

MySQL and Windows

Abstract

This is the MySQL extract for Microsoft Windows from the MySQL 8.3 Reference Manual.

For legal information, see the [Legal Notices](#).

For help with using MySQL, please visit the [MySQL Forums](#), where you can discuss your issues with other MySQL users.

Document generated on: 2024-03-27 (revision: 78201)

Table of Contents

Preface and Legal Notices	v
1 Installing MySQL on Microsoft Windows	1
1.1 Choosing an Installation Package	3
1.2 Configuration: Using MySQL Configurator	5
1.2.1 MySQL Server Configuration with MySQL Configurator	5
1.3 Configuration: Manually	11
1.3.1 Extracting the Install Archive	11
1.3.2 Creating an Option File	11
1.3.3 Selecting a MySQL Server Type	13
1.3.4 Initializing the Data Directory	13
1.3.5 Starting the Server for the First Time	13
1.3.6 Starting MySQL from the Windows Command Line	14
1.3.7 Customizing the PATH for MySQL Tools	15
1.3.8 Starting MySQL as a Windows Service	16
1.3.9 Testing The MySQL Installation	19
1.4 Troubleshooting a Microsoft Windows MySQL Server Installation	19
1.5 Windows Postinstallation Procedures	21
1.6 Windows Platform Restrictions	23
2 Upgrading MySQL on Windows	25
3 Connection to MySQL Server Failing on Windows	27
4 Resetting the Root Password: Windows Systems	29

Preface and Legal Notices

This is the MySQL extract for Microsoft Windows from the MySQL 8.3 Reference Manual.

Licensing information—MySQL 8.1. This product may include third-party software, used under license. If you are using a *Commercial* release of MySQL 8.1, see the [MySQL 8.1 Commercial Release License Information User Manual](#) for licensing information, including licensing information relating to third-party software that may be included in this Commercial release. If you are using a *Community* release of MySQL 8.1, see the [MySQL 8.1 Community Release License Information User Manual](#) for licensing information, including licensing information relating to third-party software that may be included in this Community release.

Licensing information—MySQL NDB Cluster 8.1. This product may include third-party software, used under license. If you are using a *Commercial* release of MySQL NDB Cluster 8.1, see the [MySQL NDB Cluster 8.1 Commercial Release License Information User Manual](#) for licensing information, including licensing information relating to third-party software that may be included in this Commercial release. If you are using a *Community* release of MySQL NDB Cluster 8.1, see the [MySQL NDB Cluster 8.1 Community Release License Information User Manual](#) for licensing information, including licensing information relating to third-party software that may be included in this Community release.

Legal Notices

Copyright © 1997, 2024, Oracle and/or its affiliates.

License Restrictions

This software and related documentation are provided under a license agreement containing restrictions on use and disclosure and are protected by intellectual property laws. Except as expressly permitted in your license agreement or allowed by law, you may not use, copy, reproduce, translate, broadcast, modify, license, transmit, distribute, exhibit, perform, publish, or display any part, in any form, or by any means. Reverse engineering, disassembly, or decompilation of this software, unless required by law for interoperability, is prohibited.

Warranty Disclaimer

The information contained herein is subject to change without notice and is not warranted to be error-free. If you find any errors, please report them to us in writing.

Restricted Rights Notice

If this is software, software documentation, data (as defined in the Federal Acquisition Regulation), or related documentation that is delivered to the U.S. Government or anyone licensing it on behalf of the U.S. Government, then the following notice is applicable:

U.S. GOVERNMENT END USERS: Oracle programs (including any operating system, integrated software, any programs embedded, installed, or activated on delivered hardware, and modifications of such programs) and Oracle computer documentation or other Oracle data delivered to or accessed by U.S. Government end users are "commercial computer software," "commercial computer software documentation," or "limited rights data" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, the use, reproduction, duplication, release, display, disclosure, modification, preparation of derivative works, and/or adaptation of i) Oracle programs (including any operating system, integrated software, any programs embedded, installed, or activated on delivered hardware, and modifications of such programs), ii) Oracle computer documentation and/or iii) other Oracle data, is subject to the rights and limitations specified in the license contained in the applicable contract.

The terms governing the U.S. Government's use of Oracle cloud services are defined by the applicable contract for such services. No other rights are granted to the U.S. Government.

Hazardous Applications Notice

This software or hardware is developed for general use in a variety of information management applications. It is not developed or intended for use in any inherently dangerous applications, including applications that may create a risk of personal injury. If you use this software or hardware in dangerous applications, then you shall be responsible to take all appropriate fail-safe, backup, redundancy, and other measures to ensure its safe use. Oracle Corporation and its affiliates disclaim any liability for any damages caused by use of this software or hardware in dangerous applications.

Trademark Notice

Oracle, Java, MySQL, and NetSuite are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

Intel and Intel Inside are trademarks or registered trademarks of Intel Corporation. All SPARC trademarks are used under license and are trademarks or registered trademarks of SPARC International, Inc. AMD, Epyc, and the AMD logo are trademarks or registered trademarks of Advanced Micro Devices. UNIX is a registered trademark of The Open Group.

Third-Party Content, Products, and Services Disclaimer

This software or hardware and documentation may provide access to or information about content, products, and services from third parties. Oracle Corporation and its affiliates are not responsible for and expressly disclaim all warranties of any kind with respect to third-party content, products, and services unless otherwise set forth in an applicable agreement between you and Oracle. Oracle Corporation and its affiliates will not be responsible for any loss, costs, or damages incurred due to your access to or use of third-party content, products, or services, except as set forth in an applicable agreement between you and Oracle.

Use of This Documentation

This documentation is NOT distributed under a GPL license. Use of this documentation is subject to the following terms:

You may create a printed copy of this documentation solely for your own personal use. Conversion to other formats is allowed as long as the actual content is not altered or edited in any way. You shall not publish or distribute this documentation in any form or on any media, except if you distribute the documentation in a manner similar to how Oracle disseminates it (that is, electronically for download on a Web site with the software) or on a CD-ROM or similar medium, provided however that the documentation is disseminated together with the software on the same medium. Any other use, such as any dissemination of printed copies or use of this documentation, in whole or in part, in another publication, requires the prior written consent from an authorized representative of Oracle. Oracle and/or its affiliates reserve any and all rights to this documentation not expressly granted above.

