

MySQL Workbench

MySQL Workbench

Abstract

This is the MySQL™ Workbench Reference Manual. It documents both MySQL Workbench Commercial and MySQL Workbench Community editions 6.0 through 6.0.7.

If you have not yet installed MySQL Workbench Community please download your free copy from the [download site](#). MySQL Workbench Community is available for Windows, Mac OS X, and Linux.

For release notes detailing the changes in each release, see the [MySQL Workbench Release Notes](#).

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

Document generated on: 2013-11-08 (revision: 36635)

Table of Contents

Preface and Legal Notices	ix
1. MySQL Workbench Introduction	1
2. What is new in MySQL Workbench 6.0	3
3. MySQL Workbench Editions	17
4. Installing and Launching MySQL Workbench	19
4.1. System Requirements	19
4.2. Starting MySQL Workbench	20
4.2.1. Installing MySQL Workbench on Windows	21
4.2.2. Launching MySQL Workbench on Windows	22
4.2.3. Uninstalling MySQL Workbench on Windows	22
4.2.4. Installing MySQL Workbench on Linux	23
4.2.5. Launching MySQL Workbench on Linux	23
4.2.6. Uninstalling MySQL Workbench on Linux	24
4.2.7. Installing MySQL Workbench on Mac OS X	24
4.2.8. Launching MySQL Workbench on Mac OS X	25
4.2.9. Uninstalling MySQL Workbench on Mac OS X	25
5. MySQL Connections	27
5.1. MySQL Connections Home	27
5.2. Creating A New MySQL Connection	29
5.2.1. Configure Server Management Wizard	30
5.3. Manage Server Instances Dialog	32
5.4. MySQL Connection Management Navigator	34
5.4.1. Navigator MANAGEMENT Actions	35
5.4.2. Navigator INSTANCE Actions	46
5.4.3. Navigator MySQL Enterprise Actions	50
6. Getting Started Tutorial	55
6.1. Administering a MySQL Server	55
6.2. Creating a Model	67
6.3. Adding Data to Your Database	75
7. The Home Window	81
7.1. Workbench Shortcuts	82
7.2. MySQL Connections	82
7.3. Workbench Preferences	83
7.3.1. The General Tab	84
7.3.2. The Administrator Tab	85
7.3.3. The SQL Editor Tab	86
7.3.4. The SQL Queries Tab	89
7.3.5. The Model Tab	90
7.3.6. The Model:MySQL Tab	92
7.3.7. The Diagram Tab	93
7.3.8. The Appearance Tab	94
7.3.9. The Theming Tab	96
8. SQL Development	97
8.1. Manage DB Connections Dialog	97
8.1.1. The Password Storage Vault	98
8.1.2. Standard TCP/IP Connection	99
8.1.3. Local Socket/Pipe Connection	100
8.1.4. Standard TCP/IP over SSH Connection	101
8.2. SQL Editor	101
8.2.1. Main Menu	102
8.2.2. Toolbar	103

8.2.3. SQL Query Panel	104
8.2.4. Sidebar	106
9. Data Modeling	111
9.1. Model Editor	112
9.1.1. Modeling Menus	114
9.1.2. The Toolbar	124
9.1.3. EER Diagrams	124
9.1.4. The Physical Schemata Panel	125
9.1.5. The Schema Privileges Panel	125
9.1.6. The SQL Scripts Panel	127
9.1.7. The Model Notes Panel	127
9.1.8. The History Palette	127
9.1.9. The Model Navigator Panel	127
9.1.10. The Catalog Tree Palette	128
9.1.11. The Layers Palette	128
9.1.12. The Properties Palette	129
9.2. EER Diagram Editor	129
9.2.1. The Vertical Toolbar	129
9.3. Working with Models	133
9.3.1. Creating Tables	133
9.3.2. Creating Foreign Key Relationships	145
9.3.3. Creating Views	148
9.3.4. Creating Routines and Routine Groups	150
9.3.5. Creating Layers	153
9.3.6. Creating Notes	155
9.3.7. Creating Text Objects	155
9.3.8. Creating Images	156
9.3.9. Reverse Engineering	157
9.3.10. Forward Engineering	165
9.4. Modeling Tutorials	184
9.4.1. Importing a Data Definition SQL Script	184
9.4.2. Using the Default Schema	186
9.4.3. Basic Modeling	187
9.4.4. Documenting the <code>sakila</code> Database	188
9.5. Printing	190
9.5.1. Printing Options	190
9.6. MySQL Workbench Schema Validation Plugins (Commercial Version)	190
9.6.1. General Validation	190
9.6.2. MySQL-Specific Validation	191
9.7. The DBDoc Model Reporting Dialog Window (Commercial Version)	192
9.8. Customizing DBDoc Model Reporting Templates	195
9.8.1. Supported Template Markers	199
9.8.2. Creating a Custom Template	203
10. Code Generation Overview	207
10.1. Generating SQL Statements	207
10.2. Generating PHP Code	207
11. MySQL Enterprise Features	209
11.1. MySQL Audit Inspector Interface	209
11.2. MySQL Enterprise Backup Interface	213
11.3. MySQL Workbench Backup Recovery	216
12. Database Migration Wizard	221
12.1. General installation requirements	222
12.1.1. ODBC Libraries	222
12.1.2. ODBC Drivers	223

12.2. Migration Overview	223
12.2.1. A visual guide to performing a database migration	224
12.2.2. Migrating from supported databases	238
12.2.3. Migrating from unsupported (generic) databases	239
12.3. Conceptual DBMS equivalents	239
12.4. Microsoft SQL Server migration	241
12.4.1. Preparations	241
12.4.2. Drivers	241
12.4.3. Connection Setup	243
12.4.4. Microsoft SQL Server Type Mapping	243
12.5. PostgreSQL migration	244
12.5.1. Preparations	244
12.5.2. Drivers	247
12.5.3. Connection Setup	247
12.5.4. PostgreSQL Type Mapping	247
12.6. MySQL migration	249
12.7. Using the MySQL Workbench Migration Wizard	249
12.7.1. Connecting to the databases	249
12.7.2. Schemata Retrieval and Selection	249
12.7.3. Reverse Engineering	250
12.7.4. Object Selection	250
12.7.5. Migration	250
12.7.6. Manual Editing	250
12.7.7. Target Creation Options	250
12.7.8. Schema Creation	251
12.7.9. Create Target Results	251
12.7.10. Data Migration Setup	251
12.7.11. Bulk Data Transfer	251
12.7.12. Migration Report	251
12.8. MySQL Workbench Migration Wizard FAQ	251
13. Extending Workbench	253
13.1. GRT and Workbench Data Organization	253
13.2. Modules	254
13.3. Plugins / Tools	255
13.4. Adding a GUI to a Plugin Using MForms	256
13.5. The Workbench Scripting Shell	257
13.5.1. Exploring the Workbench Scripting Shell	257
13.5.2. The Shell Window	258
13.5.3. The Files, Globals, Classes, and Modules Tabs	259
13.6. Tutorial: Writing Plugins	260
14. Keyboard Shortcuts	263
15. MySQL Utilities	267
15.1. The Preface	268
15.2. How to Install MySQL Utilities	268
15.2.1. Prerequisites	269
15.2.2. Source Code	269
15.2.3. Oracle and Red Hat Linux 6	269
15.2.4. Debian Linux	269
15.2.5. Microsoft Windows	270
15.3. Introduction	270
15.3.1. Introduction to MySQL Utilities	270
15.3.2. Connection Parameters	271
15.4. MySQL Utilities Administrative Tasks	271
15.4.1. Database Operations	272

15.4.2. General Operations	278
15.4.3. High Availability Operations	289
15.4.4. Server Operations	301
15.4.5. Specialized Operations	304
15.5. Overview of MySQL Utilities	309
15.5.1. Database Operations	309
15.5.2. General Operations	309
15.5.3. High Availability Operations	310
15.5.4. Server Operations	311
15.5.5. Specialized Operations	311
15.6. Manual Pages	312
15.6.1. mysqlauditadmin	312
15.6.2. mysqlauditgrep	316
15.6.3. mysqldbcompare	324
15.6.4. mysqldbcopy	331
15.6.5. mysqldbexport	336
15.6.6. mysqldbimport	343
15.6.7. mysqldiff	347
15.6.8. mysqldiskusage	351
15.6.9. mysqlfailover	354
15.6.10. mysqlfrm	364
15.6.11. mysqlindexcheck	367
15.6.12. mysqlmetagrep	370
15.6.13. mysqlprocgrep	374
15.6.14. mysqlreplicate	377
15.6.15. mysqlrpladmin	380
15.6.16. mysqlrplcheck	389
15.6.17. mysqlrplshow	392
15.6.18. mysqlserverclone	396
15.6.19. mysqlserverinfo	398
15.6.20. mysqluc	401
15.6.21. mysqluserclone	404
15.7. Extending MySQL Utilities	406
15.7.1. Introduction to extending the MySQL Utilities	406
15.7.2. MySQL Utilities copy_server.py sample	413
15.7.3. Specialized Operations	415
15.7.4. Parsers	418
15.8. MySQL Utilities Testing (MUT)	420
15.8.1. <code>mut</code> — MySQL Utilities Testing	420
15.9. Appendix	422
15.9.1. MySQL Utilities FAQ	422
15.9.2. MySQL Utilities Change History	424
A. Third Party Licenses	425
A.1. .NET Flat TabControl License	426
A.2. ANTLR License	427
A.3. Bitstream Vera License	427
A.4. Boost Library License	428
A.5. Cairo License	429
A.6. CTemplate (Google Template System) License	429
A.7. cURL (<code>libcurl</code>) License	430
A.8. DockPanel Suite License	430
A.9. Dojo Toolkit v1.7.0b1 License	431
A.10. GLib License (for MySQL Workbench)	431
A.11. Glitz License	432

A.12. GNU Lesser General Public License Version 2.1, February 1999	432
A.13. HtmlRenderer (System.Drawing.Html)	440
A.14. iODBC License	441
A.15. Libiconv License	442
A.16. Libintl License	442
A.17. Libxml2 License	443
A.18. Libzip License	443
A.19. Lua (liblea) License	444
A.20. Paramiko License	444
A.21. PCRE License	444
A.22. Pixman License	446
A.23. PyCrypto 2.6 License	447
A.24. PyODBC License	449
A.25. PySQLite License	450
A.26. Python License	450
A.27. Scintilla License	460
A.28. ScintillaNET License	462
A.29. SQLCipher License	462
A.30. TinyXML License	463
A.31. TreeViewAdv for .NET License	463
A.32. VSQlite++ License	464
A.33. <code>zlib</code> License	464
B. MySQL Workbench FAQ	467
C. Reporting A Useful MySQL Workbench Bug	469
D. MySQL Workbench and Utilities Change History	471
D.1. MySQL Workbench Change History	471
D.2. MySQL Utilities Change History	471

Preface and Legal Notices

This is the User Manual for the MySQL Workbench.

For license information, see the [Legal Notices](#). This product may contain third-party code. For license information on third-party code, see [Appendix A, Third Party Licenses](#).

Legal Notices

Copyright © 2006, 2013, Oracle and/or its affiliates. All rights reserved.

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.

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.

If this software or related documentation is delivered to the U.S. Government or anyone licensing it on behalf of the U.S. Government, the following notice is applicable:

U.S. GOVERNMENT RIGHTS Programs, software, databases, and related documentation and technical data delivered to U.S. Government customers are "commercial computer software" or "commercial technical data" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, the use, duplication, disclosure, modification, and adaptation shall be subject to the restrictions and license terms set forth in the applicable Government contract, and, to the extent applicable by the terms of the Government contract, the additional rights set forth in FAR 52.227-19, Commercial Computer Software License (December 2007). Oracle USA, Inc., 500 Oracle Parkway, Redwood City, CA 94065.

This software 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 which may create a risk of personal injury. If you use this software in dangerous applications, then you shall be responsible to take all appropriate fail-safe, backup, redundancy, and other measures to ensure the safe use of this software. Oracle Corporation and its affiliates disclaim any liability for any damages caused by use of this software in dangerous applications.

Oracle is a registered trademark of Oracle Corporation and/or its affiliates. MySQL is a trademark of Oracle Corporation and/or its affiliates, and shall not be used without Oracle's express written authorization. Other names may be trademarks of their respective owners.

This software and documentation may provide access to or information on 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. 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.

This document in any form, software or printed matter, contains proprietary information that is the exclusive property of Oracle. Your access to and use of this material is subject to the terms and conditions of your Oracle Software License and Service Agreement, which has been executed and with which you agree to comply. This document and information contained herein may not be disclosed, copied, reproduced, or distributed to anyone outside Oracle without prior written consent of Oracle or as specifically provided

below. This document is not part of your license agreement nor can it be incorporated into any contractual agreement with Oracle or its subsidiaries or affiliates.

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.

For more information on the terms of this license, or for details on how the MySQL documentation is built and produced, please visit [MySQL Contact & Questions](#).

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

For additional documentation on MySQL products, including translations of the documentation into other languages, and downloadable versions in variety of formats, including HTML and PDF formats, see the [MySQL Documentation Library](#).

Chapter 1. MySQL Workbench Introduction

MySQL Workbench provides a graphical tool for working with MySQL Servers and databases. MySQL Workbench fully supports MySQL Server versions 5.1 and above. It is also compatible with MySQL Server 5.0, but not every feature of 5.0 may be supported. It does not support MySQL Server versions 4.x.

MySQL Workbench provides five main areas of functionality:

- **SQL Development:** Enables you to create and manage connections to database servers. As well as enabling you to configure connection parameters, MySQL Workbench provides the capability to execute SQL queries on the database connections using the built-in SQL Editor. This functionality replaces that previously provided by the Query Browser standalone application.
- **Data Modeling:** Enables you to create models of your database schema graphically, reverse and forward engineer between a schema and a live database, and edit all aspects of your database using the comprehensive Table Editor. The Table Editor provides easy-to-use facilities for editing Tables, Columns, Indexes, Triggers, Partitioning, Options, Inserts and Privileges, Routines and Views.
- **Server Administration:** Enables you to create and administer server instances.
- **Data Migration:** Allows you to migrate from Microsoft SQL Server, Sybase ASE, SQLite, SQL Anywhere, PostgreSQL, and other RDBMS tables, objects and data to MySQL. Migration also supports migrating from earlier versions of MySQL to the latest releases.
- **MySQL Enterprise Support:** Support for Enterprise products such as MySQL Enterprise Backup and MySQL Audit.

MySQL Workbench is available in two editions, the Community Edition and the Commercial Edition. The Community Edition is available free of charge. The Commercial Edition provides additional Enterprise features, such as database documentation generation, at low cost.

For release notes detailing changes made in each release of MySQL Workbench, see [Appendix D, MySQL Workbench and Utilities Change History](#).

Chapter 2. What is new in MySQL Workbench 6.0

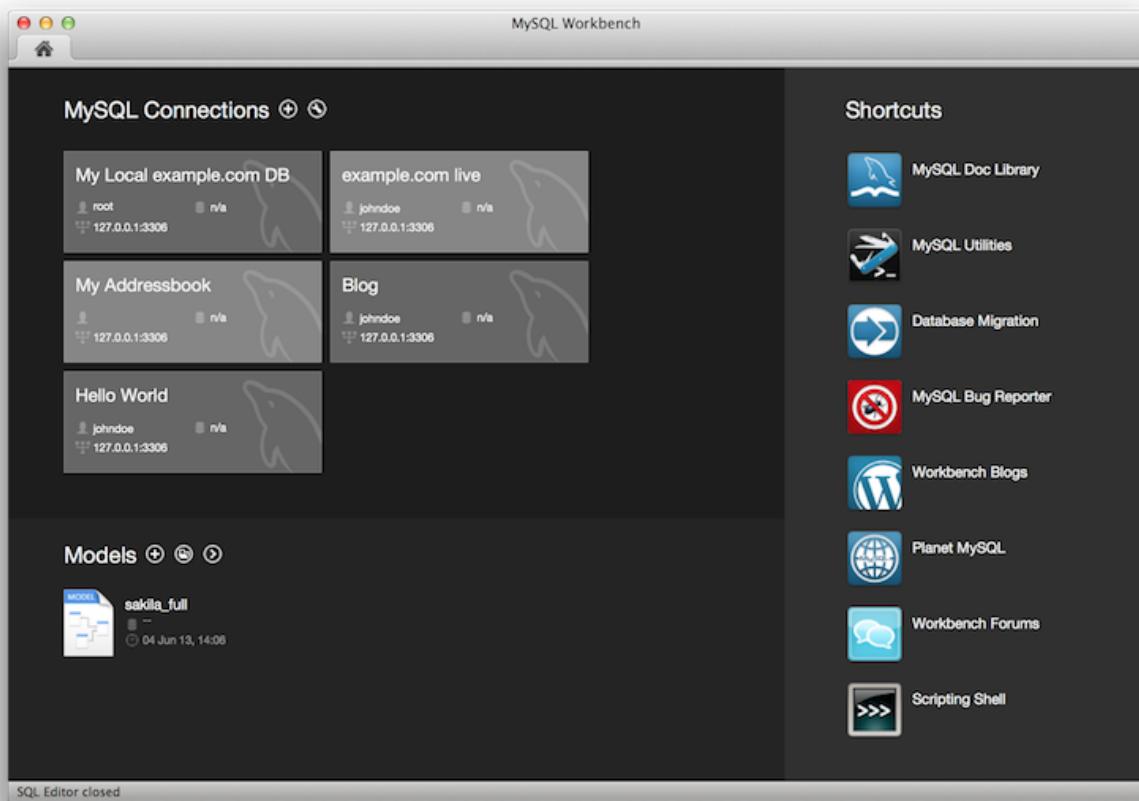
This section summarizes how MySQL Workbench 6.0 is different than MySQL Workbench 5.2.

A new home screen

A new, modernized "Home" screen where major functionality of MySQL Workbench can be accessed, including connections to MySQL servers, modeling, migration, and the command-line utilities.

Figure 2.1. Home Screen: Workbench 5.2



Figure 2.2. Home Screen: Workbench 6.0

Unified SQL Editor and Administration interface

In the new user interface, the Server Administration functionality (such as start/stop server, managing user accounts etc) is now accessible directly from the SQL Editor interface, located near where the schema information can be browsed and queries executed.

The image below contains three screenshots of the Schema window in the SQL Editor. The first is from MySQL Workbench 5.2, the second is MySQL Workbench 6.0 with the management tab collapsed, and the third shows what the merged management tab looks like. Toggle the merged and tabbed views by clicking the new merge button next to the refresh button.

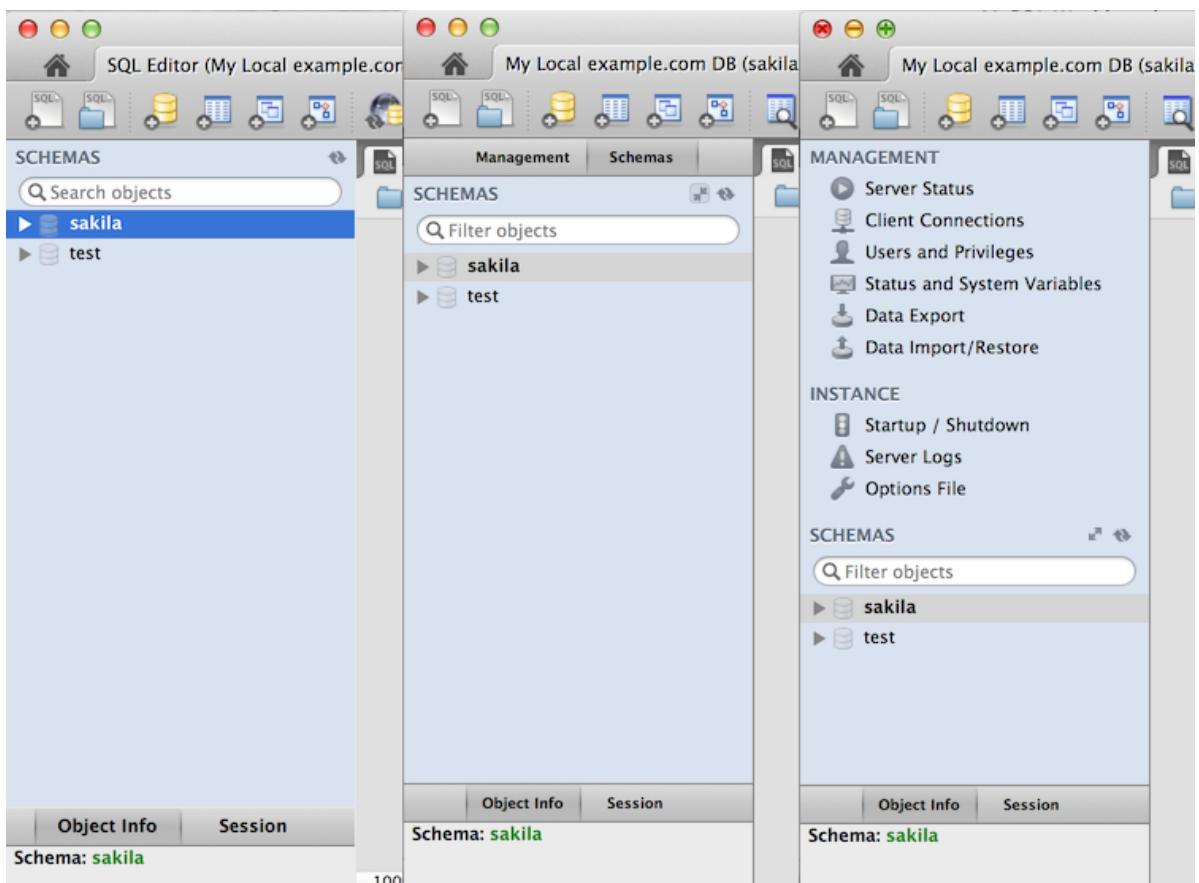
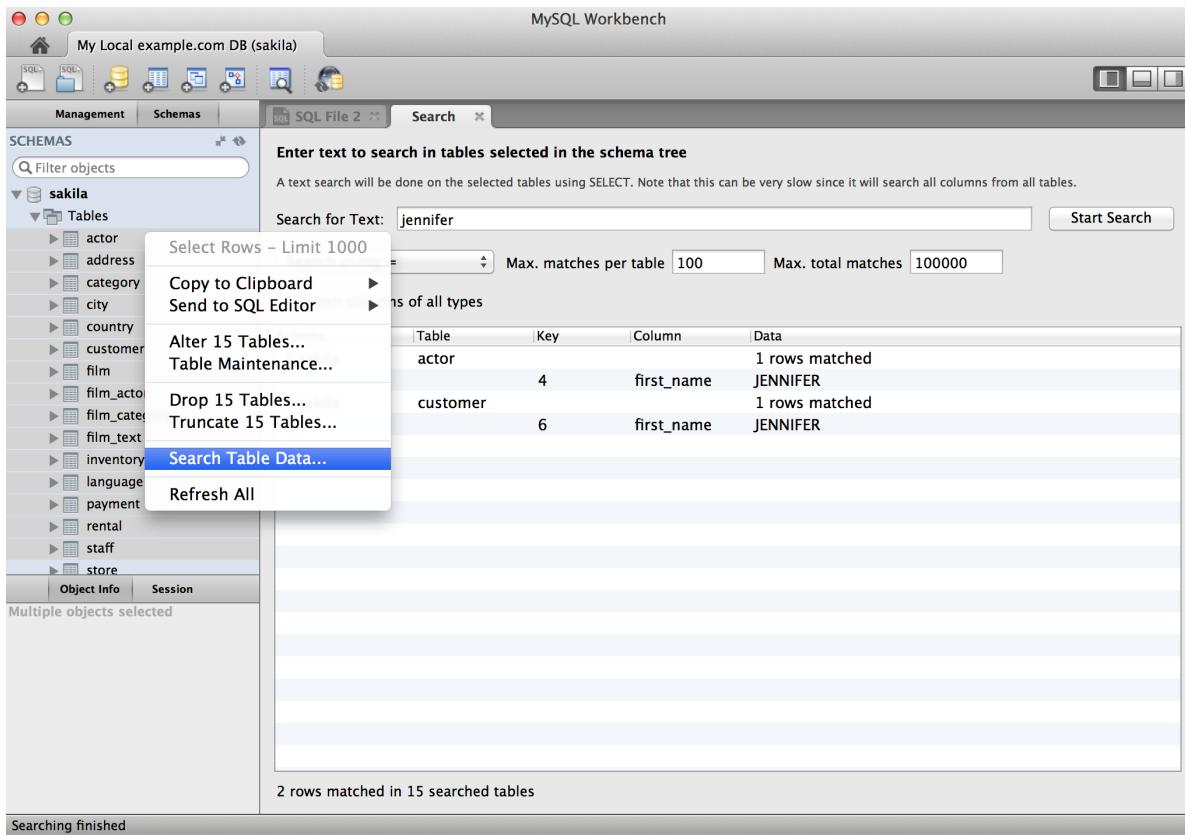
Figure 2.3. Comparing the SQL Editor interface for Workbench 5.2 and 6.0

Table data search

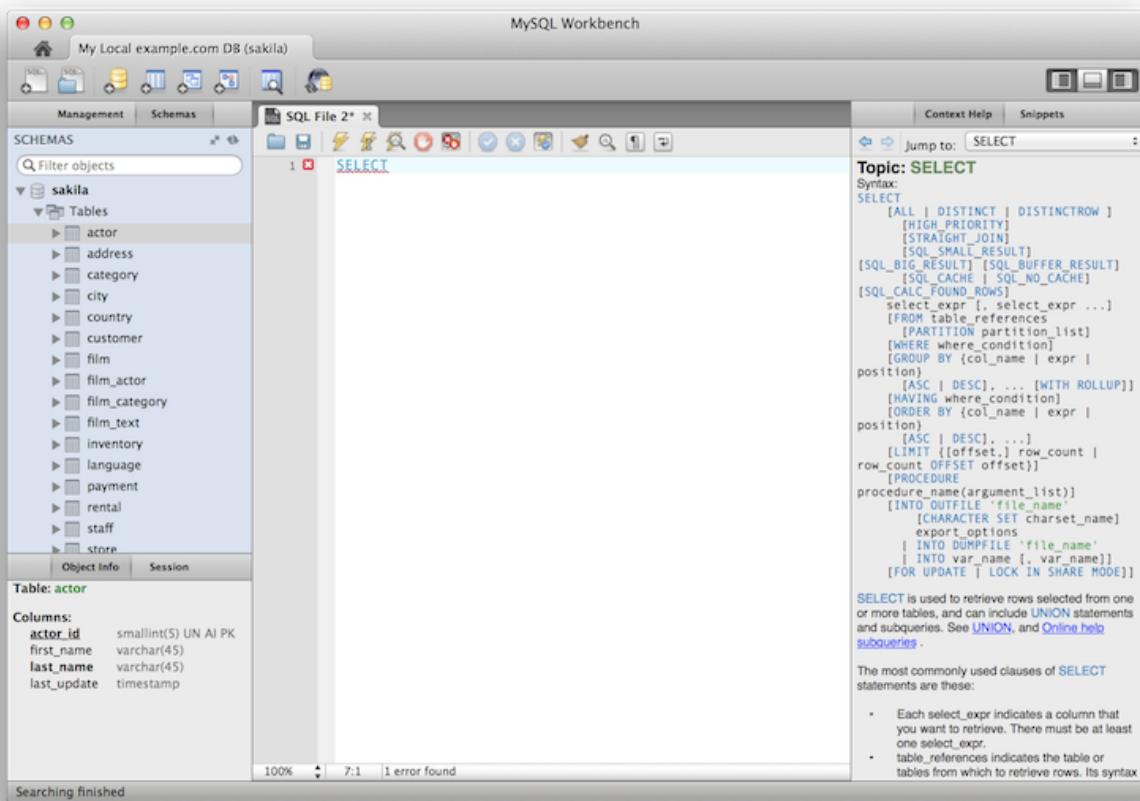
You can select schemas and/or tables to perform client-side searches for user specified strings and patterns. To access this new search feature, right click select a schema or a table in the left sidebar and select Search Table Data....

This screenshot demonstrates the search feature, along with an example search. Multiple tables were selected and searched in this example:

Figure 2.4. Table search functionality

Context Sensitive help for the SQL Editor

Select a keyword or function in your query and after a delay it will show formatted help information from the MySQL Server (equivalent to using the help command from the command-line MySQL Client).

Figure 2.5. Context Sensitive Help

Schema Inspector

New Schema Inspector feature allows you to browse general information from schema objects. For tables, it's also possible to perform maintenance tasks such as [ANALYZE](#), [OPTIMIZE](#), [CHECK](#), and [CHECKSUM TABLE](#). To access the inspector, right-click a schema and select the Schema Inspector

Schema Inspector

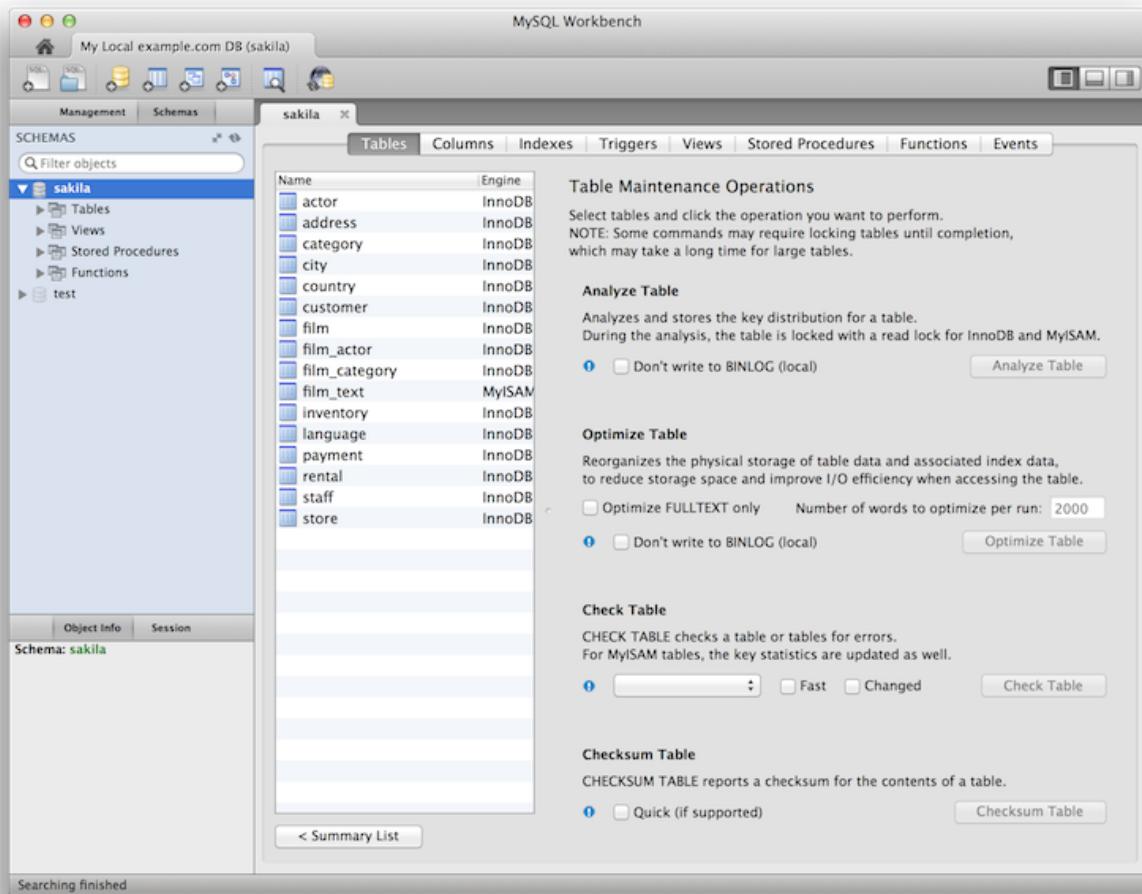
Figure 2.6. Schema Inspector

The screenshot shows the MySQL Workbench interface with the title bar "MySQL Workbench". In the top left, there's a toolbar with icons for file operations, schemas, and databases. The main area is titled "sakila" and contains a "Tables" tab. Below the tab are several tabs: Columns, Indexes, Triggers, Views, Stored Procedures, Functions, and Events. The "Tables" tab is active, displaying a list of tables from the sakila schema. The table data is as follows:

Name	Engine	Version	Row Format	Rows	Avg Row Length	Data Length
actor	InnoDB	10	Compact	200	81	16384
address	InnoDB	10	Compact	603	135	81920
category	InnoDB	10	Compact	16	1024	16384
city	InnoDB	10	Compact	600	81	49152
country	InnoDB	10	Compact	109	150	16384
customer	InnoDB	10	Compact	599	136	81920
film	InnoDB	10	Compact	1000	196	196608
film_actor	InnoDB	10	Compact	5462	35	196608
film_category	InnoDB	10	Compact	1000	65	65536
film_text	MyISAM	10	Dynamic	1000	119	119616
inventory	InnoDB	10	Compact	4581	39	180224
language	InnoDB	10	Compact	6	2730	16384
payment	InnoDB	10	Compact	16086	98	1589248
rental	InnoDB	10	Compact	16008	99	1589248
staff	InnoDB	10	Compact	2	32768	65536
store	InnoDB	10	Compact	2	8192	16384

At the bottom of the main window, there are buttons for "Maintenance >" and "Refresh". The status bar at the bottom left says "Searching finished".

And choosing Maintenance for a table:

Figure 2.7. Schema Inspector: Maintenance

Cascaded `DELETE` statements generator

You can generate a series of `DELETE` statements needed to delete a row from that table, which includes rows from other tables that reference it, recursively. The `SELECT` version allows you to preview what rows would be deleted. Right click a table and select Copy to Clipboard, Delete with References. TODO: Also, `DELETE>`

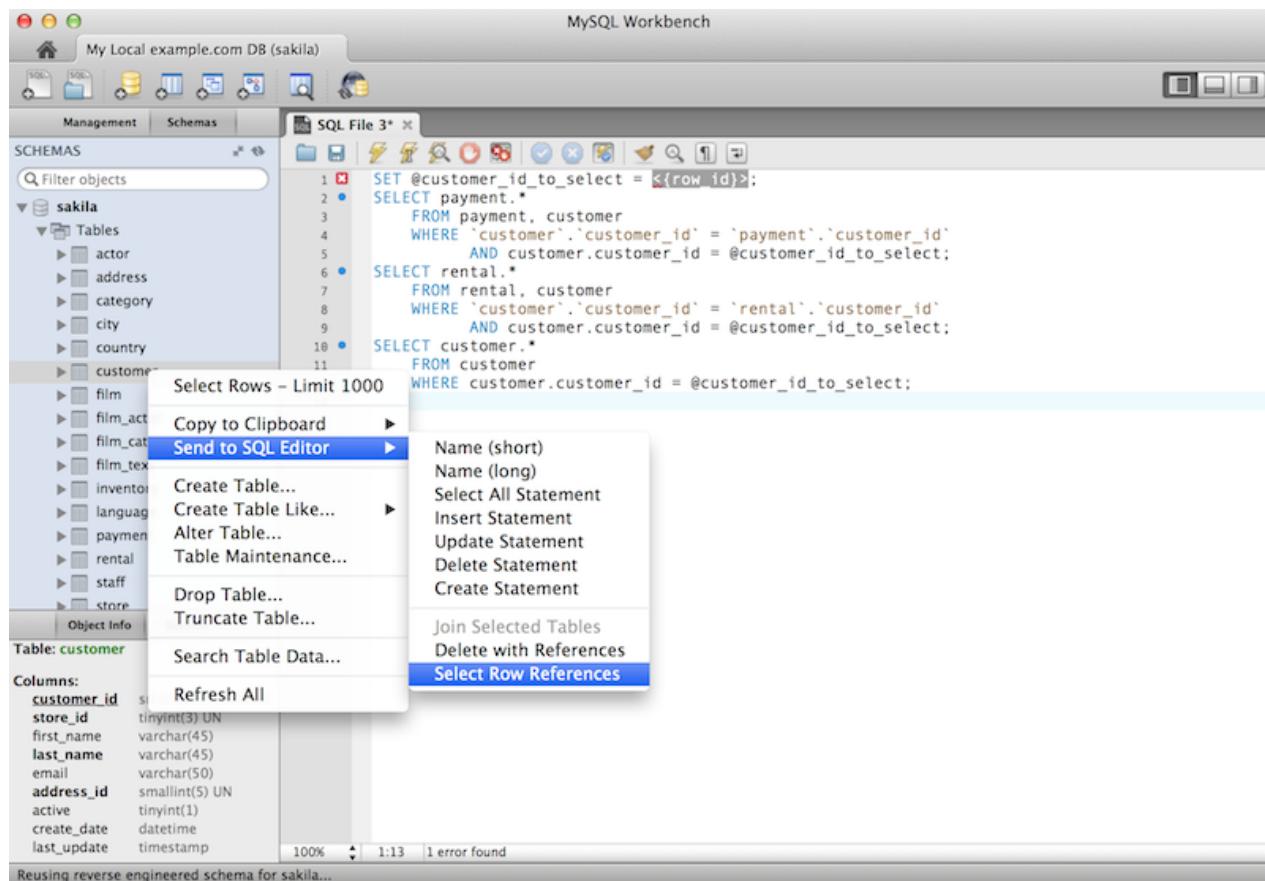
Figure 2.8. Cascading SELECT

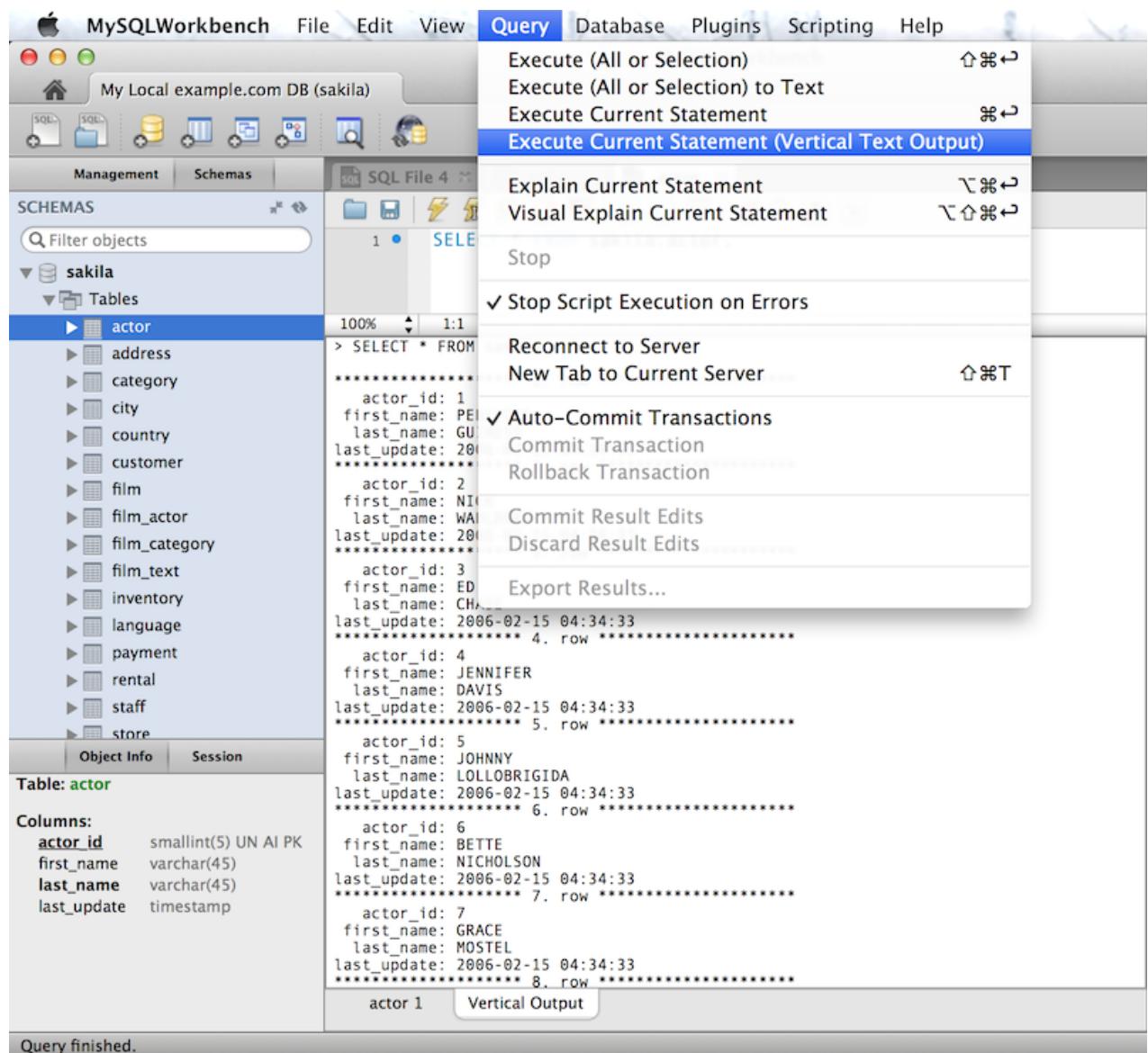
Table templates

Define templates of tables with commonly used columns, to be used to create new tables in a live connection or in an EER model. In the SQL Editor, choose **Create Table Like...**, or in Modeling, use the right sidebar.

Vertical Text

A Vertical Text output option for queries (equivalent to \G from the command-line Client) was added. To execute, choose [Query](#), Execute Current Statement (Vertical Text Output).

Figure 2.9. Vertical Text (\G)



Improved Visual Explain

The Visual Explain output was improved. TODO: Additions? Colors, numbers on lines, more information, ... specifics?

Figure 2.10. Visual Explain: Workbench 5.2

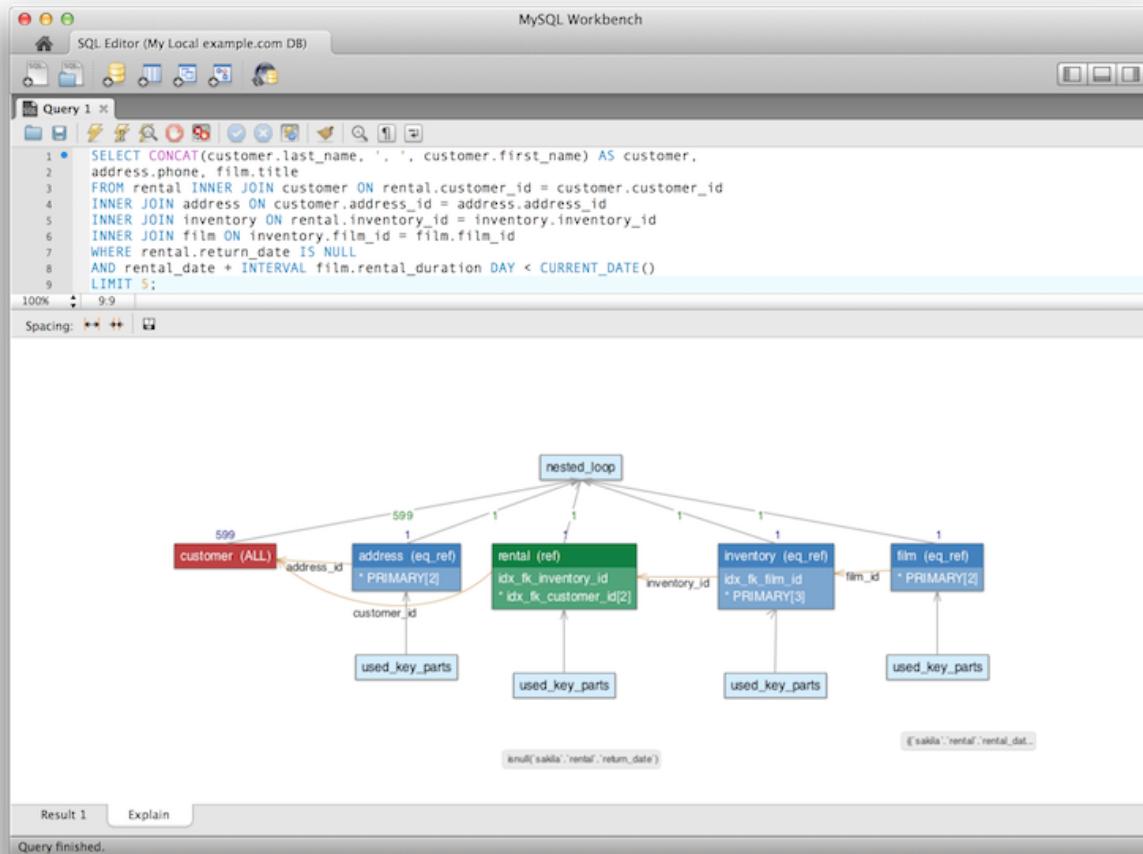
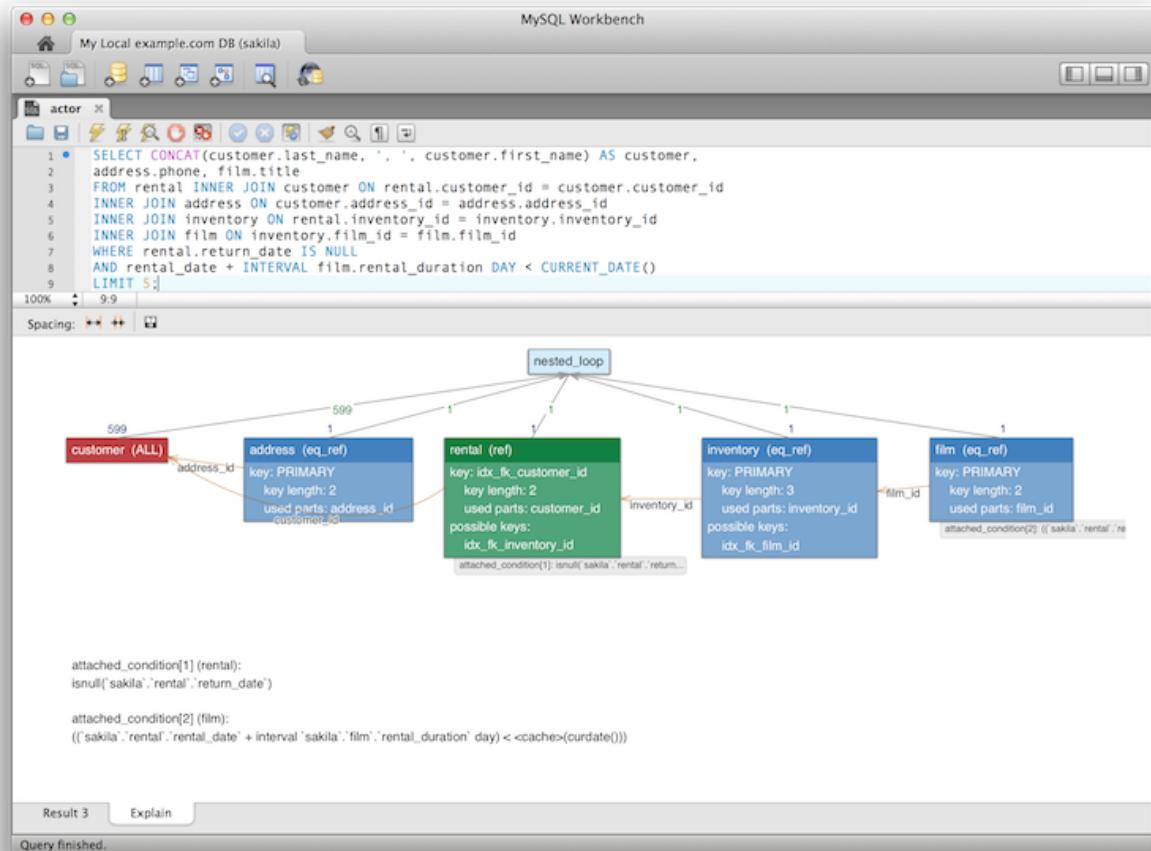


Figure 2.11. Visual Explain: Workbench 6.0



Improved Server Status

Additional server status information was added, and the user interface was improved. Select **Server Status** from the **Management** tab to open this window.

