js options
   ```

   These options are available for `configure.js`:

   - `WITH_INNOBASE_STORAGE_ENGINE`: Enable the `InnoDB` storage engine.
   - `WITH_PARTITION_STORAGE_ENGINE`: Enable user-defined partitioning.
   - `WITH_ARCHIVE_STORAGE_ENGINE`: Enable the `ARCHIVE` storage engine.
   - `WITH_BLACKHOLE_STORAGE_ENGINE`: Enable the `BLACKHOLE` storage engine.
   - `WITH_EXAMPLE_STORAGE_ENGINE`: Enable the `EXAMPLE` storage engine.
   - `WITH_FEDERATED_STORAGE_ENGINE`: Enable the `FEDERATED` storage engine.
   - `MYSQL_SERVER_SUFFIX=suffix`: Server suffix, default none.
   - `COMPILATION_COMMENT=comment`: Server comment, default "Source distribution".
   - `MYSQL_TCP_PORT=port`: Server port, default 3306.
   - `DISABLE_GRANT_OPTIONS`: Disables the the `--bootstrap`, `--skip-grant-tables`, and `--init-file` options for `mysqld`. This option is available as of MySQL 5.0.36.

   For example (type the command on one line):

   ```
   C:\workdir>win\configure.js WITH_INNOBASE_STORAGE_ENGINE »
              WITH_PARTITION_STORAGE_ENGINE MYSQL_SERVER_SUFFIX=-pro
   ```

4. From the work directory, execute the `win\build-vs8.bat` or `win\build-vs71.bat` file, depending on the version of Visual Studio you have installed. The script invokes CMake, which generates the `mysql.sln` solution file you will need to build MySQL using Visual Studio..

   You can also use `win\build-vs8_x64.bat` to build the 64-bit version of MySQL. However, you cannot build the 64-bit version with Visual Studio Express Edition. You must use Visual Studio 2005 (8.0) or higher.

5. From the work directory, open the generated `mysql.sln` file with Visual Studio and select the proper configuration using the CONFIGURATION menu. The menu provides Debug, Release, RelwithDebInfo, MinRelInfo options. Then select SOLUTION > Build to build the solution.

The build process will take some time. Please be patient.

Remember the configuration that you use in this step. It is important later when you run the test script because that script needs to know which configuration you used.

6. You should test you build before installation. See Section 4.4, “Testing a Windows Source Build”.

7. To install, use the instructions in Section 4.3, “Installing MySQL from a Source Build on Windows”.

4.2. Building MySQL from a Windows Source Distribution

The Windows source distribution includes the necessary solution file and the `vcproj` files required to build each component. Using this method you are not able to select the storage engines that are included in your build.

> **Note**
>
> VC++ workspace files for MySQL 4.1 and above are compatible with Microsoft Visual Studio 7.1 and tested by MySQL AB staff before each release.

Follow this procedure to build MySQL:

1. Create a work directory (for example, `C:\workdir`).

2. Unpack the source distribution in the aforementioned directory using `WinZip` or another Windows tool that can read `.zip` files.

3. Start Visual Studio .Net 2003 (7.1).

4. From the FILE menu, select Open Solution....

5. Open the `mysql.sln` solution you find in the work directory.

6. From the BUILD menu, select Configuration Manager....

7. In the ACTIVE SOLUTION CONFIGURATION pop-up menu, select the configuration to use. You likely want to use one of nt (normal server), Max nt (more engines and features), or Debug configuration.

8. From the BUILD menu, select Build Solution.

9. Debug versions of the programs and libraries are placed in the `client_debug` and `lib_debug` directories. Release versions of the programs and libraries are placed in the `client_release` and `lib_release` directories.

10. You should test you build before installation. See Section 4.4, “Testing a Windows Source Build”.

11. To install, use the instructions in Section 4.3, “Installing MySQL from a Source Build on Windows”.

4.3. Installing MySQL from a Source Build on Windows

When you are satisfied that the program you have built is working correctly, stop the server. Now you can install the distribution. There are two ways to do this, either by using the supplied installation script or by copying the files individually by hand.

To use the script method you must have Cygwin installed as the script is a Shell script. To execute the installation process, run the `make_win_bin_dist` script in the `scripts` directory of the MySQL source distribution (see `make_win_bin_dist`). This is a shell script, so you must have Cygwin installed if you want to use it. It creates a Zip archive of the built executables and support files that you can unpack to your desired installation location.

It is also possible to install MySQL by copying directories and files manually:

1. Create the directories where you want to install MySQL. For example, to install into `C:\mysql`, use these commands:

```
shell> mkdir C:\mysql
shell> mkdir C:\mysql\bin
shell> mkdir C:\mysql\data
shell> mkdir C:\mysql\share
shell> mkdir C:\mysql\scripts
```

If you want to compile other clients and link them to MySQL, you should also create several additional directories:

```
shell> mkdir C:\mysql\include
shell> mkdir C:\mysql\lib
shell> mkdir C:\mysql\lib\debug
shell> mkdir C:\mysql\lib\opt
```

If you want to benchmark MySQL, create this directory:

```
shell> mkdir C:\mysql\sql-bench
```

Benchmarking requires Perl support. See Perl Installation Notes.

2. From the work directory, copy into the `C:\mysql` directory the following directories:

```
shell> cd \workdir
C:\workdir> copy client_release\*.exe C:\mysql\bin
C:\workdir> copy client_debug\mysqld.exe C:\mysql\bin\mysqld-debug.exe
C:\workdir> xcopy scripts\*.* C:\mysql\scripts /E
C:\workdir> xcopy share\*.* C:\mysql\share /E
```

If you want to compile other clients and link them to MySQL, you should also copy several libraries and header files:

```
C:\workdir> copy lib_debug\mysqlclient.lib C:\mysql\lib\debug
C:\workdir> copy lib_debug\libmysql.* C:\mysql\lib\debug
C:\workdir> copy lib_debug\zlib.* C:\mysql\lib\debug
C:\workdir> copy lib_release\mysqlclient.lib C:\mysql\lib\opt
C:\workdir> copy lib_release\libmysql.* C:\mysql\lib\opt
C:\workdir> copy lib_release\zlib.* C:\mysql\lib\opt
C:\workdir> copy include\*.h C:\mysql\include
C:\workdir> copy libmysql\libmysql.def C:\mysql\include
```

If you want to benchmark MySQL, you should also do this:

```
C:\workdir> xcopy sql-bench\*.* C:\mysql\bench /E
```

After installation, set up and start the server in the same way as for binary Windows distributions. See Chapter 1, *Installing MySQL on Windows*.

4.4. Testing a Windows Source Build

You should test the server that you have built from source before using the distribution.

To test the server you need to run the built `mysqld`. By default, using the source build examples, the MySQL base directory and data directory are `C:\mysql` and `C:\mysql\data`. If you want to test your server using the source tree root directory and its data directory as the base directory and data directory, you need to tell the server their path names. You can either do this on the command line with the `--basedir` and `--datadir` options, or by placing appropriate options in an option file. (See Using Option Files.) If you have an existing data directory elsewhere that you want to use, you can specify its path name instead.

When the server is running in standalone fashion or as a service based on your configuration, try to connect to it from the `mysql` interactive command-line utility.

You can also run the standard test script, `mysql-test-run.pl`. This script is written in Perl, so you'll need either Cygwin or ActiveState Perl to run it. You may also need to install the modules required by the script. To run the test script, change location into the `mysql-test` directory under the work directory, set the `MTR_VS_CONFIG` environment variable to the configuration you selected earlier (or use the `--vs-config` option), and invoke `mysql-test-run.pl`. For example (using Cygwin and the `bash` shell):

```
shell> cd mysql-test
shell> export MTS_VS_CONFIG=debug
shell> ./mysql-test-run.pl --force --timer
shell> ./mysql-test-run.pl --force --timer --ps-protocol
```

4.5. Creating a Windows Source Package from the Bazaar Repository

To create a Windows source package from the current Bazaar source tree, use the instructions here. This procedure must be performed on a system running a Unix or Unix-like operating system because some of the configuration and build steps require tools

that work only on Unix. For example, the following procedure is known to work well on Linux.

1. Copy the Bazaar source tree for MySQL 5.0. For instructions on how to do this, see Installing from the Development Source Tree.

2. Configure and build the distribution so that you have a server binary to work with. One way to do this is to run the following command in the top-level directory of your source tree:

   ```
   shell> ./BUILD/compile-pentium-max
   ```

3. After making sure that the build process completed successfully, run the following utility script from top-level directory of your source tree:

   ```
   shell> ./scripts/make_win_src_distribution
   ```

   This script creates a Windows source package to be used on your Windows system. You can supply different options to the script based on your needs. See `make_win_src_distribution`, for a list of allowable options.

   By default, `make_win_src_distribution` creates a Zip-format archive with the name `mysql-`*`VERSION`*`-win-src.zip`, where *`VERSION`* represents the version of your MySQL source tree.

4. Copy or upload the Windows source package that you have just created to your Windows machine. To compile it, use the instructions in Section 4.2, “Building MySQL from a Windows Source Distribution”.