Access to Oracle Support for Accessibility

Oracle customers that have purchased support have access to electronic support through My Oracle Support. For information, visit <http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info> or visit <http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs> if you are hearing impaired.

Chapter 1 Installing MySQL on Microsoft Windows

Table of Contents

1.1	Choosing an Installation Package	3
1.2	Configuration: Using MySQL Configurator	5
1.2.1	MySQL Server Configuration with MySQL Configurator	5
1.3	Configuration: Manually	11
1.3.1	Extracting the Install Archive	11
1.3.2	Creating an Option File	11
1.3.3	Selecting a MySQL Server Type	13
1.3.4	Initializing the Data Directory	13
1.3.5	Starting the Server for the First Time	13
1.3.6	Starting MySQL from the Windows Command Line	14
1.3.7	Customizing the PATH for MySQL Tools	15
1.3.8	Starting MySQL as a Windows Service	16
1.3.9	Testing The MySQL Installation	19
1.4	Troubleshooting a Microsoft Windows MySQL Server Installation	19
1.5	Windows Postinstallation Procedures	21
1.6	Windows Platform Restrictions	23

MySQL is available for Microsoft Windows 64-bit operating systems only. For supported Windows platform information, see <https://www.mysql.com/support/supportedplatforms/database.html>.

There are different methods to install MySQL on Microsoft Windows: the MSI, the standard binary distribution (packaged as a compressed file) containing all of the necessary files that you unpack, and source files to compile MySQL yourself. For related information, see [Section 1.1, “Choosing an Installation Package”](#).

Note

MySQL 8.3 Server requires the Microsoft Visual C++ 2019 Redistributable Package to run on Windows platforms. Users should make sure the package has been installed on the system before installing the server. The package is available at the [Microsoft Download Center](#). Additionally, MySQL debug binaries require Visual Studio 2019.

Recommended MSI Installation Method

The simplest and recommended method is to download the MSI and let it install MySQL Server, and then use the MySQL Configurator it installs to configure MySQL:

1. Download the MSI from <https://dev.mysql.com/downloads/> and execute it. This installs the MySQL server, an associated MySQL Configurator application, and it adds related MySQL items to the Microsoft Windows Start menu under the [MySQL](#) group.
2. Upon completion, the installation wizard prompts to execute [MySQL Configurator](#). Execute it now (recommended) or later, or instead choose to manually configure MySQL.

Note

The MySQL server won't start until it's configured; it's recommended to execute the bundled MySQL Configurator immediately after the MSI.

MySQL is now installed. If you used MySQL Configurator to configure MySQL as a Windows service, then Windows automatically starts the MySQL server every time you restart the system. Also, the MSI installs the MySQL Configurator application on the local host, which you can use later to reconfigure MySQL server. It and other MySQL start up menu items were added by the MSI.

MySQL Installation Layout on Microsoft Windows

For MySQL 8.3 on Windows, the default installation directory is `C:\Program Files\MySQL\MySQL Server 8.3` for installations using the MSI, although the MSI **Custom** setup type allows using a different location. If you use the ZIP archive method to install MySQL, install it there are elsewhere, such as `C:\mysql`. Regardless, the layout of the subdirectories remains the same.

All of the files are located within this parent directory using the structure shown in the following table.

Table 1.1 Default MySQL Installation Layout for Microsoft Windows

Directory	Contents of Directory	Notes
<code>bin</code>	<code>mysqld</code> server, client, and utility programs	
<code>%PROGRAMDATA%\MySQL\MySQL Server 8.3\</code>	Log files, databases	The Windows system variable <code>%PROGRAMDATA%</code> defaults to <code>C:\ProgramData</code> .
<code>docs</code>	Release documentation	With the MSI, use the <code>Custom</code> type to include this optional component.
<code>include</code>	Include (header) files	
<code>lib</code>	Libraries	
<code>share</code>	Miscellaneous support files, including error messages, character set files, sample configuration files, SQL for database installation	

Additional Installation Information

By default, MySQL Configurator sets up the MySQL server as a Windows service. By using a service, you can monitor and control the operation of the server through the standard Windows service management tools. For related information about manually setting up the Windows service, see [Section 1.3.8, “Starting MySQL as a Windows Service”](#).

To accommodate the `RESTART` statement, the MySQL server forks when run as a service or standalone, to enable a monitor process to supervise the server process. In this case, there are two `mysqld` processes. If `RESTART` capability is not required, the server can be started with the `--no-monitor` option. See [RESTART Statement](#).

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](#). When installed, MySQL does not need to be executed using a user with Administrator privileges.

For a list of limitations on the use of MySQL on the Windows platform, see [Section 1.6, “Windows Platform Restrictions”](#).

In addition to the MySQL Server package, you may need or want additional components to use MySQL with your application or development environment. These include, but are not limited to:

- To connect to the MySQL server using ODBC, you must have a Connector/ODBC driver. For more information, including installation and configuration instructions, see [MySQL Connector/ODBC Developer Guide](#).
- To use MySQL server with .NET applications, you must have the Connector/.NET driver. For more information, including installation and configuration instructions, see [MySQL Connector/.NET Developer Guide](#).

MySQL distributions for Windows can be downloaded from <https://dev.mysql.com/downloads/>. See [How to Get MySQL](#).

MySQL for Windows is available in several distribution formats, detailed here. Generally speaking, you should use the MSI to install MySQL server and MySQL Configurator to configure it. The MSI is simpler to use than the compressed file, and you need no additional tools to get MySQL up and running. MySQL Configurator automatically configures MySQL Server, creates an options file, starts the server, enables you to create default user accounts, and more. For more information on choosing a package, see [Section 1.1, “Choosing an Installation Package”](#).

Note

Before MySQL 8.1, an application named MySQL Installer included functionality now present in MySQL Configurator. The MySQL Installer both installed and configured MySQL products but it does not exist as of MySQL 8.1.

MySQL on Windows Considerations

• Large Table Support

If you need tables with a size larger than 4GB, install MySQL on an NTFS or newer file system. Do not forget to use `MAX_ROWS` and `AVG_ROW_LENGTH` when you create tables. See [CREATE TABLE Statement](#).

• MySQL and Virus Checking Software

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 due to 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`) used to store your MySQL table data. There is usually a system built into the virus-scanning software to enable specific directories to be ignored.