Improved Server Status

Figure 2.12. Server Status: Workbench 5.2

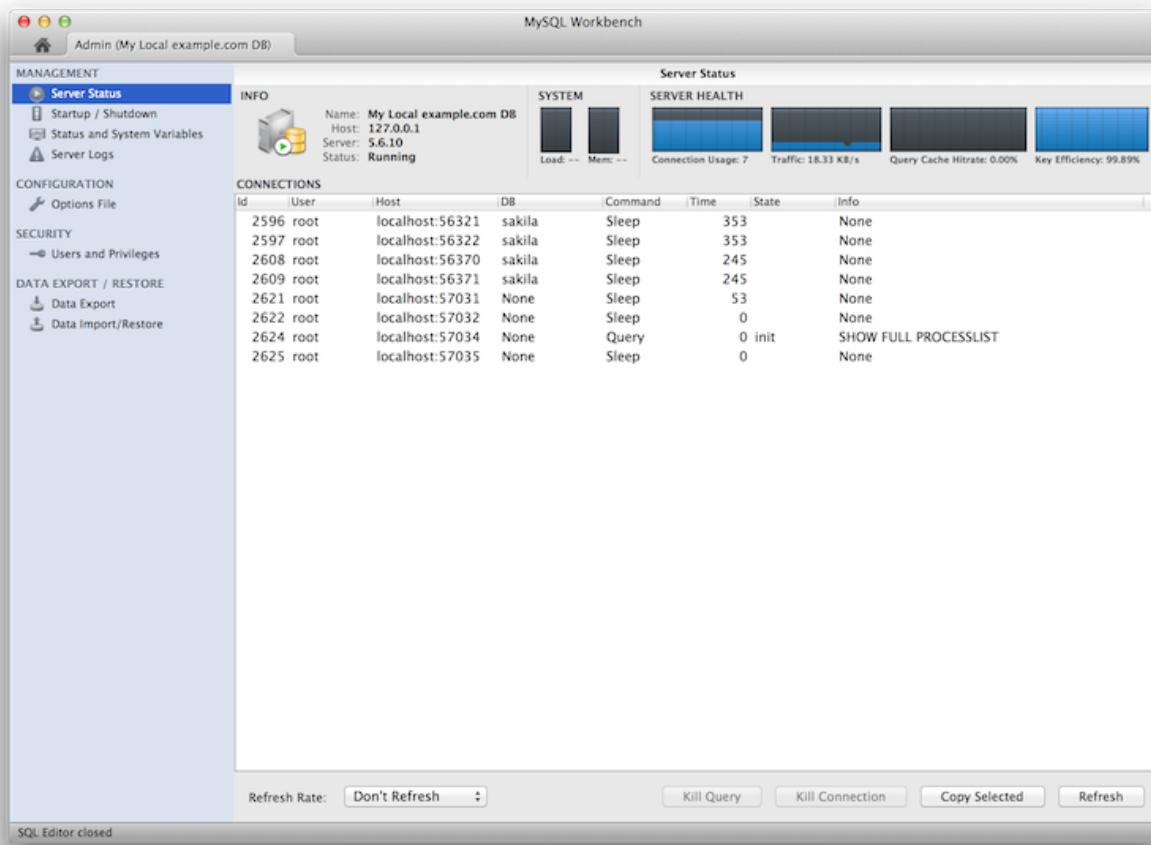
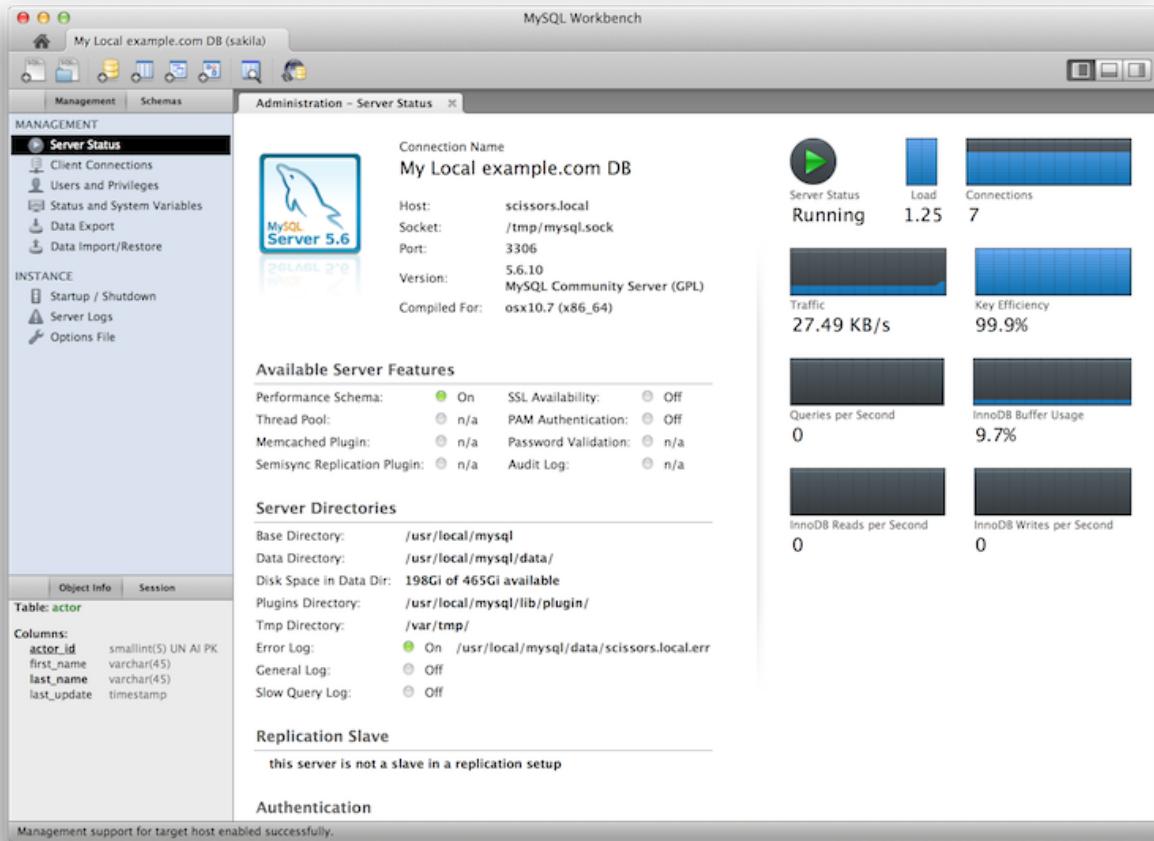


Figure 2.13. Server Status: Workbench 6.0

Enterprise Features

Support for MySQL Enterprise features in the Commercial edition of MySQL Workbench was added. From within the **Management** tab for an open connection, look for the following products under the heading **MySQL Enterprise**:

MySQL Enterprise Backup (MEB): A GUI frontend for the MEB tool. After installing a commercial version of MySQL Workbench and MySQL Enterprise Backup, MySQL Workbench will check for and handle the pre-requisites. Backup recovery is also supported. This plugin supports MEB with local and remote installations of Linux and Mac OS X, and locally for MySQL Windows.

MySQL Audit Log Inspector: A GUI for browsing the contents of generated logs by the commercial Audit Log Plugin. Powerful filtering and search capabilities are available. Fast browsing is provided by caching the log data locally in an encrypted file. This plugin supports MEB with local and remote installations of Linux and Mac OS X, and locally for MySQL Windows.

Database Migration Features

SQL Anywhere and SQLite are now supported.

Chapter 3. MySQL Workbench Editions

MySQL Workbench is available in the following editions:

- [Community Edition](#) (Open Source, GPL) -- This is the foundation for all other editions
- [Standard Edition](#) (Commercial)
- [Enterprise Edition](#) (Commercial)

For details about each edition, see <http://www.mysql.com/products/workbench/features.html>

Chapter 4. Installing and Launching MySQL Workbench

Table of Contents

4.1. System Requirements	19
4.2. Starting MySQL Workbench	20
4.2.1. Installing MySQL Workbench on Windows	21
4.2.2. Launching MySQL Workbench on Windows	22
4.2.3. Uninstalling MySQL Workbench on Windows	22
4.2.4. Installing MySQL Workbench on Linux	23
4.2.5. Launching MySQL Workbench on Linux	23
4.2.6. Uninstalling MySQL Workbench on Linux	24
4.2.7. Installing MySQL Workbench on Mac OS X	24
4.2.8. Launching MySQL Workbench on Mac OS X	25
4.2.9. Uninstalling MySQL Workbench on Mac OS X	25

MySQL Workbench is available for Windows, Linux, and Mac OS X.

Binary distributions of MySQL Workbench are available for the preceding platforms. Source code distributions are also available as a [tar.gz](#) package, or an RPM package.

Downloads are available at <http://dev.mysql.com/downloads/tools/workbench/>.



Note

Microsoft Windows users may use the [MySQL Installer](#), which bundles MySQL Workbench.

The following sections explain the installation process for each of these platforms.

4.1. System Requirements

Requirements for using MySQL Workbench.

Hardware Requirements

MySQL Workbench requires a system that runs smoothly. The minimum hardware requirements are:

- CPU: 32-bit or 64-bit
- Cores: Single (Single Core 3GHz or higher, Dual Core 2GHz or higher recommended)
- RAM: 4 GB (6 GB or higher recommended)
- Graphic Accelerators: nVidia or ATI with support of OpenGL 2 or higher
- Display Resolution: 1280x1024 minimum (1920x1200 or higher recommended)

Software Requirements

The following operating systems are officially supported:

- Apple Mac OS X v10.6.1+ (32-bit/64-bit)

- Microsoft Windows 7 (32-bit/64-bit) or greater
- Fedora 15 (32-bit/64-bit)
- Oracle Linux 6 (32-bit/64-bit)
- Ubuntu 12.04 LTS (32-bit/64-bit) or greater

MySQL Workbench also has the following general requirements:

**Note**

On startup, the application checks the OpenGL version and chooses between software and hardware rendering. To determine which rendering method is being used, open the [Help](#) menu and choose the System Info item.

Requirements for Linux:

- The requirements for Linux are embedded within their respective packages. Use the platform specific tool (for example, yum or apt) to install the package and their dependencies.
- The "Save password in keychain" functionality requires [gnome-keyring](#) to store the passwords. Note that on KDE systems, the [gnome-keyring](#) daemon is not started by default.

Requirements for Microsoft Windows:

- Microsoft .NET 4.0 Framework
- Microsoft Visual C++ 2010 Redistributable Package (x86)

**Note**

For convenience, the Windows libraries are available as the download "Dependencies for Compiling in Windows".

4.2. Starting MySQL Workbench

The procedure for launching MySQL Workbench depends on the platform. Generally, there are two ways to launch MySQL Workbench: either from the command line or from the graphical user interface of the host operating system. Using the command-line launching facility is useful when you want to customize some aspects of the way MySQL Workbench operates. The following sections describe how to launch MySQL Workbench for each of the supported platforms.

In addition to platform-specific command-line options, MySQL Workbench has the following command-line options:

**Note**

On Microsoft Windows, the command-line options contain one leading dash instead of two. For example, use `-log-level` for Microsoft Windows and `--log-level` for Linux and Mac OS X.

- `--log-level level`: Controls the verbosity level for logging output from Workbench.

With increasingly levels of verbosity, the valid values for `level` are: error, warning, info, debug1, debug2, and debug3.

The location of the generated log files, such as `wb.log`, are as follows:

Platform	Default location
Linux	<code>~/.mysql/workbench/log/</code>
Mac OS X	<code>~/Library/Application Support/Workbench/log/</code>
Microsoft Windows	<code>C:\Users\user_name\AppData\Roaming\MySQL\Workbench\log\</code>

- `--admin instance`: Load the server instance specified.
- `--query connection`: Load the connection specified.
- `--model modelfile`: Load the model specified.
- `--script script`: Run the script specified.
- `--run code`: Run the code snippet specified.
- `--quit-when-done`: Quits MySQL Workbench after `--script` or `--run` finishes.

4.2.1. Installing MySQL Workbench on Windows

MySQL Workbench for Windows can be installed using the Windows Installer package or installed manually from a Zip file.



Important

Installing MySQL Workbench using the Installer package requires either Administrator or Power User privileges. If you are using the Zip file without an installer, you do not need Administrator or Power User privileges.

Installing MySQL Workbench Using MySQL Installer

When executing [MySQL Installer](#), you may choose MySQL Workbench as one of the products to install. It is selected by default, and essentially executes the Installer Package described below.

Installing MySQL Workbench Using the Installer Package

MySQL Workbench can be installed using the Windows Installer (`.msi`) installation package. The MSI package bears the name `mysql-workbench-version-win32.msi`, where `version` indicates the MySQL Workbench version number.

1. To install MySQL Workbench, right-click the MSI file and select the Install item from the pop-up menu, or double-click the file.
2. In the **Setup Type** window you may choose a `Complete` or `Custom` installation. To use all features of MySQL Workbench choose the `Complete` option.
3. Unless you choose otherwise, MySQL Workbench is installed in `C:\%PROGRAMFILES%\MySQL\MySQL Workbench 6.0 edition_type\`, where `%PROGRAMFILES%` is the default directory for programs for your locale. The `%PROGRAMFILES%` directory may be `C:\Program Files` or `C:\programme`.

Installing from the Zip File

If you have problems running the Installer package, an alternative is to install from a Zip file without an installer. That file is called `mysql-workbench-version-win32.zip`.

To install using the Zip file, download the Zip file to a convenient location and decompress the file using a Zip utility. You can place the resulting directory anywhere on your system. You need not install or configure the application before using it. You may want to create a shortcut on your desktop or the quick launch bar.

4.2.2. Launching MySQL Workbench on Windows

To start MySQL Workbench on Windows, select Start, Programs, MySQL, then select MySQL Workbench. This executes the `MySQLWorkbench.exe` file on your system.

Alternatively, start MySQL Workbench from the command line. To view the available command-line options, issue the command `MySQLWorkbench -help` from the MySQL Workbench installation directory. You will see the following output:

```
MySQLWorkbench.exe [<options>] [<model file>]

Options:
  -swrendering      Force the diagram canvas to use software rendering instead of OpenGL
  -query [<connection>] Open a query tab to the named connection or prompt for it if none given
  -admin <instance>  Open a administration tab to the named instance
  -upgrade-mysql-dbs Open a migration wizard tab
  -model <model file> Open the given EER model file
  -open <file>       Open the given file at startup
  -run <script>       Execute the given code in default language for GRT shell
  -run-python <script> Execute the given code in Python
  -run-lua <script>   Execute the given code in Lua
  -migration         Open the Migration Wizard tab
  -quit-when-done    Quit Workbench when the script is done
  -log-to-stderr     Also log to stderr
  -help, -h          Show command line options and exit
  -log-level=<level> Valid levels are: error, warning, info, debug1, debug2, debug3
  -verbose, -v        Enable diagnostics output
  -version           Show Workbench version number and exit
```

Use the `-swrendering` option if your video card does not support OpenGL 1.5. The `-version` option can be used to display the MySQL Workbench version number. The other options are self-explanatory.

4.2.3. Uninstalling MySQL Workbench on Windows

The method for uninstalling MySQL Workbench depends on how you installed MySQL Workbench in the first place.

Removing MySQL Workbench After Installation Using the Installer Package

1. To uninstall MySQL Workbench, open the **Control Panel** and Choose **Add or Remove Programs**. Find the MySQL Workbench entry and choose the **Remove** button. This will remove MySQL Workbench.
2. Any modules added to the `C:\%PROGRAMFILES%\MySQL\MySQL Workbench version\modules` directory will **not** be deleted.



Note

If you installed MySQL Workbench using the Installer package, it is not possible to remove MySQL Workbench from the command line. Although you can manually remove some of the components, there is no command-line option for removing MySQL Workbench.

Removing the MySQL Workbench directory manually will not remove all the files belonging to MySQL Workbench.

Removing MySQL Workbench After Installation from the MySQL Installer

Open the MySQL Installer for Windows, click **Remove MySQL Products**, choose MySQL Workbench, and then **Execute**.

4.2.4. Installing MySQL Workbench on Linux

There are binary distributions of MySQL Workbench available for several variants of Linux, including Fedora, Oracle Linux, and Ubuntu.

In addition to the binary distributions, it is also possible to download the MySQL Workbench source code as a `.tar.gz` or RPM package.

Check the MySQL Workbench [download page](#) for the latest packages.

The procedure for installing on Linux depends on which Linux distribution you are using.

Installing DEB packages

On Ubuntu, and other systems that use the Debian package scheme, you can install MySQL Workbench using a command such as:

```
shell> sudo dpkg -i package.deb
```

`package.deb` is the MySQL Workbench package name; for example, `mysql-workbench-community-version_i386.deb`, where `version` is the MySQL Workbench version number.



Note

You may be warned that certain libraries are not available, depending on what you already have installed. Install the required libraries and then install the MySQL Workbench package again.

Installing RPM packages

On Red Hat-based systems, and other systems that use the RPM package format, MySQL Workbench can be installed by a command such as:

```
shell> sudo rpm -i package.rpm
```

`package.rpm` is the MySQL Workbench package name; for example, `mysql-workbench-community-version-1fc10.x86_64.rpm`, where `version` is the MySQL Workbench version number.

4.2.5. Launching MySQL Workbench on Linux

After MySQL Workbench has been installed, it can be launched by selecting **Applications**, **Programming**, **MySQL Workbench** from the main menu.

MySQL Workbench can also be launched from the command line on Linux. Type the command:

```
shell> /usr/bin/mysql-workbench --help
```

This will display the available command-line options:

```
mysql-workbench [<options>] [<model file>]
Options:
  --force-sw-render      Force Xlib rendering
  --force-opengl-render  Force OpenGL rendering
  --query [<connection>] Open a query tab to the named connection or prompt for it if none given
  --admin <instance>    Open a administration tab to the named instance
  --upgrade-mysql-dbs   Open a migration wizard tab
  --model <model file>  Open the given EER model file
  --open <file>          Open the given file at startup
  --run <script>         Execute the given code in default language for GRT shell
  --run-python <script>  Execute the given code in Python
  --run-lua <script>     Execute the given code in Lua
  --migration            Open the Migration Wizard tab
  --quit-when-done       Quit Workbench when the script is done
  --log-to-stderr        Also log to stderr
  --help, -h              Show command line options and exit
  --log-level=<level>    Valid levels are: error, warning, info, debug1, debug2, debug3
  --verbose, -v           Enable diagnostics output
  --version               Show Workbench version number and exit
```

4.2.6. Uninstalling MySQL Workbench on Linux

The procedure for uninstalling MySQL Workbench on Linux depends on the package you are using.

Uninstalling DEB packages

To uninstall a Debian package, use this command:

```
shell> sudo dpkg -r mysql-workbench-community
```

This command does not remove the configuration files. If you wish to also remove the configuration files, use this command:

```
shell> sudo dpkg --purge mysql-workbench-community
```

Uninstalling RPM packages

To uninstall an RPM package, use this command:

```
shell> sudo rpm -e mysql-workbench-community
```

This command does not remove the configuration files.

4.2.7. Installing MySQL Workbench on Mac OS X

MySQL Workbench for Mac OS X is distributed as a DMG file. The file is named `mysql-workbench-community-version-osx.dmg`, where `version` is the MySQL Workbench version.

To install MySQL Workbench on Mac OS X, download the file. Double-click the downloaded file. You will be presented with the installation window.

Figure 4.1. MySQL Workbench Mac OS X Installation Window

Drag the MySQL Workbench icon onto the Applications icon as instructed. MySQL Workbench is now installed.

You can now launch MySQL Workbench from the Applications folder.

4.2.8. Launching MySQL Workbench on Mac OS X

To launch MySQL Workbench on Mac OS X, open the Applications folder in the Finder, then double-click MySQL Workbench.

It is also possible to start MySQL Workbench from the command line:

```
shell> open MySQLWorkbench.app model_file
```

A model file must be specified.

4.2.9. Uninstalling MySQL Workbench on Mac OS X

To uninstall MySQL Workbench for Mac OS X, locate MySQL Workbench in the Applications folder, right-click, and select Move to Trash.

Chapter 5. MySQL Connections

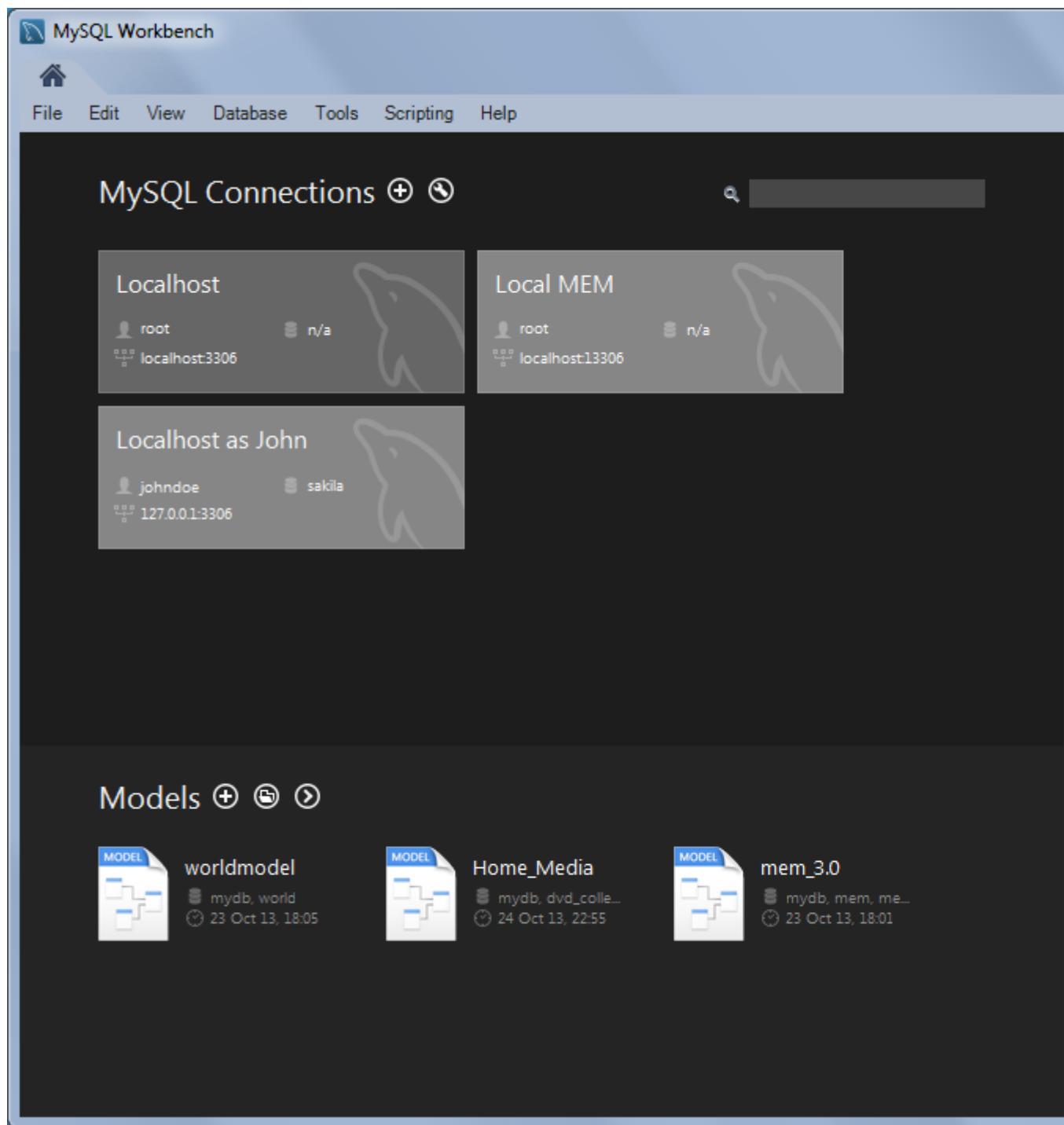
Table of Contents

5.1. MySQL Connections Home	27
5.2. Creating A New MySQL Connection	29
5.2.1. Configure Server Management Wizard	30
5.3. Manage Server Instances Dialog	32
5.4. MySQL Connection Management Navigator	34
5.4.1. Navigator MANAGEMENT Actions	35
5.4.2. Navigator INSTANCE Actions	46
5.4.3. Navigator MySQL Enterprise Actions	50

To administer your MySQL Server, you must first create a MySQL connection. These connections are listed on the MySQL Workbench Home page, and creating a MySQL connection is often the first action performed after installing MySQL Workbench.

5.1. MySQL Connections Home

The following is a typical MySQL Workbench Home page:

Figure 5.1. MySQL Workbench Home Page

Notice the three MySQL connections named "Localhost", "Local MEM", and "Localhost as John" that were created. Each tile shows the Username, Hostname, and (optionally) the default Schema used with the connection.

**Note**

The optional **Default Schema** is selected when a MySQL connection is opened by issuing a "USE database" statement.

To add a connection, click the **[+]** icon to the right of the **MySQL Connections** title. For more information about creating a MySQL connection, see [Section 5.2, "Creating A New MySQL Connection"](#).

MySQL Connection Groups

5.2. Creating A New MySQL Connection

To add a connection, click the **[+]** icon to the right of the **MySQL Connections** title. This opens the **Setup New Connection** form:

Figure 5.2. Setup New Connection Form

The screenshot shows the 'Setup New Connection' dialog box. At the top, there is a 'Connection Name:' field with a placeholder 'Type a name for the connection'. Below it is a 'Connection Method:' dropdown set to 'Standard (TCP/IP)'. Underneath these are three tabs: 'Parameters' (selected), 'SSL', and 'Advanced'. The 'Parameters' tab contains fields for 'Hostname' (127.0.0.1), 'Port' (3306), 'Username' (root), 'Password' (with 'Store in Vault...' and 'Clear' buttons), and 'Default Schema' (empty). To the right of each parameter are descriptive labels. At the bottom of the dialog are buttons for 'Configure Server Management...', 'Test Connection', and 'Cancel'.

**Important**

The **Setup New Connection** form features a **Configure Server Management** button (bottom left) that is required if a MySQL connection will be used for anything that requires shell access to the host. For example, if you want to start/stop the MySQL

instance, or edit the configuration file, then executing this wizard is required. For more information, see [Section 5.2.1, "Configure Server Management Wizard"](#).

Fill out the connection details, such as the **Connection Name**, and click **Configure Server Management** to execute the Server Management wizard.

New connections are added to the Home page as a tab, and multiple connection tabs may be open simultaneously in MySQL Workbench. You may monitor and configure a MySQL server via a MySQL connection using the Navigator as described at [Section 8.2.4.3, "Navigator"](#).

5.2.1. Configure Server Management Wizard

Clicking the [+] icon from the Home page launches the **Setup New Connection** wizard. The wizard provides a MySQL connection form to create a new MySQL connection, and includes a **Configure Server Management** option as a step-by-step approach to creating a new MySQL server connection.



Note

See [Chapter 6, Getting Started Tutorial](#) for examples of how this wizard looks.

The steps presented in the wizard are as follows:

1. Test DB Connection
2. Management and OS
3. SSH Configuration
4. Windows Management
5. Test Settings
6. Review Settings
7. MySQL Config File
8. Specify Commands

Test DB Connection

On this page, MySQL Workbench tests your database connection and displays the results. If an error occurs, you are directed to view the logs, which can be done by clicking the **Show Logs** button.

Management and OS

Used to specify a remote management type and target operating system, which is available when the Host Machine is defined as a remote host.

The SSH login based management option includes configuration entries for the Operating System and MySQL Installation Type.

SSH Configuration

If you specified a Remote Host on the Specify Host Machine page, you will be presented with the Host SSH Connection page, that enables you to use SSH for the connection to the server instance. This facility enables you to create a secure connection to remotely administer and configure the server instance. You must enter the host name and user name of the account that will be used to log in to the server for

administration and configuration activities. If you do not enter the optional SSH Key for use with the server, then you will be prompted for the password when the connection is established by MySQL Workbench.



Note

This connection is to enable remote administration and configuration of the MySQL Server itself. It is not the same as the connection used to connect to a server for general database manipulation.



Note

You must use an SSH connection type when managing a remote server if you wish to start or stop the server or edit its configuration file. Other administrative functions do not require an SSH connection.

Windows Management

If a Windows server is used, then the Windows configuration parameters must be set. Windows management requires a user account with the required privileges to query the system status, and to control services. And read/write access to the configuration file is needed to allow editing of the file.

Test Settings

On the next page your settings are tested and the wizard reports back the results after attempting to connect to the server. If an error occurs, you are directed to view the logs, which can be done by clicking the Show Logs button.

MySQL Workbench must know where the MySQL Server configuration file is located to be able to display configuration information. The wizard is able to determine the most likely location of the configuration file, based on the selection made on the Operating System page of the wizard. However, it is possible to test that this information is correct by clicking the Check path and Check section buttons. The wizard then reports whether the configuration file and server configuration section can in fact be accessed. It is also possible to manually enter the location of the configuration file, and the section pertaining to MySQL Server data; these manually entered values should be tested using the buttons provided. Click the Next button to continue.

Review Settings

The modified settings may be reviewed, which also includes the default values. Check the Change Parameters checkbox if the MySQL Config File section will be edited, and then click Next to continue.

MySQL Config File

Allows configuration of the MySQL server version. It also allows the editing and validation of the configuration file path, and validation of the server instance section. Click Next to continue.

Specify Commands

This page enables you to set the commands required to start, stop, and check the status of the running server instance. It is possible to customize the commands if required, but the defaults should be suitable in most cases. The defaults are set based on the options selected in the Operating System page of the wizard. Click Next to continue.

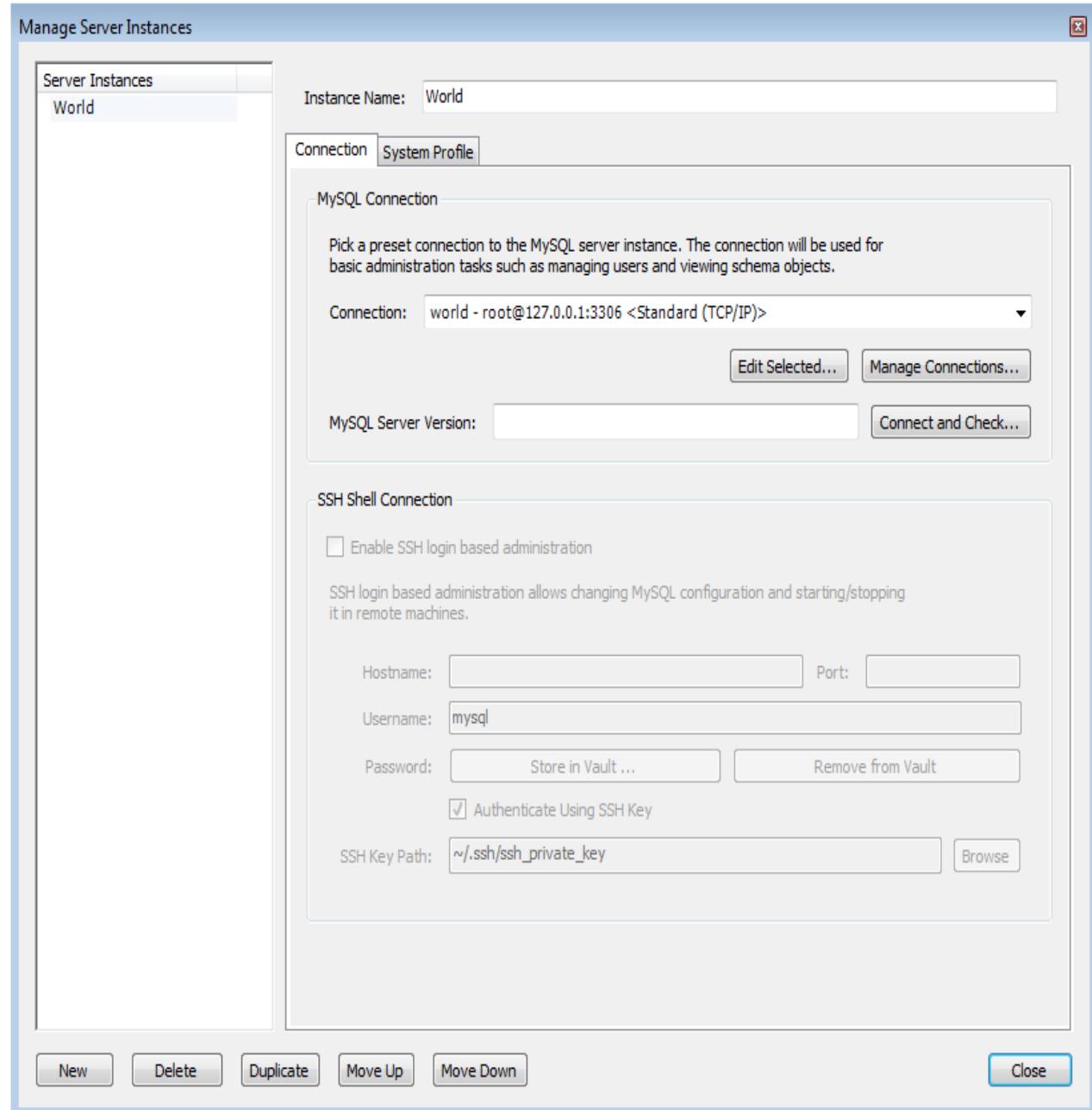
Complete Setup

On this page, you finally assign a name to the server instance. This name is used in various parts of the GUI to enable you to refer to this instance. After setting a suitable name, click Finish to save the instance.

5.3. Manage Server Instances Dialog

The Manage Server Instances dialog enables you to create, delete, and manage server instances. The **Connection** tab of the wizard enables you to select a predefined connection to use for connecting to a server to be managed. It is also possible to connect to a remote server using an SSH connection.

Figure 5.3. Manage Server Instances Dialog

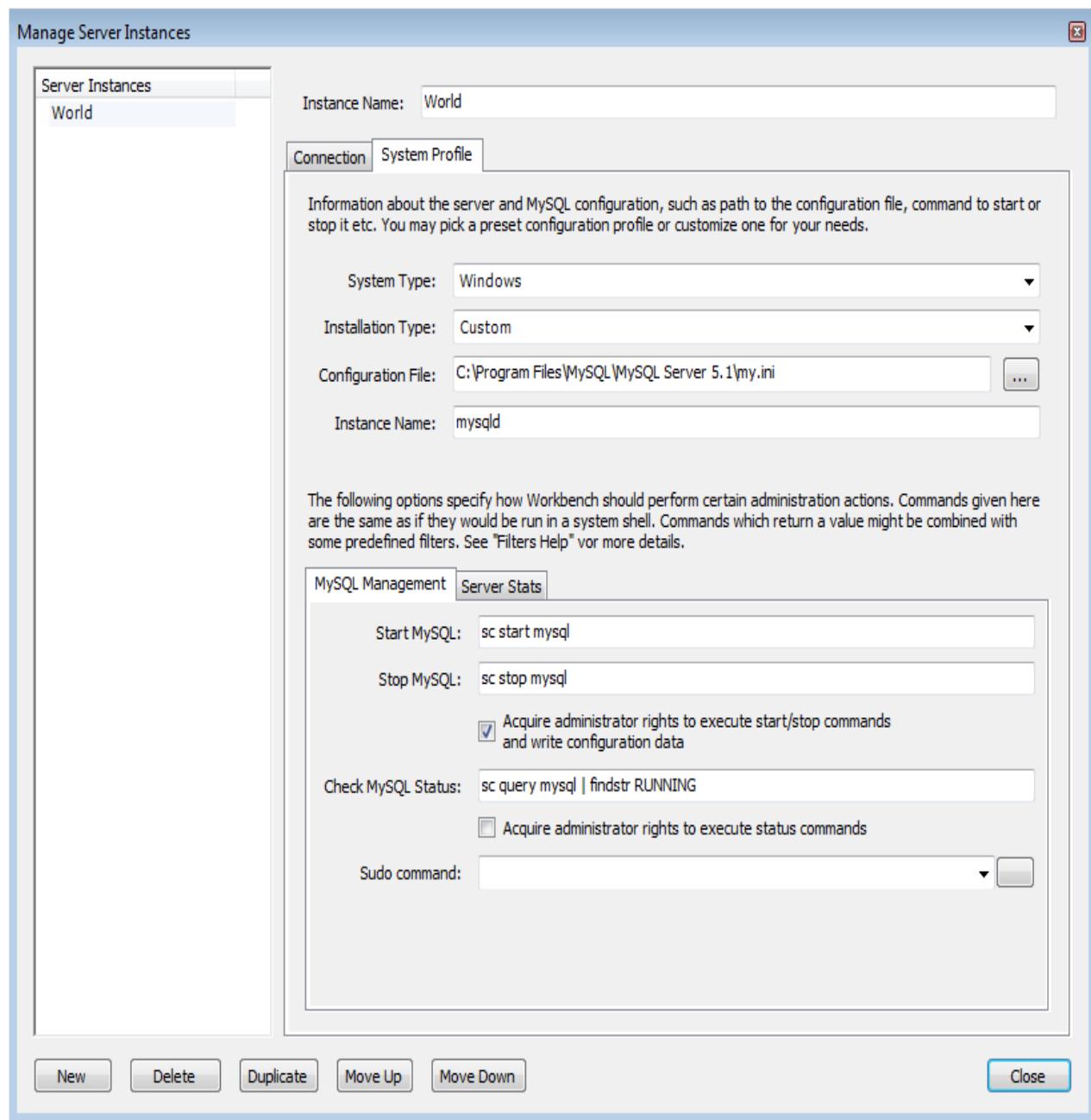


The **System Profile** tab of the wizard enables you to specify server-specific information. This is achieved primarily through selecting a Profile Template. A Profile Template contains standard information used in managing the server instance. The following Profile Templates are available:

- Fedora Linux (MySQL Package)

- Fedora Linux (Vendor Package)
- FreeBSD (MySQL Package)
- Generic Linux (MySQL tar package)
- Mac OS X (MySQL Package)
- OpenSolaris (MySQL Package)
- RHEL (MySQL Package)
- SLES (MySQL Package)
- Ubuntu Linux (MySQL Package)
- Ubuntu Linux (Vendor Package)
- Windows (MySQL 5.0 Installer Package)
- Windows (MySQL 5.1 Installer Package)
- Windows (MySQL zip package)
- Custom

After you select a profile, a number of default parameters will be set, including commands used to start and stop MySQL, commands to check server status, and the location of the `my.ini` or `my.cnf` configuration file.

Figure 5.4. Manage Server Instances Dialog

After an instance has been created, it can be launched by double-clicking its icon in the **Server Administration** panel of the **Home** page. This creates an Admin page, which has two main panels, **Server Status** and **Configuration**. The **Configuration** panel features multiple tabs: **Startup**, **Configuration**, **Accounts**, **Connections**, **Variables**, **Data Dump**, and **Logs**.

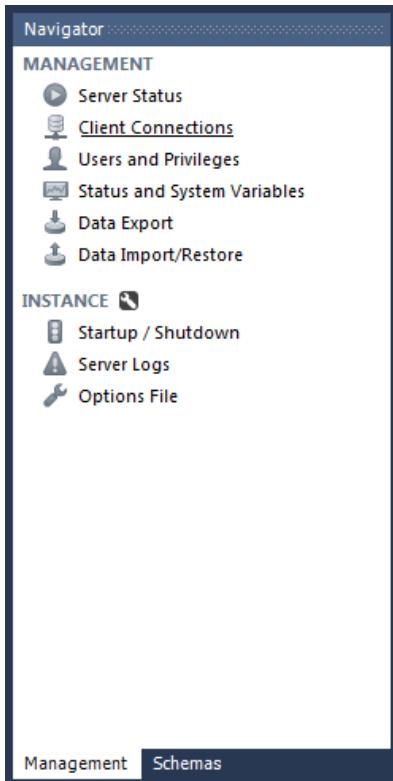
5.4. MySQL Connection Management Navigator

The Navigator panel has a **Management** tab that provides functionality to monitor and configure your MySQL connection.

**Note**

The Navigator panel also has a **Schemas** tab, for managing databases using your MySQL Connection. For information about the Schemas tab, see [Section 8.2.4.3, “Navigator”](#).

Figure 5.5. SQL Editor - Navigator Management Tab

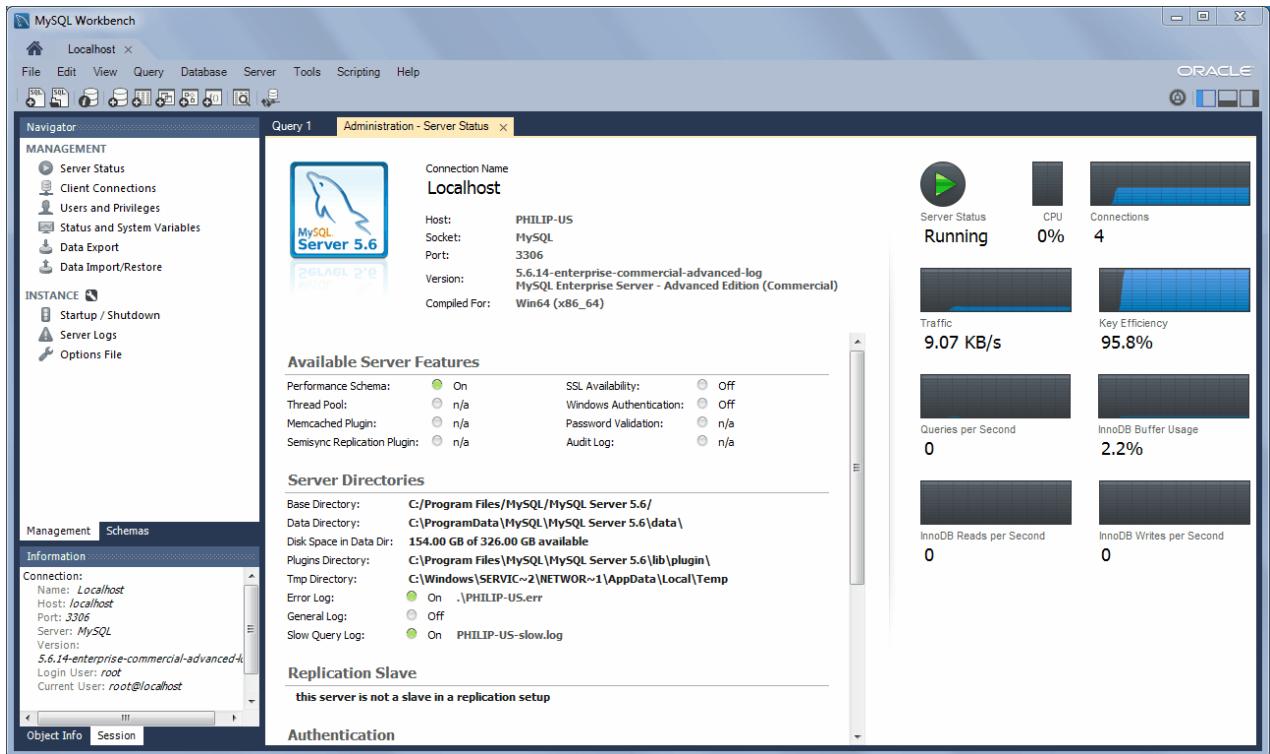


The Navigator Management tab is separated into the **MANAGEMENT** and **INSTANCE** sections, and the Commercial version of MySQL Workbench also includes the **MYSQL ENTERPRISE** section. Each section offers several actions.

5.4.1. Navigator MANAGEMENT Actions

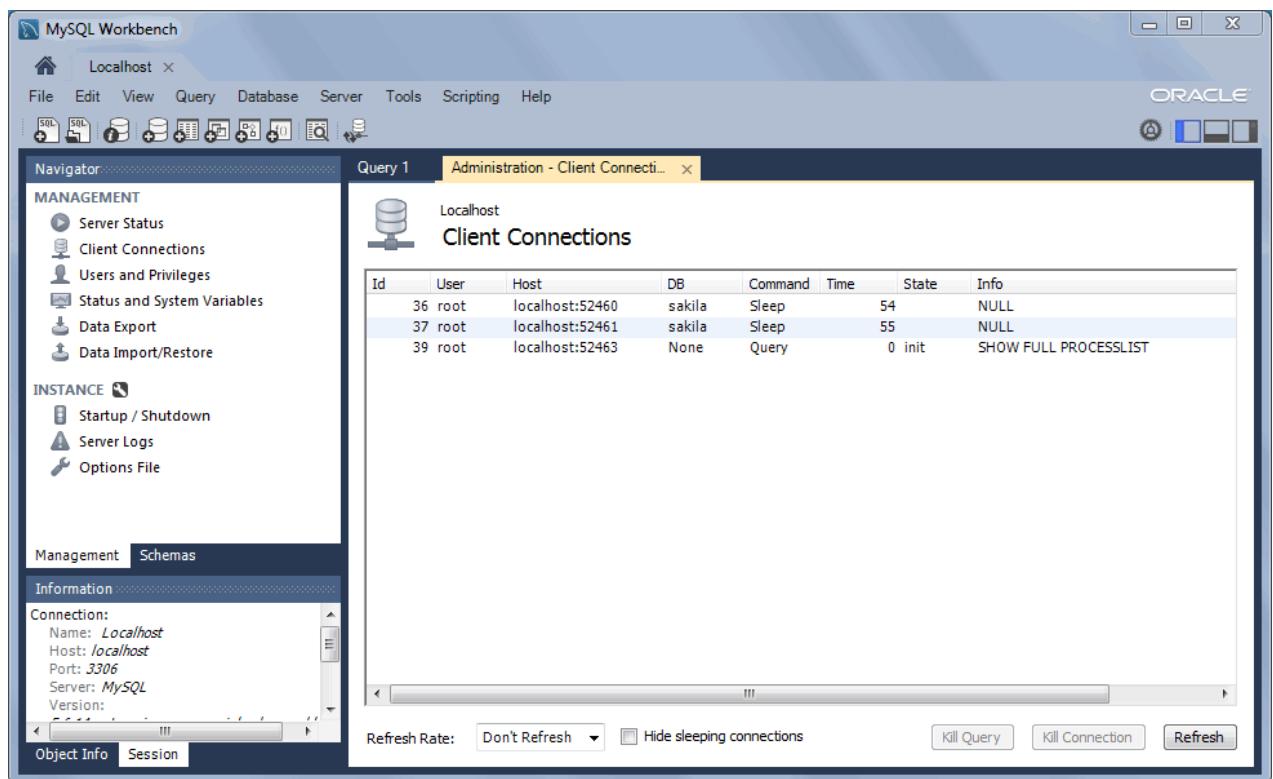
5.4.1.1. Server Status

Includes information about the connected MySQL server. This includes its running state (stopped/running), number of connections, traffic, available server features/plugins, directory paths, replication slaves, authentication keys, and SSL information.

Figure 5.6. Navigator Management: Server Status

5.4.1.2. Client Connections

The list of client connections, with the ability to kill statements and connections.

Figure 5.7. Navigator Management: Client Connections

5.4.1.3. Users and Privileges

A listing of all users and privileges that relate to the MySQL connection. You may also manage (add) user accounts, adjust privileges, and expire passwords.