In addition, by default, MySQL creates temporary files in the standard Windows temporary directory. To prevent the temporary files also being scanned, configure a separate temporary directory for MySQL temporary files and add this directory 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 1.3.2, “Creating an Option File”](#).

1.1 Choosing an Installation Package

For MySQL 8.3, there are multiple installation package formats to choose from when installing MySQL on Windows. The package formats described in this section are:

- [MySQL Installation MSI](#)
- [MySQL noinstall ZIP Archives](#)
- [MySQL Docker Images](#)

MySQL Installation MSI

This package has a file name similar to `mysql-community-8.3.0.msi` or `mysql-commercial-8.3.0.msi`, and installs MySQL server along with MySQL Configurator. The MSI includes a MySQL Configurator application that is recommended for most users to set up, configure, and reconfigure the MySQL server.

The MSI and MySQL Configurator operate on all MySQL supported versions of Windows (see <https://www.mysql.com/support/supportedplatforms/database.html>). For instructions on how to configure MySQL using MySQL Configurator, see [Section 1.2, “Configuration: Using MySQL Configurator”](#).

MySQL noinstall ZIP Archives

These packages contain the files found in the complete MySQL Server installation package, with the exception of the GUI. This format does not include an automated installer, but does include MySQL Configurator to configure the MySQL server.

The `noinstall` ZIP archives are split into two separate compressed files. The main package is named `mysql-VERSION-winx64.zip`. This contains the components needed to use MySQL on your system. The optional MySQL test suite, MySQL benchmark suite, and debugging binaries/information components (including PDB files) are in a separate compressed file named `mysql-VERSION-winx64-debug-test.zip`.

Program Database (PDB) files (with file name extension `pdb`) provide information for debugging your MySQL installation in the event of a problem. These files are included in ZIP Archive distributions (but not MSI distributions) of MySQL.

To install MySQL by extracting the Zip archive rather than use the MSI, consider the following:

1. If you are upgrading from a previous version please refer to [Chapter 2, 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`. 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 1.3.2, “Creating an Option File”](#).

Note

The MSI installs MySQL under `C:\Program Files\MySQL\MySQL Server 8.3\`.

4. Extract the install archive to the chosen installation location using your preferred file-compression 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.
5. Configure the MySQL server using either MySQL Configurator (recommended) or [Section 1.3, “Configuration: Manually”](#).

MySQL Docker Images

For information on using the MySQL Docker images provided by Oracle on Windows platform, see [Deploying MySQL on Windows and Other Non-Linux Platforms with Docker](#).

Warning

The MySQL Docker images provided by Oracle are built specifically for Linux platforms. Other platforms are not supported, and users running the MySQL Docker images from Oracle on them are doing so at their own risk.

1.2 Configuration: Using MySQL Configurator

MySQL Configurator is a standalone application designed to ease the complexity of configuring a MySQL server to run MySQL on Microsoft Windows. It is bundled with the MySQL server, in both the MSI and standalone Zip archive.

Methods to Start MySQL Configurator

MySQL Configurator can both configure and reconfigure MySQL server; and the methods to start MySQL Configurator are:

- The MySQL server MSI prompts to execute MySQL Configurator immediately after it installs the MySQL server.
- From the Start Menu: the MSI creates a MySQL Configurator start menu item.
- From the command line: the `mysql-configurator.exe` executable is located in the same directory as `mysqld.exe` and other MySQL binaries installed with the MySQL server.

Typically this location is in `C:\Program Files\MySQL\MySQL Server X.Y\bin` if installed via the MSI, or a custom directory for the Zip archive.

1.2.1 MySQL Server Configuration with MySQL Configurator

MySQL Configurator performs the initial configuration, a reconfiguration, and also functions as part of the uninstallation process.

Note

Full permissions are granted to the user executing MySQL Configurator to all generated files, such as `my.ini`. This does not apply to files and directories for specific products, such as the MySQL server data directory in `%ProgramData%` that is owned by `SYSTEM`.

MySQL Configurator performs the configuration of the MySQL server. For example:

- It creates the configuration file (`my.ini`) that is used to configure the MySQL server. The values written to this file are influenced by choices you make during the installation process. Some definitions are host dependent.
- By default, a Windows service for the MySQL server is added.
- Provides default installation and data paths for MySQL server.
- It can optionally create MySQL server user accounts with configurable permissions based on general roles, such as DB Administrator, DB Designer, and Backup Admin.

- Checking **Show Advanced Options** enables additional **Logging Options** to be set. This includes defining custom file paths for the error log, general log, slow query log (including the configuration of seconds it requires to execute a query), and the binary log.

The sections that follow describe the server configuration options that apply to MySQL server on Windows. The server version you installed will determine which steps and options you can configure. Configuring MySQL server may include some or all of the steps.

1.2.1.1 MySQL Server Installations

MySQL Configurator adds an upgrade option if it finds an existing MySQL Server installation is discovered. It offers two options:

Note

This upgrade functionality was added in MySQL 8.3.0.

In-Place Upgrade of an Existing MySQL Server Installation

This replaces the existing MySQL server installation as part of the upgrade process which may also upgrade the data schema. Upon success, the existing MySQL server installation is removed from the system.

Note

The existing MySQL server instance must be running for the in-place upgrade option to function.

While MySQL Configurator may attempt (and succeed) to perform an in-place upgrade for other scenarios, the following table lists the scenarios officially supported by the configurator:

Table 1.2 Supported Upgrade Paths

A supported upgrade scenario	Description
8.0.35+ to 8.1	From 8.0.35 or higher to the first MySQL 8 Innovation release.
8.0.35+ to 8.4	From 8.0.35 or higher to the MySQL next LTS release.
8.X to 8.Y where $Y = X + 1$	From an Innovation release to the next consecutive Innovation release.
8.3 to 8.4	From the last MySQL 8 Innovation release to the next MySQL 8 LTS release.
8.4.X to 8.4.Y where $Y > X$	From within the same LTS release.
8.4.X to 9.0.0	From an LTS release to the first consecutive Innovation release.
8.4 to 9.7	From an LTS release to the next consecutive LTS release.

This dialogue prompts for the protocol (default: TCP/IP), port (default: 3306), and root password for the existing installation. Execute connect and then review and confirm the MySQL instance's information (such as version, paths, and configuration file) before proceeding with the upgrade.

This upgrade may replace the file paths. For example, "MySQL Server 8.2\Data\" changes to "MySQL Server 8.3\Data\" when upgrading 8.2 to 8.3.

This upgrade functionality also provides these additional options: "Backup Data" allows running `mysqldump` before performing the upgrade, and "Server File Permissions" to optionally customize file permissions.

Add a Separate MySQL Server Installation

Configure a standard side-by-side installation with the new MySQL server installation. This means having multiple MySQL server installations installed and running on the system.

1.2.1.2 Type and Networking

- Server Configuration Type


Choose the MySQL server configuration type that describes your setup. This setting defines the amount of system resources (memory) to assign to your MySQL server instance.

- **Development:** A computer that hosts many other applications, and typically this is your personal workstation. This setting configures MySQL to use the least amount of memory.
- **Server:** Several other applications are expected to run on this computer, such as a web server. The Server setting configures MySQL to use a medium amount of memory.
- **Dedicated:** A computer that is dedicated to running the MySQL server. Because no other major applications run on this server, this setting configures MySQL to use the majority of available memory.
- **Manual:** Prevents MySQL Configurator from attempting to optimize the server installation, and instead, sets the default values to the server variables included in the `my.ini` configuration file. With the **Manual** type selected, MySQL Configurator uses the default value of 16M for the `tmp_table_size` variable assignment.

- Connectivity

Connectivity options control how the connection to MySQL is made. Options include:

- **TCP/IP:** This option is selected by default. You may disable TCP/IP Networking to permit local host connections only. With the TCP/IP connection option selected, you can modify the following items:
 - **Port** for classic MySQL protocol connections. The default value is `3306`.
 - **X Protocol Port** defaults to `33060`
 - **Open Windows Firewall port for network access**, which is selected by default for TCP/IP connections.

If a port number is in use already, you will see the error icon () next to the default value and **Next** is disabled until you provide a new port number.

- **Named Pipe:** Enable and define the pipe name, similar to setting the `named_pipe` system variable. The default name is `MySQL`.

When you select **Named Pipe** connectivity, and then proceed to the next step, you are prompted to set the level of access control granted to client software on named-pipe connections. Some clients require only minimum access control for communication, while other clients require full access to the named pipe.

You can set the level of access control based on the Windows user (or users) running the client as follows:

- **Minimum access to all users (RECOMMENDED).** This level is enabled by default because it is the most secure.
- **Full access to members of a local group.** If the minimum-access option is too restrictive for the client software, use this option to reduce the number of users who have full access on the named pipe. The group must be established on Windows before you can select it from the list. Membership in this group should be limited and managed. Windows requires a newly added member to first log out and then log in again to join a local group.
- **Full access to all users (NOT RECOMMENDED).** This option is less secure and should be set only when other safeguards are implemented.
- **Shared Memory:** Enable and define the memory name, similar to setting the `shared_memory` system variable. The default name is `MySQL`.
- **Advanced Configuration**

Check **Show Advanced and Logging Options** to set custom logging and advanced options in later steps. The Logging Options step enables you to define custom file paths for the error log, general log, slow query log (including the configuration of seconds it requires to execute a query), and the binary log. The Advanced Options step enables you to set the unique server ID required when binary logging is enabled in a replication topology.


- **MySQL Enterprise Firewall (Enterprise Edition only)**

The **Enable MySQL Enterprise Firewall** check box is deselected by default. Select this option to enable a security list that offers protection against certain types of attacks. Additional post-installation configuration is required (see [MySQL Enterprise Firewall](#)).

1.2.1.3 Accounts and Roles

- **Root Account Password**

Assigning a root password is required and you will be asked for it when reconfiguring with MySQL Configurator in the future. Password strength is evaluated when you repeat the password in the box provided. For descriptive information regarding password requirements or status, move your mouse

pointer over the information icon () when it appears.

- **MySQL User Accounts (Optional)**

Click **Add User** or **Edit User** to create or modify MySQL user accounts with predefined roles. Next, enter the required account credentials:

- **User Name:** MySQL user names can be up to 32 characters long.
- **Host:** Select `localhost` for local connections only or `<All Hosts (%)>` when remote connections to the server are required.
- **Role:** Each predefined role, such as `DB Admin`, is configured with its own set of privileges. For example, the `DB Admin` role has more privileges than the `DB Designer` role. The **Role** drop-down list contains a description of each role.

- **Password:** Password strength assessment is performed while you type the password. Passwords must be confirmed. MySQL permits a blank or empty password (considered to be insecure).

MySQL Configurator Commercial Release Only: MySQL Enterprise Edition for Windows, a commercial product, also supports an authentication method that performs external authentication on Windows. Accounts authenticated by the Windows operating system can access the MySQL server without providing an additional password.

To create a new MySQL account that uses Windows authentication, enter the user name and then select a value for **Host** and **Role**. Click **Windows** authentication to enable the [authentication_windows](#) plugin. In the Windows Security Tokens area, enter a token for each Windows user (or group) who can authenticate with the MySQL user name. MySQL accounts can include security tokens for both local Windows users and Windows users that belong to a domain. Multiple security tokens are separated by the semicolon character (;) and use the following format for local and domain accounts:

- Local account

Enter the simple Windows user name as the security token for each local user or group; for example, [finley;jeffrey;admin](#).

- Domain account

Use standard Windows syntax ([domain\domainuser](#)) or MySQL syntax ([domain\domainuser](#)) to enter Windows domain users and groups.

For domain accounts, you may need to use the credentials of an administrator within the domain if the account running MySQL Configurator lacks the permissions to query the Active Directory. If this is the case, select **Validate Active Directory users with** to activate the domain administrator credentials.

Windows authentication permits you to test all of the security tokens each time you add or modify a token. Click **Test Security Tokens** to validate (or revalidate) each token. Invalid tokens generate a descriptive error message along with a red [x](#) icon and red token text. When all tokens resolve as valid (green text without an [x](#) icon), you can click **OK** to save the changes.

1.2.1.4 Windows Service

On the Windows platform, MySQL server can run as a named service managed by the operating system and be configured to start up automatically when Windows starts. Alternatively, you can configure MySQL server to run as an executable program that requires manual configuration.

- **Configure MySQL server as a Windows service** (Selected by default.)

When the default configuration option is selected, you can also select the following:

- **Windows Service Name**

Defaults to MySQL`XY` where XY is 81 for MySQL 8.1.