The **Users and Privileges** page has several sections:

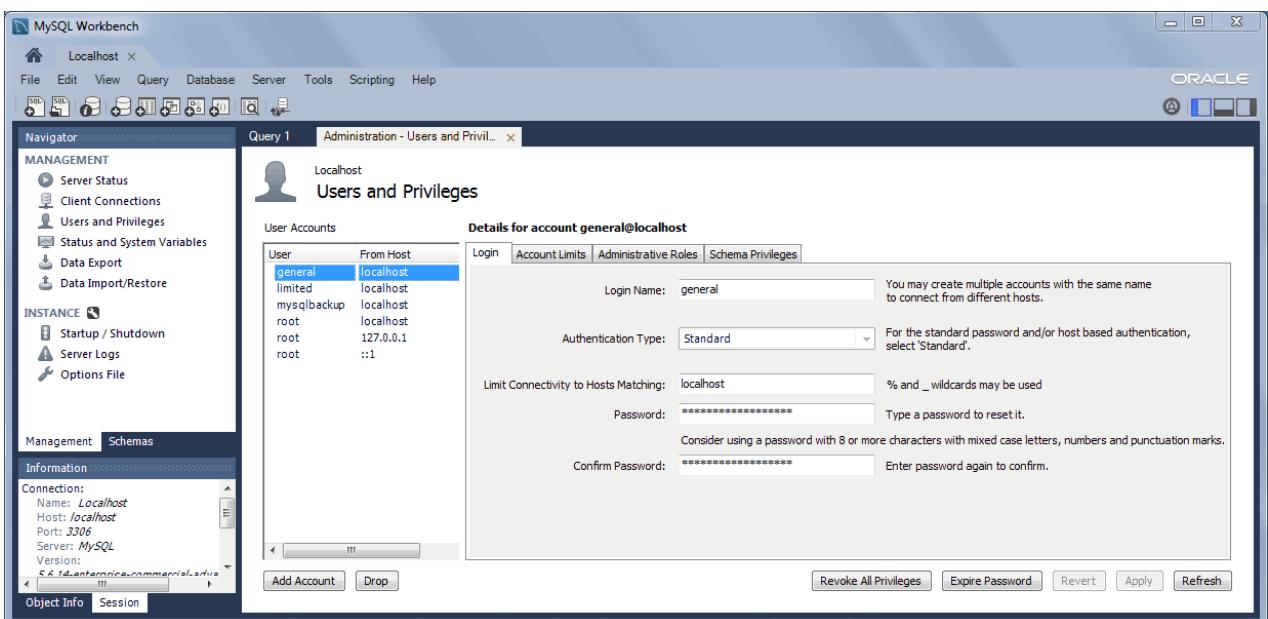
User Accounts

Lists each user account that is associated to the active MySQL connection.

Login

Login information related to the selected user account.

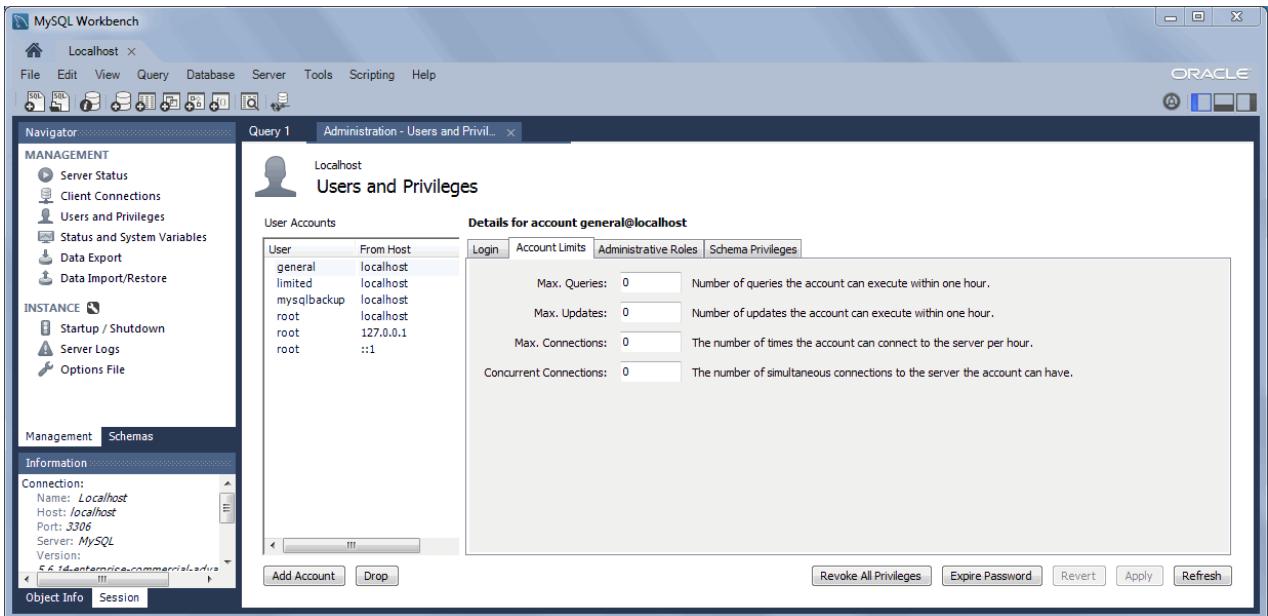
Figure 5.8. Navigator Management: User And Privileges: Login



Account Limits

Define limits for the user account, such as the maximum number of queries, updates, connections, and concurrent connections that an account can execute in one hour.

Figure 5.9. Navigator Management: User And Privileges: Account Limits



Administrative Roles

To aid in assigning privileges to MySQL Server users, MySQL Workbench introduces the concept of Administrative Roles. Roles are a quick way of granting a set of privileges to a user, based on the work the user must carry out on the server. It is also possible to assign multiple roles to a user. To assign roles, click the User Account you wish to modify, then click the **Administrative Roles** tab. Then click the check boxes

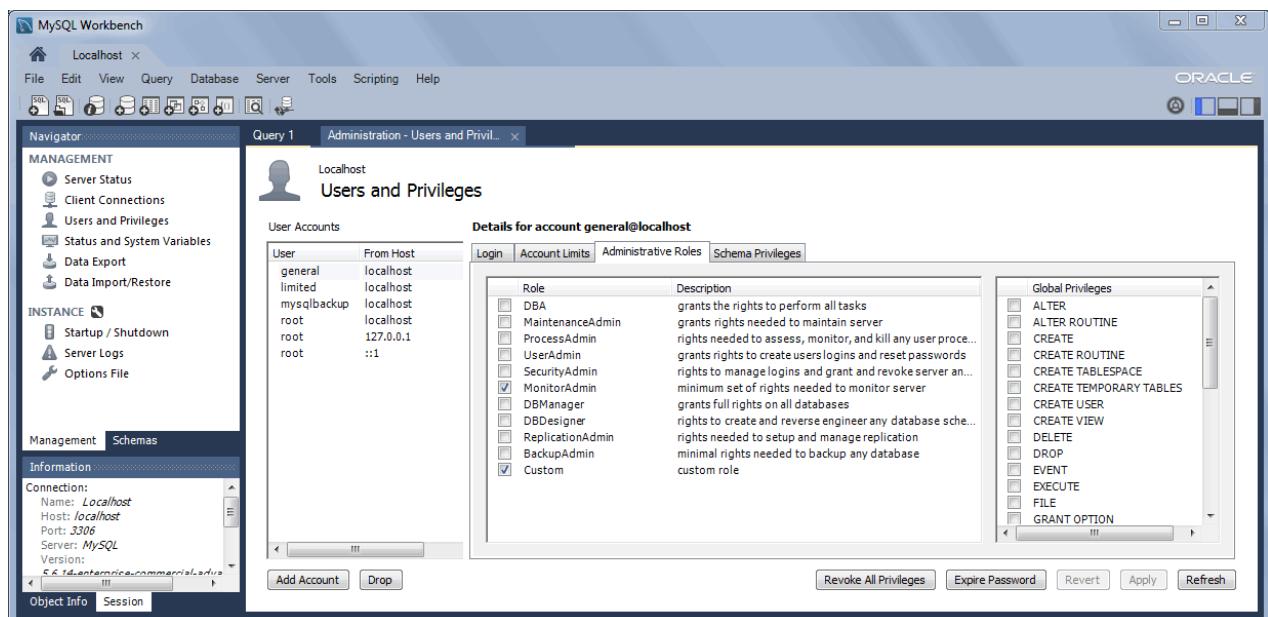
according to the roles you wish to allocate to the user. After you select a role to a user, you will see the accumulated privileges in the **Global Privileges Assigned to User** panel. For example, if you select the role **BackupAdmin**, the privileges granted include **EVENT, LOCK TABLES, SELECT, SHOW DATABASES**. If you also select the role of **ReplicationAdmin**, the list of privileges expands to include **REPLICATION CLIENT, REPLICATION SLAVE** and **SUPER**.

These roles are available:

- **DBA:** Grants all privileges
- **MaintenanceAdmin:** Grants privileges to maintain the server
- **ProcessAdmin:** Grants privileges to monitor and kill user processes
- **UserAdmin:** Grants privileges to create users and reset passwords
- **SecurityAdmin:** Grants privileges to manage logins and grant and revoke server privileges
- **MonitorAdmin:** Grants privileges to monitor the server
- **DBManager:** Grants privileges to manage databases
- **DBDesigner:** Grants privileges to create and reverse engineer any database schema
- **ReplicationAdmin:** Grants privileges to set up and manage replication
- **BackupAdmin:** Grants privileges required to back up databases
- **Custom:** Lists other (custom) privileges that are assigned to the user account

The *Password Validation Plugin* (available as of MySQL Server 5.6.6) is supported in Workbench. For information about what these settings mean, see [The Password Validation Plugin](#).

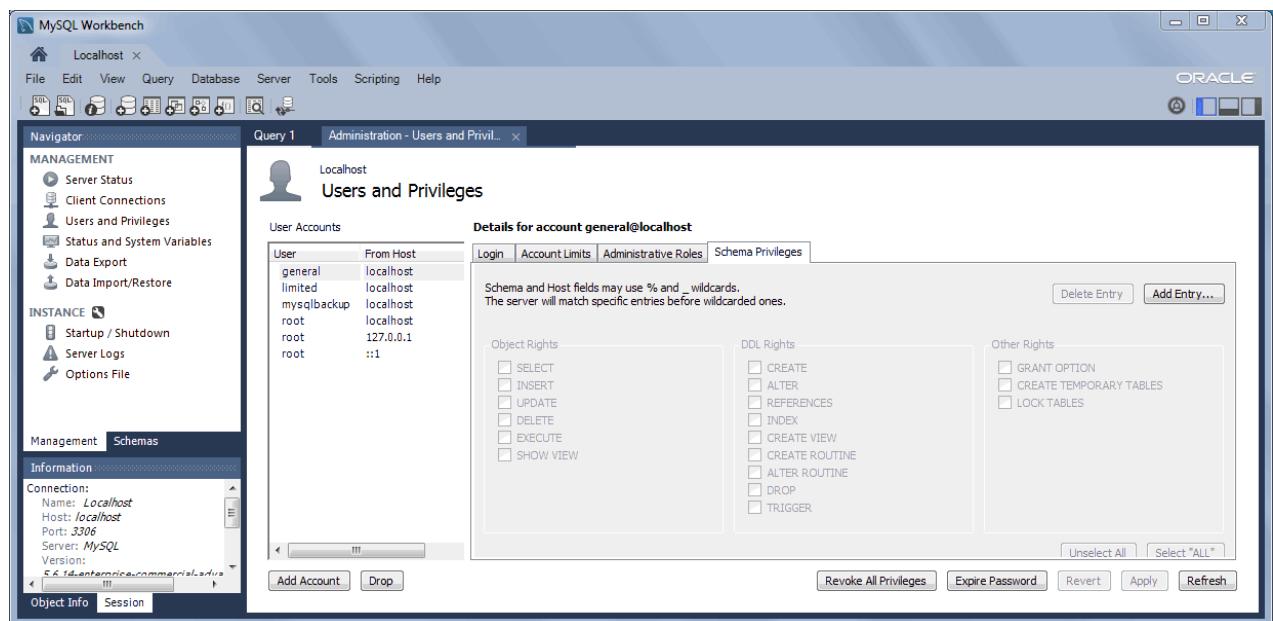
Figure 5.10. Navigator Management: User And Privileges: Administrative Roles



Schema Privileges

Additional schema privileges that the account can use. For example, the standard `mysqlbackup` user has "CREATE TEMPORARY TABLES" on the `mysql` schema.

Figure 5.11. Navigator Management: User And Privileges: Schema Privileges



5.4.1.4. Status and System Variables

Status and System Variables: Lists all server variables for the MySQL connection. You may also copy all or selected variables to your clipboard.

Figure 5.12. Navigator Management: Status Variables

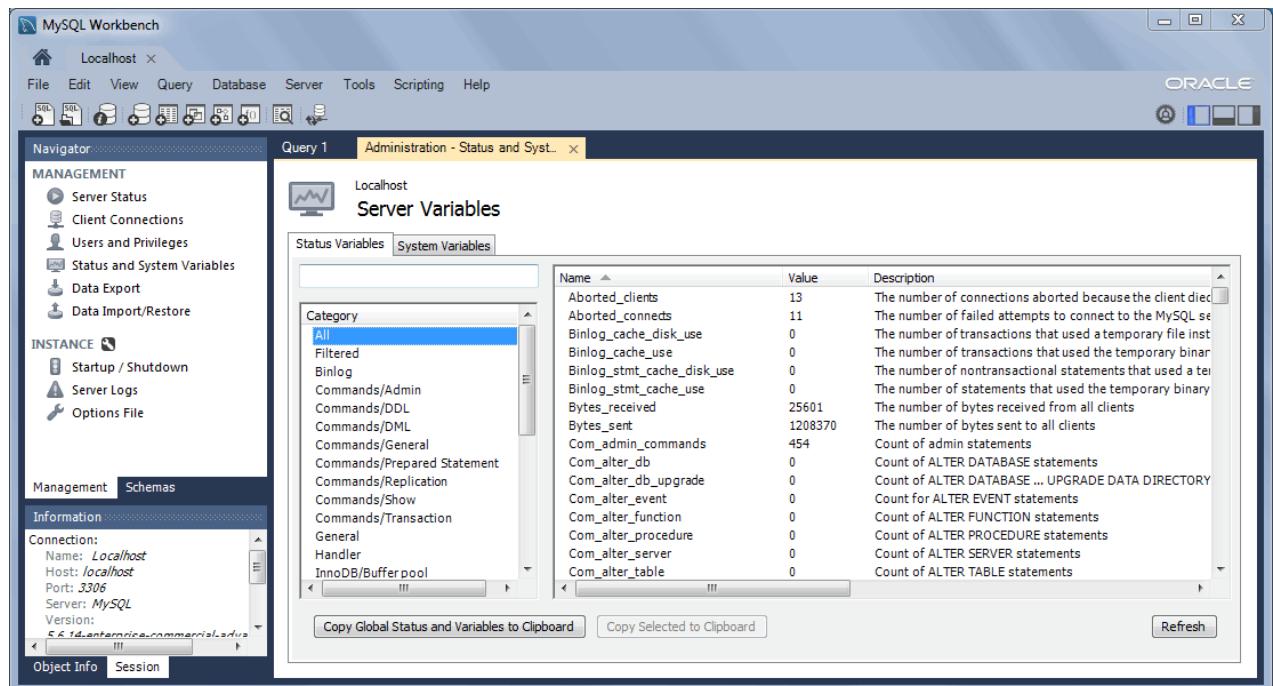
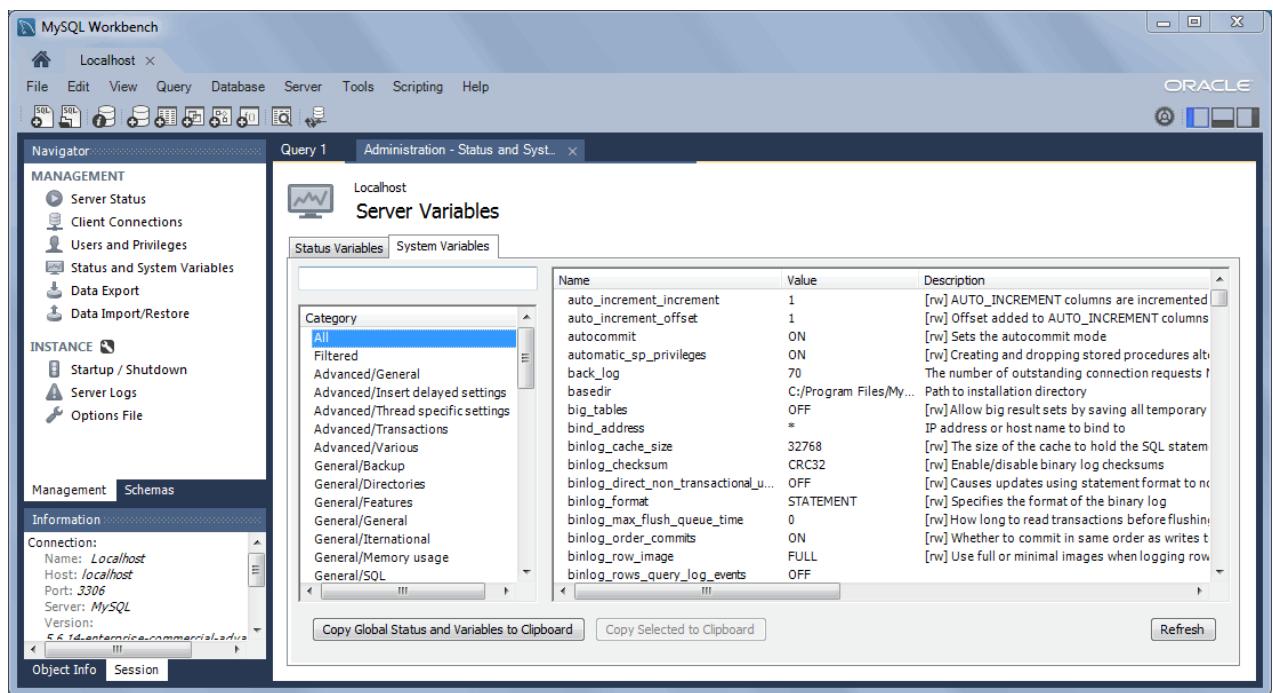
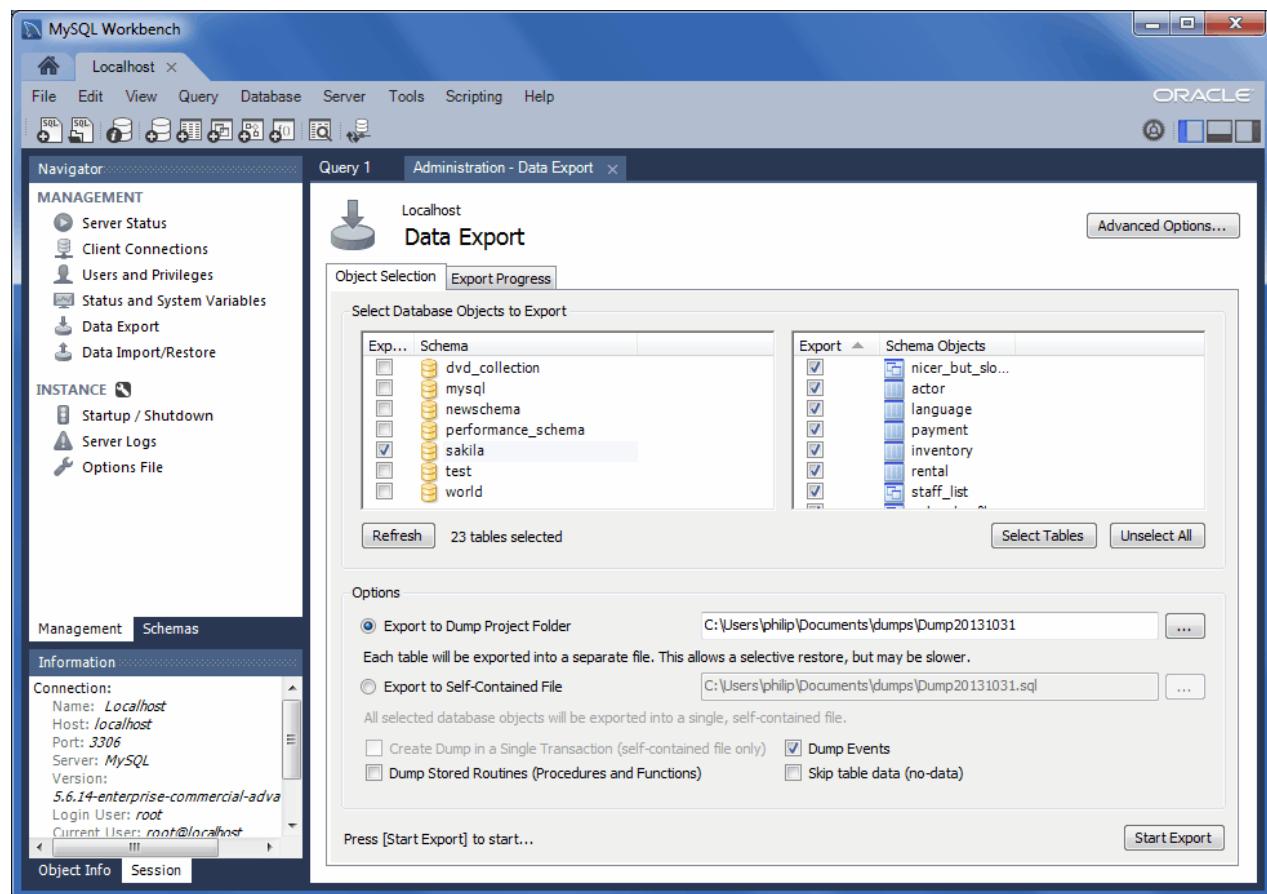


Figure 5.13. Navigator Management: Status Variables

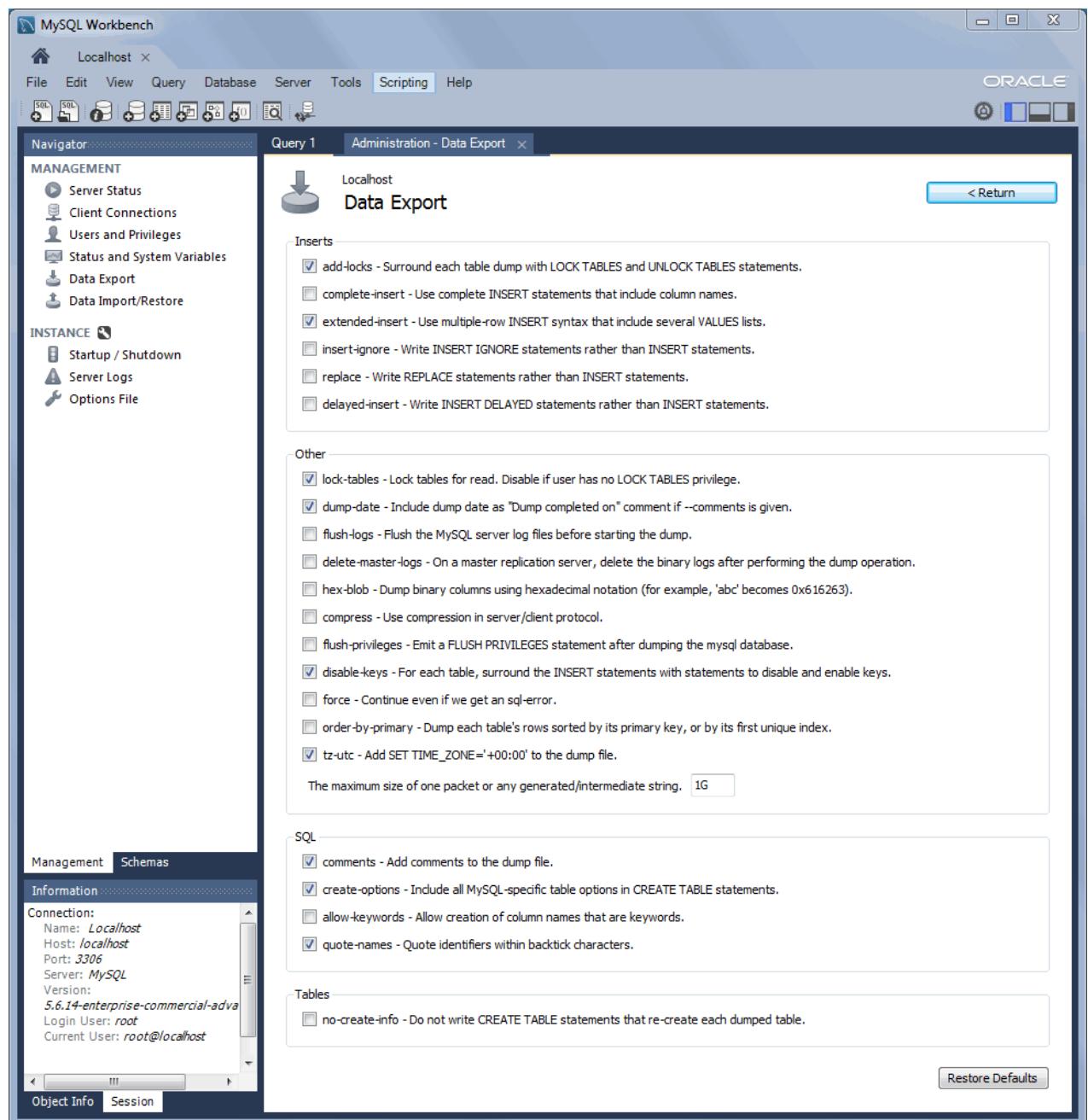
5.4.1.5. Data Export

This tab allows you to export your MySQL data. Select each schema you want to export, optionally choose specific schema objects/tables from each schema, and generate the export. Configuration options include exporting to a project folder or self-contained SQL file, optionally dump stored routines and events, or skip table data.

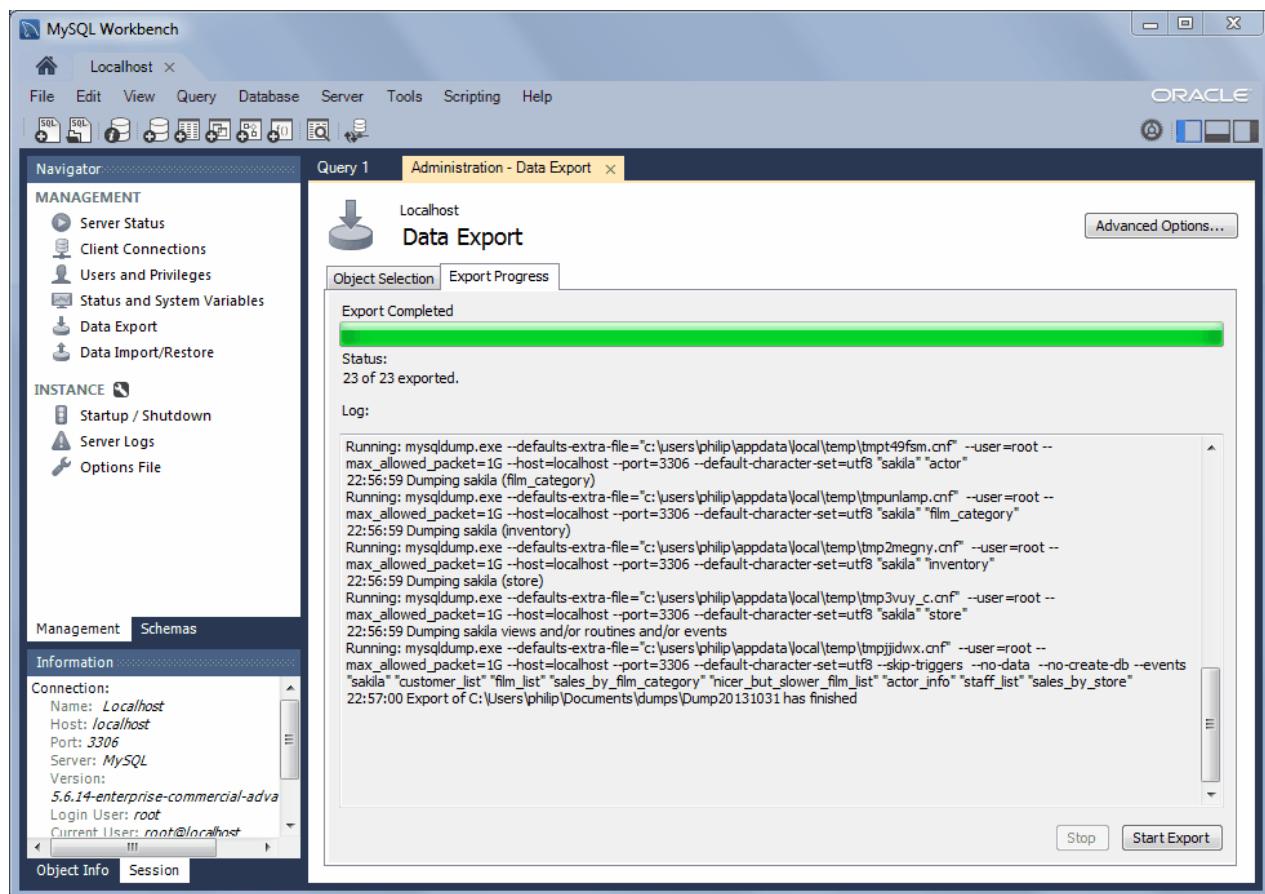
Select the Database objects to export, and configure the related options:

Figure 5.14. Navigator Management: Data Export: Object Selection

Optionally open the Advanced Options tab that allows you to refine the export operation. For example, add table locks, use replace instead of insert statements, quote identifiers with backtick characters, and more.

Figure 5.15. Navigator Management: Data Export: Advanced Options

Click Start Export to begin the export process:

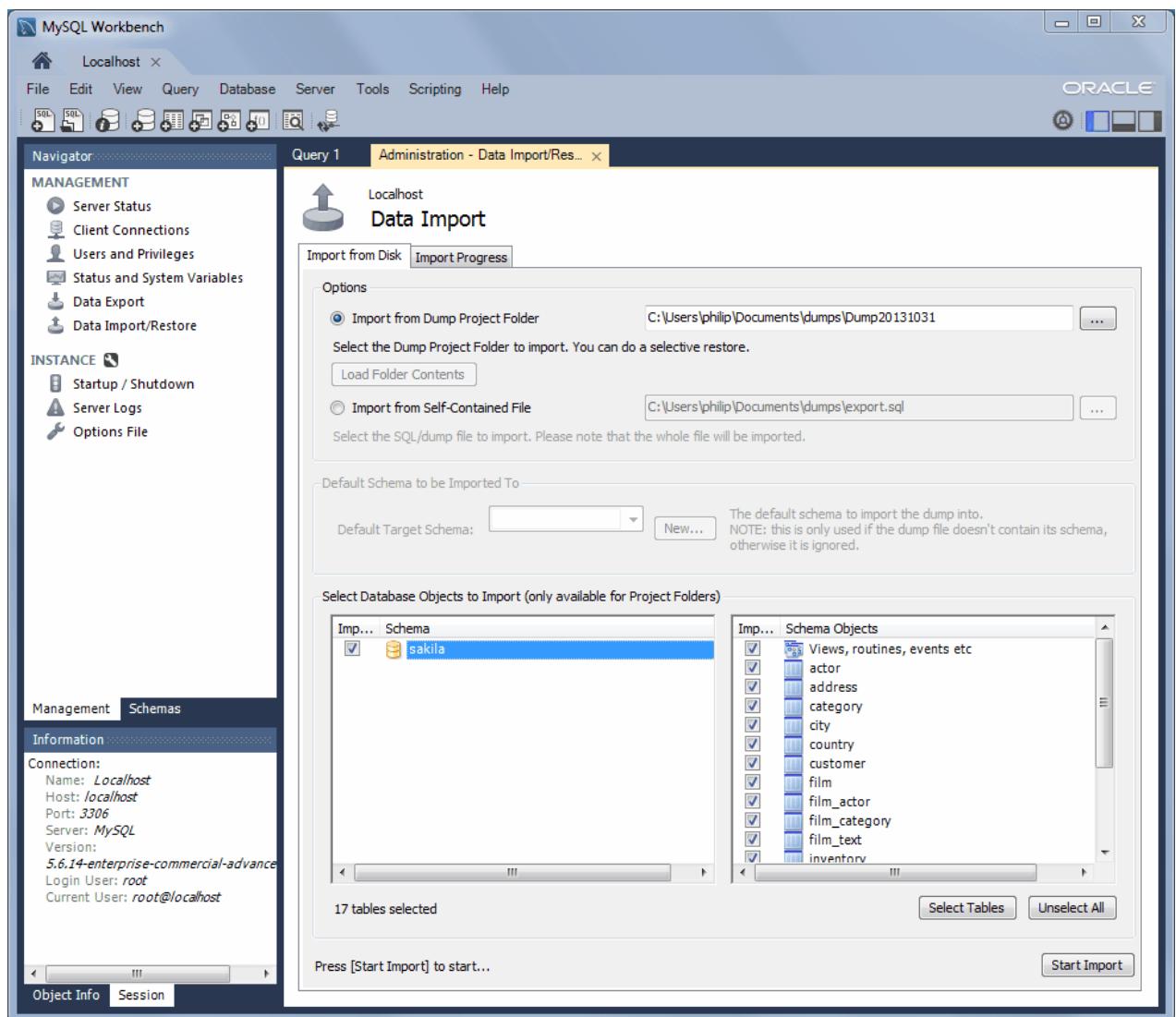
Figure 5.16. Navigator Management: Data Export: Export Progress

This functionality uses the [mysqldump](#) command.

5.4.1.6. Data Import/Restore

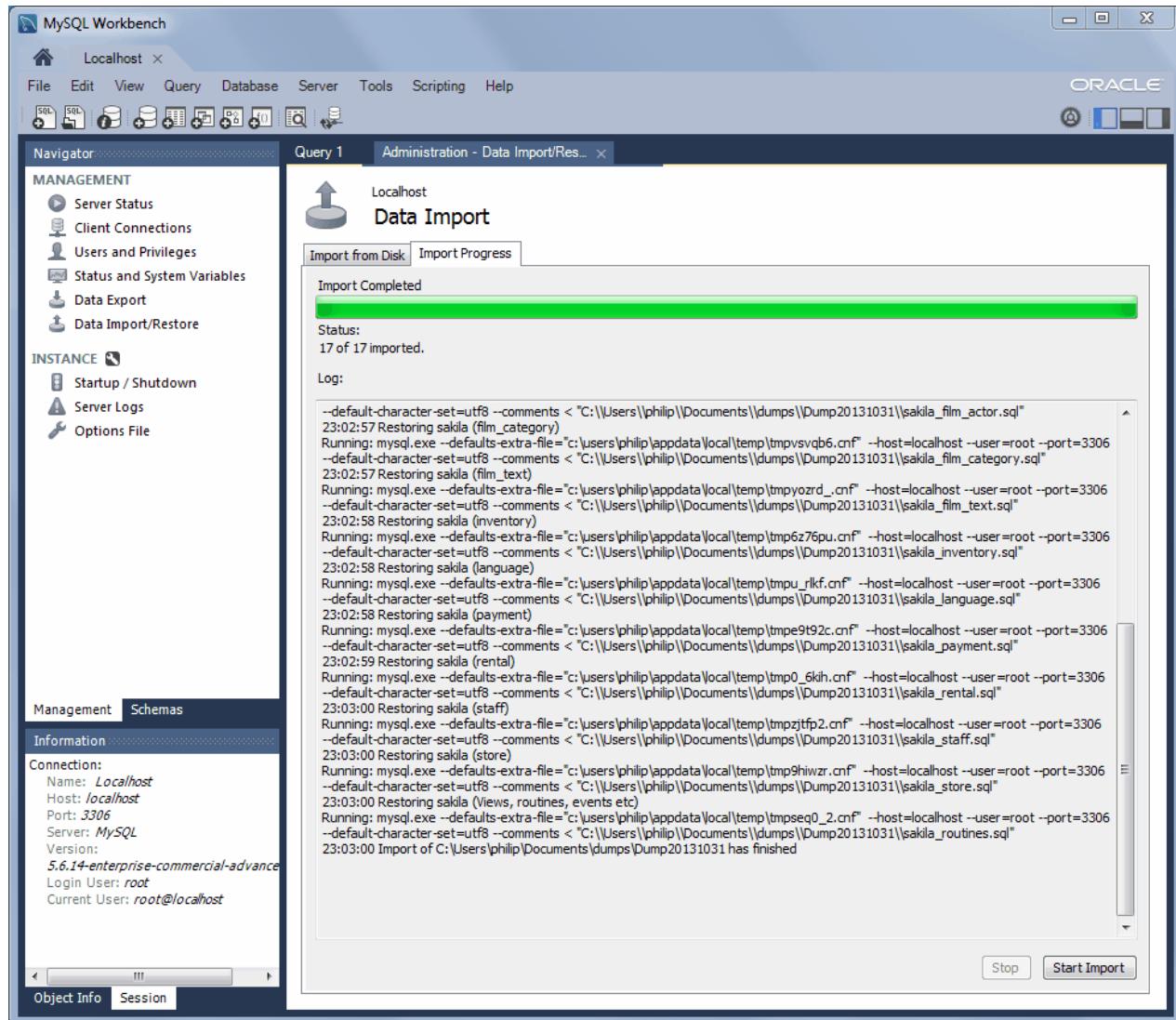
Restore exported data from the **Data Export** operation, or from other exported data from the [mysqldump](#) command.

Choose the project folder or self-contained SQL file, choose the schema that the data will be imported to, or choose **New** to define a new schema.

Figure 5.17. Navigator Management: Data Import: Import From Disk**Note**

You may only select specific data objects (tables) to import if the data export operation used project folders instead of a self-contained SQL file.

Click Start Import to begin the import process:

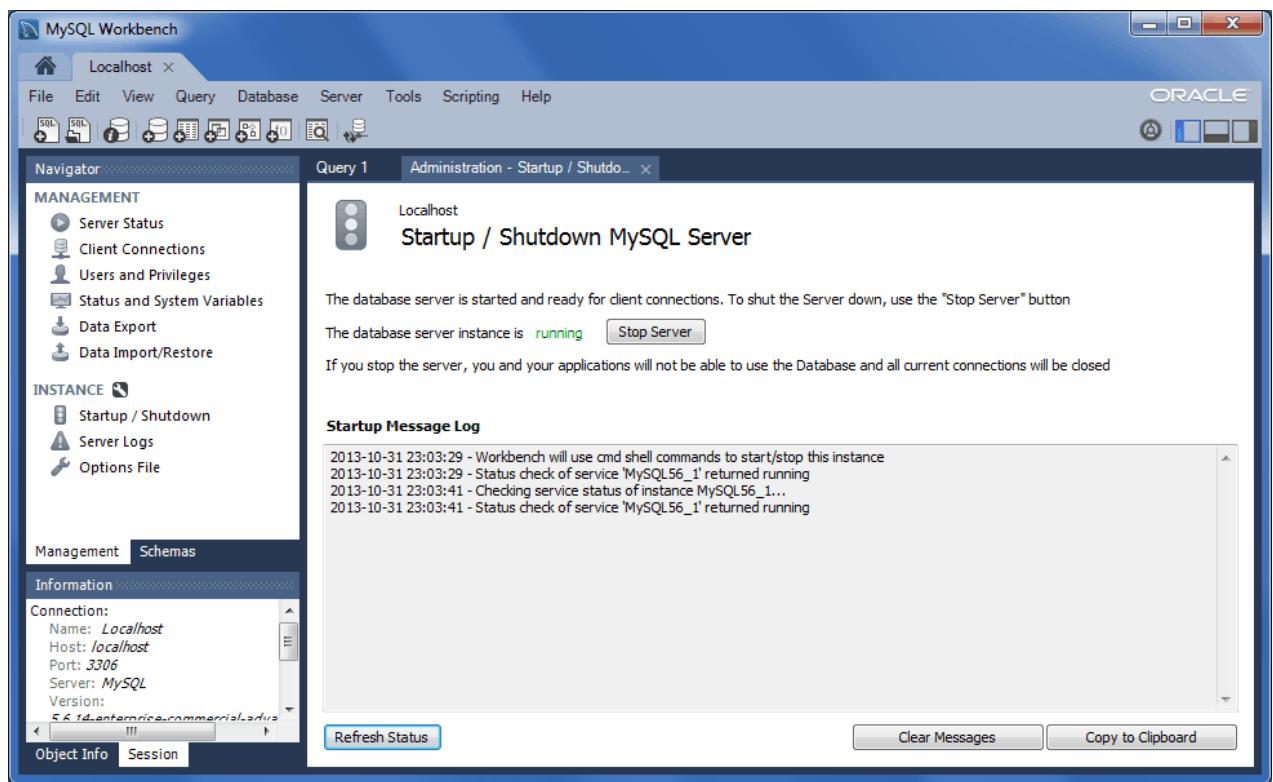
Figure 5.18. Navigator Management: Data Import: Import Progress

5.4.2. Navigator INSTANCE Actions

5.4.2.1. Startup / Shutdown

Functionality includes:

- Viewing the **Startup Message Log**
- Start up and shut down the MySQL instance
- View the current status of the MySQL instance

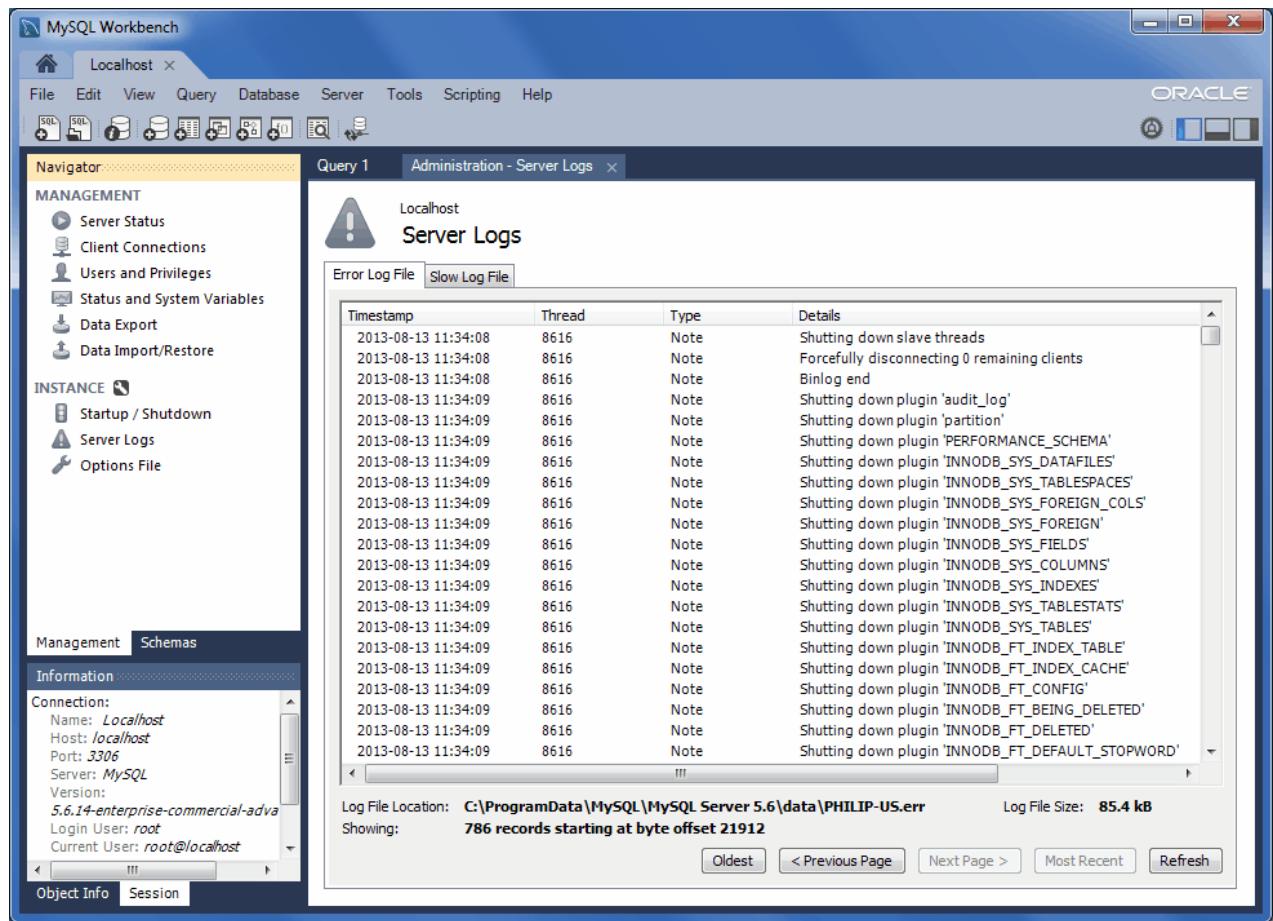
Figure 5.19. Navigator Management: Instance: Startup / Shutdown

5.4.2.2. Server Logs

The Server Logs page features two subtabs:

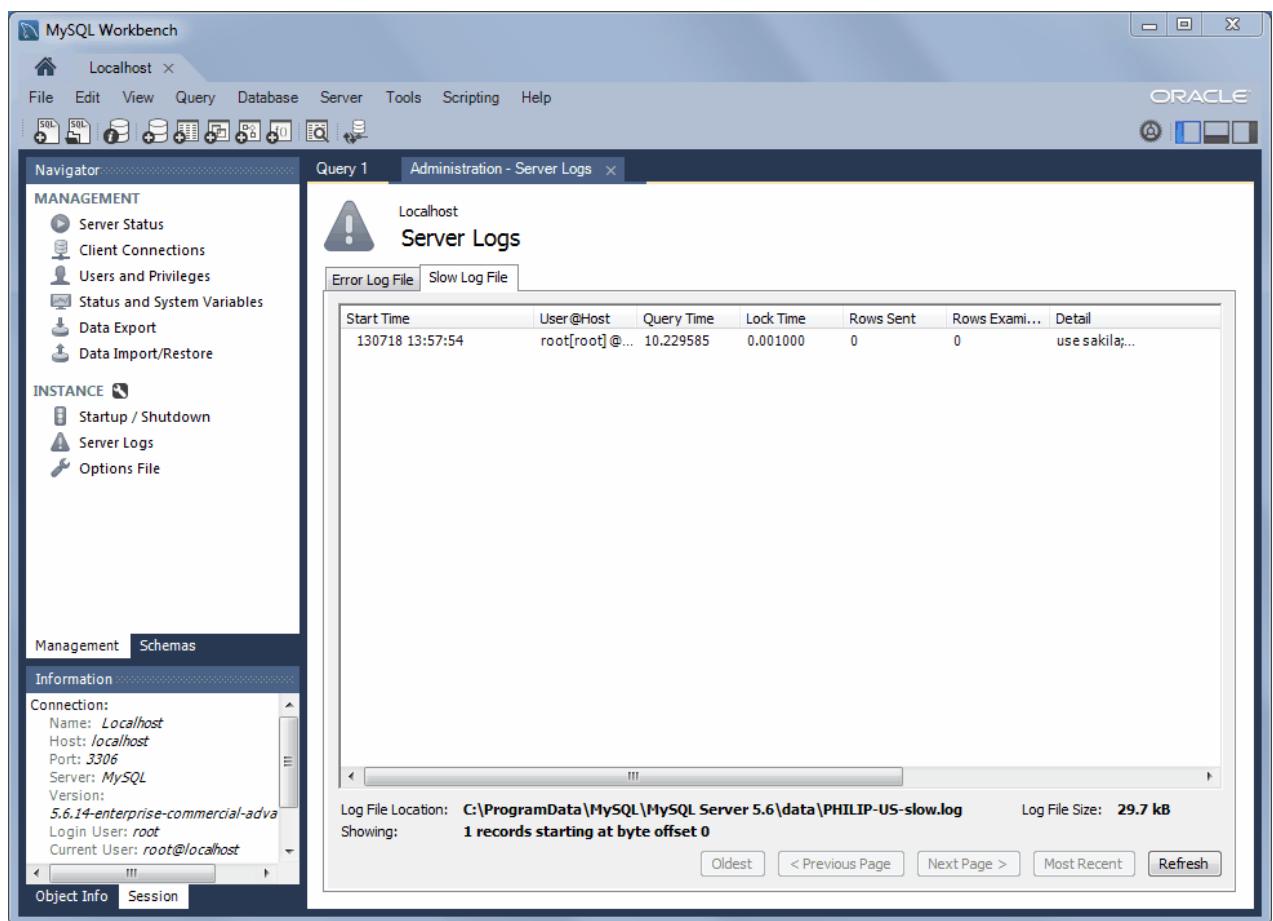
Error Log File

General MySQL errors, for more information see [The Error Log](#)