- **Start the MySQL Server at System Startup**

When selected (default), the service startup type is set to Automatic; otherwise, the startup type is set to Manual.

- **Run Windows Service as**

When **Standard System Account** is selected (default), the service logs on as Network Service.

The **Custom User** option must have privileges to log on to Microsoft Windows as a service. The **Next** button will be disabled until this user is configured with the required privileges.

A custom user account is configured in Windows by searching for "local security policy" in the Start menu. In the Local Security Policy window, select **Local Policies, User Rights Assignment**, and then **Log On As A Service** to open the property dialog. Click **Add User or Group** to add the custom user and then click **OK** in each dialog to save the changes.

1.2.1.5 Server File Permissions

Optionally, permissions set on the folders and files located at `C:\ProgramData\MySQL\MySQL Server X.Y\Data` can be managed during the server configuration operation. You have the following options:

- MySQL Configurator can configure the folders and files with full control granted exclusively to the user running the Windows service, if applicable, and to the Administrators group.

All other groups and users are denied access. This is the default option.

- Have MySQL Configurator use a configuration option similar to the one just described, but also have MySQL Configurator show which users could have full control.

You are then able to decide if a group or user should be given full control. If not, you can move the qualified members from this list to a second list that restricts all access.

- Have MySQL Configurator skip making file-permission changes during the configuration operation.

If you select this option, you are responsible for securing the `Data` folder and its related files manually after the server configuration finishes.

1.2.1.6 Logging Options

This step is available if the **Show Advanced Configuration** check box was selected during the **Type and Networking** step. To enable this step now, click **Back** to return to the **Type and Networking** step and select the check box.

Advanced configuration options are related to the following MySQL log files:

- [Error Log](#)
- [General Log](#)
- [Slow Query Log](#)
- [Binary Log](#)

1.2.1.7 Advanced Options

This step is available if the **Show Advanced Configuration** check box was selected during the **Type and Networking** step. To enable this step now, click **Back** to return to the **Type and Networking** step and select the check box.

The advanced-configuration options include:

- **Server ID**

Set the unique identifier used in a replication topology. If binary logging is enabled, you must specify a server ID. The default ID value depends on the server version. For more information, see the description of the `server_id` system variable.

- **Table Names Case**

These options only apply to the initial configuration of the MySQL server.

- Lower Case

Sets the `lower_case_table_names` option value to 1 (default), in which table names are stored in lowercase on disk and comparisons are not case-sensitive.

- Preserve Given Case

Sets the `lower_case_table_names` option value to 2, in which table names are stored as given but compared in lowercase.

1.2.1.8 Sample Databases

Optionally install sample databases that include test data to help develop applications with MySQL. The options include the `sakila` and `world` databases.

1.2.1.9 Apply Configuration

All configuration settings are applied to the MySQL server when you click **Execute**. Use the **Configuration Steps** tab to follow the progress of each action; the icon for each toggles from white to green (with a check mark) on success. Otherwise, the process stops and displays an error message if an individual action times out. Click the **Log** tab to view the log.

1.3 Configuration: Manually

These instructions apply to those *not* using the MySQL Configurator application to configure and set up the MySQL server; or for manually making additional changes. Using MySQL Configurator is recommended.

1.3.1 Extracting the Install Archive

To install MySQL manually, do the following:

1. If you are upgrading from a previous version then refer to [Chapter 2, 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`. 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 1.3.2, “Creating an Option File”](#).

Note

The MSI installs MySQL under `C:\Program Files\MySQL\MySQL Server 8.3`.

4. Extract the install archive to the chosen installation location using your preferred file-compression 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.

1.3.2 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 8.3` and `C:\Program Files\MySQL\MySQL Server 8.3\data`).
- You need to tune the server settings, such as memory, cache, or InnoDB configuration information.

When the MySQL server starts on Windows, it looks for option files in several locations, such as the Windows directory, `C:\`, and the MySQL installation directory (for the full list of locations, see [Using Option Files](#)). 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:

```
C:\> echo %WINDIR%
```

MySQL looks for options in each location 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.

Note

When using MySQL Configurator to configure MySQL Server, it creates the `my.ini` at the default location, and the user executing MySQL Configurator is granted full permissions to this new `my.ini` file.

In other words, be sure that the MySQL Server user has permission to read the `my.ini` file.

You can also make use of the example option files included with your MySQL distribution; see [Server Configuration Defaults](#).

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
```

Microsoft Windows path names are specified in option files using (forward) slashes rather than backslashes. If you do use backslashes, 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
```

The rules for use of backslash in option file values are given in [Using Option Files](#).

The ZIP archive does not include a `data` directory. To initialize a MySQL installation by creating the data directory and populating the tables in the `mysql` system database, initialize MySQL using either `--initialize` or `--initialize-insecure`. For additional information, see [Initializing the Data Directory](#).

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 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 the default location (for example `C:\Program Files\MySQL\MySQL Server 8.3\data`) to `E:\mydata`.
2. Use a `--datadir` option to specify the new data directory location each time you start the server.

1.3.3 Selecting a MySQL Server Type

The following table shows the available servers for Windows in MySQL 8.3.

Binary	Description
<code>mysqld</code>	Optimized binary with named-pipe support
<code>mysqld-debug</code>	Like <code>mysqld</code> , but compiled with full debugging and automatic memory allocation checking

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 8.3 servers have support for symbolic linking of database directories.

MySQL supports TCP/IP on all Windows platforms. MySQL servers on Windows also support named pipes, if you start the server with the `named_pipe` system variable enabled. It is necessary to enable this variable explicitly because some users have experienced problems with shutting down the MySQL server when named pipes were used. The default is to use TCP/IP regardless of platform because named pipes are slower than TCP/IP in many Windows configurations.

1.3.4 Initializing the Data Directory

If you installed MySQL using the `noinstall` package, no data directory is included. To initialize the data directory, use the instructions at [Initializing the Data Directory](#).

1.3.5 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 using MySQL Configurator.

The examples in these sections assume that MySQL is installed under the default location of `C:\Program Files\MySQL\MySQL Server 8.3`. 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` system variable enabled. Clients can connect through shared memory by using the `--protocol=MEMORY` option.

For information about which server binary to run, see [Section 1.3.3, “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.

Note

The database must be initialized before MySQL can be started. For additional information about the initialization process, see [Initializing the Data Directory](#).

To start the server, enter this command:

```
C:\> "C:\Program Files\MySQL\MySQL Server 8.3\bin\mysqld" --console
```

You should see messages similar to those following as it starts (the path names and sizes may differ). The `ready for connections` messages indicate that the server is ready to service client connections.

```
[Server] C:\mysql\bin\mysqld.exe (mysqld 8.0.30) starting as process 21236
[InnoDB] InnoDB initialization has started.
[InnoDB] InnoDB initialization has ended.
[Server] CA certificate ca.pem is self signed.
[Server] Channel mysql_main configured to support TLS.
Encrypted connections are now supported for this channel.
[Server] X Plugin ready for connections. Bind-address: '::' port: 33060
[Server] C:\mysql\bin\mysqld.exe: ready for connections.
Version: '8.0.30' socket: '' port: 3306 MySQL Community Server - GPL.
```

You can now 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 8.3\data` by default). The error log is the file with the `.err` extension, and may be set using the `--log-error` option.

Note

The initial `root` account in the MySQL grant tables has no password. After starting the server, you should set up a password for it using the instructions in [Securing the Initial MySQL Account](#).

1.3.6 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:

```
C:\> "C:\Program Files\MySQL\MySQL Server 8.3\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:

```
C:\> "C:\Program Files\MySQL\MySQL Server 8.3\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

Users in the MySQL grant system are wholly independent from any operating system users under Microsoft 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. By default, the error log is located in the `C:\Program Files\MySQL\MySQL Server 8.3\data` directory. It is the file with a suffix of `.err`, or may be specified by passing in the `--log-error` option. Alternatively, you can try to start the server with the `--console` option; in this case, the server may display some useful information on the screen to 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 [The DBUG Package](#).

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

1.3.7 Customizing the PATH for MySQL Tools

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.

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 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 8.3\bin`)

Note

There must 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. The new `PATH` value should now be available to any new command shell you open, allowing you 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.

1.3.8 Starting MySQL as a Windows Service

On Windows, the recommended way to run MySQL is to install it as a Windows service, so that 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. To avoid conflicts, it is advisable to close the `Services` utility while performing server installation or removal operations from the command line.

Installing the service

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

```
C:\> "C:\Program Files\MySQL\MySQL Server 8.3\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

Users in the MySQL grant system are wholly independent from any operating system users under Windows.

Install the server as a service using this command:

```
C:\> "C:\Program Files\MySQL\MySQL Server 8.3\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 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 8.3\bin`), and 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 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 `--defaults-file=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.

- 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. 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 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 enables 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 the same way as described in the previous item, except that it reads options only from the named file and ignores the standard option files.

As a more complex example, consider the following command:

```
C:\> "C:\Program Files\MySQL\MySQL Server 8.3\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.

Note

On Windows, if the server is started with the `--defaults-file` and `--install` options, `--install` must be first. Otherwise, `mysqld.exe` attempts to start the MySQL server.

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

Finally, before trying to start the MySQL service, make sure the user variables `%TEMP%` and `%TMP%` (and also `%TMPDIR%`, if it has ever been set) for the operating system user who is to run the service are pointing to a folder to which the user has write access. The default user for running the MySQL service is `LocalSystem`, and the default value for its `%TEMP%` and `%TMP%` is `C:\Windows\Temp`, a directory `LocalSystem` has write access to by default. However, if there are any changes to that default setup (for example, changes to the user who runs the service or to the mentioned user variables, or the `--tmpdir` option has been used to put the temporary directory somewhere else), the MySQL service might fail to run because write access to the temporary directory has not been granted to the proper user.

Starting the service

After a MySQL server instance 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 an `sc start mysql_d_service_name` or `NET START mysql_d_service_name` command. `SC` and `NET` commands are 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 8.3\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 using the `Services` utility, the `sc stop mysql_d_service_name` command, the `NET STOP mysql_d_service_name` 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:

```
C:\> "C:\Program Files\MySQL\MySQL Server 8.3\bin\mysqld" --install-manual
```

Removing the service

To remove a server that is installed as a service, first stop it if it is running by executing `SC STOP mysql_d_service_name` or `NET STOP mysql_d_service_name`. Then use `SC DELETE mysql_d_service_name` to remove it:

```
C:\> SC DELETE mysql
```

Alternatively, use the `mysqld --remove` option to remove the service.

```
C:\> "C:\Program Files\MySQL\MySQL Server 8.3\bin\mysqld" --remove
```

If `mysqld` is not running as a service, you can start it from the command line. For instructions, see [Section 1.3.6, "Starting MySQL from the Windows Command Line"](#).

If you encounter difficulties during installation, see [Section 1.4, “Troubleshooting a Microsoft Windows MySQL Server Installation”](#).

For more information about stopping or removing a Windows service, see [Starting Multiple MySQL Instances as Windows Services](#).

1.3.9 Testing The MySQL Installation

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

```
C:\> "C:\Program Files\MySQL\MySQL Server 8.3\bin\mysqlshow"  
C:\> "C:\Program Files\MySQL\MySQL Server 8.3\bin\mysqlshow" -u root mysql  
C:\> "C:\Program Files\MySQL\MySQL Server 8.3\bin\mysqladmin" version status proc  
C:\> "C:\Program Files\MySQL\MySQL Server 8.3\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` system variable enabled and use only `localhost` and IP addresses in the `Host` column of the MySQL grant tables. (Be sure that an account exists that specifies an IP address or you may not be able to connect.)

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.

If you have set a password for the `root` account, deleted the anonymous account, or created a new user account, then to connect to the MySQL server you must use the appropriate `-u` and `-p` options with the commands shown previously. See [Connecting to the MySQL Server Using Command Options](#).

For more information about `mysqlshow`, see [mysqlshow — Display Database, Table, and Column Information](#).

1.4 Troubleshooting a Microsoft Windows MySQL Server Installation

When installing and running MySQL for the first time, you may encounter certain errors that prevent the MySQL server from starting. This section helps 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 8.3\data`, or `C:\ProgramData\MySQL` on Windows 7 and Windows Server 2008. The `C:\ProgramData` directory is hidden by default. You need to change your folder options to see the directory and contents. For more information on the error log and understanding the content, see [The Error Log](#).