Figure 5.20. Navigator Management: Instance: Server Logs: Error Log

Slow Log File

Slow queries (when available), for more information see [The Slow Query Log](#)

Figure 5.21. Navigator Management: Instance: Server Logs: Slow Log

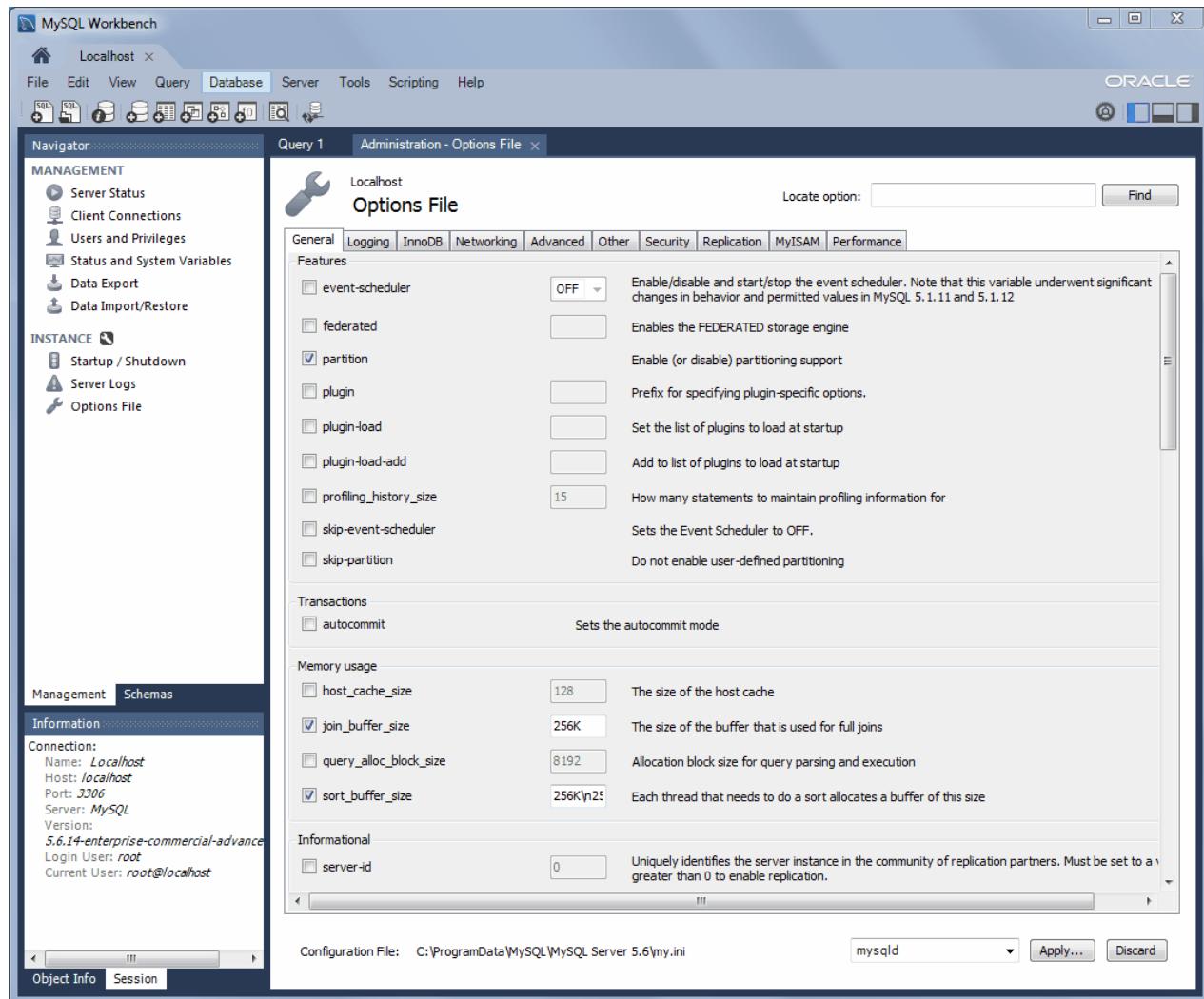
5.4.2.3. Options File

The Options tab enables you to view and edit the `my.ini` or `my.cnf` configuration file by selecting check boxes and other GUI controls. This tab also features a number of subtabs, which provide access to various sub-sections within the configuration file. The subtabs are:

- General
- Logging
- InnoDB
- Networking
- Advanced
- Other
- Security
- Replication
- MyISAM
- Performance

A screenshot with the General tab selected:

Figure 5.22. Navigator Management: Instance: Options File: General



5.4.3. Navigator MySQL Enterprise Actions

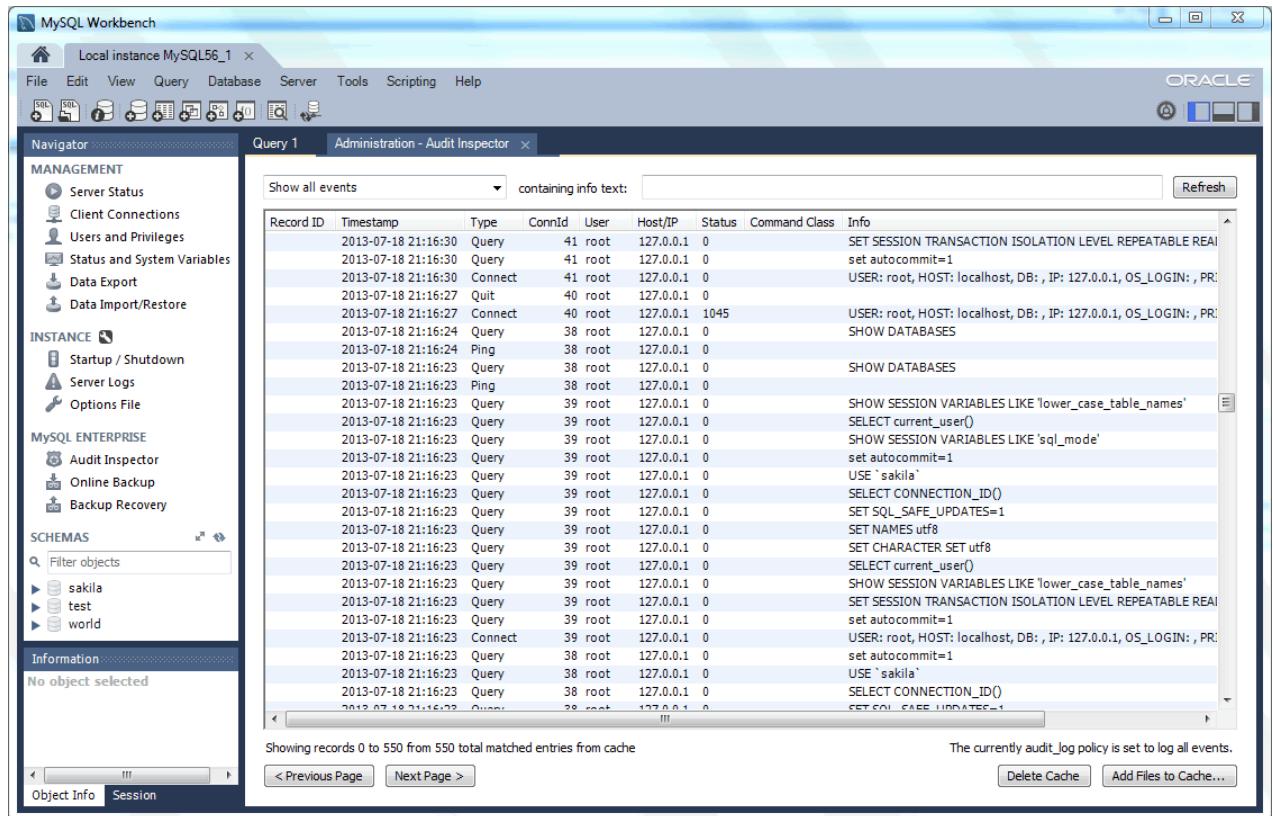


Note

This functionality provides an interface to MySQL Enterprise features, so these options are only available with the Commercial download of MySQL Workbench.

5.4.3.1. Audit Inspector

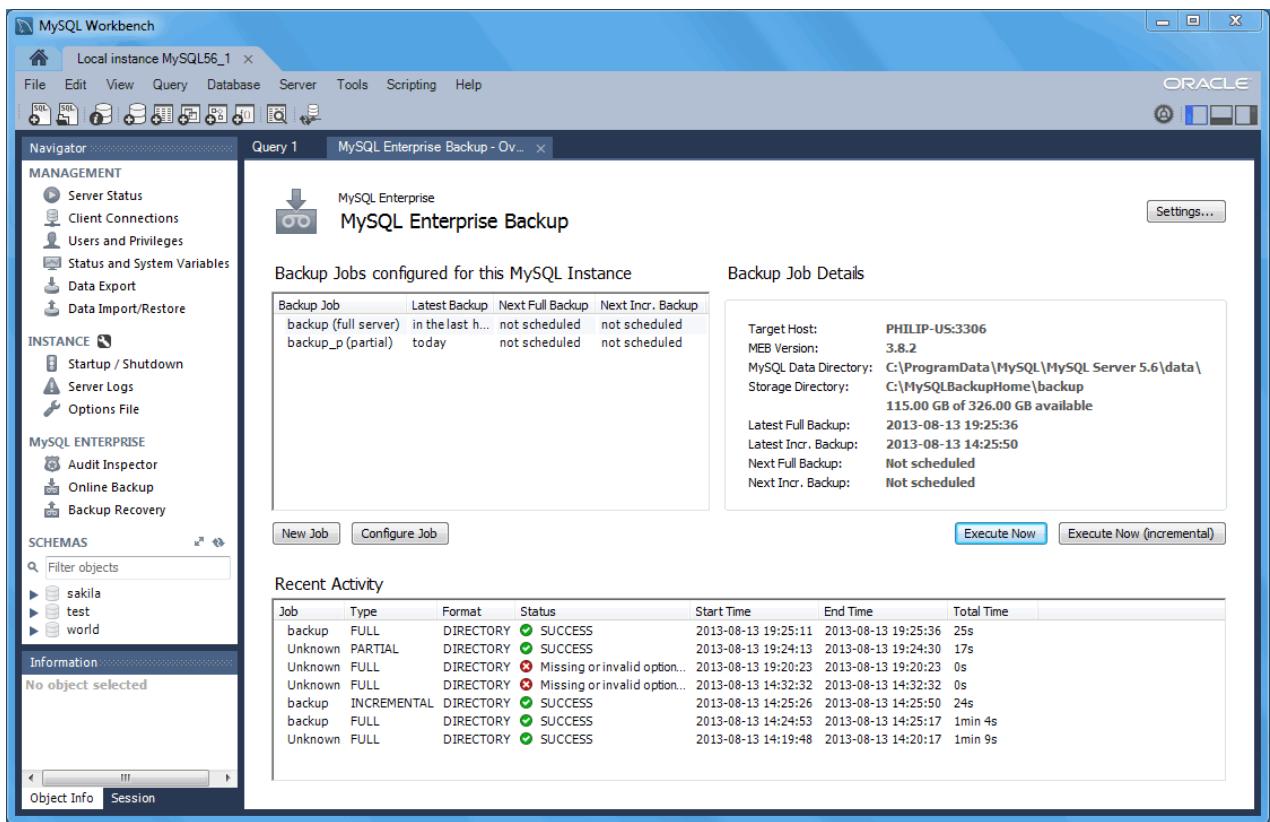
An interface to the MySQL Enterprise Audit plugin, which uses the MySQL server plugin named [audit_log](#).

Figure 5.23. Workbench: Audit Inspector

For information about how the Audit Inspector UI functions in MySQL Workbench, see [Section 11.1, “MySQL Audit Inspector Interface”](#).

5.4.3.2. Online Backup

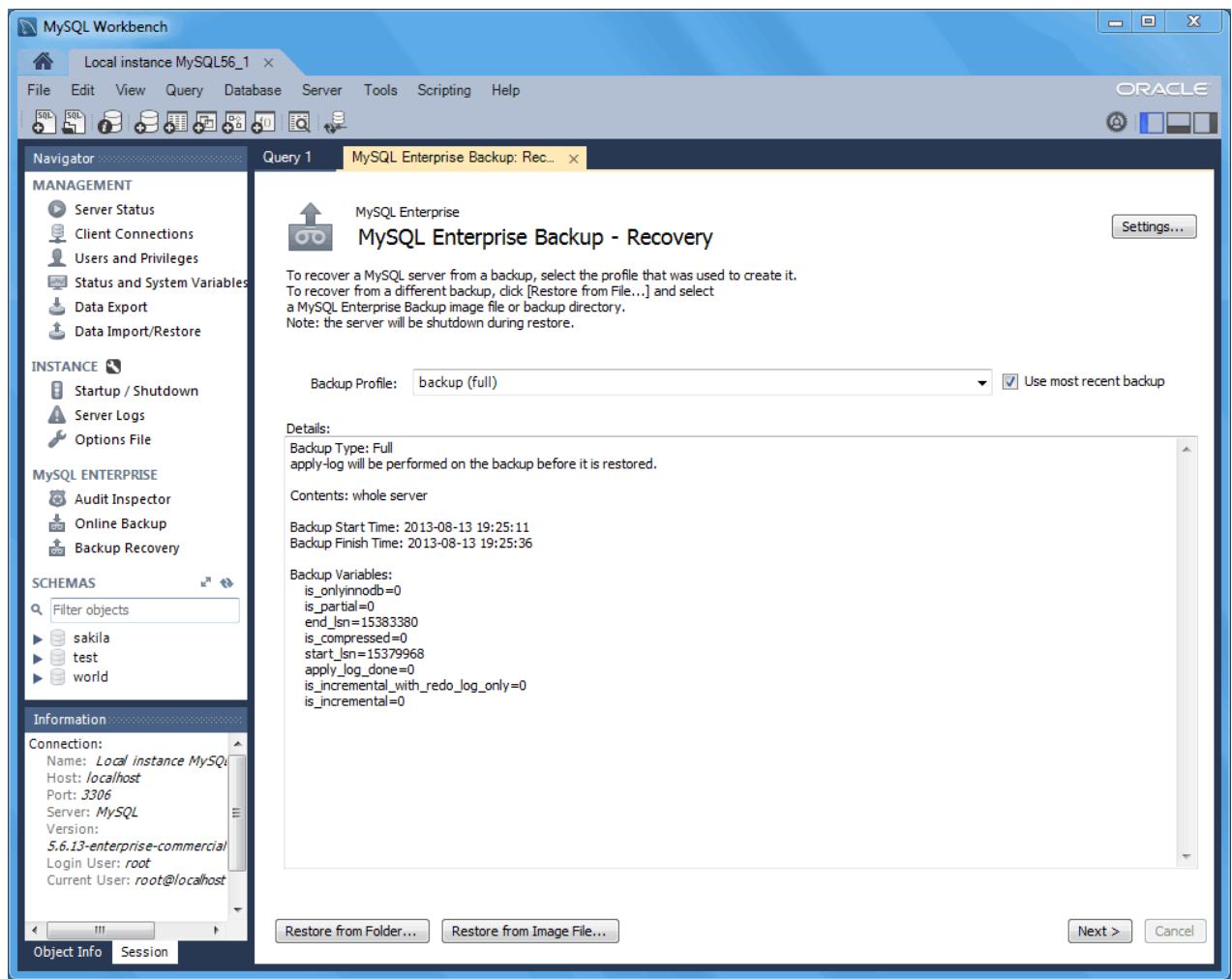
An interface to the MySQL Enterprise Backup tool. Schedule new or configure backup jobs, view the recent MySQL Enterprise Backup activity, execute full or incremental backups, and view details for each backup operation.

Figure 5.24. Workbench: MySQL Enterprise Backup

For information about how the MySQL Enterprise Backup UI functions in MySQL Workbench, see [Section 11.2, “MySQL Enterprise Backup Interface”](#).

5.4.3.3. Backup Recovery

Restore backups from a folder or image file.

Figure 5.25. Workbench: Backup Recovery: Main page

For information about how the Backup Recovery UI functions in MySQL Workbench, see [Section 11.3, “MySQL Workbench Backup Recovery”](#).

Chapter 6. Getting Started Tutorial

Table of Contents

6.1. Administering a MySQL Server	55
6.2. Creating a Model	67
6.3. Adding Data to Your Database	75

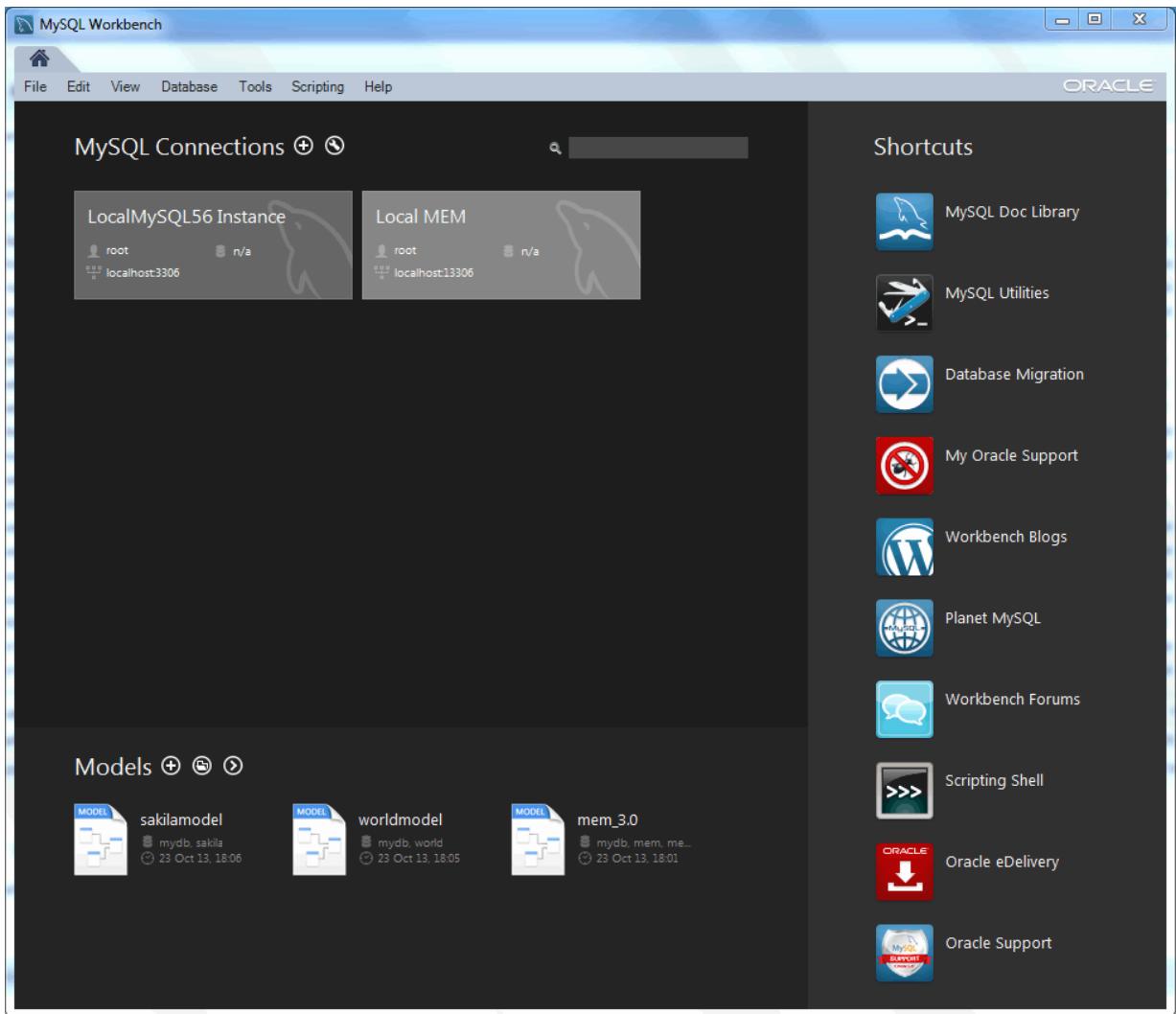
This tutorial provides a quick hands-on introduction to using MySQL Workbench for beginners. If you have used MySQL Workbench before you can safely skip this tutorial.

This tutorial uses a locally installed MySQL Server. If you only have access to a remote MySQL server, you must enter appropriate connection parameters as necessary. Although this tutorial requires MySQL Workbench 6.0.0 or above, the concepts are similar in MySQL Workbench 5.2.x. It is assumed that you have a basic understanding of MySQL concepts. This tutorial demonstrates the procedures on Microsoft Windows, but they are the same for all supported platforms.

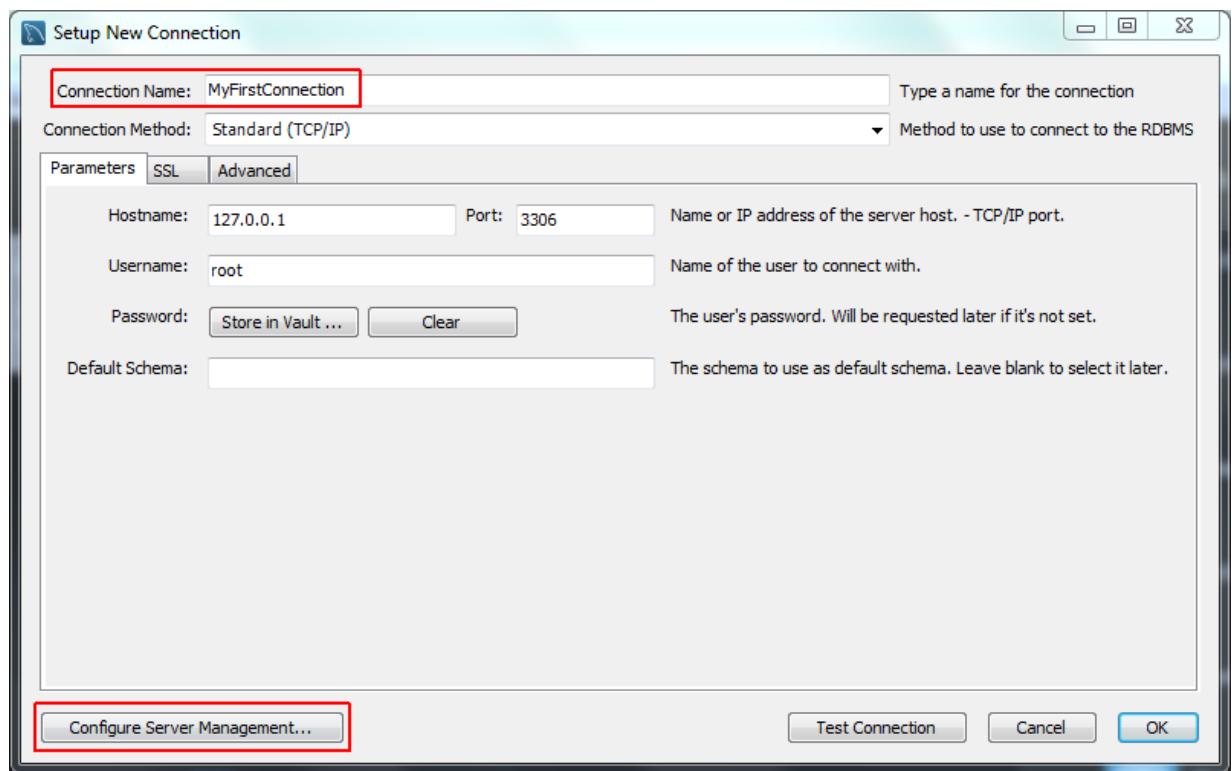
6.1. Administering a MySQL Server

In this section, you will use MySQL Workbench to carry out administrative functions, such as starting and stopping the server.

1. Launch MySQL Workbench. You will be presented with the Home window.

Figure 6.1. Getting Started Tutorial - Home Window

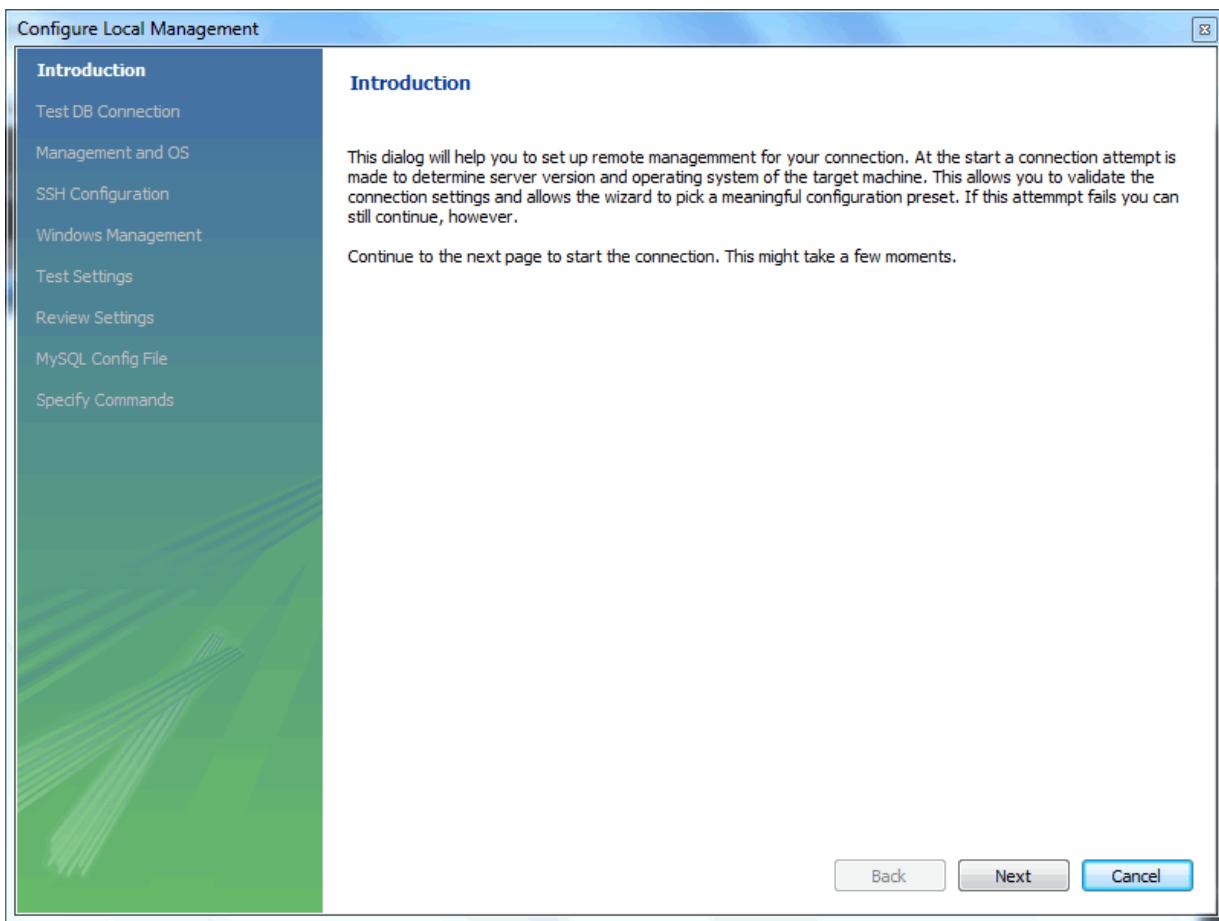
2. To administer your MySQL Server, you must first create a MySQL Connection. Our example already has two connections created, but let us create a new connection. From the MySQL Workbench Home window, click the **[+]** icon near the **MySQL Connections** label. This opens the **Setup New Connection** wizard.
3. Define the **Connection Name** value, which we will set to "MyFirstConnection" in this example.

Figure 6.2. Getting Started Tutorial - Setup New Connection: MyFirstConnection

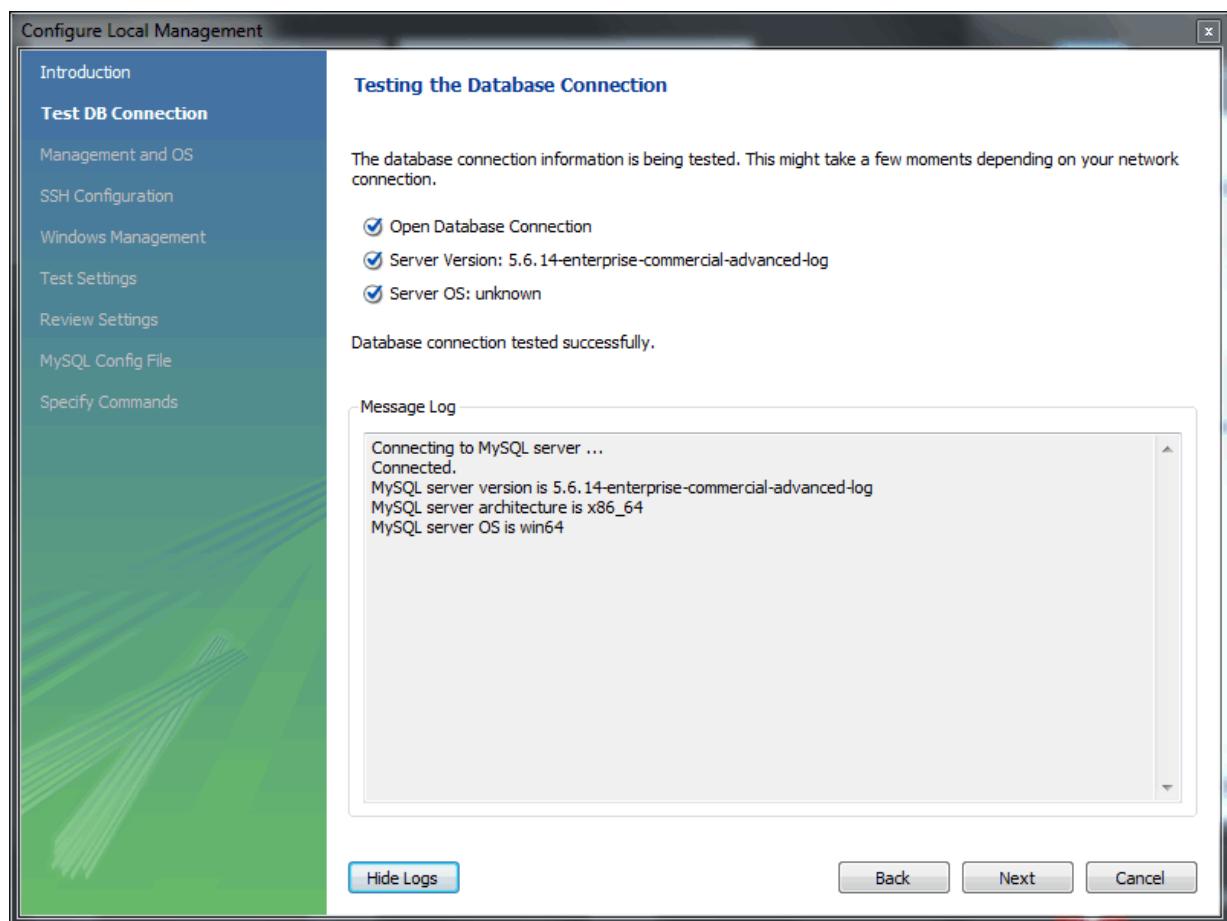
The default connection values are for a typical local setup, so check them and enter the appropriate values. If you are unsure, click the **Test Connection** button to check the connection parameters. Do not press **OK**.

Next, click **Configure Server Management...**, which opens up the **Configure Local Management** wizard:

4. Read the **Configure Local Management** introduction, and press **Next** to begin defining the new connection parameters.

Figure 6.3. Getting Started Tutorial - Configure Local Management Introduction

5. The connection will now be tested. You should see that the connection was successful. If not, click Back and check that you have entered the information correctly.

Figure 6.4. Getting Started Tutorial - Test Database Connection

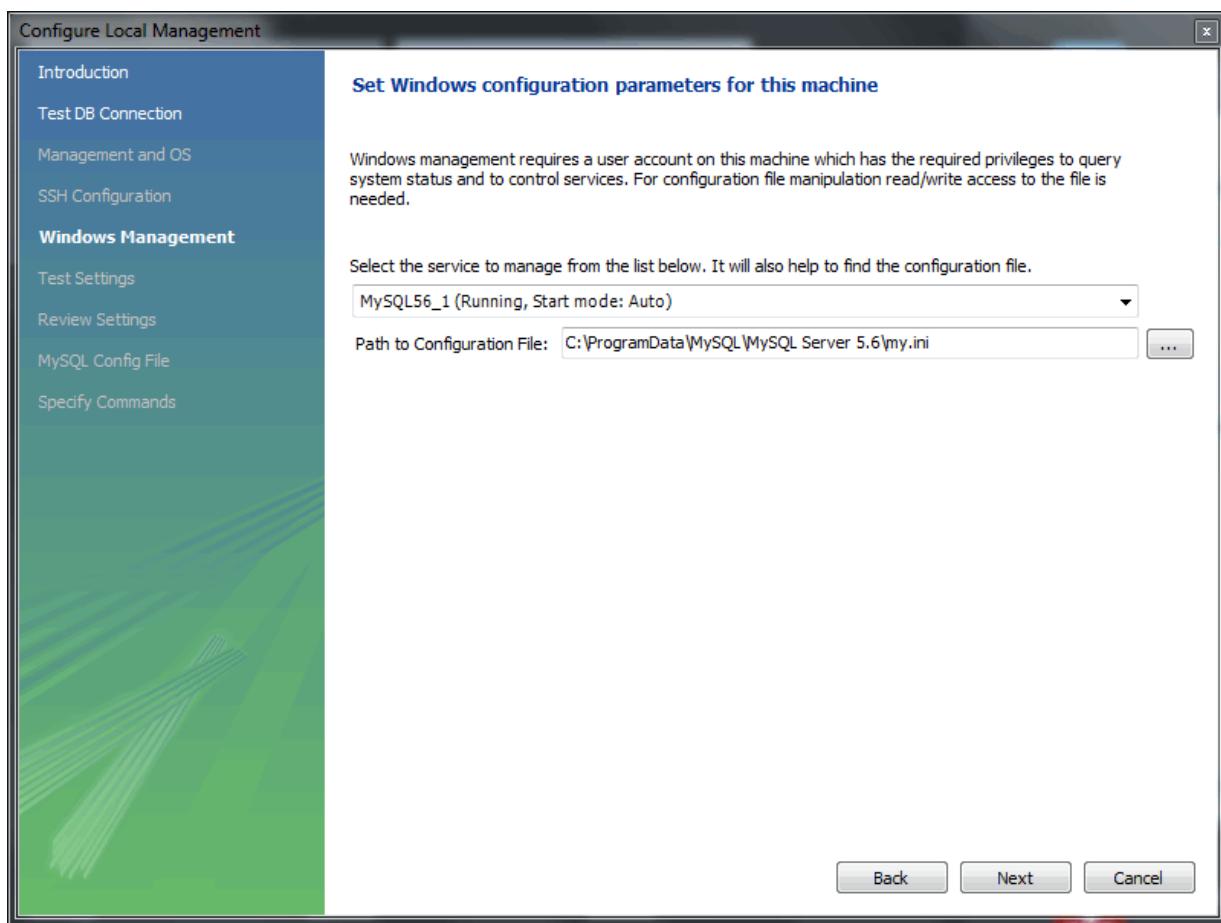
Toggle the Show Logs to view additional details about the tested connection, then click Next.

6. Optionally, you may configure a method for remote management if a Remote Host was specified. Setting these options enables MySQL Workbench to determine the location of configuration files, and the correct start and stop commands to use for the connection.

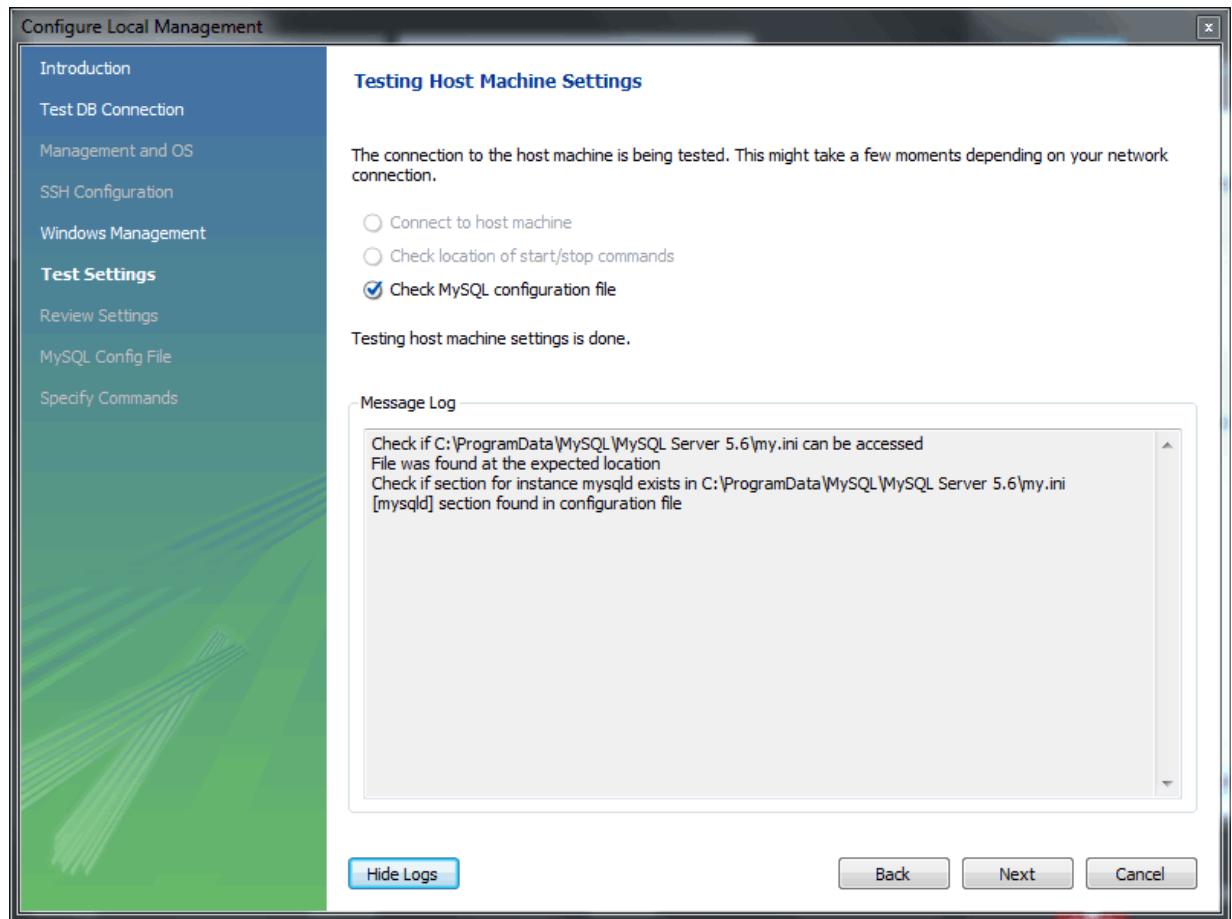
SSH login based management and Native Windows remote management types are available. The Operating System and MySQL Installation Type are configured for the SSH login variant.

We are creating a local MySQL connection in this tutorial, so are skipping the **Management and OS** and **SSH Configuration** options, as they are used for configuring a remote MySQL connection.

7. On Microsoft Windows, select the appropriate MySQL service for the MySQL connection.

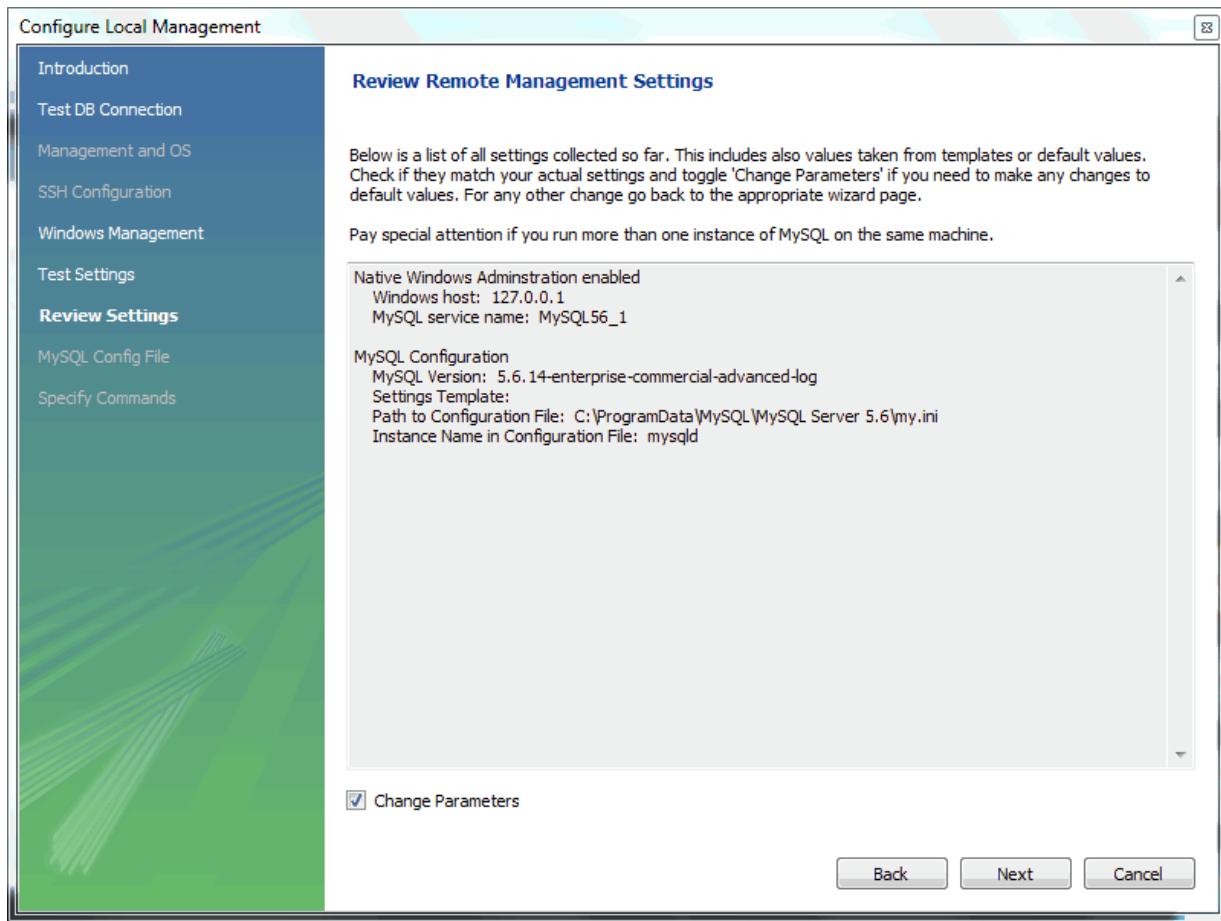
Figure 6.5. Getting Started Tutorial - Windows Management

8. The wizard will now check its ability to access the start and stop commands, and check access to the MySQL Server configuration file.

Figure 6.6. Getting Started Tutorial - Test Settings

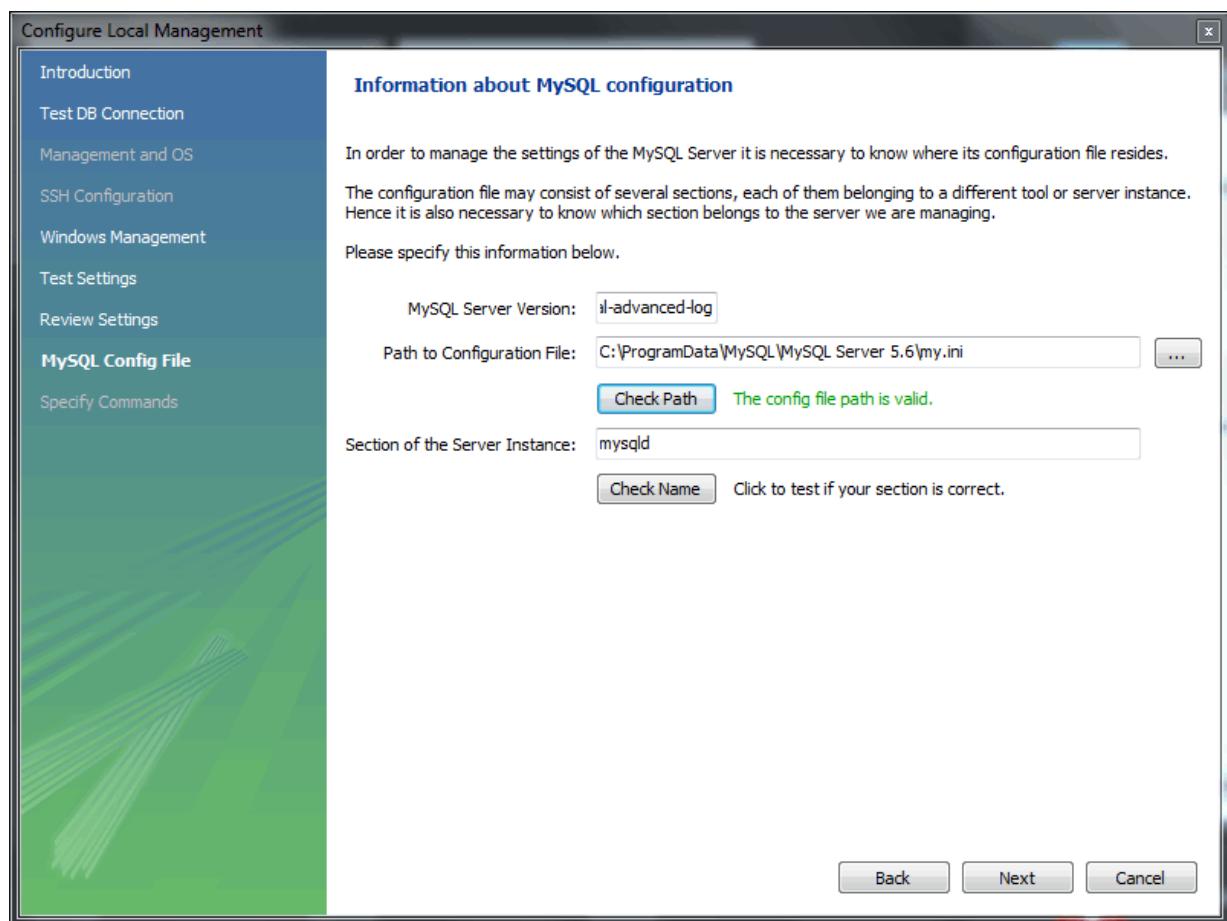
9. You now have a chance to review the configuration settings. The information displayed varies slightly depending on platform, connection method, and installation type.

At the **Review Settings** prompt, choose "I'd like to review the settings again" to review the settings. Choosing "Continue" closes the "Configure Server Management" dialog.

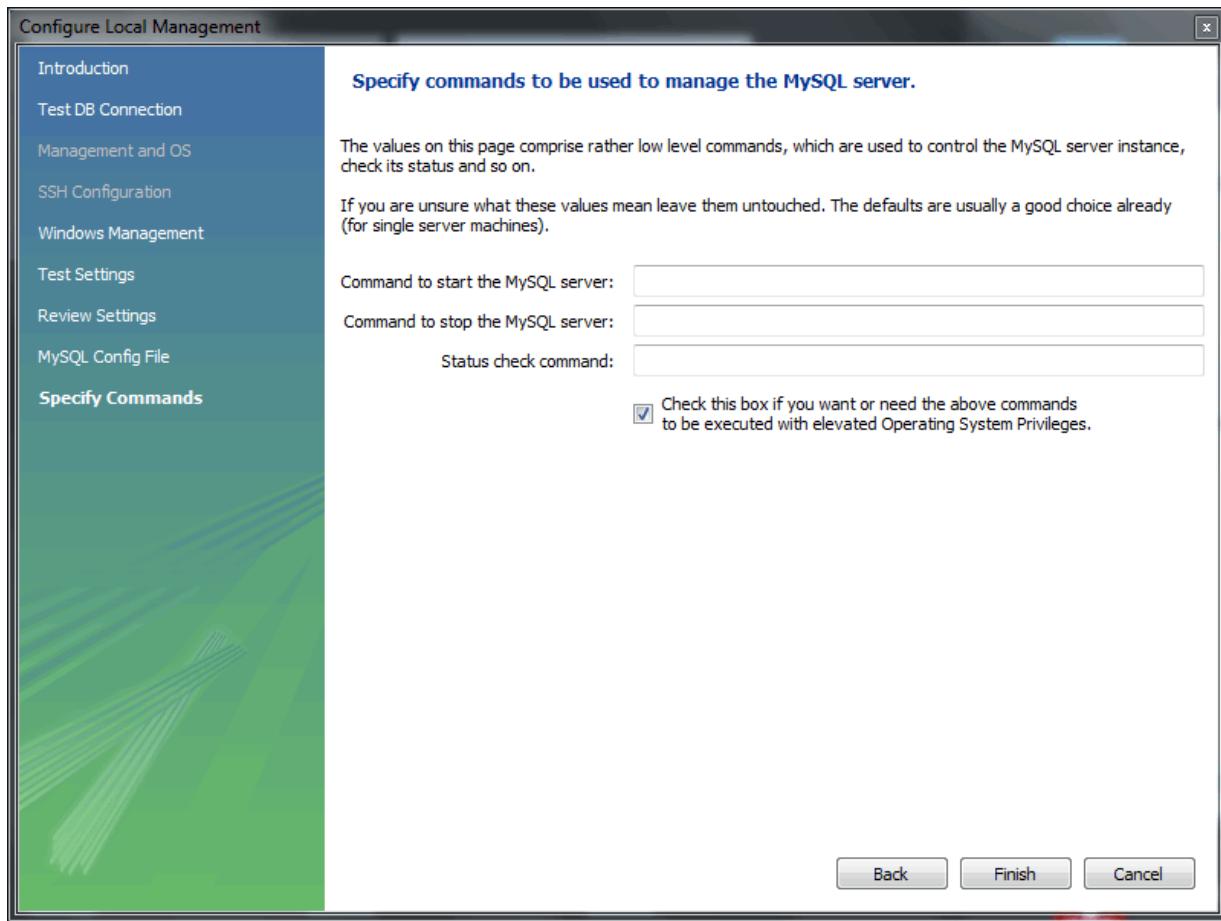
Figure 6.7. Getting Started Tutorial - Review Settings

Check the **Change Parameters** if you want to check or edit information about the MySQL configuration file. In our example we will check it, and click Next to continue.

10. Review the MySQL configuration file information. Click the **Check** buttons to perform the described checks, or optionally change the configuration file path.

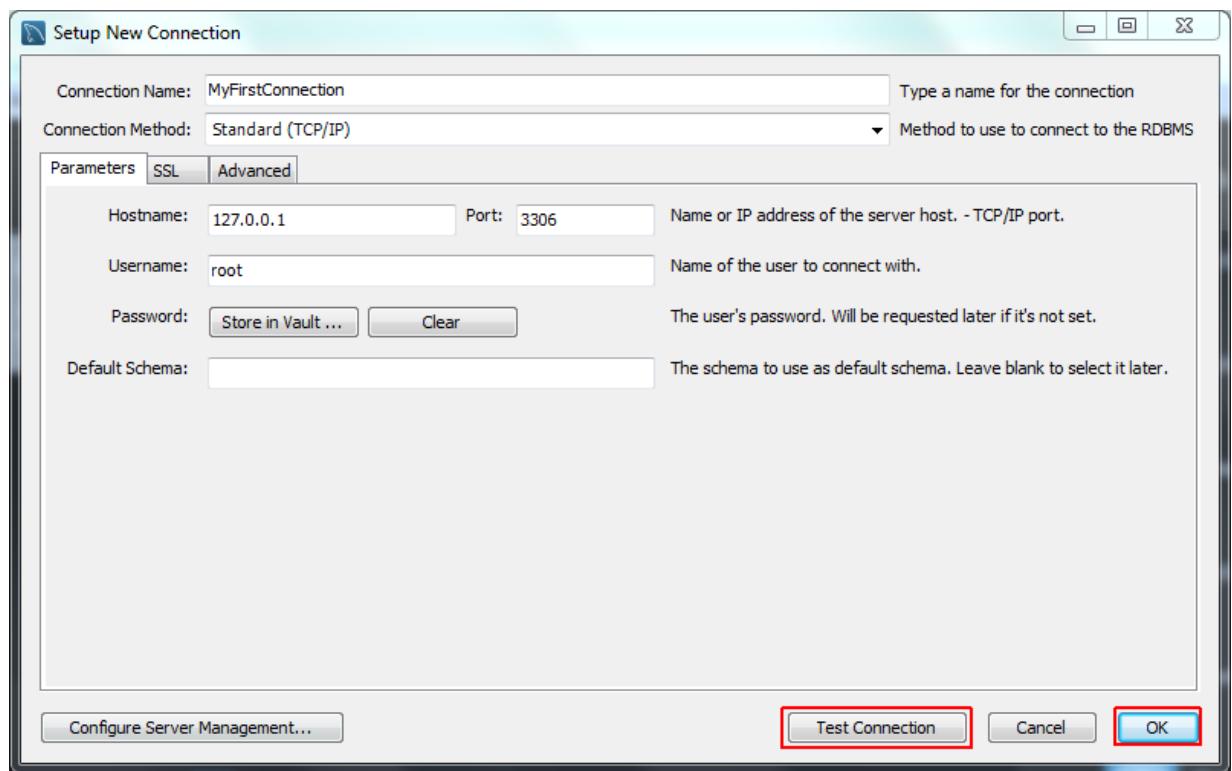
Figure 6.8. Getting Started Tutorial - MySQL Config File

11. Optionally, enter your own commands for starting, stopping, and checking the MySQL connection. Typically the default values are used, which means leaving these optional values blank.

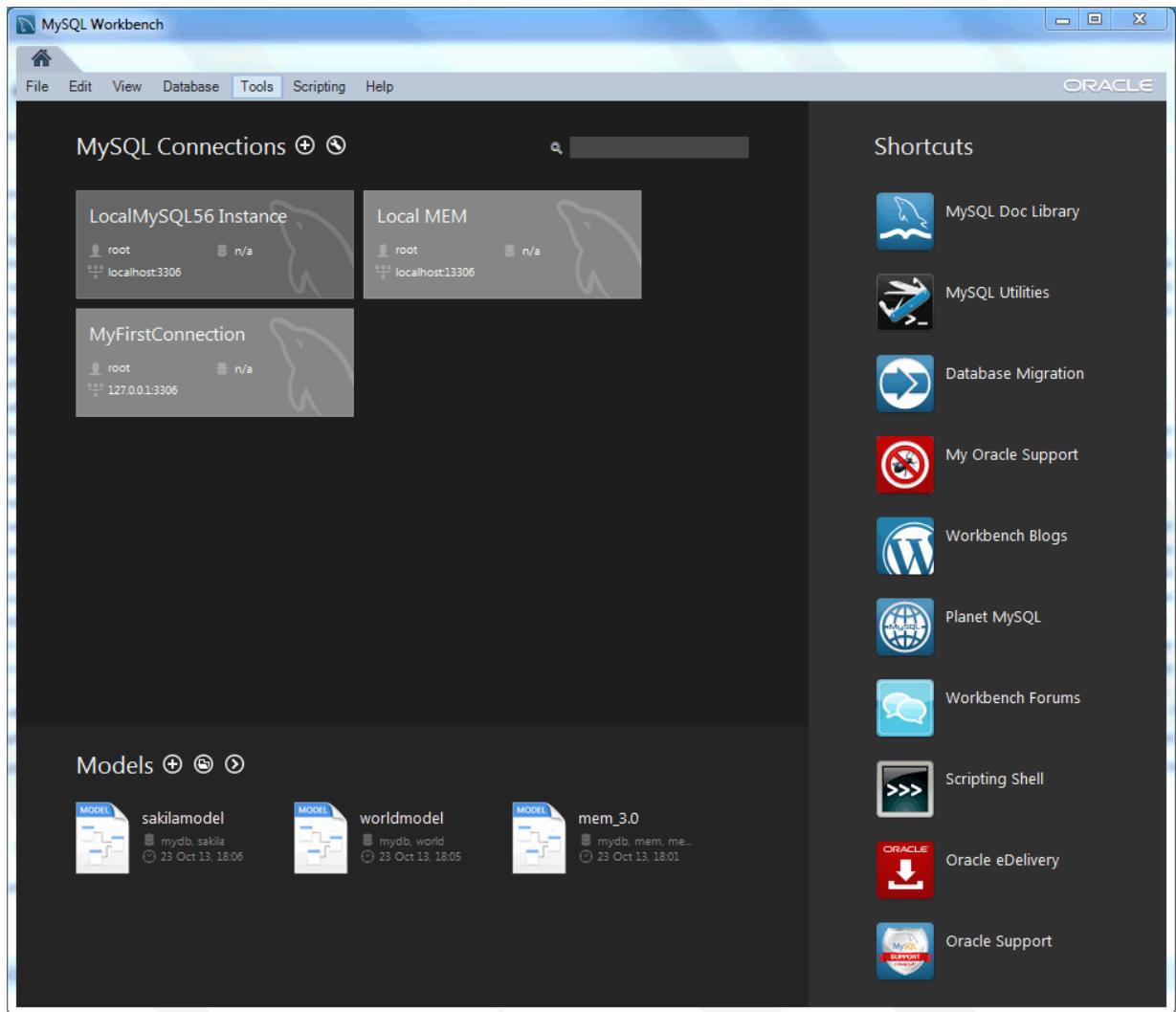
Figure 6.9. Getting Started Tutorial - Specify Commands

Click **Finish** to close the "Configure Server Management" dialog, which reveals the original **Setup New Connection** window.

12. After reviewing the **Setup New Connection** information, press Test Connection again to make sure it still functions, and then **OK** to create the new MySQL connection.

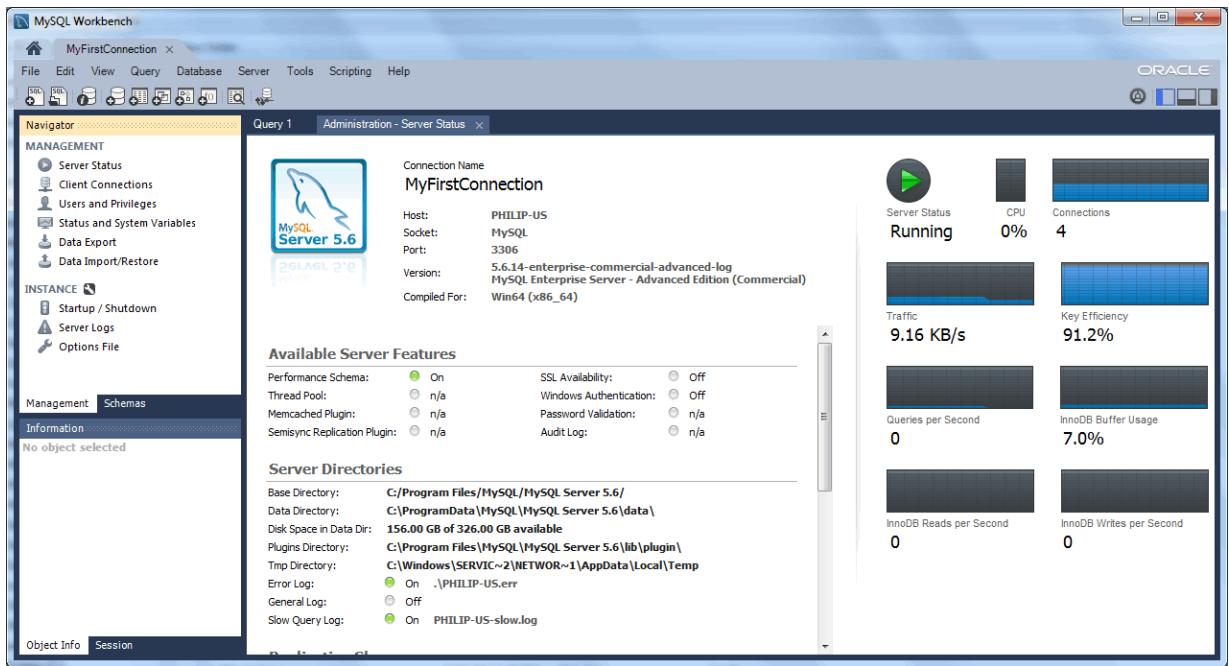
Figure 6.10. Getting Started Tutorial - Setup New Connection

13. You will now be returned to the Home window. You will see the new MySQL connection that you created, named MyFirstConnection.

Figure 6.11. Getting Started Tutorial - Home Window Instance

You are now ready to test your new MySQL connection.

14. From the Home window, double-click the MySQL connection you created to open up the SQL editor for this connection. The SQL editor is the default page, so now select the **Server Status** from the left **Navigator** panel. This displays the status of the MySQL server from your new MySQL connection.

Figure 6.12. Getting Started Tutorial - Server Status

15. Click around the **Navigator** panel and view the other **MANAGEMENT** and **INSTANCE** pages that relate to your new MySQL connection.

Notice the **Management** and **Schemas** tabs on the bottom of the **Navigator** panel. The **Schemas** view displays the schemas that are associated with your new MySQL connection. Alternatively, you can merge the Schemas and Management tabs by either clicking the merge icon on the top right of the Navigator panel, or by setting the **Show Management Tools and Schema Tree in a single tab** SQL Editor preference.

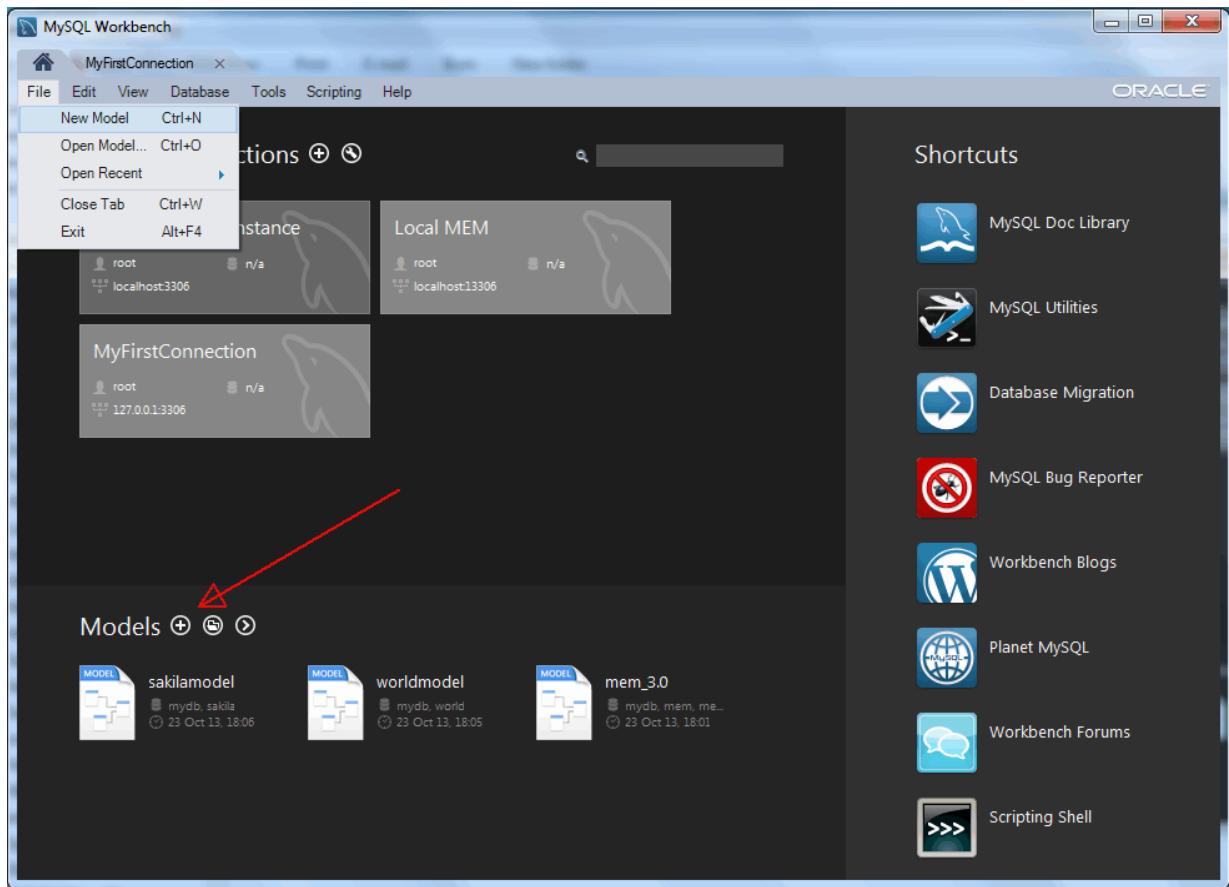
You have now seen how to create a MySQL connection to enable you to manage a MySQL server.

For further information, see [Chapter 5, MySQL Connections](#).

6.2. Creating a Model

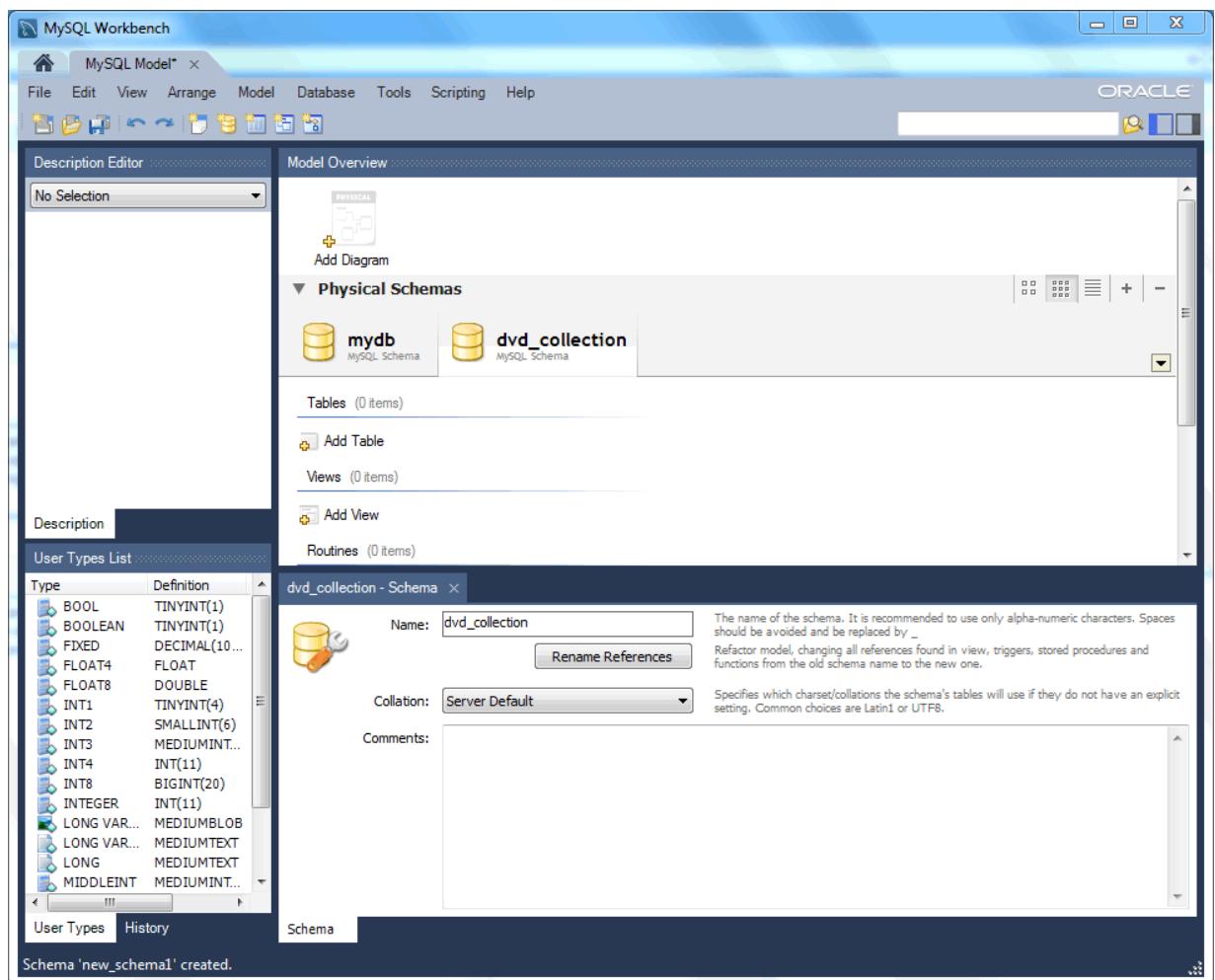
In this section, you will learn how to create a new database model, create a table, create an EER Diagram of your model, and then forward engineer your model to the live database server.

1. Start MySQL Workbench. On the Home window, click the **[+]** icon next to the **Models** section on the bottom of the page, or select **File**, **New Model**. A model can contain multiple schemata. Note that when you create a new model, it contains the `mydb` schema by default. You can change the name of this schema to serve your own purposes, or delete it.

Figure 6.13. Getting Started Tutorial - Home Window

2. On the Physical Schemata toolbar, click the button **+** to add a new schema. This will create a new schema and display a tabsheet for the schema. In the tabsheet, change the name of the schema to “dvd_collection”, by typing into the field called **Name**. Ensure that this change is reflected on the Physical Schemata tab. Now you are ready to add a table to your schema.

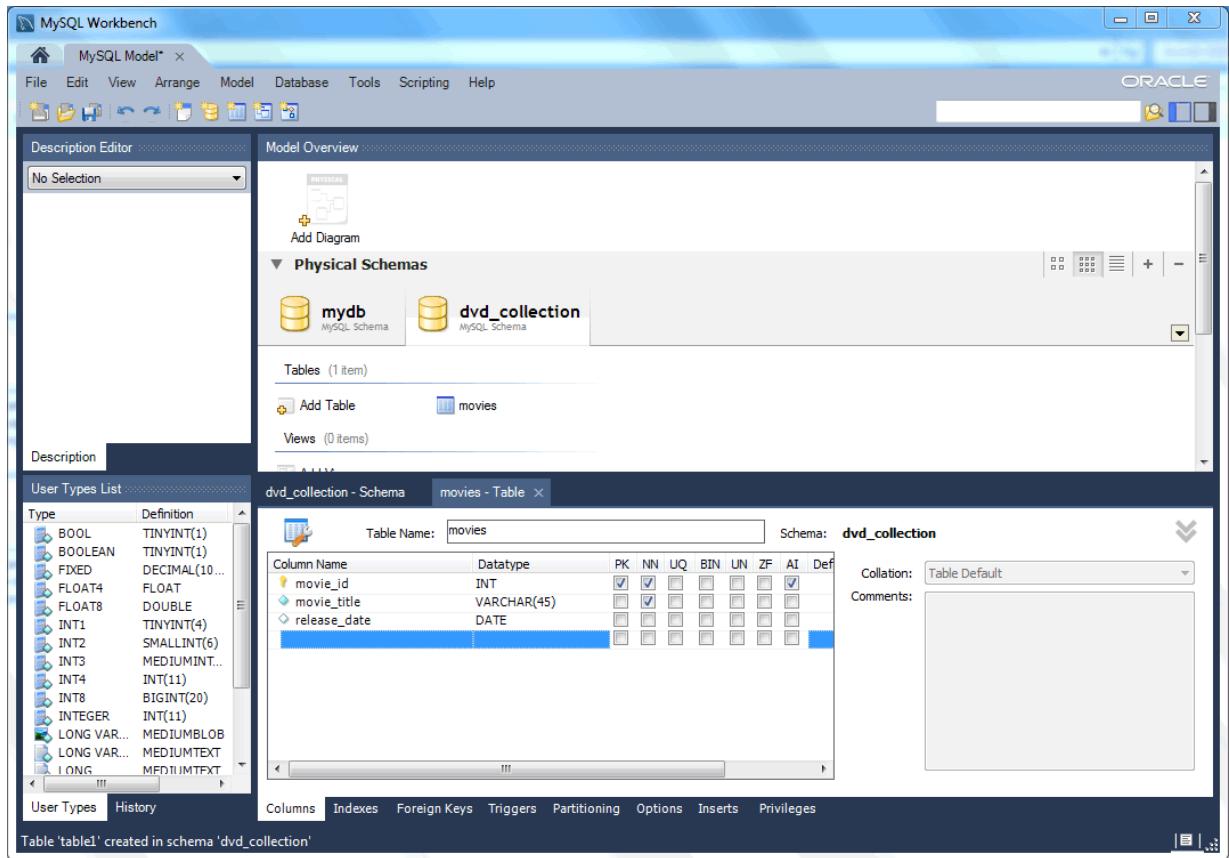
Figure 6.14. Getting Started Tutorial - New Schema



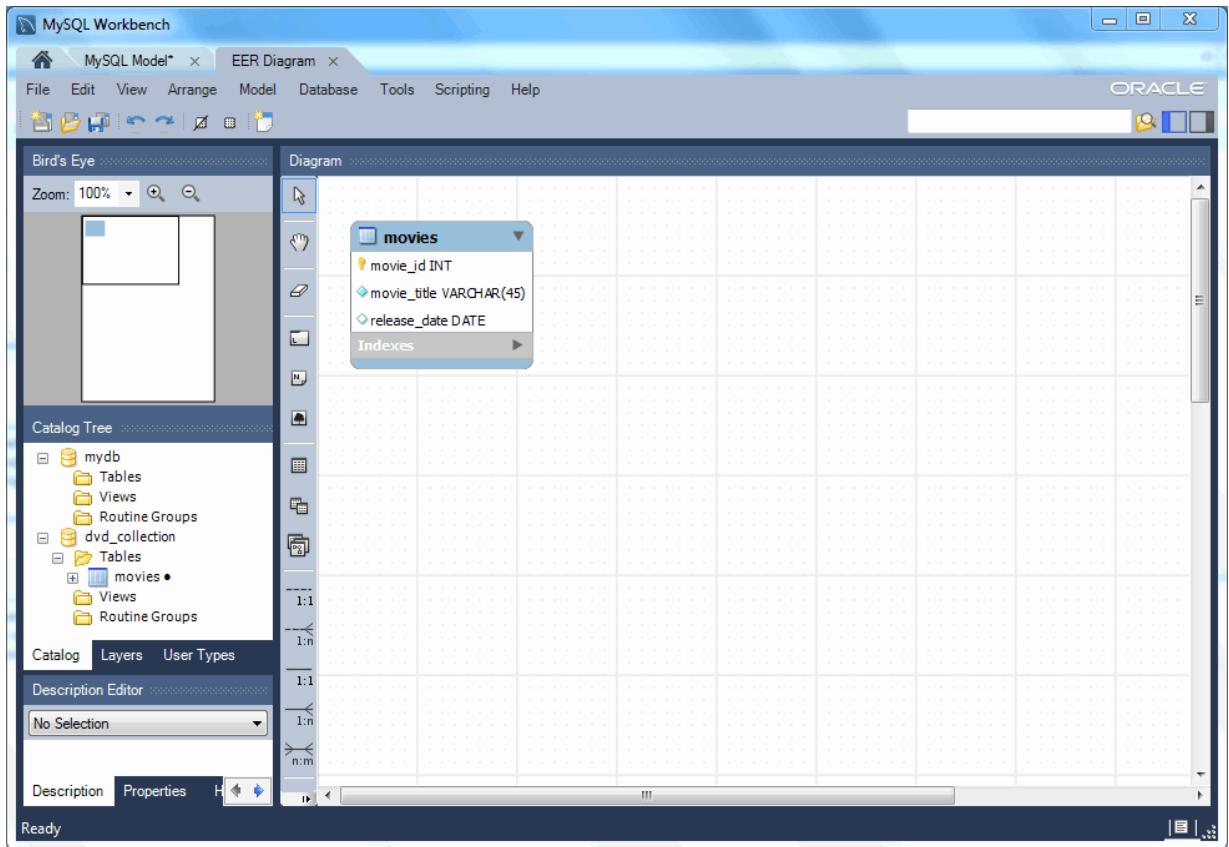
3. In the Physical Schemata section, double-click **Add Table**.
4. This will automatically load the table editor, with the default table name being **table1**. In the table editor, change the name of the table from “table1” to “movies”.
5. Next, add several columns. Double click a cell within the **Column Name** column, and the first field will default to “moviesid” because MySQL Workbench appends “id” to the table name as the default for the initial field. Change the name to “movie_id” and keep the **Datatype** as **INT**. Then, be sure **PK** (PRIMARY KEY), **NN** (NOT NULL), and **AI** (AUTO_INCREMENT) are all checked.
6. Add two additional columns using the same method as described above:

Column Name	Data Type	Column Properties
movie_title	VARCHAR(45)	NN
release_date	DATE (YYYY-MM-DD)	None

Figure 6.15. Getting Started Tutorial - Editing table columns



- Now you can obtain a visual representation of this schema so far. From the main menu, select Model, Create Diagram from Catalog Objects. The EER Diagram will be created and displayed.

Figure 6.16. Getting Started Tutorial - EER Diagram

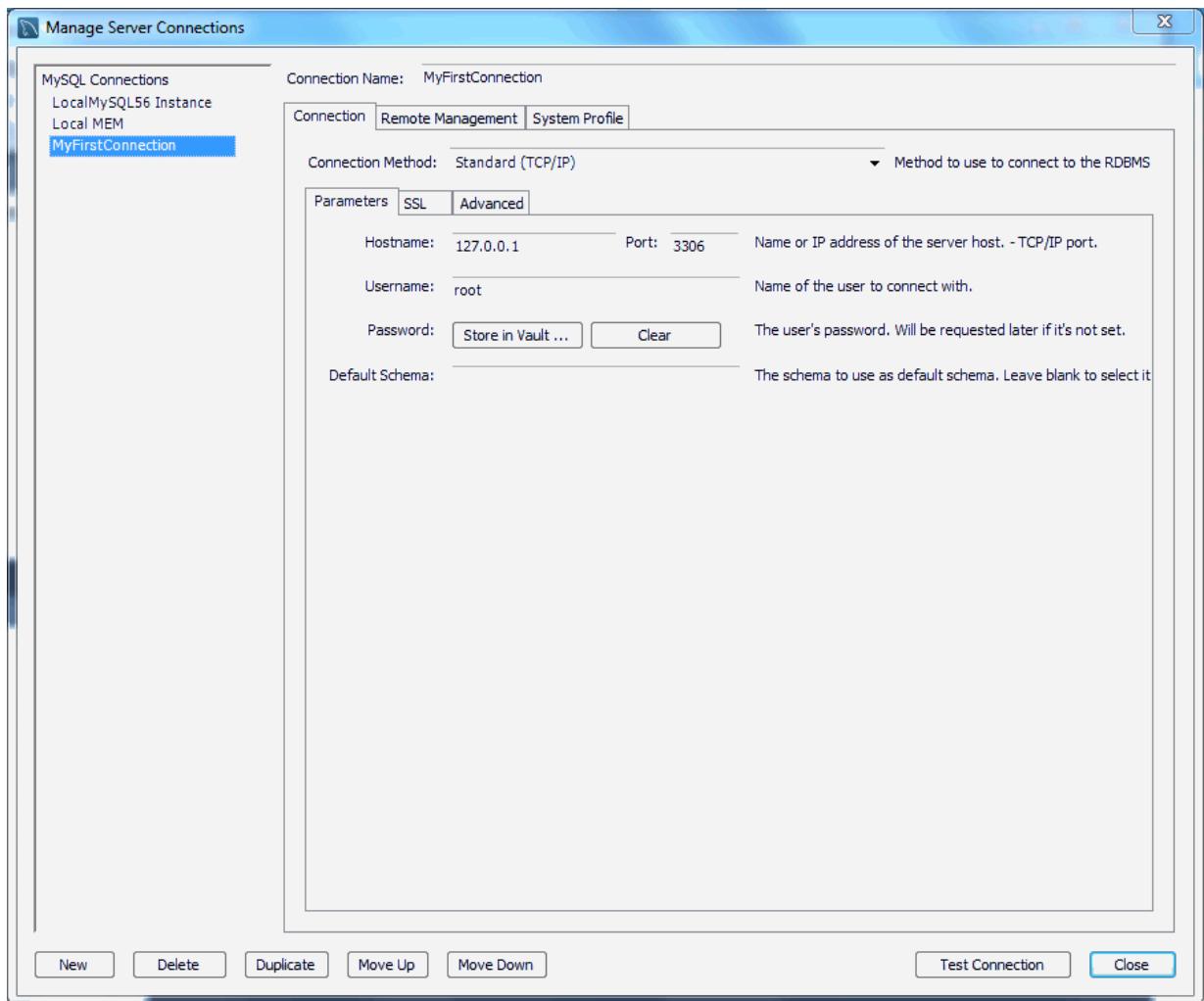
8. In the table editor, change the name of the column "movie_title" to "title". Note that the EER Diagram is automatically updated to reflect this change.

Note



There are several ways to open the table editor. Either change back to the **MySQL Model** tab and right-click on the `movies` table, or right-click on `movies` in the EER diagram and select an Edit 'movies' option.

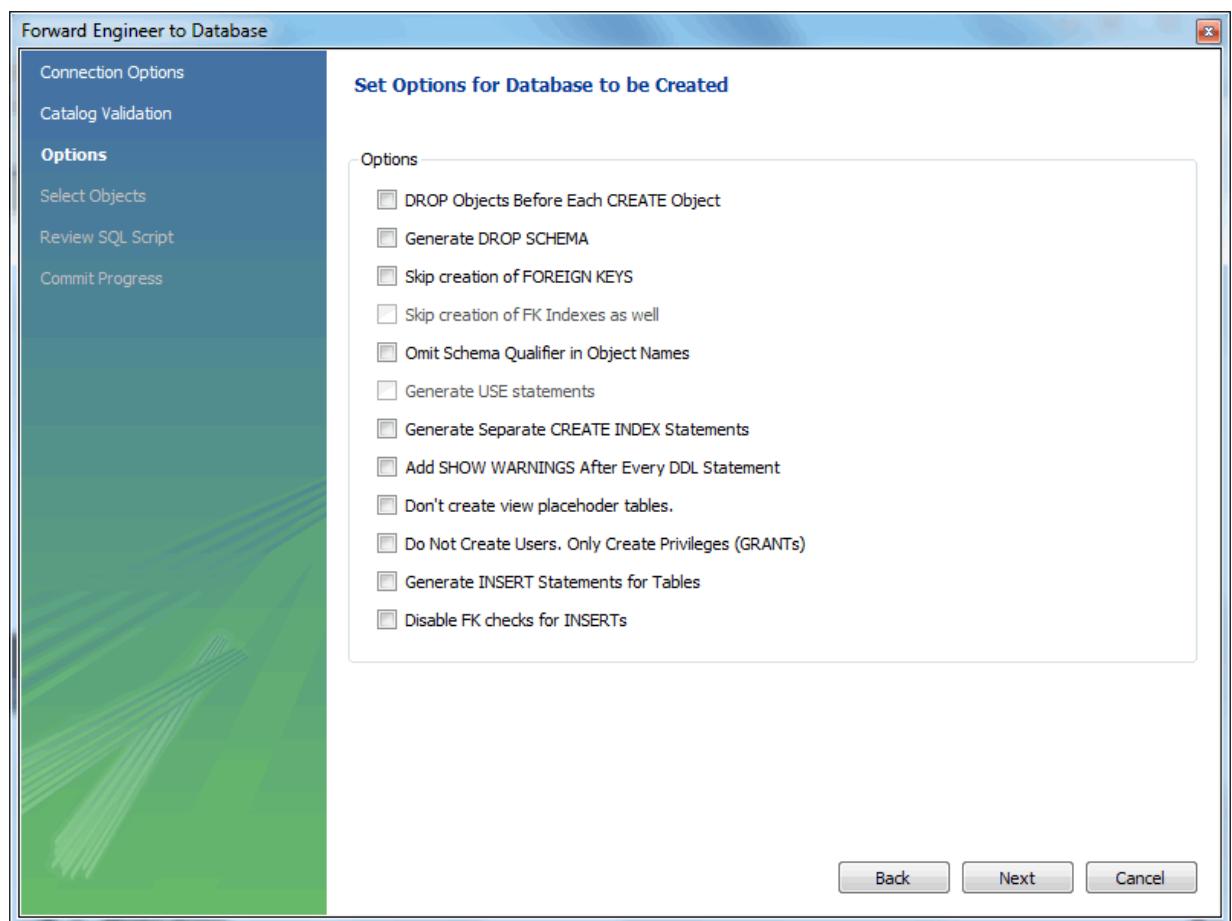
9. At this point, you can save your model. Click the main toolbar button **Save Model to Current File**. You have not yet saved this file so you will be prompted to enter a model file name. For this tutorial, enter "Home_Media". The Home_Media model may contain further schemata in addition to `dvd_collection`, such as `cd_collection`. Click **Save** to save the model.
10. You can synchronize your model with the live database server. First, tell MySQL Workbench how to connect to the live server. From the main menu, select **Database**, **Manage Connections....**
11. Here you can create a new MySQL connection, or use a MySQL connection that you created previously. For our example, we will use the **MyFirstConnection** from the previous tutorial. To do this, select `MyFirstConnection` from the list of MySQL connections on the left.
12. Click **Test Connection** to test your connection parameters. If everything is okay at this point, you can click **Close**.

Figure 6.17. Getting Started Tutorial - Manage Connections

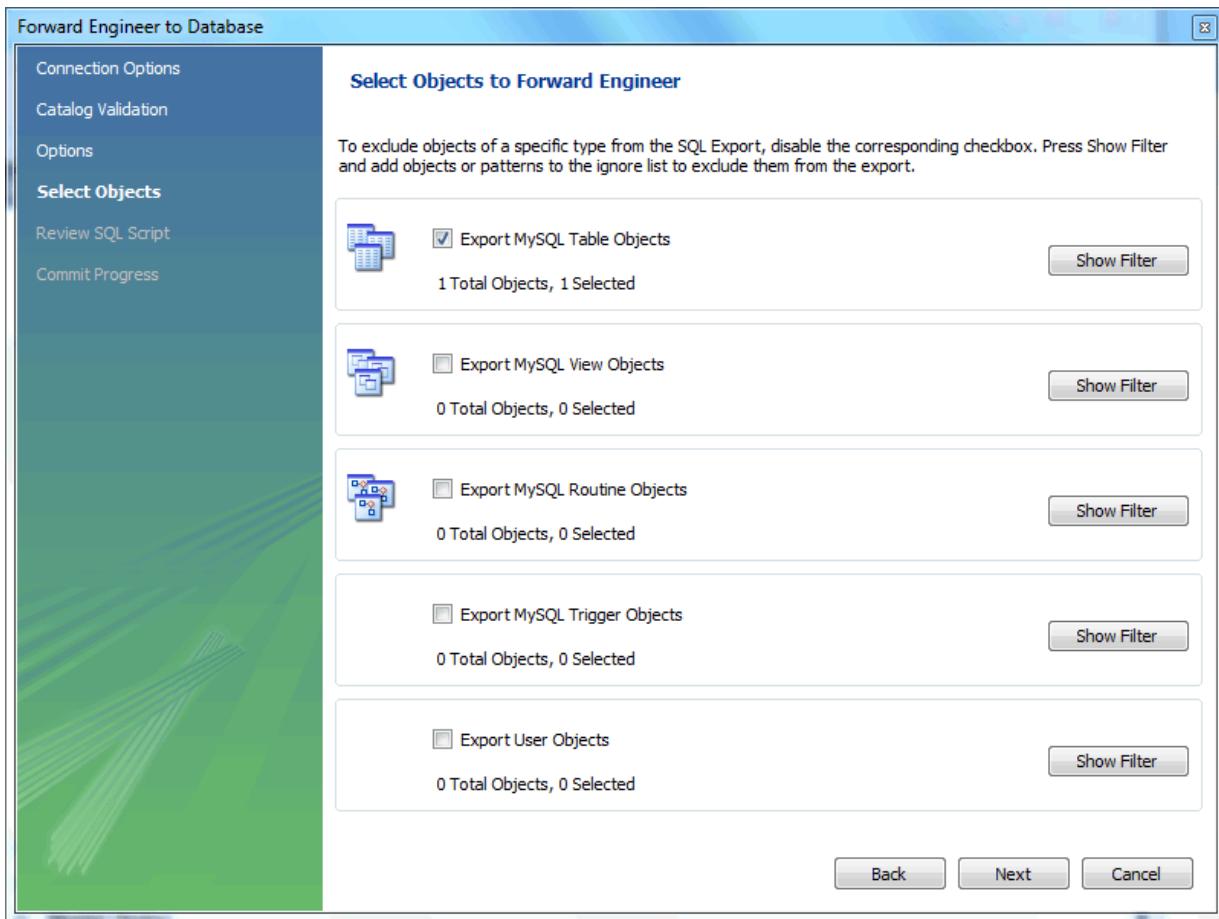
13. You are now ready to forward engineer your model to the live server. From the main menu, select Database, Forward Engineer.... The **Forward Engineer to Database** wizard will be displayed.
 14. The **Connection Options** page of the wizard lets you set additional options for your selected MySQL connection. Or, you may decided to choose a different MySQL connection. We do not require any connection changes, so click **Next**.
- 💡

Note

Notice that your **MyFirstConnection** connection is selected.
15. Optionally, you can execute a **Catalog Validation**. Click **Run Validations** and it should display "Validation Finished Successfully" without any errors.
 16. The **Options** page of the wizard shows various advanced options. For this tutorial, you can ignore these and simply click **Next**.

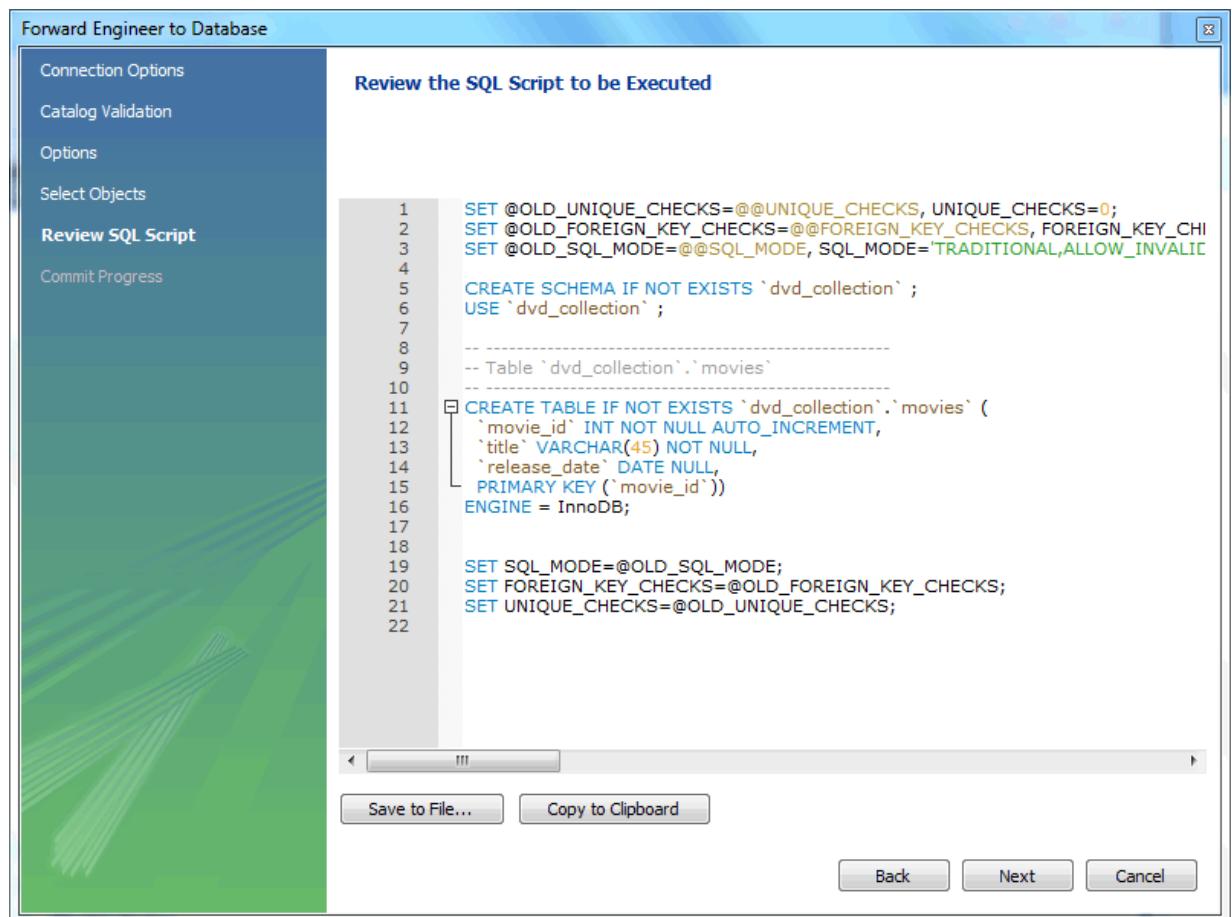
Figure 6.18. Getting Started Tutorial - Options

17. On the next page, you can select the object you want to export to the live server. In this case, we only have one table, so no other objects need be selected. Click **Next**.

Figure 6.19. Getting Started Tutorial - Select Objects

18. The **Review SQL Script** page displays the script that will be run on the live server to create your schema. Review the script to make sure that you understand the operations that will be carried out.

Clicking **Next** will execute the Forward Engineering process.