For information regarding possible errors, also consult the console messages displayed when the MySQL service is starting. Use the `SC START mysqld_service_name` or `NET START mysqld_service_name` 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 1.3.8, “Starting MySQL as a Windows Service”](#).

The following examples show other common error messages you might 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, it displays these messages:

```
System error 1067 has occurred.
```

```
Fatal error: Can't open and lock privilege tables:
Table 'mysql.user' 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 8.3](#) and [C:\Program Files\MySQL\MySQL Server 8.3\data](#), respectively).

This situation can 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, old and new configuration files might 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 8.3](#), ensure that the MySQL server is aware of this through the use of a configuration ([my.ini](#)) file. Put the [my.ini](#) file in your Windows directory, typically [C:\WINDOWS](#). To determine its exact location from the value of the [WINDIR](#) environment variable, issue the following command from the command prompt:

```
C:\> echo %WINDIR%
```

You can create or modify an option file 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
```

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

```
[mysqld]
# set basedir to your installation path
basedir=C:\\Program Files\\MySQL\\MySQL Server 8.3
# set datadir to the location of your data directory
datadir=D:\\MySQLdata
```

The rules for use of backslash in option file values are given in [Using Option Files](#).

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 1.3.2, "Creating an Option File"](#).

- If you reinstall or upgrade MySQL without first stopping and removing the existing MySQL service, and then configure MySQL using MySQL Configurator, you might 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 enables 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:

```
C:\> 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.

1.5 Windows Postinstallation Procedures

GUI tools exist that perform most of the tasks described in this section, including:

- [MySQL Configurator](#): Used to configure the MySQL server.
- [MySQL Workbench](#): Manages the MySQL server and edits SQL statements.

If necessary, initialize the data directory and create the MySQL grant tables. Windows installation operations performed by MySQL Configurator can initialize the data directory automatically. For installation from a ZIP Archive package, initialize the data directory as described at [Initializing the Data Directory](#).

Regarding passwords, if you configured MySQL using the MySQL Configurator, you may have already assigned a password to the initial `root` account. (See [Section 1.2, "Configuration: Using MySQL Configurator"](#).) Otherwise, use the password-assignment procedure given in [Securing the Initial MySQL Account](#).

Before assigning a password, you might want to try running some client programs to make sure that you can connect to the server and that it is operating properly. Make sure that the server is running (see [Section 1.3.5, "Starting the Server for the First Time"](#)). You can also set up a MySQL service that runs automatically when Windows starts (see [Section 1.3.8, "Starting MySQL as a Windows Service"](#)).

These instructions assume that your current location is the MySQL installation directory and that it has a `bin` subdirectory containing the MySQL programs used here. If that is not true, adjust the command path names accordingly.

If you installed MySQL using the MSI, the default installation directory is `C:\Program Files\MySQL\MySQL Server 8.3`:

```
C:\> cd "C:\Program Files\MySQL\MySQL Server 8.3"
```

A common installation location for installation from a ZIP archive is `C:\mysql`:

```
C:\> cd C:\mysql
```

Alternatively, add the `bin` directory to your `PATH` environment variable setting. That enables your command interpreter to find MySQL programs properly, so that you can run a program by typing only its name, not its path name. See [Section 1.3.7, "Customizing the PATH for MySQL Tools"](#).

With the server running, issue the following commands to verify that you can retrieve information from the server. The output should be similar to that shown here.

Use `mysqlshow` to see what databases exist:

```
C:\> bin\mysqlshow
+-----+
| Databases |
+-----+
| information_schema |
| mysql |
```

```
| performance_schema |
| sys                 |
+-----+

```

The list of installed databases may vary, but always includes at least `mysql` and `information_schema`.

The preceding command (and commands for other MySQL programs such as `mysql`) may not work if the correct MySQL account does not exist. For example, the program may fail with an error, or you may not be able to view all databases. If you configured MySQL using MySQL Configurator, the `root` user is created automatically with the password you supplied. In this case, you should use the `-u root` and `-p` options. (You must use those options if you have already secured the initial MySQL accounts.) With `-p`, the client program prompts for the `root` password. For example:

```
C:\> bin\mysqlshow -u root -p
Enter password: (enter root password here)
+-----+
| Databases |
+-----+
| information_schema |
| mysql             |
| performance_schema |
| sys               |
+-----+

```

If you specify a database name, `mysqlshow` displays a list of the tables within the database:

```
C:\> bin\mysqlshow mysql
Database: mysql
+-----+
| Tables |
+-----+
| columns_priv |
| component |
| db |
| default_roles |
| engine_cost |
| func |
| general_log |
| global_grants |
| gtid_executed |
| help_category |
| help_keyword |
| help_relation |
| help_topic |
| innodb_index_stats |
| innodb_table_stats |
| ndb_binlog_index |
| password_history |
| plugin |
| procs_priv |
| proxies_priv |
| role_edges |
| server_cost |
| servers |
| slave_master_info |
| slave_relay_log_info |
| slave_worker_info |
| slow_log |
| tables_priv |
| time_zone |
| time_zone_leap_second |
| time_zone_name |
| time_zone_transition |
| time_zone_transition_type |
| user |

```

Use the `mysql` program to select information from a table in the `mysql` database:

```
C:\> bin\mysql -e "SELECT User, Host, plugin FROM mysql.user" mysql
+-----+-----+-----+
| User | Host      | plugin                |
+-----+-----+-----+
| root | localhost | caching_sha2_password |
+-----+-----+-----+
```

For more information about `mysql` and `mysqlshow`, see [mysql — The MySQL Command-Line Client](#), and [mysqlshow — Display Database, Table, and Column Information](#).

1.6 Windows Platform Restrictions

The following restrictions apply to use of MySQL on the Windows platform:

- **Process memory**

On Windows 32-bit platforms, it is not possible by default to use more than 2GB of RAM within a single process, including MySQL. This is because the physical address limit on Windows 32-bit is 4GB and the default setting within Windows is to split the virtual address space between kernel (2GB) and user/applications (2GB).

Some versions of Windows have a boot time setting to enable larger applications by reducing the kernel application. Alternatively, to use more than 2GB, use a 64-bit version of Windows.

- **File system aliases**

When using `MyISAM` tables, you cannot use aliases within Windows link to the data files on another volume and then link back to the main MySQL `datadir` location.

This facility is often used to move the data and index files to a RAID or other fast solution.

- **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. 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 <https://support.microsoft.com/kb/196271>.

- **DATA DIRECTORY and INDEX DIRECTORY**

The `DATA DIRECTORY` clause of the `CREATE TABLE` statement is supported on Windows for `InnoDB` tables only, as described in [Creating Tables Externally](#). For `MyISAM` and other storage engines, the `DATA DIRECTORY` and `INDEX DIRECTORY` clauses for `CREATE TABLE` are ignored on Windows and any other platforms with a nonfunctional `realpath()` call.

- **DROP DATABASE**

You cannot drop a database that is in use by another session.

- **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 does not work in the Western locale (code page 1252):

```
datadir="C:/私たちのプロジェクトのデータ"
```

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

- **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` 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:

```
C:\> 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:

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

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

Chapter 2 Upgrading MySQL on Windows

To upgrade MySQL on Windows, either [download and execute the latest MySQL Server MSI](#) or [use the Windows ZIP archive distribution](#).