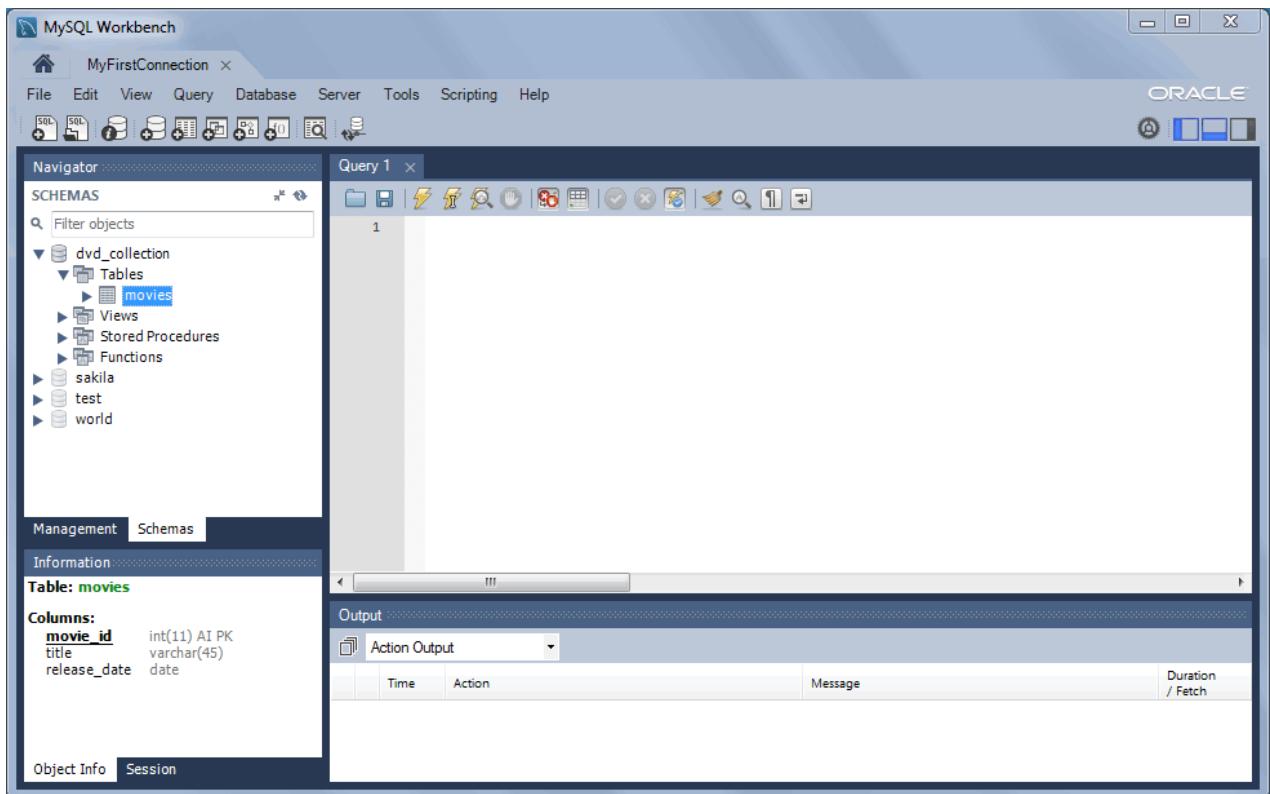
Figure 6.20. Getting Started Tutorial - Review SQL Script

19. The **Commit Progress** page shows each executed step, and its associated status. It is recommended to view the logs on this page, so click **Show Logs**.
20. After ensuring that the script ran without error on the server, then click **Close**. As a simple test that the script worked, launch the MySQL Command Line Client (`mysql`). Enter `SHOW DATABASES;` and identify your schema. Enter `USE dvd_collection;` to select your schema. Now enter `SHOW TABLES;`. Enter `SELECT * FROM movies;`, this will return the empty set as you have not yet entered any data into your database. Note that it is possible to use MySQL Workbench to carry out such checks, and you will see how to do this later, but the MySQL Command Line Client has been used here as you have probably used it previously.
21. Ensure that your model is saved. Click **Save Model to Current File** on the main toolbar.

6.3. Adding Data to Your Database

In the previous section, you created a model, schema, and table. You also forward engineered your model to the live server. In this section, you will see how you can use MySQL Workbench to add data into your database on the live server.

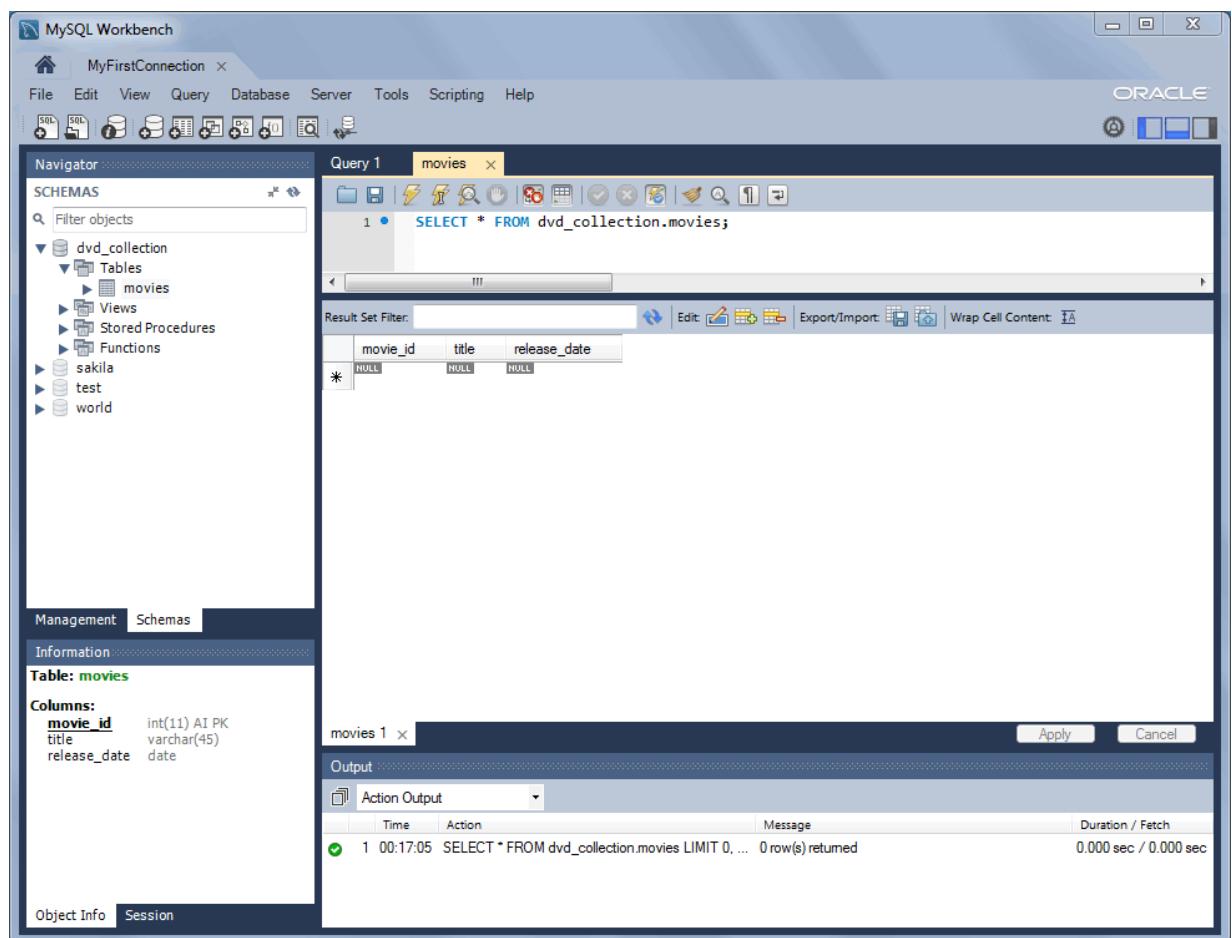
Open the SQL Editor by opening a MySQL connection.

Figure 6.21. Getting Started Tutorial - SQL Editor

1. From the **Navigator** panel on the left, select the `movies` table from the `dvd_collection` schema. Right-click on the `movies` table and choose Select Rows - Limit 1000.

**Note**

The **Navigator** panel has both **Management** and **Schemas** tabs.

Figure 6.22. Getting Started Tutorial - Adding Data from the SQL Editor

2. You will see the query and its associated data grid. This is where you can enter the data for your database. Remember that the `movie_id` was set to auto increment, so values are not needed for this column. In the data grid, enter the movie information shown in the following table.

title	release_date
Gone with the Wind	1939-04-17
The Hound of the Baskervilles	1939-03-31
The Matrix	1999-06-11
Above the Law	1988-04-08
Iron Man 2	2010-05-07

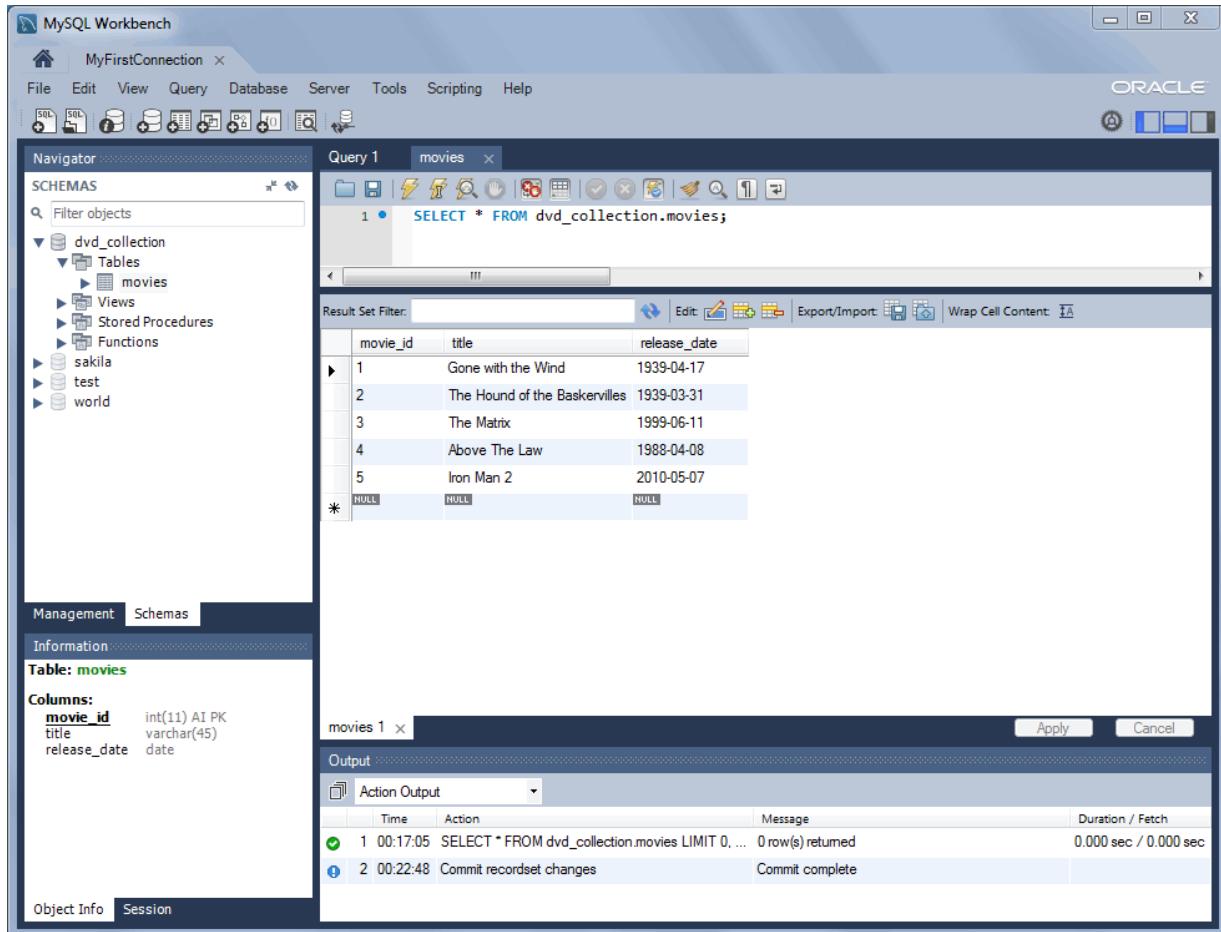
**Note**

Do not modify any values in the `movie_id` column.

3. Now click the Apply button in the toolbar located in the bottom right corner. A list of SQL statements will be displayed. Confirm that you understand the operations to be carried out. Click **Apply** to apply these changes to the live server.

4. Confirm that the script was executed correctly, then click **Finish**.
5. View the data grid again and observe that the autoincrement values have been generated.

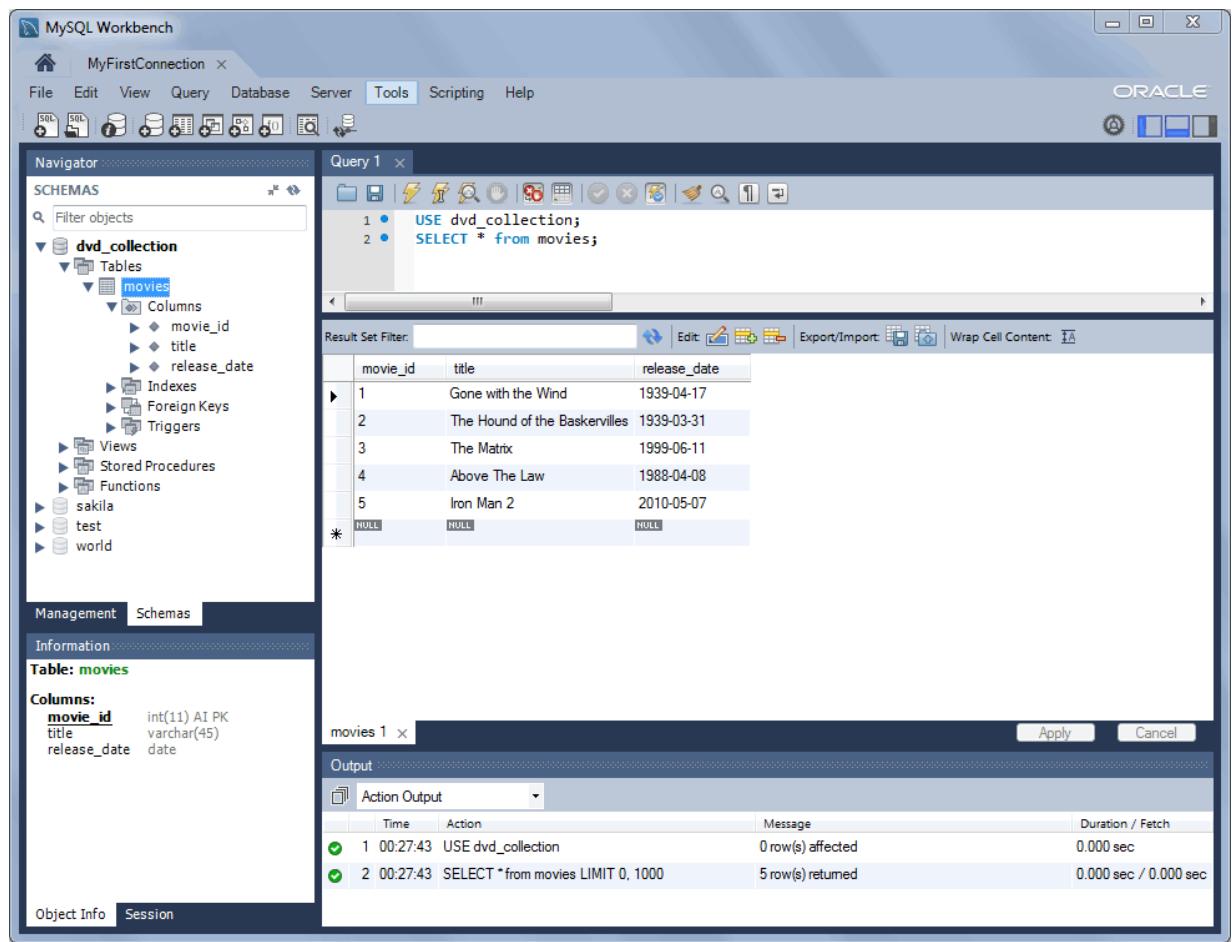
Figure 6.23. Getting Started Tutorial - Edit Data



6. Now you will check that the data really has been applied to the live server. Launch the MySQL Command Line Client. Enter `SELECT * FROM movies;` to see the data just entered.
7. You can also carry out a similar check from within MySQL Workbench. Close the `MyFirstConnection` tab, which should load the Home page. Now, click on the `MyFirstConnection` connection to load the SQL editor.
8. A new SQL Editor tab will be displayed. In the Query area, enter the following code:

```
USE dvd_collection;
SELECT * FROM movies;
```

9. Now click the **Execute** toolbar button. This resembles a small lightning bolt. The SQL Editor will display a new Result tab contain the result of executing the SQL statements.

Figure 6.24. Getting Started Tutorial - Results

In this section of the tutorial, you have learned how to add data to your database, and also how to execute SQL statements using MySQL Workbench.

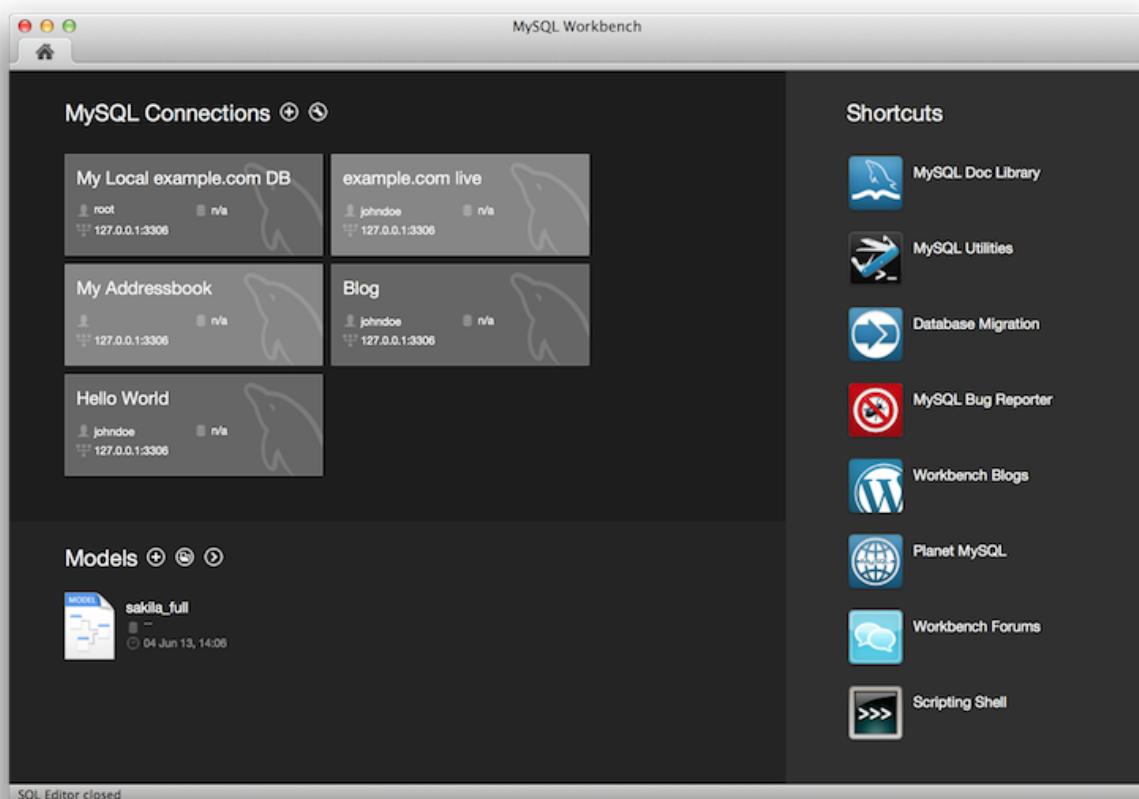
Chapter 7. The Home Window

Table of Contents

7.1. Workbench Shortcuts	82
7.2. MySQL Connections	82
7.3. Workbench Preferences	83
7.3.1. The General Tab	84
7.3.2. The Administrator Tab	85
7.3.3. The SQL Editor Tab	86
7.3.4. The SQL Queries Tab	89
7.3.5. The Model Tab	90
7.3.6. The Model:MySQL Tab	92
7.3.7. The Diagram Tab	93
7.3.8. The Appearance Tab	94
7.3.9. The Theming Tab	96

When MySQL Workbench first starts, it presents the **Home** window, which has three main sections; a list of MySQL Connections, Models, and Shortcuts.

Figure 7.1. The Home Window



7.1. Workbench Shortcuts

This includes shortcuts for following facilities:

- MySQL Doc Library: Built-in documentation
- MySQL Utilities: Command-line utilities, for further information see [Chapter 15, MySQL Utilities](#).
- Database Migration: Migrate databases to MySQL, for further information see [Chapter 12, Database Migration Wizard](#).
- MySQL Bug Reporter: Links to the MySQL bug system, where you can report bugs
- Workbench Blogs: Links to the Workbench team blog
- Planet MySQL: Links to MySQL-related blogs and news
- Workbench Forums: Links to the MySQL user and developer forums
- Scripting Shell: Execute scripts from within MySQL Workbench
- Oracle eDelivery: Links to your MySQL Workbench downloads (Commercial)
- Oracle Support: Links to your MOS (My Oracle Support) page (Commercial)

7.2. MySQL Connections

This section lists connections to all of your MySQL servers, and allows you to load, configure, group, and view information about each MySQL connection.

Connection Groups

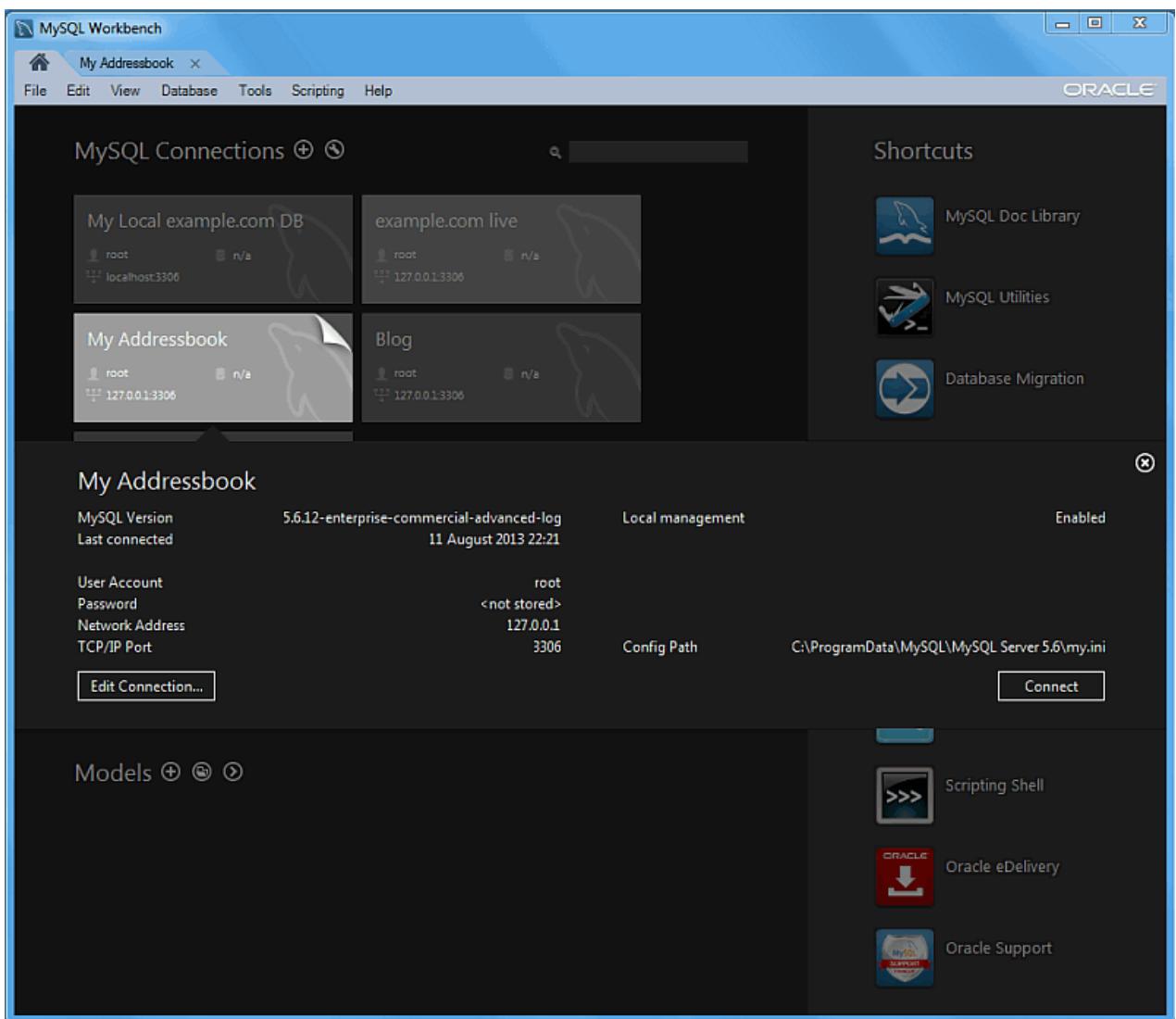
You may also create groups of connections. Create a group by either right-clicking a connection and choosing the Move to group... context menu option, or you may prefix your connection name with the group name separated by a forward slash (for example, "QA\TestBox") when you create or configure the connection.

Connection Information

The method to view connection information depends on the operating system.

- On Microsoft Windows and Linux: hover over the right side of a connection title and click the title
- On Mac OS X: hover over a connection title and click the little **(i)** in appears in the bottom right corner

The information will be displayed under the connection tiles, and will appear similar to:

Figure 7.2. Viewing Connection Information

Models

The **Models** panel lists your most recently used models. Each entry shows the date and time it was last opened, and its associated database. For further information about modeling, see [Chapter 9, Data Modeling](#).

To the right of the **Models** title are three icons. The (+) adds a new model, the (folder) opens an existing model from the disk, and the (>) opens a context menu for additional commands, such as [Create EER Model from Database](#).

7.3. Workbench Preferences

MySQL Workbench preferences and default settings are set by the [Preferences](#) menu. This menu is divided into sections, as described below:

- [General](#): Configuration of general-purpose options
- [Administrator](#): Configuration of tools used by the Administrator functionality

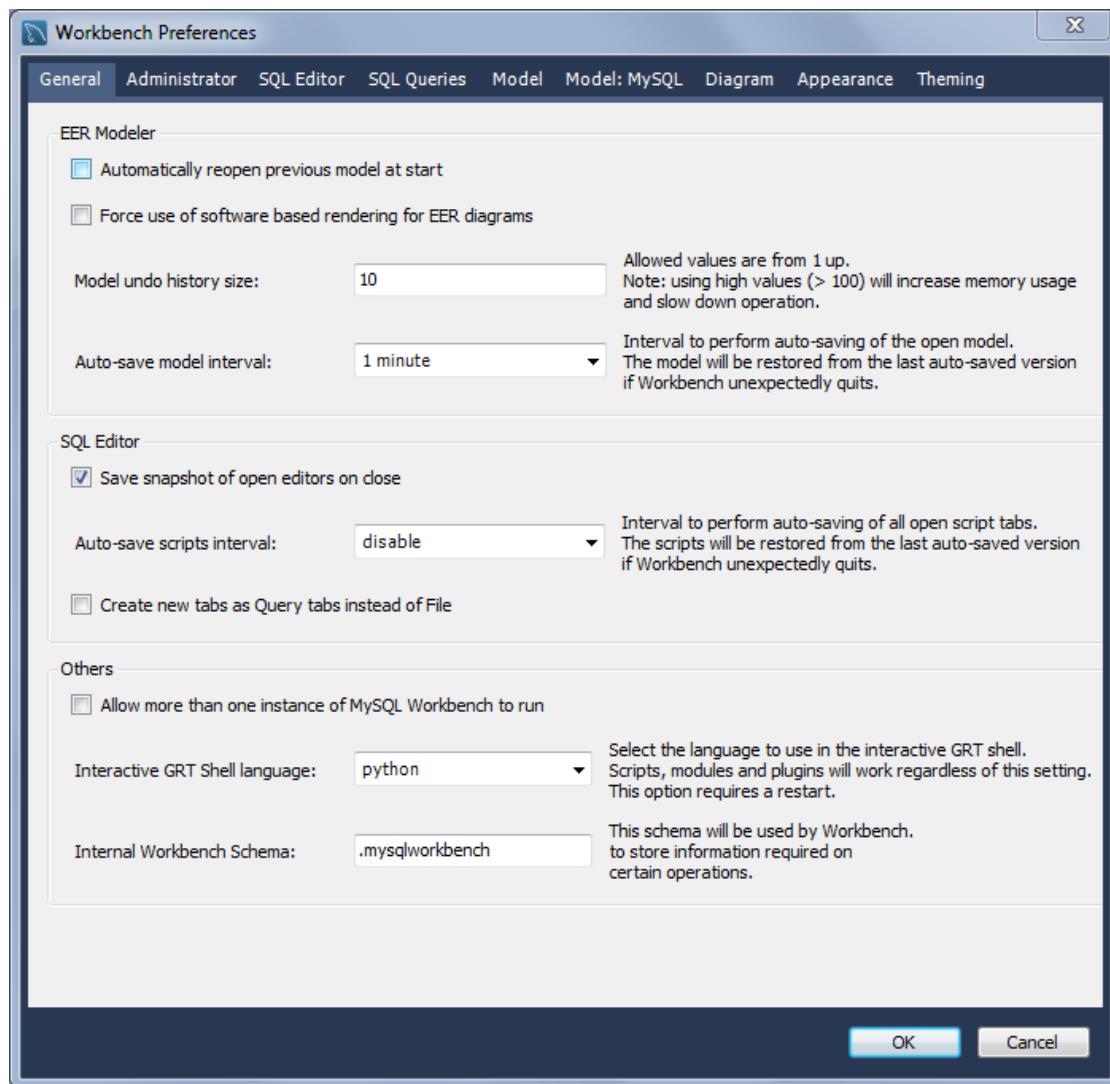
- SQL Editor: Configuration of the SQL Editor
- SQL Queries: Preferences specific to the queries
- Model: Default model attributes
- Model: MySQL: MySQL specific model attributes, including the MySQL storage engine and targeted MySQL version
- Diagram: EER diagram settings
- Appearance: Change colors and fonts used by various Workbench components
- Theming: Select a scheme that determines the core colors

A more detailed discussion of these options follows.

7.3.1. The General Tab

The General tab enables you to set the following options:

Figure 7.3. The General Preferences Dialog Box



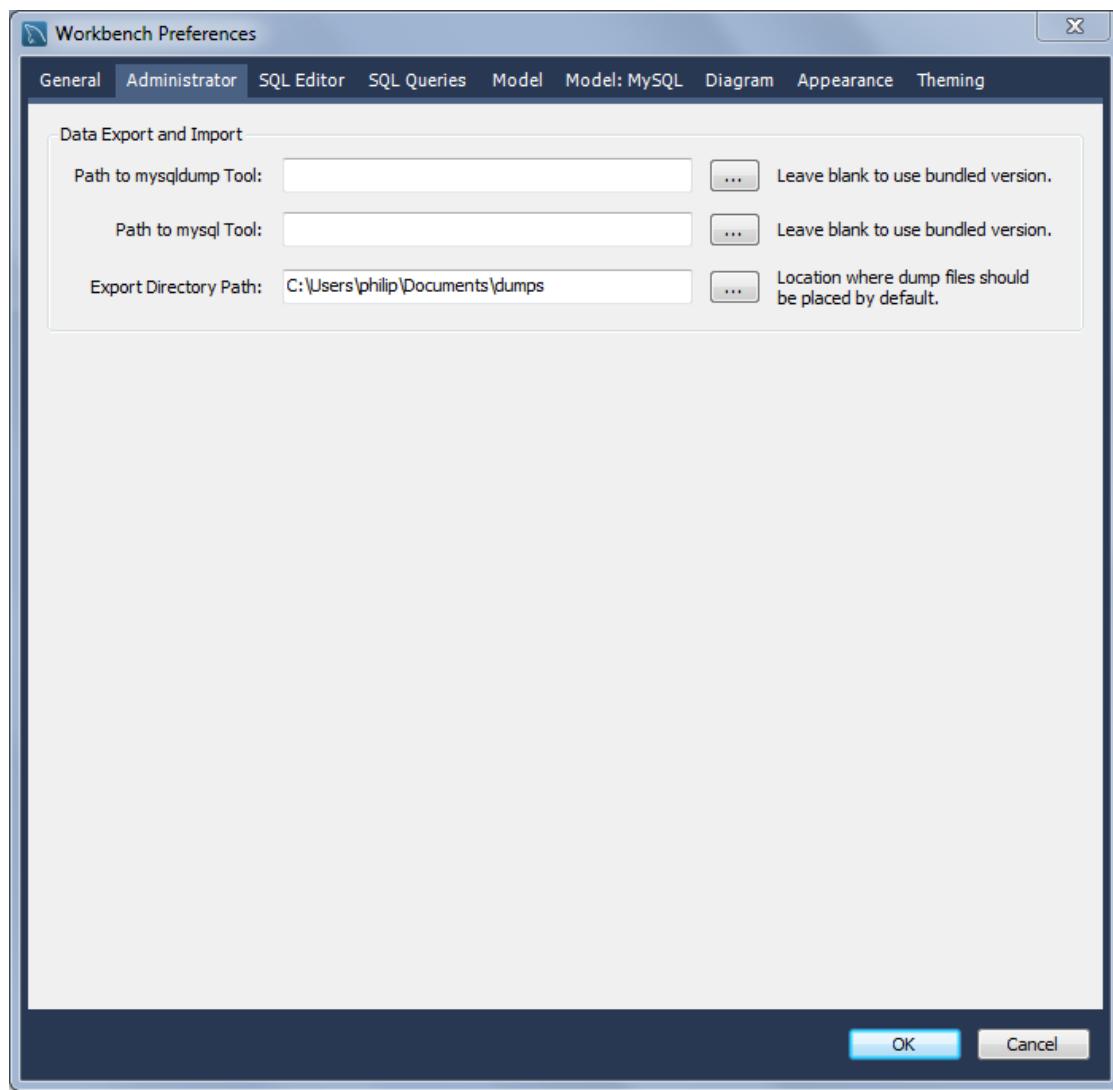
- **Automatically reopen previous model at start:** Check this if you want the model on which you previously worked to be automatically reopened when you start MySQL Workbench.
- **Force use of software based rendering for EER diagrams:** MySQL Workbench will use OpenGL for rendering when available. However, due to faulty drivers, problems do occasionally occur. These issues can be resolved by selecting the software rendering option here.
- **Model undo history size:** You can limit the size of the undo history here. Set this value to 0 to have an unlimited undo history.
- **Auto-save model interval:** An open model that has not been saved will automatically be saved after this period. On loading a model file, MySQL Workbench will notify the user if the file was not previously saved correctly, due to a crash or power failure. MySQL Workbench can then attempt to recover the last auto-saved version. For automatic recovery to be available for a new file, it will have to have been saved at least once by the user.
- **Save snapshot of open editors on close:** Enabling will save and reload the SQL Editor tabs after closing/opening MySQL Workbench (including after an unexpected crash).
- **Auto-save scripts interval:** Frequency of the auto-saves.
- **Create new tabs as Query tabs instead of File:** By default, opening a new SQL Editor tab opens as an SQL File tab. Check this option if you prefer the simpler Query tabs that, for example, will not prompt to be saved when closed. Added as of MySQL Workbench 5.2.45.
- **Allow more than once instance of MySQL Workbench to run:** Disabled by default.
- **Interactive GRT Shell Language:** You can select the language to be used in the GRT (Generic RunTime) shell by choosing a language from the list **Interactive GRT Shell Language**. Currently, the choices are Lua and Python. Python is the recommended option.
- **Internal Workbench Schema:** The schema used by MySQL Workbench to store information required on certain operations. Defaults to `.mysqlworkbench`

7.3.2. The Administrator Tab

This section provides configuration options that affect the Administrator functionality in MySQL Workbench.

- **Path to mysqldump tool:** Path to your local mysqldump binary. Leave it blank to use the bundled mysqldump binary.
- **Path to mysql tool:** Path to your local mysql client binary. Leave it blank to use the bundled mysql binary.
- **Export Directory Path:** Directory where your exported mysql dumps are located.

Figure 7.4. The **Administrator Preferences** Dialog Box



7.3.3. The SQL Editor Tab

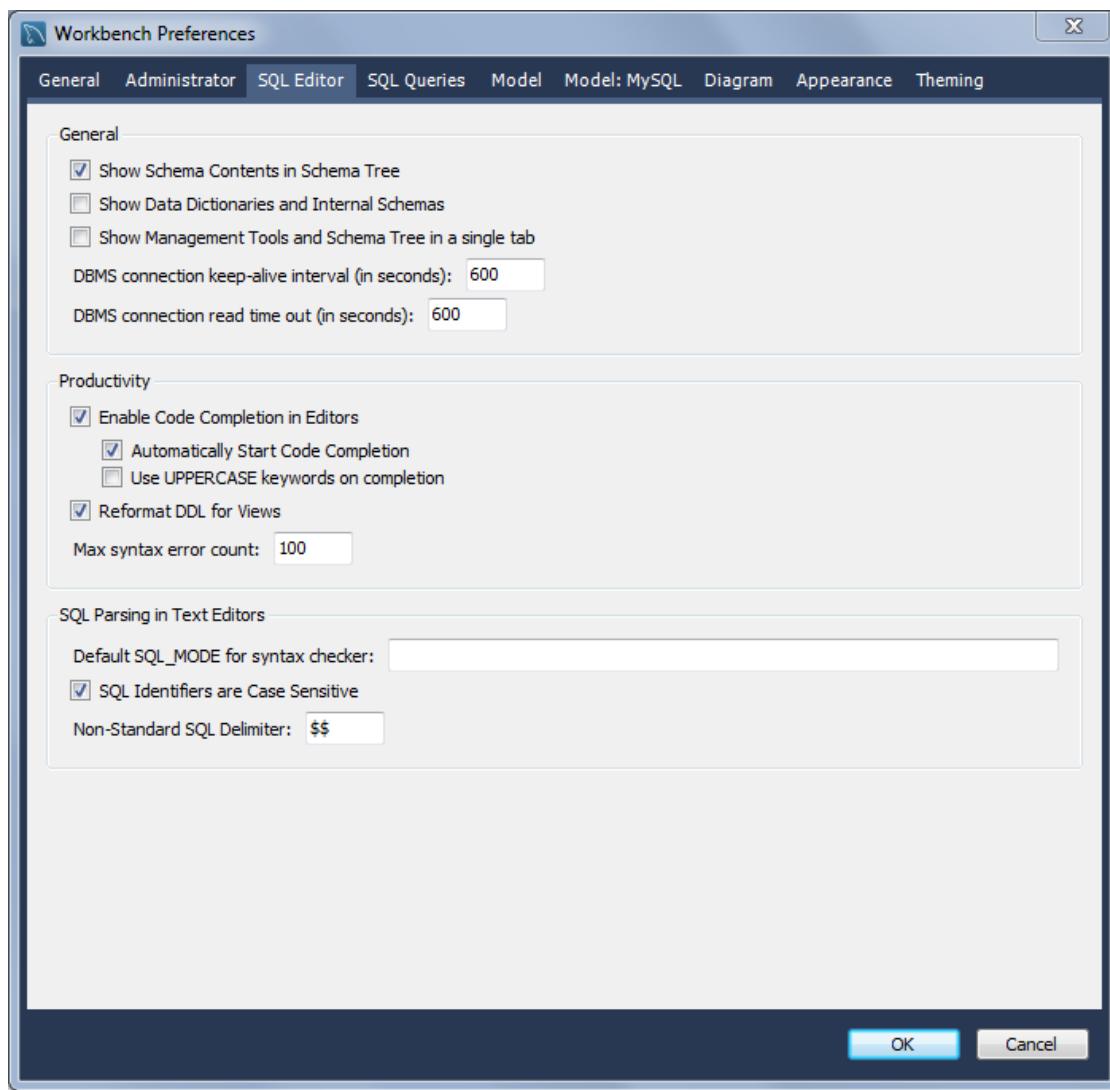
This section provides configuration options that affect the SQL Editor functionality in MySQL Workbench.



Note

There is also an **SQL Editor** section in the **General preference** tab, with preference settings for how the SQL Editor tabs are saved, and opened.

The SQL Editor is divided into three sections: General, Productivity, and SQL Parsing in Text Editors, as seen in the following screenshot:

Figure 7.5. The SQL Editor Preferences

General

- **Show Schema Contents in Schema Tree:** Enumerating, populating, and drawing large numbers of items can significantly increase loading times. For this reason, this facility can be switched off for models containing large numbers of schemata and tables.
- **Show Data Dictionaries and Internal Schemas:** Whether to show data directories and internal schemas in the schema tree (such as `INFORMATION_SCHEMA`, `mysql`, and schemas starting with `".`).
- **Show Management Tools and Schema Tree in a single tab:** This affects the Object Browser in the left sidebar, and this option can also be toggled from the sidebar. The management tools and schema tree can be viewable as separate tabs, or as a single long list.
- **DBMS connection keep-alive interval (in seconds):** When executing long running queries over a slow connection, you may need to increase this value to prevent the connection being lost. Defaults to 600.
- **DBMS connection read time out (in seconds):** Maximum amount of time that the query can take to return data from the DBMS. Defaults to 600.

Productivity

The query results properties that can be set include the following:

- **Enable Code Completion in Editors:** The SQL Editor offers Auto-complete functionality by either pressing the keyboard shortcut (**Modifier + Space**), or it will start automatically if the **Automatically Start Code Completion** preference is enabled.
- **Automatically Start Code Completion:** Starts code completion automatically when you type something and wait a moment.
- **Use UPPERCASE keywords on completion:** Normally keywords are shown and inserted as they come from the code editor's configuration file. This setting will always write completed keywords as uppercase.
- **Automatically Start Code Completion:** Enabled by default, this will automatically execute the code auto-completion feature while editing SQL in the SQL Editor. If disabled, you will instead use the keyboard shortcut **Modifier + Space** to execute the auto-completion routine.
- **Reformat DDL for Views:** Whether to automatically reformat the View DDL that is returned by the MySQL Server.



Note

The MySQL Server does not store the formatting information for View definitions.

- **Max syntax error count:** Large complex scripts may contain errors. Further, a syntax error early on can lead to subsequent syntax errors. For these reasons, it is possible to limit the number of errors displayed using this option. The default is 100 error messages.

SQL

SQL properties that can be set include the `SQL_MODE`, case sensitivity of identifiers, and the SQL delimiter used.

The document property `SqlMode` defines `SQL_MODE` for all operations affecting SQL parsing at the document scope. The purpose of this option is to preserve the consistency of SQL statements within the document.

The property has the following functions:

- Sets the `SQL_MODE` DBMS session variable to the value stored in the `SqlMode` property of the document when performing reverse engineering, forward engineering, or synchronization operations.
- Honors the `SQL_MODE` values defined in `SqlMode` so that SQL parsing is correct.

Only a subset of all possible `SQL_MODE` values affect the MySQL Workbench SQL parser. These values are: `ANSI_QUOTES`, `HIGH_NOT_PRECEDENCE`, `IGNORE_SPACE`, `NO_BACKSLASH_ESCAPES`, `PIPES_AS_CONCAT`. Other values do not affect the MySQL Workbench SQL parser and are ignored.

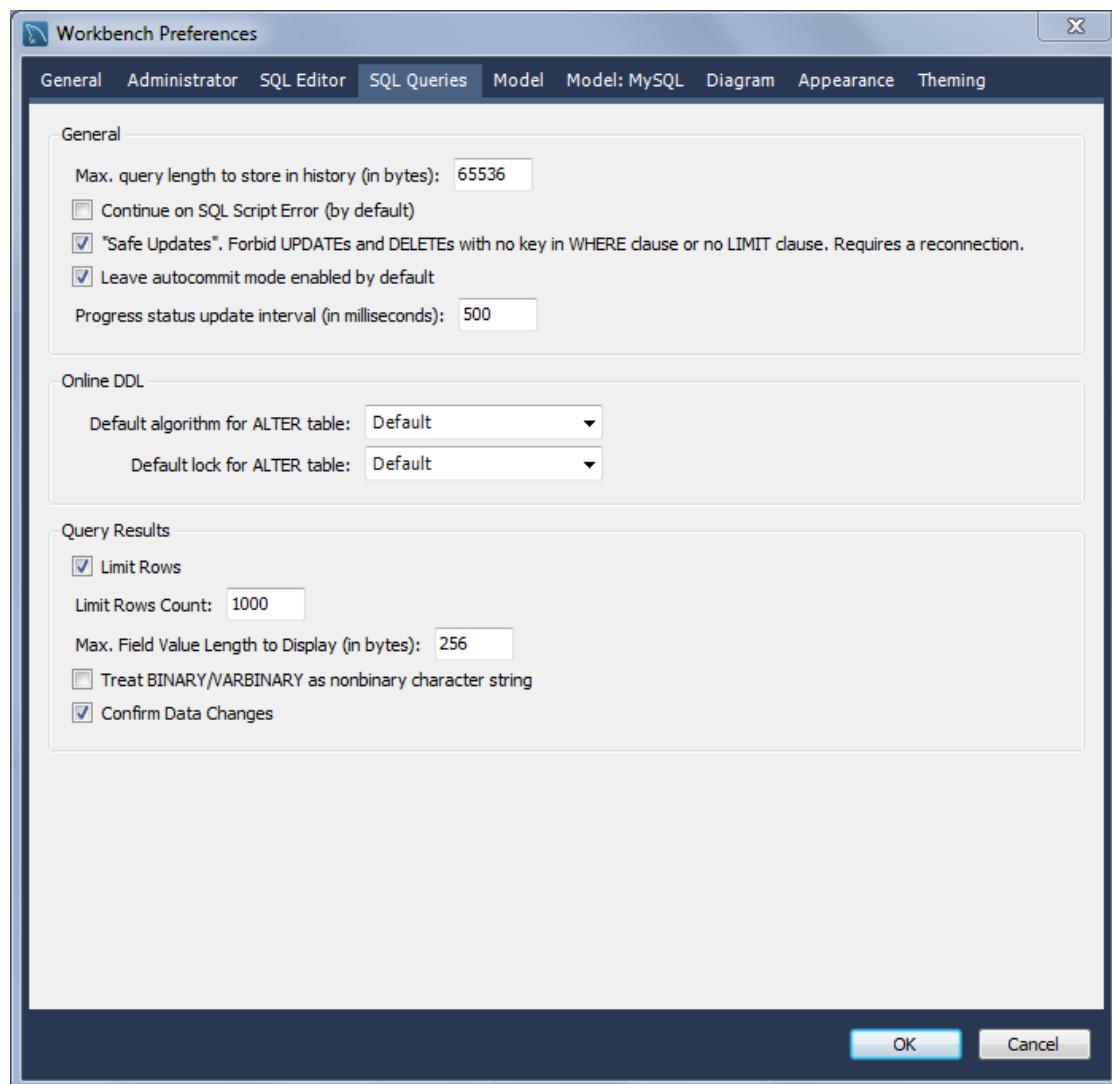
If the value of `SqlMode` is not set, the default value of the `SQL_MODE` session variable defined by the server stays unchanged during operations with the server. However, the MySQL Workbench SQL parser behaves as if `SQL_MODE` is also not set. This may potentially lead to inconsistencies in parsing of SQL statements stored in the document. If you choose to not set the `SqlMode` property, ensure that the default `SQL_MODE` variable defined by the server does not contain any values from the following list: `ANSI_QUOTES`, `HIGH_NOT_PRECEDENCE`, `IGNORE_SPACE`, `NO_BACKSLASH_ESCAPES`, `PIPES_AS_CONCAT`.

The `SqlMode` property is defined in two locations: globally and at document scope. MySQL Workbench uses the global property to initialize the document property for each new document created. For each document, the property value defined at document scope always has higher priority over the one defined globally.

7.3.4. The SQL Queries Tab

Preferences related to the SQL Editor, and how query viewing behaves.

Figure 7.6. The SQL Queries Preferences



General

- **Max query length to store in history (in bytes):** Queries that exceed this size will not be saved in the history when executed. The default is 65536 bytes, and setting to 0 means there is no limit (all queries will be saved).
- **Continue on SQL Script Error:** Should an error occur while executing a script, this option causes execution to continue for the remainder of the script.

- **"Safe Updates". Forbid UPDATEs and DELETEs with no key in WHERE clause, or no LIMIT clause. Requires connection:** This enables the `sql_safe_updates` MySQL Server option for the MySQL Workbench session. Changing this option requires a reconnection to the server, which can be performed by [Query](#), [Reconnect to Server](#).
- **Leave autocommit mode enabled by default:** Toggles the default autocommit mode for connections. When enabled, each statement will be committed immediately.

**Note**

All query tabs in the same connection share the same transaction. To have independent transactions, you must open a new connection.

- **Progress status update interval:** When executing long running queries over a slow connection, you may need to increase this value to prevent excess load on the connection. Defaults to 500 milliseconds.

Online DDL

- **Default algorithm for ALTER table:** The default algorithm selected when performing `ALTER TABLE` operations in MySQL Workbench. The setting can also be adjusted for each `ALTER TABLE` operation. Options include "In-Place" (preferred) and "Copy", see the [online DDL](#) documentation for more information.
- **Default lock for ALTER table:** The default lock setting for allowing concurrent queries with `ALTER TABLE` in MySQL Workbench. This setting can also be adjusted for each `ALTER TABLE` operation. Options include "None", "Shared", and "Exclusive", see the [online DDL](#) documentation for more information.

Query Results

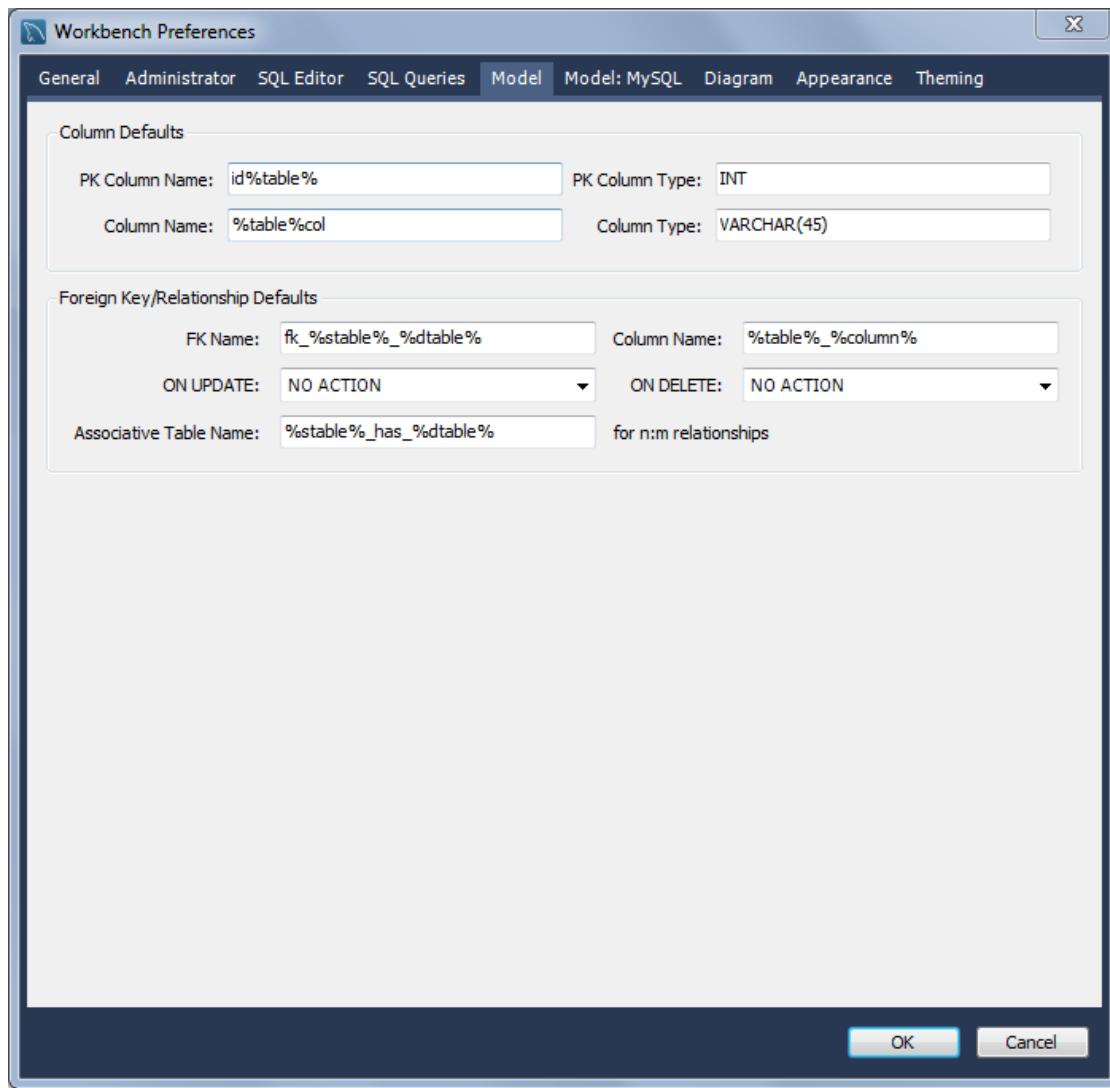
- **Limit Rows:** Queries can sometimes return an excessive number of rows, which can heavily load the connection, and take time to display in MySQL Workbench. To prevent this, you can set a more moderate value here. This limit is defined by the **Limit Rows Count** option.
- **Limit Rows Count:** Specify the maximum number of result rows to return. Defaults to 1000.
- **Max. Field Value Length to Display:** To avoid display problems due to excessive field length, it is possible to set the maximum field length to display (in bytes). Defaults to 256.
- **Treat BINARY/VARBINARY as non-binary character string:** Binary byte string values are not displayed by default in the results grid, but are instead marked as `BLOB` values. These can then be viewed or edited with the `BLOB` editor. Nonbinary character string values are displayed in the results grid, and can be edited in the grid cell or using the `BLOB` editor.

If this option is turned on, data truncation may result: Binary byte string values may contain null bytes as part of their valid data, whereas for nonbinary character strings, a null byte terminates the string.

- **Confirm Data Changes:** In the SQL Editor, if you edit table data and then click the [Applying changes to data](#) button, MySQL Workbench launches a wizard to step you through applying your changes. This gives you a chance to review the SQL that will be applied to the live server to make the requested changes. If this option is deselected, the changes will be applied to the server without the wizard being displayed and without giving you a chance to review the changes that will be made.

7.3.5. The Model Tab

This section provides configuration options that affect the Modeling functionality in MySQL Workbench.

Figure 7.7. The Model Preferences

Use the **When Deleting Physical Model Figures in Diagram** section to determine the behavior when deleting objects from the EER diagram canvas. Choose [Ask](#) and whenever you delete an object, you will be asked whether you wish to remove the object from an EER diagram only or also from the catalog. The [Keep Database Object in Catalog](#) is the safest option. You also have the option of deleting the object from both the EER diagram and the catalog.



Note

If you choose the [Ask](#) option, a confirmation dialog box opens only when you are deleting an object from an EER Diagram. When deleting in the MySQL Model view, there is **no** confirmation dialog window and the delete action always removes the object from the catalog.

There are a variety of ways to delete an object from an EER canvas: using the [eraser](#) tool; choosing a pop-up menu item; using the delete key; and by choosing the delete option from the [Edit](#) menu. In each case, the action performed by the delete key is determined by the option chosen from the **When Deleting Physical Model Figures in Diagram** section.

Use the Model tab to set the default value for various object names and the primary key data type. The following table shows the object names and their default values.

Object Name	Default Value
PK Column Name	<code>id%table%</code>
PK Column Type	<code>INT</code>
Column Name	<code>%table%col</code>
Column Type	<code>VARCHAR(45)</code>
FK Name	<code>fk%stable_%dtable%</code>
Foreign Key Column Name	<code>%table%_column%</code>
ON UPDATE	<code>NO ACTION</code>
ON DELETE	<code>NO ACTION</code>
Associative Table Name	<code>%stable%_has_%dtable%</code>

The **PK Column Name**, **PK Column Type**, **Column Name**, and **Column Type** values are the defaults used by the table editor, and only function on Microsoft Windows and Mac OS X. The others are the default names used when using the relationship tools on an EER diagram.

Within object values items enclosed by percentage signs are variables. Their meanings are as follows:

- `%table%`: The table associated with the object
- `%stable%`: The source table associated with the object
- `%dtable%`: The destination table associated with the object
- `%column%`: The column associated with the object

Legitimate values for the foreign key update or delete rules are:

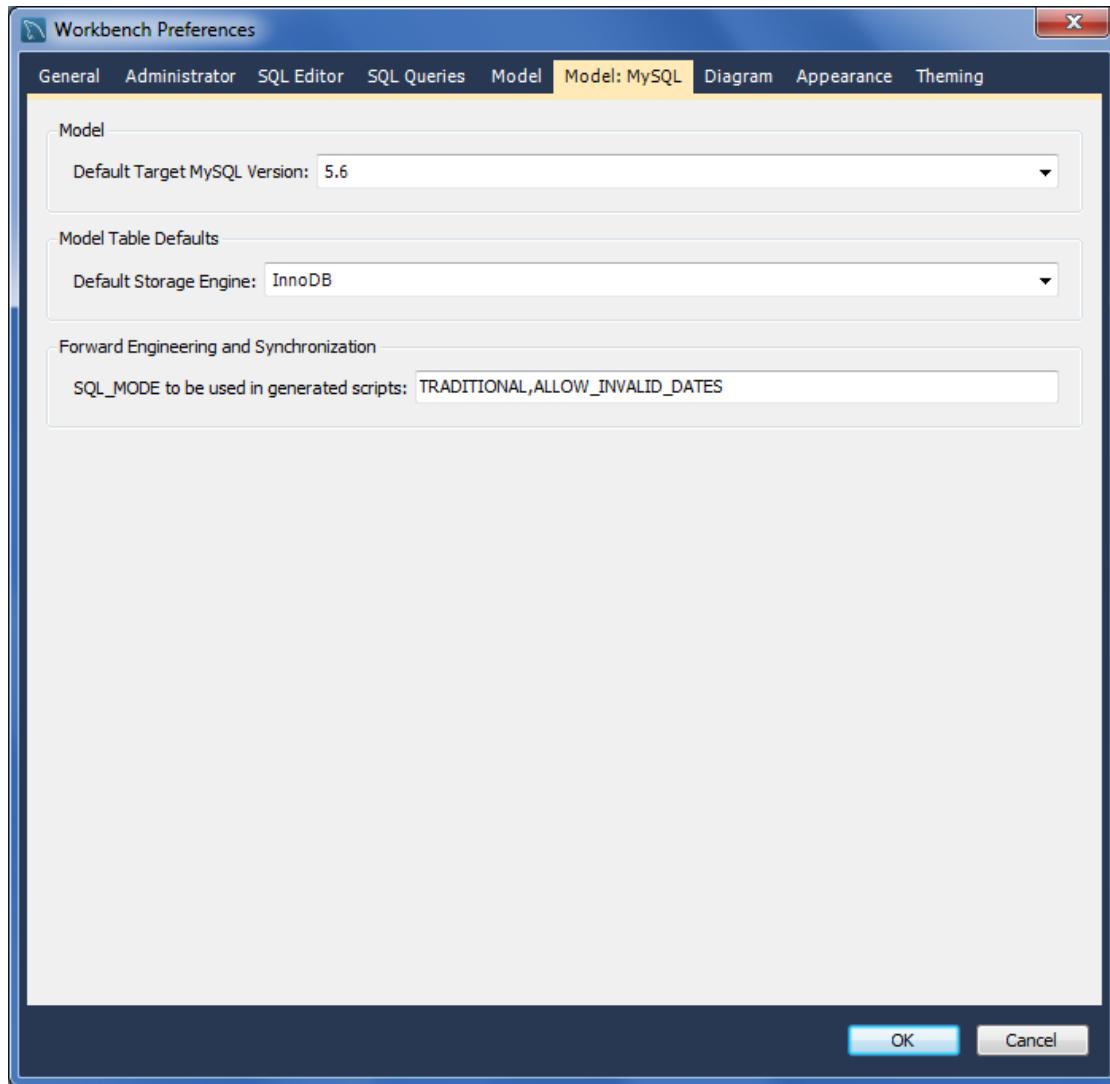
- `RESTRICT`
- `CASCADE`
- `SET NULL`
- `NO ACTION` (default)

For more information about these actions, see [Section 9.3.1.3.4, “The Foreign Keys Tab”](#).

7.3.6. The Model:MySQL Tab

This enables you to set the default table storage engine, MySQL Server version, and `SQL_MODE` to be used for generated scripts.

Figure 7.8. The **Model:MySQL** Preferences

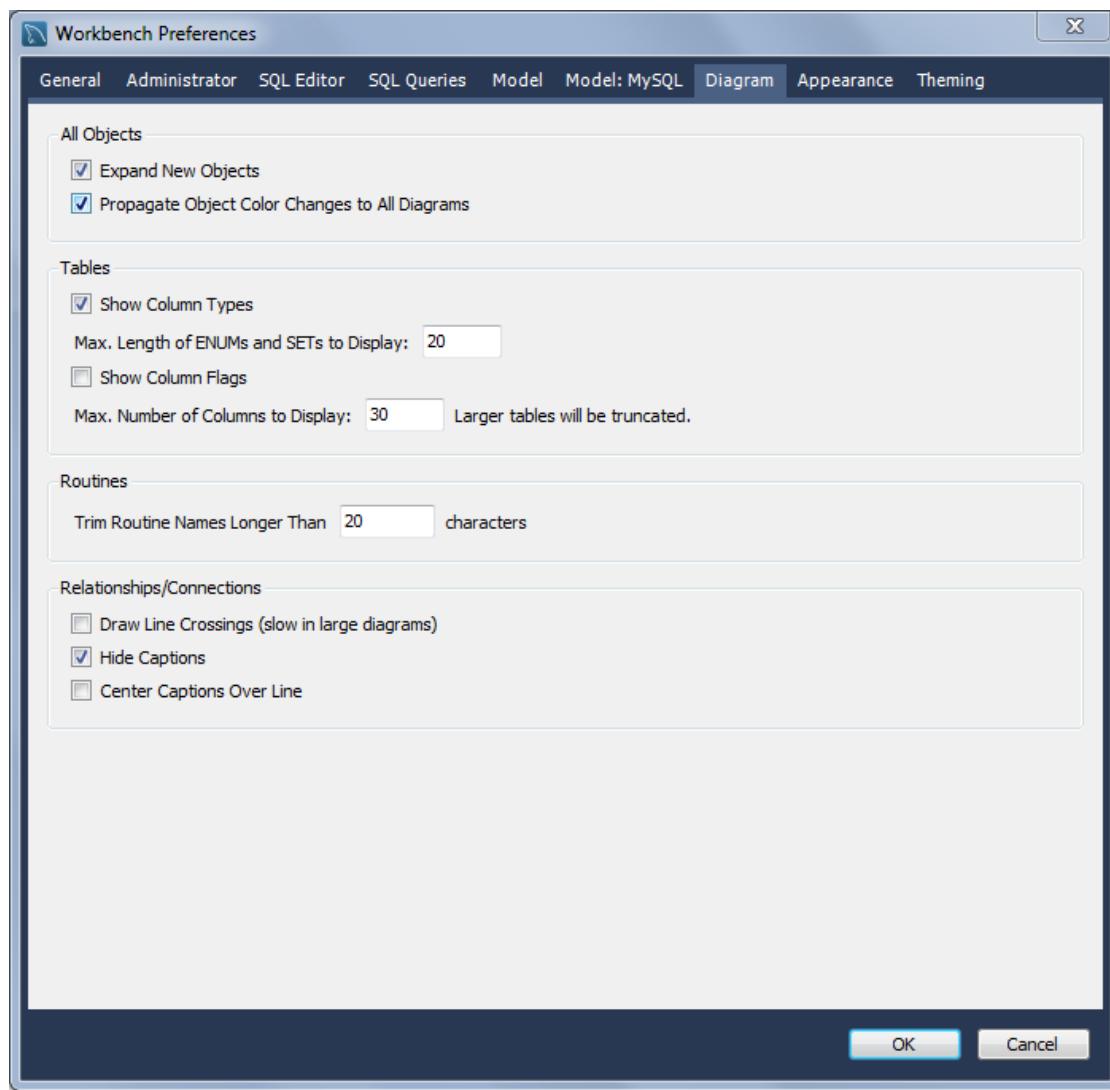


- **Default Target MySQL Version:** Certain features will utilize options for this MySQL Version.
- **Default Storage Engine:** Tables created in MySQL Workbench will be defined using this default storage engine.
- **SQL_MODE to be used in generated scripts:** Defaults to "TRADITIONAL,ALLOW_INVALID_DATES", this defines the `SQL_MODE` used by Forward Engineering and Synchronization.

7.3.7. The Diagram Tab

Use this tab to determine display settings for an EER diagram.

Figure 7.9. The **Diagram** Preferences



Select whether to expand new objects by checking the **Expand New Objects** check box and select whether to draw line crossings by checking the **Draw Line Crossings** check box.

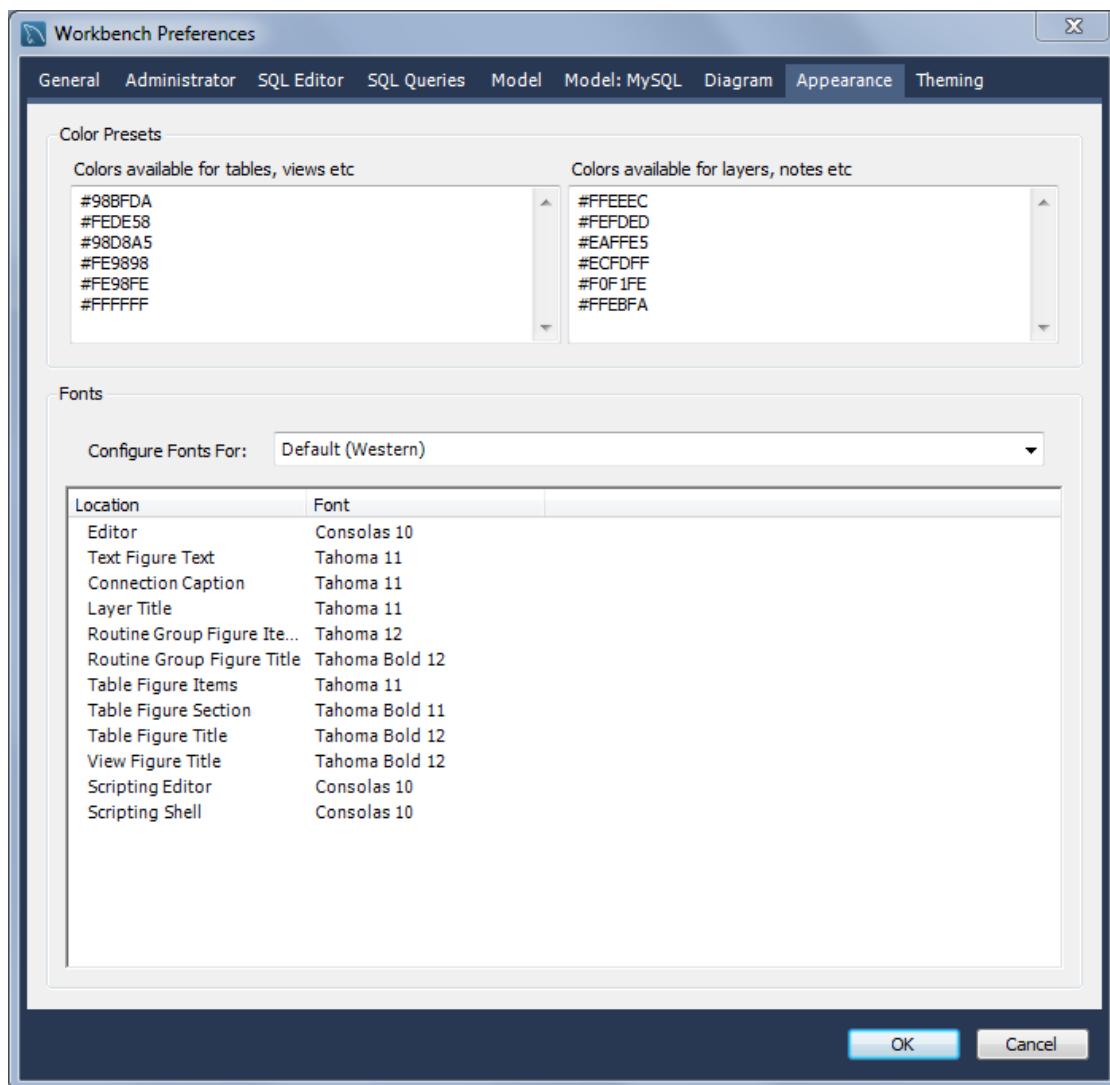
This tab also enables you to set the maximum number of characters for the following items:

- Column Names
- Column Types
- Routine Names

Changes to these values change the display properties only, not the objects themselves.

7.3.8. The Appearance Tab

Use this tab to set the available colors for the objects that appear on an EER diagram canvas. You can also add colors if you wish.

Figure 7.10. The Appearance Preferences

Changes made here affect the list of colors that appears on the toolbar when adding objects to an EER diagram canvas. For information about using this list, see [Section 9.1.2.1, “Tool-Specific Toolbar Items”](#).

You can also use this tab to set the font face, size, and style for the following items:

- Editor
- Layer Title
- Text Figure Text
- Text Figure Title
- Connection Caption
- Routine Group Figure Item
- Routine Group Figure Title
- Table Figure Items

- Table Figure Section
- Table Figure Title
- View Figure Title



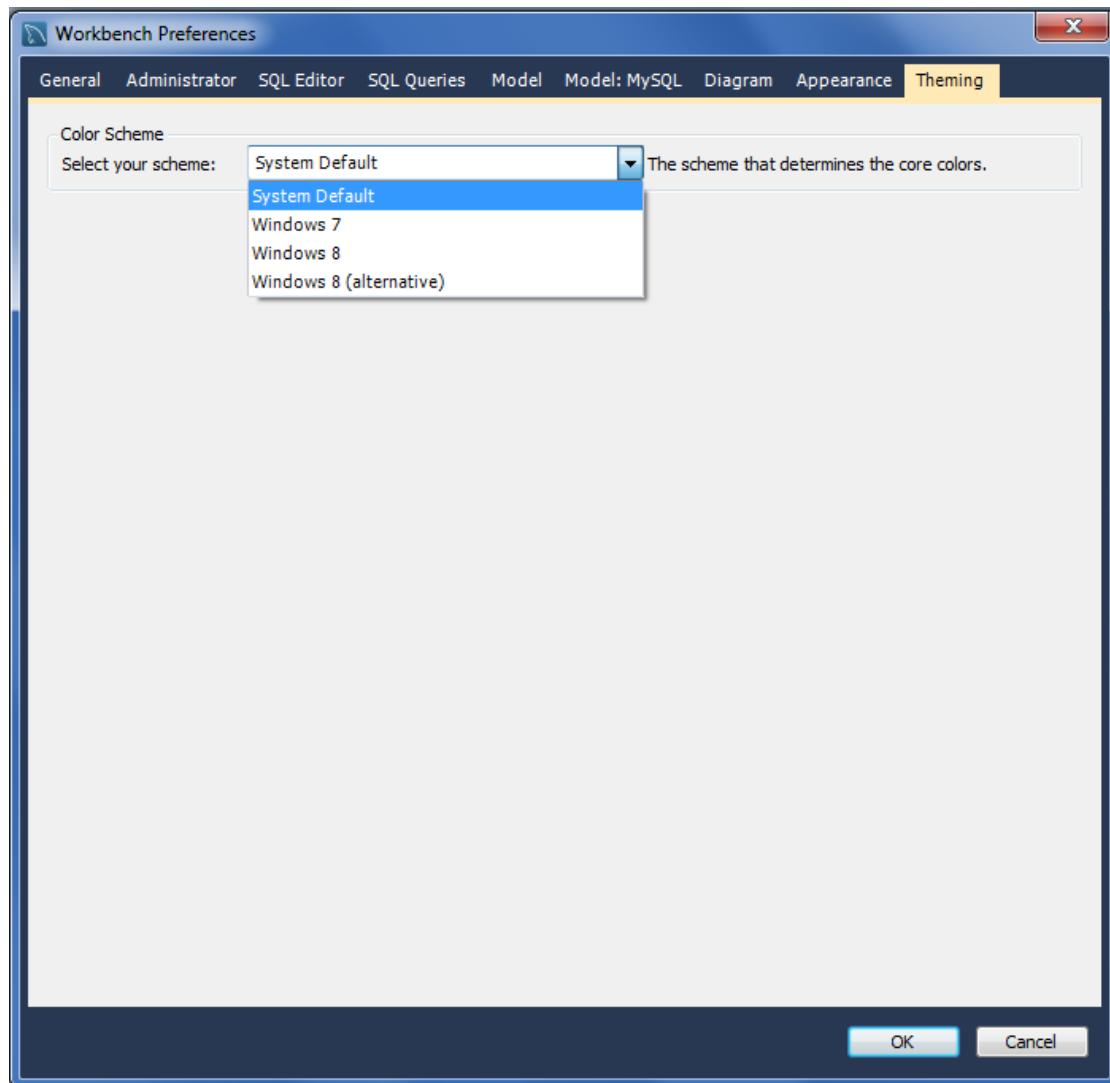
Note

On Windows, the default font for the editor supports only `latin-1` characters. If you need to use characters not supported by the `latin-1` character set, you must change the font here.

7.3.9. The Theming Tab

On Microsoft Windows, set the scheme that determines the code colors.

Figure 7.11. The Theming Preferences



Chapter 8. SQL Development

Table of Contents

8.1. Manage DB Connections Dialog	97
8.1.1. The Password Storage Vault	98
8.1.2. Standard TCP/IP Connection	99
8.1.3. Local Socket/Pipe Connection	100
8.1.4. Standard TCP/IP over SSH Connection	101
8.2. SQL Editor	101
8.2.1. Main Menu	102
8.2.2. Toolbar	103
8.2.3. SQL Query Panel	104
8.2.4. Sidebar	106

MySQL Workbench provides extensive facilities for working directly with SQL code. Before working directly with a live server, a connection must be created. After a connection is established, it is possible to execute SQL code directly on the server and manipulate the server using SQL code.

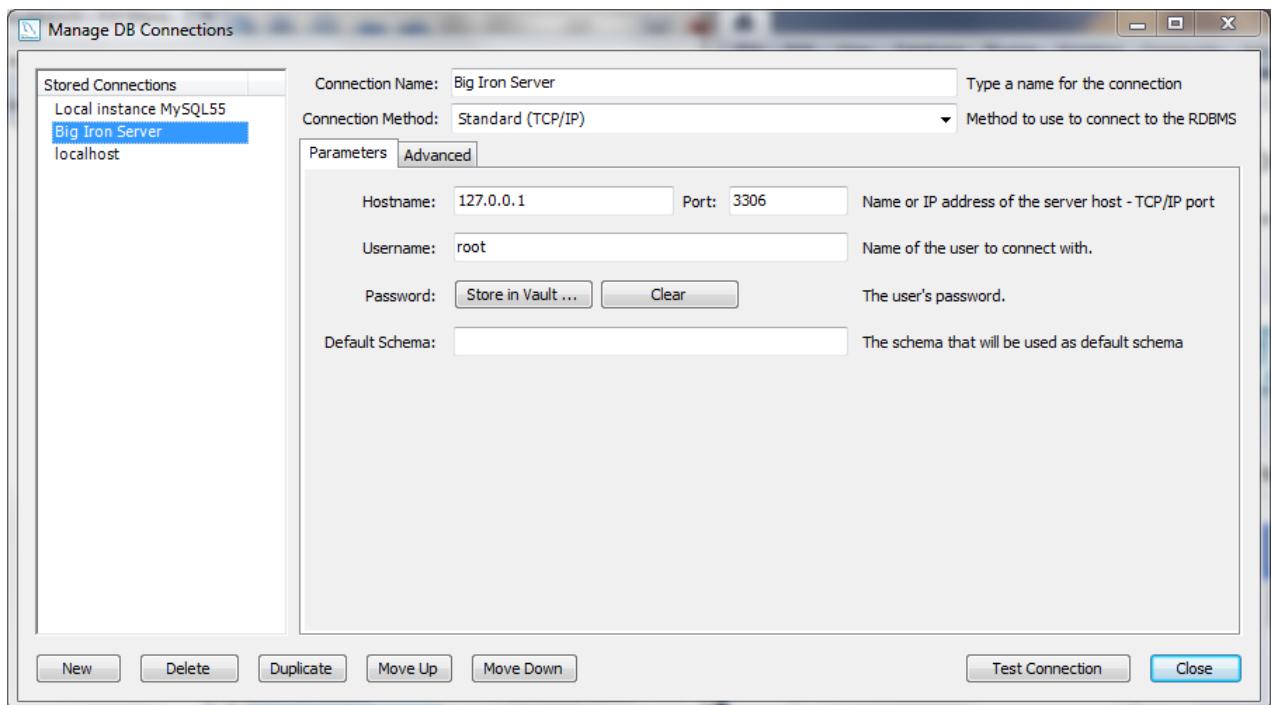
The starting point for embarking on SQL Development work is the SQL Development area of the [Home](#) window that lists your connections.

8.1. Manage DB Connections Dialog

MySQL Workbench provides a Manage DB Connections dialog for creating and managing connections to servers. The connections created can then be used from the wizards that must connect to a server, such as the wizard used to reverse engineer a live database. However, it is still possible to set connection parameters from these wizards if required, without invoking the Manage DB Connections dialog directly.

The Manage DB Connections dialog is invoked by selecting [Database](#), Manage Connections from the main menu. It can also be invoked from any of the wizards requiring access to a live database. This is achieved by using the **Manage Stored Connections** item, found in the wizard's **Stored Connection** list.

After the Manage DB Connections dialog is launched, you are presented with the following dialog, which enables you to create or delete connections.

Figure 8.1. Manage DB Connections - Dialog

Click **New** to create a new connection. Once created, the connection can be selected from the **Stored Connections** list. You can then set various parameters for the connection, including the following:

- **Connection Name:** The name used to refer to this connection. This connection can then be selected from a list in other wizards requiring a connection.
- **Connection Method:** The methods available are Standard TCP/IP, Local Socket/Pipe, and Standard TCP/IP over SSH.

After you select a connection method, the fields available in the **Parameters** tab and the **Advanced** tab of the dialog changes accordingly. More details about these options and parameters are available in the following sections.

After all parameters have been set as required, you can click the **Test Connection** button to test the connection to the live server. After you are satisfied that the connection works as expected, you can close the wizard by clicking the **Close** button. The stored connection then is available for use from any of the wizards requiring a connection to a live server.

You can duplicate an existing connection using the **Duplicate** button. This is an easy way to begin setting up a new connection that differs only slightly from an existing one.

8.1.1. The Password Storage Vault

The vault provides a convenient secure storage for passwords used to access MySQL servers. By using the vault, you need not enter credentials every time MySQL Workbench attempts to connect to a server. The vault is implemented differently on each platform:

- **Windows:** The vault is an encrypted file in the MySQL Workbench `data` directory. This is where `connections.xml` and related files are located. The file is encrypted using a Windows API which performs the encryption based on the current user, so only the current user can decrypt it. As a result it is not possible to decrypt the file on any other computer. It is possible to delete the file, in which case all

stored passwords are lost, but MySQL Workbench will otherwise perform as expected. You then must re-enter passwords as required.

- **Mac OS X:** The vault is implemented using the Mac OS X Secure Keychain. The keychain contents can be viewed using the [Keychain Access.app](#) utility.
- **Linux:** The vault works by storing passwords using the [gnome-keyring](#) daemon, which must be running for password persistency to work. The daemon is automatically started in GNOME desktops, but normally is not in KDE and others. The [gnome-keyring](#) daemon can be used for password storage in MySQL Workbench on non-GNOME platforms, but must be started manually.

8.1.2. Standard TCP/IP Connection

This connection method enables MySQL Workbench to connect to MySQL Server using TCP/IP.

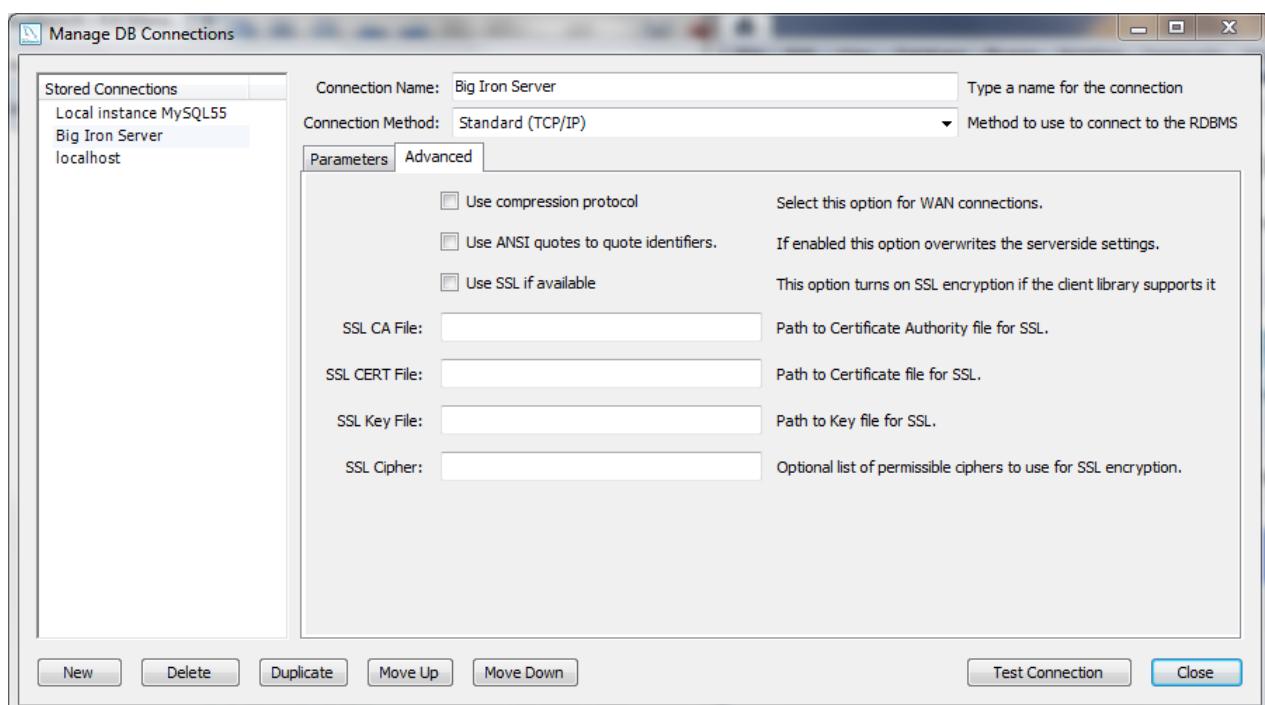
Parameters tab

- **Hostname:** The host name or IP address of the MySQL server.
- **Username:** User name to use for the connection.
- **Password:** Optional password for the account used. If you enter no password here, you will be prompted to enter the password when MySQL Workbench attempts to establish the connection. MySQL Workbench can store the password in a vault (see [Section 8.1.1, “The Password Storage Vault”](#)).
- **Port:** The TCP/IP port on which the MySQL server is listening (the default is 3306).
- **Default Schema:** When the connection to the server is established, this is the schema that will be used by default. It becomes the default schema for use in other parts of MySQL Workbench.

Advanced tab

More parameters can be set for the connection by using the **Advanced** tab.

Figure 8.2. Manage DB Connections - Advanced Tab



The **Advanced** tab includes these check boxes:

- **Use compression protocol:** If checked, the communication between the application and the MySQL server will be compressed, which may increase transfer rates. This corresponds to starting a MySQL command-line client with the `--compress` option.
- **Use SSL if available:** This option turns on SSL encryption. The client library must support this option. Note: This feature is currently not supported.
- **Use ANSI quotes to quote identifiers:** Treat ` as an identifier quote character (like the ` quote character) and not as a string quote character. You can still use ` to quote identifiers with this mode enabled. With this option enabled, you cannot use double quotation marks to quote literal strings, because it is interpreted as an identifier. Note: If this option is selected, it overrides the server setting.

8.1.3. Local Socket/Pipe Connection

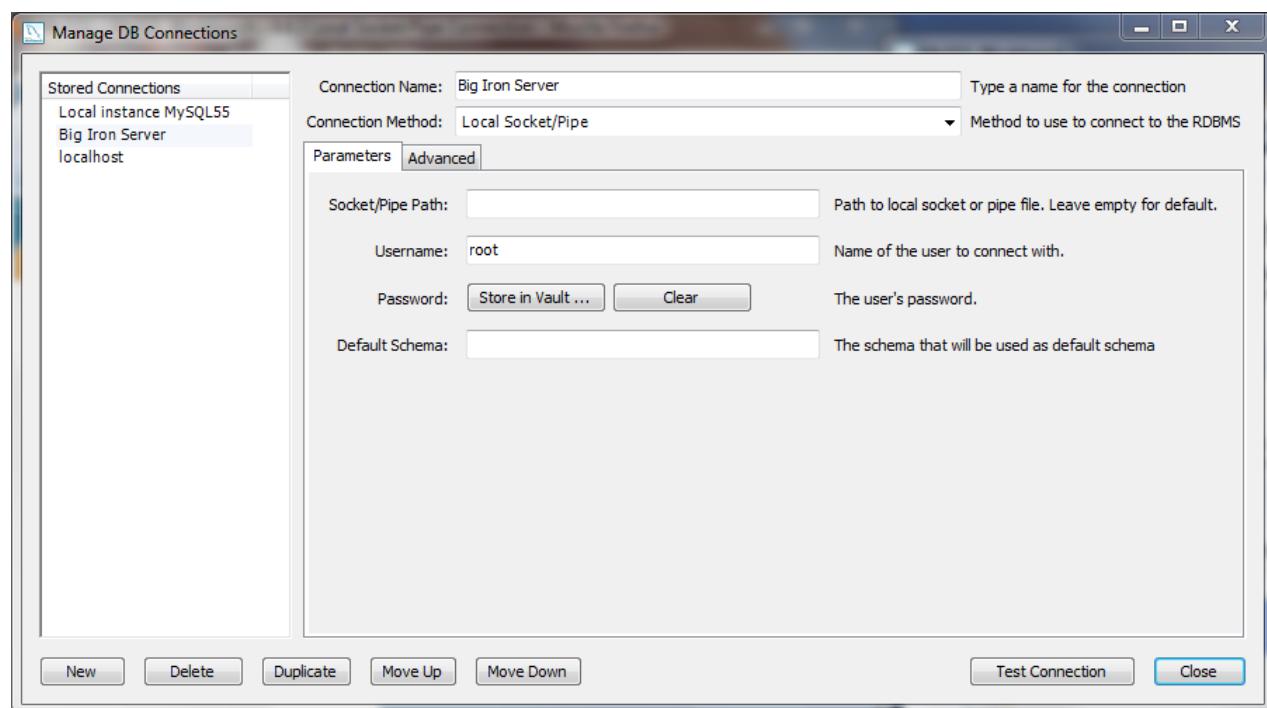
This connection method enables MySQL Workbench to connect to MySQL Server using a socket file (on Unix) or a named pipe (on Windows).

Parameters

The unique field here is **Socket/Pipe Path**. Enter the name of the socket or pipe here. If the field is left blank, the default socket or pipe name is used. On Unix, the default socket name is `/tmp/mysql.sock`. On Microsoft Windows, the default pipe name is `MySQL`.

This option can be seen in the following screenshot.

Figure 8.3. Manage DB Connections - Socket/Pipe Parameters



Advanced

The only option available in this tab is **Use ANSI quotes to quote identifiers**. This option was discussed in [Section 8.1.2, "Standard TCP/IP Connection"](#).

8.1.4. Standard TCP/IP over SSH Connection

This connection method enables MySQL Workbench to connect to MySQL Server using TCP/IP over an SSH connection.

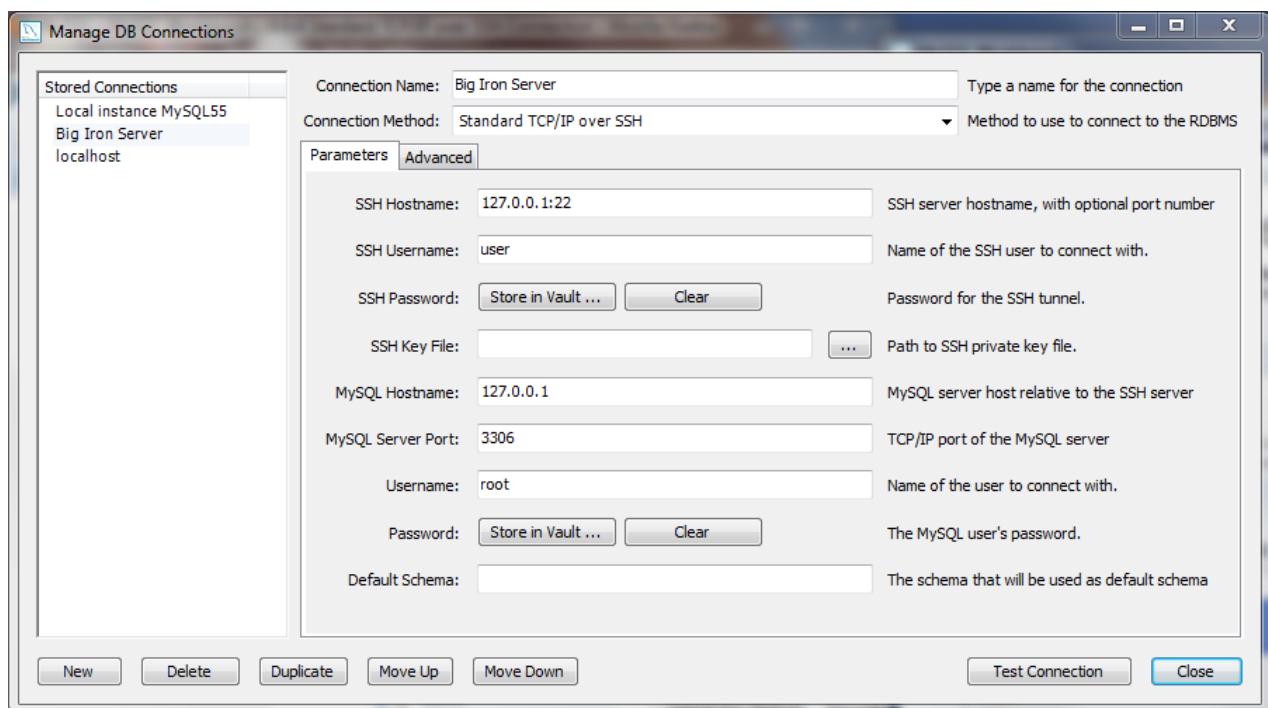
Parameters

In addition to a number of parameters that are in common with Standard TCP/IP connections, this connection method features a number of specialized parameters. These are listed here:

- **SSH Hostname:** This is the name of the SSH server. An optional port number can also be provided.
- **SSH Username:** This is the name of the SSH user name to connect with.
- **SSH Password:** The SSH password. It is recommended that an SSH key file is also used.
- **SSH Key File:** A path to the SSH key file. Note: Only key files in OpenSSH format are currently supported.

These options can be seen in the following screenshot.

Figure 8.4. Manage DB Connections - SSH Parameters



Advanced

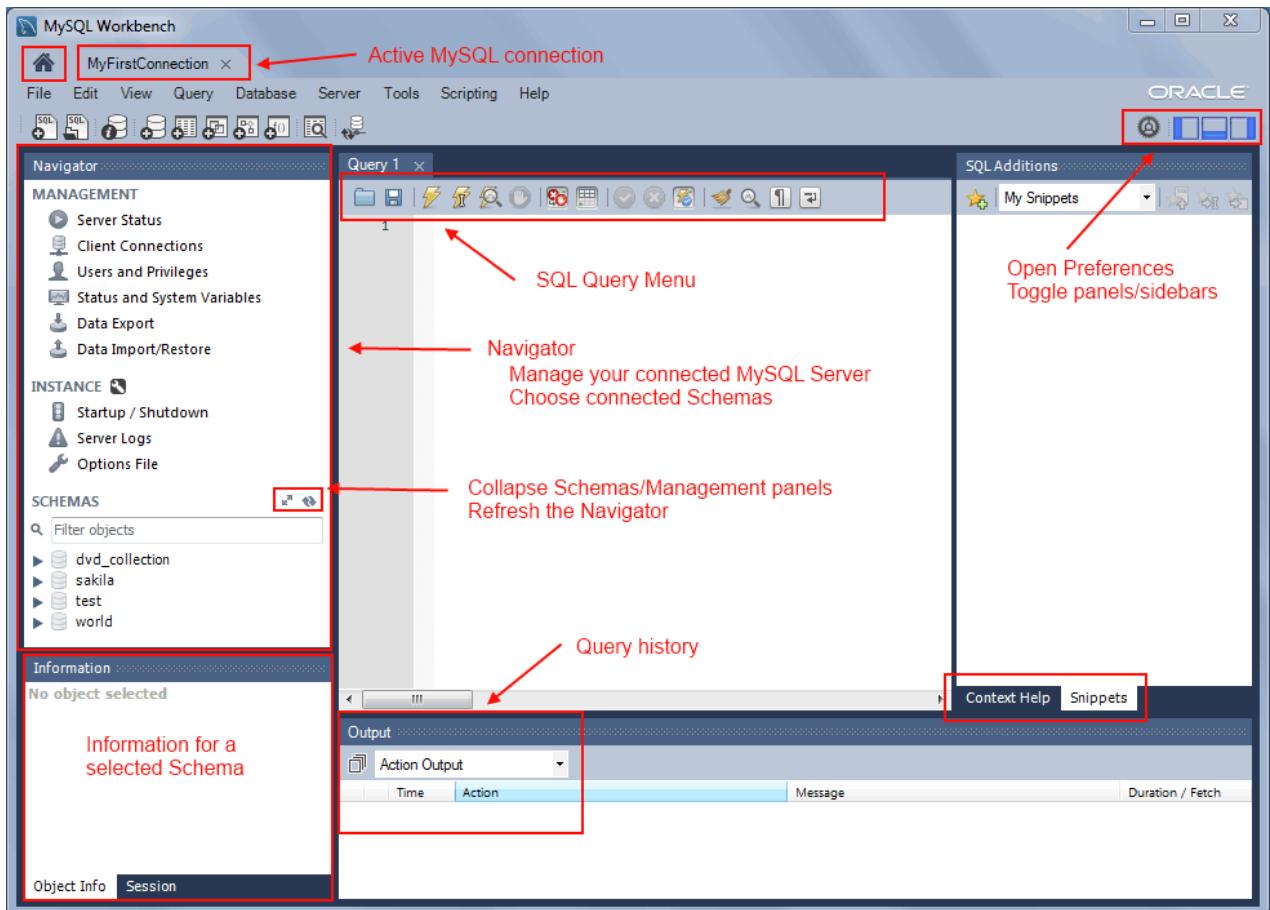
The options here are the same as for the Standard TCP/IP connection. See [Section 8.1.2, “Standard TCP/IP Connection”](#).