Note

Unlike MySQL 8.3, MySQL 8.0 uses MySQL Installer to install and upgrade MySQL Server along with most other MySQL products; but MySQL Installer is not available with MySQL 8.1 and higher. However, the configuration functionality used in MySQL Installer is available as of MySQL 8.1 using [Section 1.2, "Configuration: Using MySQL Configurator"](#) that's bundled with both the MSI and Zip archive.

Considering 8.3 is an innovation release, there likely won't be a 8.3 point release to upgrade to. Upgrading beyond MySQL 8.3 involves a new series number, such as 8.1 to 8.2, so it's especially important to always back up your current MySQL installation before performing an upgrade. See [Database Backup Methods](#).

The approach you select depends on how the existing installation was performed. Before proceeding, review [Upgrading MySQL](#) for additional information on upgrading MySQL that is not specific to Windows.

Upgrading MySQL with MSI

Download and execute the latest MSI. Although upgrading between release series is not directly supported, the "Custom Setup" option allows defining an installation location as otherwise the MSI installs to the standard location, such as `C:\Program Files\MySQL\MySQL Server 8.3\`.

Execute [MySQL Configurator](#) to configure your installation.

Upgrading MySQL Using the Windows ZIP Distribution

To perform an upgrade using the Windows ZIP archive distribution:

1. Download the latest Windows ZIP Archive distribution of MySQL from <https://dev.mysql.com/downloads/>.
2. If the server is running, stop it. If the server is installed as a service, stop the service with the following command from the command prompt:

```
C:\> SC STOP mysql_d_service_name
```

Alternatively, use `NET STOP mysql_d_service_name .`

If you are not running the MySQL server as a service, use `mysqladmin` to stop it. For example, before upgrading from MySQL 8.2 to 8.3, use `mysqladmin` from MySQL 8.2 as follows:

```
C:\> "C:\Program Files\MySQL\MySQL Server 8.2\bin\mysqladmin" -u root shutdown
```

Note

If the MySQL `root` user account has a password, invoke `mysqladmin` with the `-p` option and enter the password when prompted.

3. Extract the ZIP archive. You may either overwrite your existing MySQL installation (usually located at `C:\mysql`), or install it into a different directory, such as `C:\mysql8`. Overwriting the existing installation is recommended.

4. Restart the server. For example, use the `SC START mysqld_service_name` or `NET START mysqld_service_name` command if you run MySQL as a service, or invoke `mysqld` directly otherwise.
5. If you encounter errors, see [Section 1.4, "Troubleshooting a Microsoft Windows MySQL Server Installation"](#).

Chapter 3 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 does not allow for enough ephemeral (short-lived) ports to serve those connections.

The purpose of `TIME_WAIT` is to keep a connection accepting packets even after the connection has been closed. This is because Internet routing can cause a packet to take a slow route to its destination and it may arrive after both sides have agreed to close. If the port is in use for a new connection, that packet from the old connection could break the protocol or compromise personal information from the original connection. The `TIME_WAIT` delay prevents this by ensuring that the port cannot be reused until after some time has been permitted for those delayed packets to arrive.

It is safe to reduce `TIME_WAIT` greatly on LAN connections because there is little chance of packets arriving at very long delays, as they could through the Internet with its comparatively large distances and latencies.

Windows permits ephemeral (short-lived) TCP ports to the user. After any port is closed, it remains in a `TIME_WAIT` status for 120 seconds. The port is not available again until this time expires. The default range of port numbers depends on the version of Windows, with a more limited number of ports in older versions:

- Windows through Server 2003: Ports in range 1025–5000
- Windows Vista, Server 2008, and newer: Ports in range 49152–65535

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 it 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 30 and 300 decimal, although you may wish to check with Microsoft for the latest permitted values. 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 4 Resetting the Root Password: Windows Systems

On Windows, use the following procedure to reset the password for the MySQL `'root'@'localhost'` account. To change the password for a `root` account with a different host name part, modify the instructions to use that host name.

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: From the **Start** menu, select **Control Panel**, then **Administrative Tools**, then **Services**. 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 containing the password-assignment statement on a single line. Replace the password with the password that you want to use.

```
ALTER USER 'root'@'localhost' IDENTIFIED BY 'MyNewPass';
```

4. Save the file. This example assumes that you name the file `C:\mysql-init.txt`.
5. Open a console window to get to the command prompt: From the **Start** menu, select **Run**, then enter `cmd` as the command to be run.
6. Start the MySQL server with the `init_file` system variable set to name the file (notice that the backslash in the option value is doubled):

```
C:\> cd "C:\Program Files\MySQL\MySQL Server 8.3\bin"
C:\> mysqld --init-file=C:\\mysql-init.txt
```

If you installed MySQL to a different location, adjust the `cd` command accordingly.

The server executes the contents of the file named by the `init_file` system variable at startup, changing the `'root'@'localhost'` account password.

To have server output to appear in the console window rather than in a log file, add the `--console` option to the `mysqld` command.

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

```
C:\> mysqld
      --defaults-file="C:\\ProgramData\\MySQL\\MySQL Server 8.3\\my.ini"
      --init-file=C:\\mysql-init.txt
```

The appropriate `--defaults-file` setting can be found using the Services Manager: From the **Start** menu, select **Control Panel**, then **Administrative Tools**, then **Services**. Find the MySQL service in the list, right-click 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`.

You should now be able to connect to the MySQL server as `root` using the new password. Stop the MySQL server and restart it normally. 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.