8.2. SQL Editor

The SQL Editor can be launched by clicking a connection tile on Home window, or by selecting Database, Connect to Database... from the main menu, or by using the keyboard shortcut **Control+U** on Windows, or

Command+U on Mac OS X. At this point, you will be asked to either select a stored connection or enter the details for a new connection. After a connection has been made to the server, a new tab called **SQL Editor (schema)** is displayed.

Figure 8.5. SQL Editor



The SQL Editor user interface, like most MySQL Workbench pages, has these main elements:

- Main Menu
- Toolbar
- SQL Query Panel
- Navigator sidebar - with Management, Instance, and Schema views
- Toggable Sidebar panels on the left (Navigator), right (SQL Additions), and bottom (Output)
- Auto-completion (in the Main Menu)

The following sections describe each of these elements.

8.2.1. Main Menu

When an SQL Editor tab is selected, the most important items on the main menu bar are the Query and Edit menus.

SQL Query Menu

The Query menu features the following items:

- Execute (All or Selection): Executes all statements in the SQL Query area, or only the selected statements.
- Execute Current Statement: Executes the current SQL statement.
- Explain (All or Selection): Describes all statements, or the selected statement.
- Explain Current Statement: Describes the current statement.
- Visual Explain Current Statement: Visually describes the current statement, based on EXPLAIN information provided by MySQL Server 5.6 and above.
- Stop: Stops executing the currently running script.
- Reconnect to Server: Reconnects to the MySQL server.
- New Tab: Creates a duplicate of the current SQL Editor tab.
- Commit Transaction: Commits a database transaction.
- Rollback Transaction: Rolls back a database transaction.
- Refresh: Synchronizes with the live server and refreshes views such as the live Overview tabsheet.
- Commit Result Edits: Commits any changes you have made to the server.
- Discard Result Edits: Discards any changes you have made.
- Export Results: Exports result sets to a file. Selecting this option displays the **Export Query Results to File** dialog. The dialog enables you to select which result set you wish to export, the file format (CSV, HTML, XML), and the name and location of the output file. Then click **Export** to export the data.

Edit Menu

The Edit menu features the Format submenu. The Format submenu includes the following menu items that are of importance when in SQL Editor mode:

- Beautify Query: Reformats the query selected in the query tab and lays it out in nicely indented fashion.
- UPCASE Keywords: Converts keywords to uppercase in the currently selected query in the query tab.
- lowercase Keywords: Converts keywords to lowercase in the currently selected query in the query tab.
- Un/Comment Selection: Comments the lines currently selected in the query tab. If the lines are already commented, this operation removes the comments.
- Auto-complete: Triggers the auto-completion wizard. This is enabled (and triggered) by default, and can be disabled with Preferences, SQL Editor, **Automatically Start Code Completion**. Auto-completion will list functions, keywords, schema names, table names and column names.

8.2.2. Toolbar

The toolbar features buttons in two locations, in the main toolbar and within the SQL Editor itself. The SQL Editor buttons are described below.

Figure 8.6. SQL Editor - Toolbar

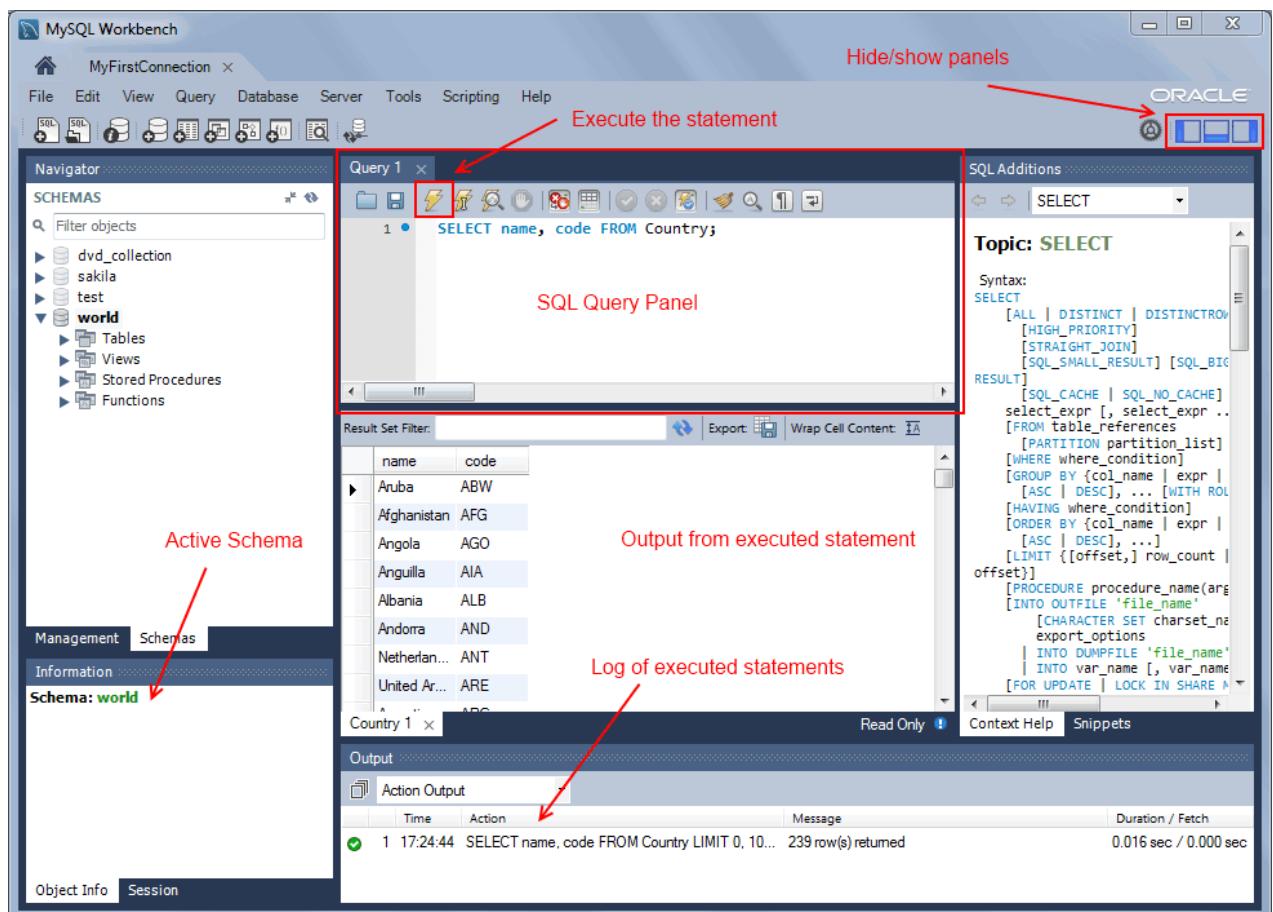
From left to right, these buttons are:

- **Open a SQL Script File:** Loads a saved SQL script to be ready for execution. The script is displayed in the **SQL Query** area.
- **Save SQL Script to File:** Saves the currently loaded SQL script to a file specified by the user.
- **Execute SQL Script:** Executes the selected portion of the query, or the entire query if nothing is selected.
- **Execute Current SQL script:** Execute the statement under the keyboard cursor.
- **Explain (All or Selection):** Execute the `EXPLAIN` command on the query under the keyboard cursor.
- **Stop the query being executed:** Halts execution of the currently executing SQL script. Note: the database connection will not be restarted, and open transactions will remain open.
- **Toggle whether execution of SQL script should continue after failed statements:** If the red “breakpoint” circle is displayed, the script terminates on a statement that fails. If the button is depressed so that the green arrow is displayed, execution continues past the failed code, possibly generating additional result sets. In either case, any error generated from attempting to execute the faulty statement is recorded in the Output tabsheet.
- **Commit:** Commits the current transaction. Note: All query tabs in the same connection share the same transactions. To have independent transactions, a new connection must be opened.
- **Rollback:** Rolls back the current transaction. Note: All query tabs in the same connection share the same transactions. To have independent transactions, a new connection must be opened.
- **Toggle Auto-Commit Mode:** If selected, each statement will be committed independently. Note: All query tabs in the same connection share the same transactions. To have independent transactions, a new connection must be opened.
- **Beautify SQL:** Beautify/reformat the SQL script.
- **Find panel:** Show the Find panel for the editor.
- **Invisible characters:** Toggle display of invisible characters, such as newlines, tabs, spaces.
- **Wrapping:** Toggles the wrapping of long lines in the SQL editor window.

8.2.3. SQL Query Panel

In this area, you can enter SQL statements directly. The statements entered can be saved to a file or snippet for later use. At any point, you can also execute the statements you have entered.

To save a snippet of code entered into the SQL Query panel, click the `Save SQL to Snippets List` icon in the Snippets panel, enter a name (optional), and click `OK`. The snippet can be inserted into the SQL Query panel at any time by double-clicking the snippet in the SQL Snippets panel.

Figure 8.7. SQL Editor - SQL Query Panel

Executing a `SELECT` query will display the associated result set in the SQL View panel, directly below the SQL Query panel. These cells are editable if MySQL Workbench is able to determine how, as for example they are editable if a Primary or Unique key exists within the result set. If not, MySQL Workbench will display a "read-only" icon at the bottom-right corner of the SQL View panel, and hovering the mouse cursor over this icon will provide a hint as to why it's not editable.



Note

To quickly enter the name of a table, view, or column, double-click the item in the Schemata Palette. The item name will be inserted into the SQL Query panel.

The SQL Editor has several configurable panels and windows, as described in the screenshot above.

Output and History

The **Output and History** panel is located at the bottom of MySQL Workbench. Its select box includes the `Action Output`, `Text Output`, and `History` options.

The **Action Output** panel displays a summary of the communication between the script and the server, and can refer to errors or general information. Each message displays the time, action, and server response. This output is useful for troubleshooting scripts.

The **Text Output** panel displays a textual representation of the query, as displayed using the MySQL Console. Use `Query`, `Execute (All or Selection)` to `Text` to send output to this panel.

The **History** panel provides a history of SQL operations carried out. The time and SQL code for each operation is recorded. To view the executed SQL statement, click the time, and the SQL code executed will be displayed in the **SQL** column.

Results Panel

The results area of the screen shows the results from executed statements. If the script contains multiple statements, a result tab will be generated for each statement that returned results.

The results panel offers the following options:

- **Result Set Filter**: performs a case-insensitive search of all cells. It automatically refreshes, and there is also the refresh button to perform this action manually.
- **Export**: Writes a result set to a CSV, HTML, or XML file as required.
- **Wrap Cell Content**: If the contents of a cell exceeds the cell width, then the data will be cut off with an ellipsis. This option will instead wrap the contents within the cell, and adjust the cell height accordingly.



Note

The "Refresh" button automatically adjusts the column width to match the longest string one of its cells. You may also manually adjust the column width.

8.2.4. Sidebar

The SQL Editor sidebar contains several different panels, and each panel has multiple tabs. The top-level panels are:

- **Navigator**: On the left side, it contains several options including:
 - **Management**: Options for managing the MySQL connection and its associated data, such as **Server Status**, **Users and Privileges**, and **Data Export**.
 - **Instance**: Options to interact with the MySQL instance, such as Start/Stop, and viewing the logs and options file.
 - **Schemas**: Schema information for the MySQL Connection. Optionally, the **Schemas** panel can be its own tab, or combined (default) with the **Navigator** sidebar.
 - MySQL Enterprise: With the Commercial version of MySQL Workbench, options include using the **Audit Inspector**, **Online Backup**, and **Backup Recovery** features.
- **Information**: This panel, on the bottom left sidebar, includes two tabs. The **Object Info** displays information about the selected Schema Object. For example, it displays the column structure if a table is selected, or a function definition for a function. The **Session** tab summarizes information about the object, such as the Connection name, login user, and port.
- **SQL Additions**: This includes the **Context Help** and **Snippets** tabs. For more information, see [Section 8.2.4.1, "Snippets tab"](#).

The following sections describe each panel in more detail.

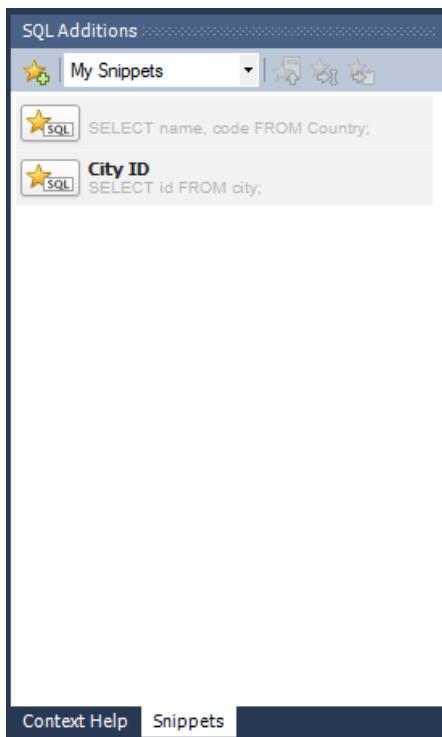
8.2.4.1. Snippets tab

The Snippets tab, which is a tab in the **SQL Additions** sidebar, offers both built-in and custom snippets. The tab contains a select box, with **My Snippets** for custom snippets, and built-in options titled **DB**

Mgmt (Database Management), **SQL DDL** (SQL Data Definition Language), and **SQL DML** (SQL Data Manipulation Language).

Snippets may be given names, and these snippets can be viewed and edited from the Snippets tab. To load a snippet into the SQL Query area, either choose the Snippets Insert icon or right-click on the desired snippet and choose Insert. Double-click a snippet to open an edit context, to edit the snippet body or title. This example shows two snippets, with only the first having defined a name.

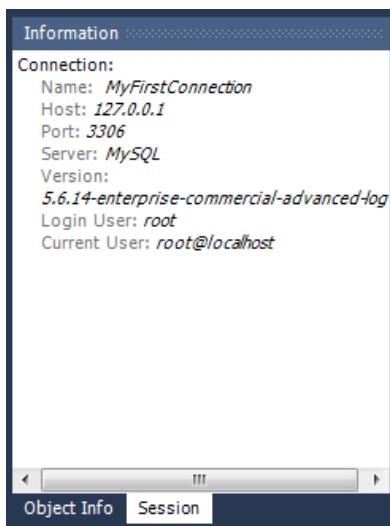
Figure 8.8. SQL Editor - Snippets Palette



8.2.4.2. Session and Object Information Panel

This panel summarizes the current connection to the server.

Figure 8.9. SQL Editor - Connection Information Palette



This panel also summarizes information about the object.

Figure 8.10. SQL Editor - Object Info



8.2.4.3. Navigator

The Navigator contains options to manage the active MySQL connection, and also lists the schemas available to that connection.

The **Navigator** is separated into two main sections that default to their own tabs within the Navigator. The **Management** tab offers management actions for the selected MySQL connection, and the **Schemas** tab lists the available schemas.

Navigator Schemas Tab

The Schemata list shows available schemata on the currently connected server. These can be explored to show tables, views, and routines within the schema.

Figure 8.11. SQL Editor - Navigator Schemas Tab



It is possible to set a schema as the default schema by right-clicking the schema and selecting the Set As Default Schema menu item. This executes a `USE schema_name` statement so that subsequent statements without schema qualifiers are executed against this schema. This setting applies only to the query session. To set a default schema for multiple MySQL Workbench sessions, you must set the default schema for the stored connection. From the Home screen, right-click on a MySQL connection, choose Edit Connection, and set the desired default schema on the **Default Schema** box.



Note

The selected schema is displayed as **bold** in the Schema navigator.

Double-clicking a table, view, or column name in the schemata explorer inserts the name into the SQL Query area. This reduces typing significantly when entering SQL statements containing references to several tables, views, or columns.

The Schema Navigator also features a context menu which can be displayed by right-clicking an object. For example, right-clicking a table displays the following menu items:

- Select Rows - Limit 1000: Pulls up to 1000 rows of table data from the live server into a Results tabsheet, and enables editing. Data can be saved directly to the live server.
- Copy to Clipboard: There are various submenus, each of which copies information to the clipboard:
 - Name (short): Copies the table name.
 - Name (long): Copies the qualified table name in the form ``schema`.`table``.
 - Select All Statement: Copies a statement to select all columns in this form:

```

SELECT
`table`.`column1`,
`table`.`column2`,
...
FROM `schema`.`table`;

```

- Insert Statement: Copies an `INSERT` statement to insert all columns.
- Update Statement: Copies an `UPDATE` statement to update all columns.
- Delete Statement: Copies a `DELETE` statement in the form `DELETE FROM `world`.`country` WHERE <{where_condition}>;`.
- Create Statement: Copies a `CREATE` statement in the form `DELETE FROM `world`.`country` WHERE <{where_condition}>;`.
- Delete with References: Copies a `DELETE` statement, in the form of a transaction, that deletes all objects that reference the row (directly or indirectly).

Use Select with References first to preview this operation.

- Select with References: Copies a `SELECT` statement that selects all objects that reference the row (directly or indirectly).

Use Delete with References to generate a `DELETE` statement for this operation.

- Send to SQL Editor: Provides functionality similar to Copy to Clipboard. However, this item inserts the SQL code directly into the SQL Query panel, where it can be edited further as required.
- Create Table: Launches a dialog to enable you to create a new table.
- Create Table Like...: Launches a dialog to enable you to create a new table, and to also apply predefined templates.
- Alter Table: Displays the table editor loaded with the details of the table.
- Table Maintenance: Opens a new tab for performing table maintenance operations. Operations include "Anylyze Table", "Optimize Table", "Check Table", and "Checksum Table". Additional information about the table may also be viewed from this tab.
- Drop Table: Drops the table. All data in the table will be lost if this operation is carried out.
- Truncate Table: Truncates the table.
- Search Table Data: Opens a new tab for performing table searches. It performs a search on all columns, and offers additional options to limit the search.
- Refresh All: Refreshes all schemata in the explorer by resynchronizing with the server.

Right-clicking on a schema provides similar options to the table context menu described above, but the operations refer to the Schema. For example, the **Table Maintenance** in the table context menu selects the table in the **Schema Inspector**, which is a schema context menu option.

Chapter 9. Data Modeling

Table of Contents

9.1. Model Editor	112
9.1.1. Modeling Menus	114
9.1.2. The Toolbar	124
9.1.3. EER Diagrams	124
9.1.4. The Physical Schemata Panel	125
9.1.5. The Schema Privileges Panel	125
9.1.6. The SQL Scripts Panel	127
9.1.7. The Model Notes Panel	127
9.1.8. The History Palette	127
9.1.9. The Model Navigator Panel	127
9.1.10. The Catalog Tree Palette	128
9.1.11. The Layers Palette	128
9.1.12. The Properties Palette	129
9.2. EER Diagram Editor	129
9.2.1. The Vertical Toolbar	129
9.3. Working with Models	133
9.3.1. Creating Tables	133
9.3.2. Creating Foreign Key Relationships	145
9.3.3. Creating Views	148
9.3.4. Creating Routines and Routine Groups	150
9.3.5. Creating Layers	153
9.3.6. Creating Notes	155
9.3.7. Creating Text Objects	155
9.3.8. Creating Images	156
9.3.9. Reverse Engineering	157
9.3.10. Forward Engineering	165
9.4. Modeling Tutorials	184
9.4.1. Importing a Data Definition SQL Script	184
9.4.2. Using the Default Schema	186
9.4.3. Basic Modeling	187
9.4.4. Documenting the <code>sakila</code> Database	188
9.5. Printing	190
9.5.1. Printing Options	190
9.6. MySQL Workbench Schema Validation Plugins (Commercial Version)	190
9.6.1. General Validation	190
9.6.2. MySQL-Specific Validation	191
9.7. The DBDoc Model Reporting Dialog Window (Commercial Version)	192
9.8. Customizing DBDoc Model Reporting Templates	195
9.8.1. Supported Template Markers	199
9.8.2. Creating a Custom Template	203

MySQL Workbench provides extensive capabilities for creating and manipulating database models, including these:

- Create and manipulate a model graphically
- Reverse engineer a live database to a model
- Forward engineer a model to a script or live database

- Create and edit tables and insert data

This is not an exhaustive list. The following sections discuss these and additional data-modeling capabilities.

The Home window is the typical starting point for work with data modeling. In the Data Modeling section of the Workspace, you can use the action items there to create and manage models, forward and reverse engineer, and compare and synchronize schemata:

- [Open an Existing EER Model](#): Either click on the model, or launch the [Open Workbench Model](#) and locate the model file to open.

If you have already created one or more model files, each will appear in the Model section of the Home page as an icon. Clicking the item of the model you wish to load creates a new MySQL Model tab and displays your model.

- [Create a new EER Model](#): Click the [+] icon, or choose [File](#), New Model from the main window.
- [Create EER Model from an Existing Database](#): Click the [>] icon from the Home window and choose this option.

This enables you to create an EER Model from an existing live database by launching the **Reverse Engineer Database** wizard. This is a multi-stage wizard that enables you to select a connection to a live server, and select the schema and objects you wish to reverse engineer into your new model. This is a convenient way to see how an existing database is structured.

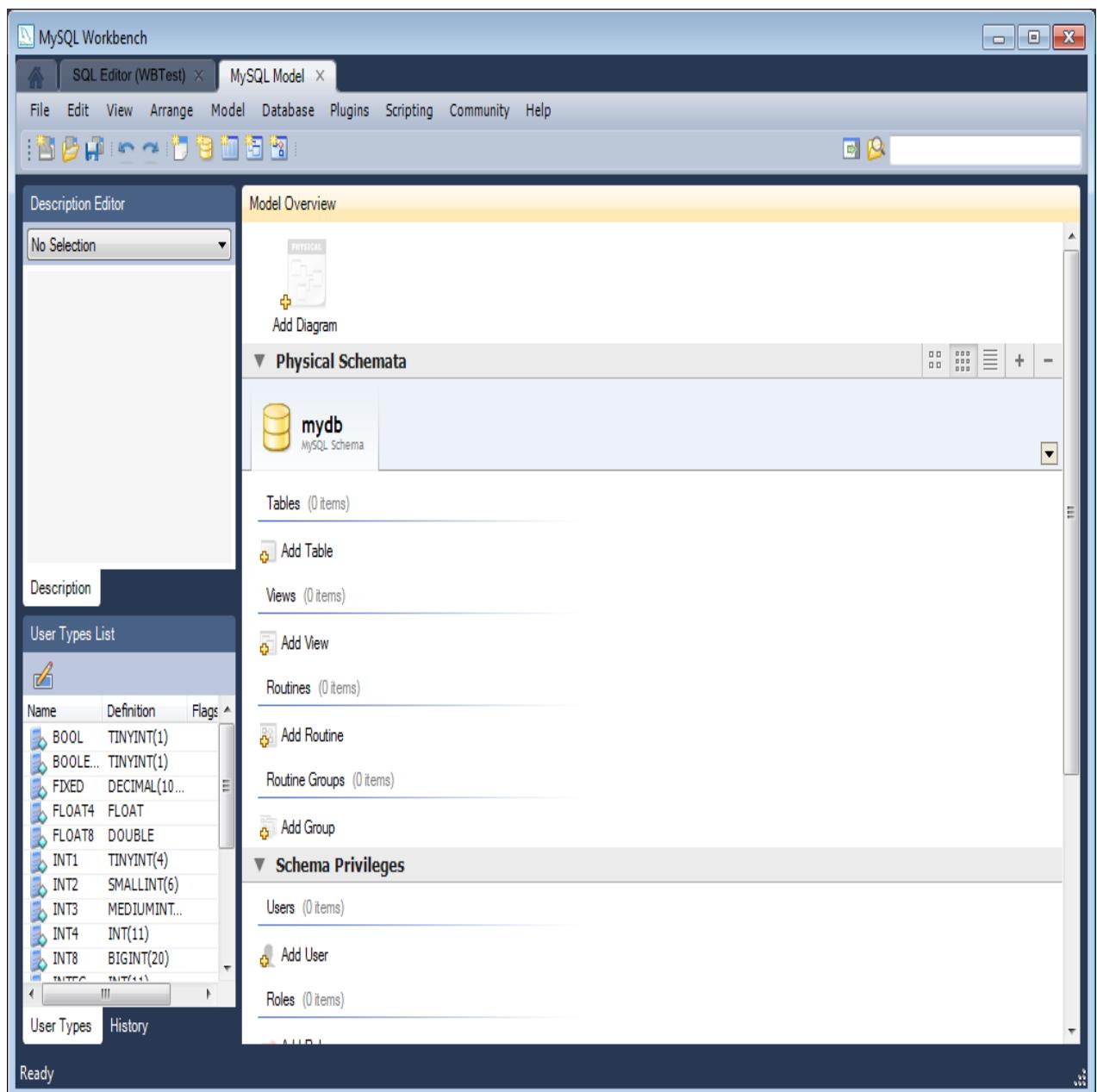
- [Create EER Model from SQL Script](#): Click the [>] icon from the Home window and choose this option.

This allows you to create a model from an SQL Create script. Such a script may have been created by hand or as a result of reverse engineering an existing database. The script may then be modified according to requirements. This also launches the **Reverse Engineer SQL Script** wizard. This is a multi-stage wizard that enables you to select the script you want to create your model from.

To read more about modeling, see [Section 9.1, “Model Editor”](#).

9.1. Model Editor

When the Model Editor is executed from the Home window, MySQL Workbench displays the MySQL Model page. The MySQL Model page has three main panels, as shown in the following screenshot: Description Editor, User Types List/History panel, and Model Overview.

Figure 9.1. The MySQL Model Page

The Description Editor and User Types List/History panel are contained within the Sidebar. The Sidebar is located on the left by default, but can be relocated to the right using a setting in the Workbench Preferences dialog.

The Model Overview panel has several sections:

- EER Diagrams
- Physical Schemata
- Schema Privileges
- SQL Scripts

- Model Notes

For each of these sections, add objects to a project by clicking the appropriate add-object icon. You may also rename, edit, cut, copy, or delete objects on this page by right-clicking to open a pop-up menu.

The following sections further discuss the MySQL Model page.

9.1.1. Modeling Menus

Some menu items are not available in the MySQL Workbench Community edition of this application, and are available only in MySQL Workbench Commercial. This is indicated where applicable.

9.1.1.1. The File Menu

Use the [File](#) menu to open a project, begin a new project, or save a project. Choosing New Model opens the default schema, `mydb`. Choosing Open Model opens a file dialog box with the default file type set to MySQL Workbench Models (`mwb` extension). To display a list of recently opened MWB files, choose the Open Recent menu item. The keyboard shortcut to create a new project is **Control+N** and the command to open an existing project is **Control+O**.

To close the currently active [MySQL Model](#) or [EER Diagram](#) tab, use the Close Tab menu item. You can also do this from the keyboard by pressing **Control+W**. To reopen the [MySQL Model](#) tab, see [Section 9.1.1.3, “The View Menu”](#). To reopen an [EER Diagram](#) tab, double-click the [EER Diagram](#) icon in the [EER Diagrams](#) section of the [MySQL Model](#) page.

Use the Save Model or Save Model As menu items to save a model. When you save a model, its name appears in the title bar of the application. If you have made changes to a project and have not saved those changes, an asterisk appears in the title bar following the model name. When you save a model, it is saved as a MySQL Workbench file with the extension `mwb`.

Use the Import menu item to import a MySQL data definition (DDL) script file. For example, this might be a file created by issuing the command `mysqldump --no-data`. MySQL Workbench handles the script as follows:

- If the script does not contain a `CREATE DATABASE db_name;` statement, the schema objects are copied to the default schema, `mydb`.
- If the script creates a database, a new tab bearing the database name is added to the [Physical Schemata](#) section of the [MySQL Model](#) page.
- If the script contains data, the data is ignored.

For details about importing a DDL script, see [Section 9.3.9.1, “Reverse Engineering Using a Create Script”](#).

Under the Import submenu, you can also import [DBDesigner4](#) files.

There are variety of items under the Export submenu. You may generate the SQL statements necessary to create a new database or alter an existing one. For more information about these menu items, see [Section 9.3.10.1, “Forward Engineering Using an SQL Script”](#).

Using the Export submenu, you can also export an EER diagram as a PNG, SVG, PDF, or Postscript file. For an example of a PNG file, see [Figure 9.48, ‘The sakila Database EER Diagram’](#).

The Page Setup menu item enables you to set the paper size, orientation, and margins for printing purposes.

The printing options are enabled only if the **EER Diagrams** tab is selected. You have the choice of printing your model directly to your printer, printing it as a PDF file, or creating a PostScript file. For more information, see [Section 9.5, “Printing”](#).

**Note**

The printing options are available only in commercial versions of MySQL Workbench.

Use the Document Properties menu item to set the following properties of your project:

- **Name**: The model name (default is `MySQL Model`)
- **Version**: The project version number
- **Author**: The project author
- **Project**: The project name
- **Created**: Not editable; determined by the MWB file attributes
- **Last Changed**: Not editable; determined by the MWB file attributes
- **Description**: A description of your project

9.1.1.2. The Edit Menu

Use the **Edit** menu to make changes to objects. The text description for several of the menu items changes to reflect the name of the currently selected object.

This menu has items for cutting, copying, and pasting. These actions can also be performed using the **Control+X**, **Control+C**, and **Control+V** key combinations. Undo a deletion using the Undo Delete '*object_name*' item. The **Control+Z** key combination can also be used to undo an operation. It is also possible to carry out a Redo operation using either the menu item, or the key combination **Control+Y**.

Also find a Delete '*object_name*' menu item for removing the currently selected object. The keyboard command for this action is **Control+Delete**. You can also right-click an object and choose the delete option from the pop-up menu.

The Delete '*object_name*' menu item behaves differently depending upon circumstances. For example, if an **EER Diagram** is active and a table on the canvas is the currently selected object, a dialog box may open asking whether you want to remove the table from the canvas only or from the database as well. For information about setting the default behavior when deleting from an EER Diagram, see [Section 7.3.5, “The Model Tab”](#).

**Warning**

If the `MySQL Model` page is active, the selected object is deleted from the catalog and there will be *no confirmation dialog box*.

Choose **Edit Selected** to edit the currently selected object. You can also perform edits in a new window by selecting **Edit Selected in New Window**. The keyboard shortcuts for **Edit Selected** and **Edit Selected in New Window** are **Control+E** and **Control+Shift+E**, respectively.

The **Select** item has the following submenus:

- **Select All** (Keyboard shortcut, **Control+A**): Selects all the objects on the active EER diagram.

- Similar Figures (Objects of the same type): Finds objects similar to the currently selected object.
- Connected Figures: Finds all the objects connected to the currently selected object.

These menu items are active only when an **EER Diagram** tab is selected. The Similar Figures and the Connected Figures menu items are disabled if no object is currently selected on an EER diagram.

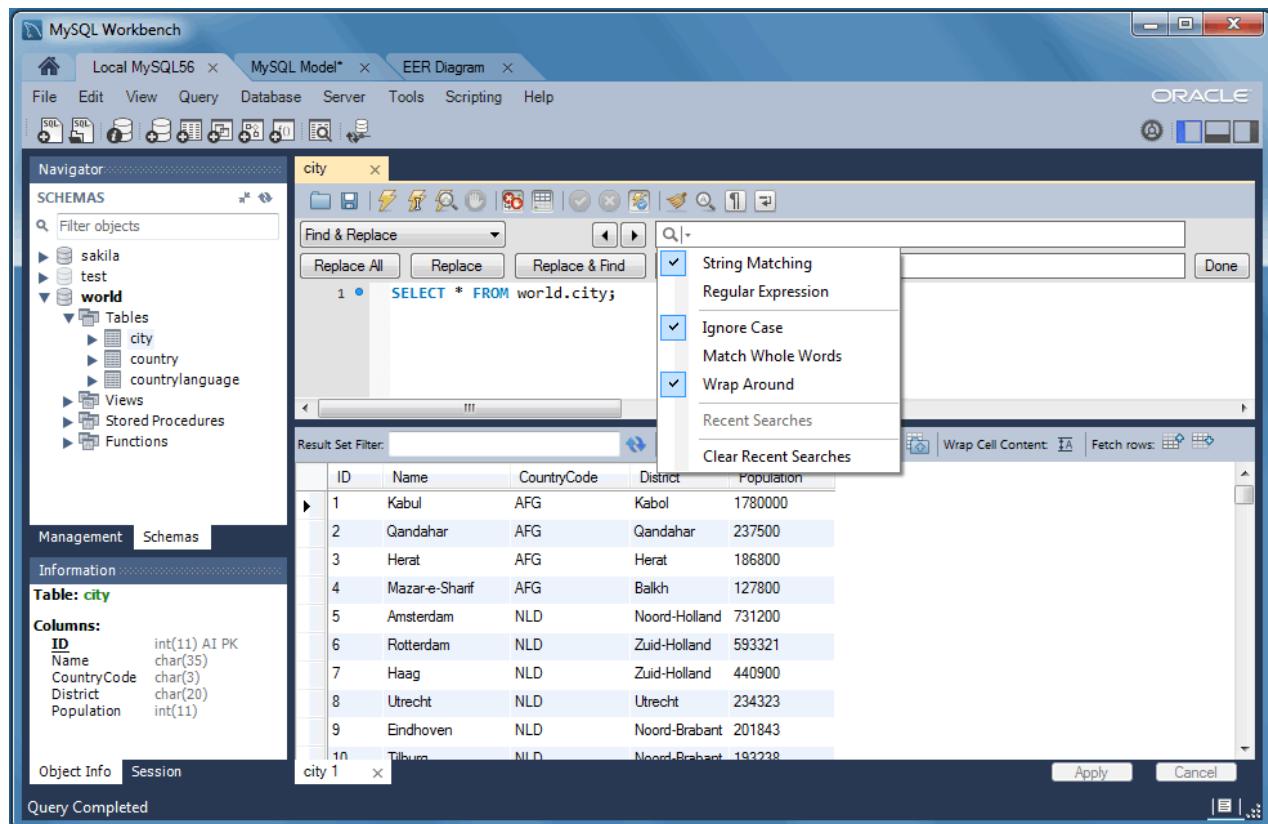
When multiple objects have been selected using one of these menu items, you can navigate between selected items by choosing the Go to Next Selected or Go to previous Selected menu item.

Selecting objects changes some of the **Edit** menu items. If only one object is selected, that object's name appears after the Cut, Copy and Delete menu items. If more than one object is selected, these menu items show the number of objects selected.

9.1.1.2.1. Find Dialog Window

Each MySQL Workbench window includes search functionality. The Find panel with Find & Replace enabled is shown below:

Figure 9.2. The Find Panel with Find & Replace



Find options

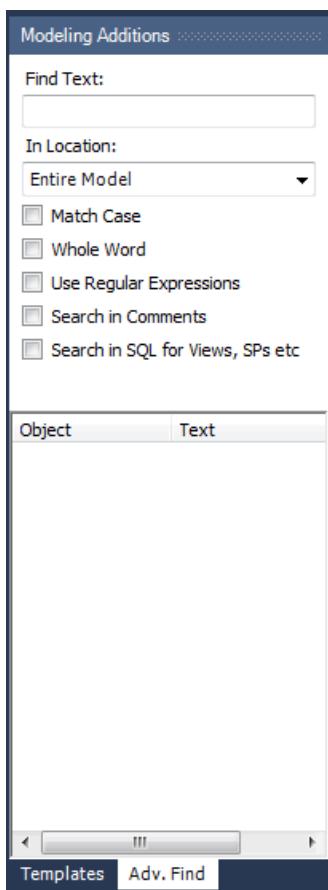
The Find dialogue options are described below:

- String Matching (default) or Regular Expression: Search by matching a string, or a PCRE regular expression.

- Ignore Case: A case-insensitive search. Works with both the String Matching and Regular Expression search methods. Enabled by default.
- Match Whole Words: If enabled, only whole strings are matched. For example, a search for "home" would not match "home_id". Disabled by default.
- Wrap Around: The search will wrap around to the beginning of the document, as otherwise it will only search from the cursor position to the end of the document. Enabled by default.
- And the arrows jump to the discovered search terms, and behave according to the Wrap Around option.

The MySQL Workbench Commercial edition includes an advanced Find facility for models:

Figure 9.3. The Find Window



You can search the following locations:

- Entire Model: Searches the entire model.
- Current View: Searches the current view only. This may be the [MySQL Model](#) page.
- All Views: Searches the [MySQL Model Page](#) and all EER diagrams.
- Database Objects: Searches database objects only.
- Selected Figures: Searches the currently selected objects. This feature works only for EER diagrams.

Enter the text you wish to search for in the **Find Text** list. You may also select any or all of the following check boxes:

- Match Case
- Whole Word
- Use Regular Expression
- Search in Comments
- Search in SQL for Views, SPs etc.

Any text you enter into the **Find Text** list is retained for the duration of your session. Use the **Next** or **Previous** buttons to find occurrences of your search criterion.

Clicking the **Find All** button opens a **Find Results** window anchored at the bottom of the application. If you wish, you may undock this window as you would any other.

Use this window to navigate to objects. For example, double-clicking the **Description** of an object located on an EER diagram navigates to the specific diagram and selects the object. Notice that the properties of the object are displayed in the **Properties** palette.

The **Find** dialog window can also be opened using the **Control+F** key combination. Use **Control+G** to find the next occurrence and **Control+Shift+G** to find a previous occurrence. Close the **Find** dialog window by clicking the **x** in the top right corner or by pressing the **Esc** key.

9.1.1.2.2. Workbench Preferences

This menu item enables you to set global preferences for the MySQL Workbench application.

For further information, see [Section 7.3, “Workbench Preferences”](#).

9.1.1.3. The View Menu

The **View** menu has these items:

- Home: Selects the Home window
- Windows: A submenu with items that provide a means for opening the windows associated with them:
 - Model Navigator: Opens the **Model Navigator** palette
 - Catalog: Opens the **Catalog** palette
 - Layers: Opens the **Layers** palette
 - User Datatypes: Opens the **User Datatypes** palette
 - Object Descriptions: Opens the **Description** palette
 - Object Properties: Opens the **Properties** palette
 - Undo History: Opens the **History** palette
- Output: Displays the console output. The keyboard shortcut for this menu item is **Control+F2**.
- Reset Window Layout: Resets all windows to their default layout

- Zoom 100%: The default level of detail of an EER diagram
- Zoom In: Zooms in on an EER diagram.
- Zoom Out: Zooms out from an EER diagram.

The ability to zoom in on an EER diagram is also available using the slider tool in the [Model Navigator](#) palette. See [Section 9.1.9, “The Model Navigator Panel”](#).

- Set Marker: Bookmarks an object. From the keyboard, select the object you wish to bookmark, then use the key combination **Control+Shift** and the number of the marker (1 through 9). You may create up to nine markers.
- Go To Marker: Returns to a marker. From the keyboard, use the **Control** key and the number of the marker.
- Toggle Grid: Displays grid lines on an EER diagram.
- Toggle Page Guides: Toggles Page Guides.

9.1.1.4. The Arrange Menu

The Arrange menu items apply only to objects on an EER diagram canvas and are enabled only if an EER diagram view is active. The Arrange menu has these items:

- Align to Grid: Aligns items on the canvas to the grid lines
- Bring to Front: Brings objects to the foreground
- Send to Back: Sends objects to the background
- Center Diagram Contents: Centers objects on the canvas
- Autolayout: Automatically arranges objects on the canvas
- Reset Object Size: Expands an object on an EER diagram. For example, if a table has a long column name that is not fully displayed, this menu item expands the table to make the column visible. This menu item is not enabled unless an object is selected.
- Expand All: Use this item to expand all objects on an EER diagram. This item will display a table's columns if the object notation supports expansion. Some object notations, such as [Classic](#), do not permit expansion or contraction. Indexes will not automatically be expanded unless they were previously expanded and have been collapsed using the Collapse All menu item.
- Collapse All: Undo the operation performed by Expand All.

9.1.1.5. The Model Menu

The Model menu has these items:

- Add Diagram: Creates a new EER Diagram. The keyboard shortcut is **Control+T**.
- Create Diagram From Catalog Objects: Creates an EER diagram from all the objects in the catalog.
- DBDoc – Model Reporting...: For information about this menu item, see [Section 9.1.1.5.1, “The DBDoc Model Reporting Dialog Window \(Commercial Version\)”](#). Commercial version only.
- User Defined Types: Presents a dialog box that enables you to add and delete user defined data types.

- Object Notation: For information about this menu item, see [Section 9.1.1.5.3, “The Object Notation Submenu”](#).
- Relationship Notation: For information about this menu item, see [Section 9.1.1.5.4, “The Relationship Notation Submenu”](#).
- Diagram Properties and Size: Opens a diagram size dialog box that enables you to adjust the width or height of the canvas. The unit of measure is pages; the default value is two.

When you have tables with numerous columns, use this menu item to increase the size of the EER.

- Validation: For information about this menu item, see [Section 9.1.1.5.2, “The Validation Submenus \(Commercial Version\)”](#). Commercial version only.
- Model Options: Sets options at the model level. These options should not be confused with the options that are set globally for the Workbench application, and which are referred to as Workbench Preferences. The available model options are a subset of the Workbench Preferences options.

For more information about Workbench Preferences, see [Section 7.3.5, “The Model Tab”](#).

9.1.1.5.1. The DBDoc Model Reporting Dialog Window (Commercial Version)

This dialog window is found by navigating to the [Model](#) menu and choosing the DBDoc - Model Reporting... item.



Note

The DBDoc - Model Reporting... item is not available in the MySQL Workbench Community version.

Use this dialog window to set the options for creating documentation of your database models. For more information, see [Section 9.7, “The DBDoc Model Reporting Dialog Window \(Commercial Version\)”](#).

9.1.1.5.2. The Validation Submenus (Commercial Version)

The [Model](#) menu has two validation submenus, Validation and Validation (MySQL). Use these submenus for general validation and MySQL-specific validation of the objects and relationships defined in your model.



Note

These items are not available in the MySQL Workbench Community version.

The Validation submenu has these items:

- Validate All: Performs all available validation checks
- Empty Content Validation: Checks for objects with no content, such as a table with no columns
- Table Efficiency Validation: Checks the efficiency of tables, such as a table with no primary key defined
- Duplicate Identifiers Validation: Checks for duplicate identifiers, such as two tables with the same name
- Consistency Validation: Checks for consistent naming conventions
- Logic Validation: Checks, for example, that a foreign key does not reference a nonprimary key column in the source table

The Validation (MySQL) submenu has these items:

- Validate All: Performs all available validation checks
- Integrity Validation: Checks for invalid references, such as a table name longer than the maximum permitted
- Syntax validation: Checks for correct SQL syntax
- Duplicate Identifiers Validation (Additions): Checks for objects with the same name

For detailed information about validation, see [Section 9.6, “MySQL Workbench Schema Validation Plugins \(Commercial Version\)”](#).

9.1.1.5.3. The Object Notation Submenu

The items under the Object Notation submenu apply exclusively to an EER diagram. They are not enabled unless an EER diagram tab is selected.

The Object Notation submenu has these items:

- Workbench (Default): Displays table columns, indexes, and triggers
- Workbench (Simplified): Shows only a table's columns
- Classic: Similar to the [Workbench \(Simplified\)](#) style showing only the table's columns
- IDEF1X: The ICAM DEFinition language information modeling style

The object notation style that you choose persists for the duration of your MySQL Workbench session and is saved along with your model. When MySQL Workbench is restarted, the object notation reverts to the default.



Note

If you plan to export or print an EER diagram be sure to decide on a notation style first. Changing notation styles after objects have been placed on a diagram can significantly change the appearance of the diagram.

9.1.1.5.4. The Relationship Notation Submenu

The items under the Relationship Notation submenu apply exclusively to an EER diagram. They are not enabled unless an EER diagram tab is selected.

The Relationship Notation submenu has these items:

- Crow's Foot (IE): The default modeling style. For an example, see [Figure 9.45, “Adding Tables to the Canvas”](#).
- Classic: Uses a diamond shape to indicate cardinality.
- Connect to Columns
- UML: Universal Modeling Language style.
- IDEF1X: The ICAM DEFinition language information modeling method

To view the different styles, set up a relationship between two or more tables and choose the different menu items.

The relationship notation style that you choose persists for the duration of your MySQL Workbench session and is saved along with your model. When MySQL Workbench is restarted, the relationship notation reverts to the default, the [Crow's Foot](#) style.

**Note**

If you plan to export or print an EER diagram, be sure to decide on a notation style first. Changing notation styles after objects have been placed on a diagram can significantly change the appearance of the diagram.

9.1.1.6. The Database Menu

The [Database](#) menu has these items:

- Query Database: Launches the SQL Editor, which enables you to create SQL code and execute it on a live server. For more information, see [Section 8.2, “SQL Editor”](#).
- Manage Connections: Launches the Manage DB Connections dialog, which enables you to create and manage multiple connections. For more information, see [Section 8.1, “Manage DB Connections Dialog”](#).
- Reverse Engineer: Creates a model from an existing database. For more information, see [Section 9.3.9.2, “Reverse Engineering a Live Database”](#).
- Forward Engineer: Creates a database from a model. For more information, see [Section 9.3.10.2, “Forward Engineering to a Live Server”](#).
- Synchronize with Any Source: Allows you to compare a target database or script with the open model, external script, or a second database, and apply these changes back to the target.
- Synchronize Model: Synchronizes your database model with an existing database. For more information, see [Section 9.3.10.3, “Database Synchronization”](#).
- Compare Schemas: Compares your schema model with a live database or a script file. [Section 9.3.10.4, “Compare and Report Differences in Catalogs”](#).

9.1.1.7. The Tools Menu

The [Tools](#) menu lists any plugins that you may have installed. For more information about this menu, see [Section 13.3, “Plugins / Tools”](#).

9.1.1.8. The Scripting Menu

The [Scripting](#) menu has these items:

- Scripting Shell: Launches the MySQL Workbench Scripting Shell
- New Script: Opens a **New Script File** dialogue, with options to create a Python Script, Lua Script, Python Plugin, or Python Module.
- Open Script: Opens a **Open GRT Script** dialogue, which defaults to the Workbench scripts directory. Files are opened into the **Workbench Scripting Shell** window.
- Run Workbench Script File: Executes the specified script
- Install Plugin/Module File: Loads and installs a plugin or module file
- Plugin Manager: Displays information about the plugins that are installed, and allows disabling and uninstalling the plugins.

9.1.1.9. The Community Menu

The Community menu has the following items. Use them to go online and learn more about MySQL Workbench.

- Workbench Blog
- FAQs About Workbench
- Learn How To Code For Workbench
- Discuss Workbench Topics
- Contribute To Workbench

9.1.1.10. The Help Menu

The [Help](#) menu has the following items. Use them to go online and learn more about MySQL Workbench.

- Help Index: Opens a window showing the MySQL Workbench documentation. Read, search, or print the documentation from this window.
- MySQL.com Website: Opens your default browser on the MySQL Web site home page.
- Workbench Product Page: Opens your default browser on the MySQL Workbench product page.
- System Info: Displays information about your system, which is useful when reporting a bug. For more information, see [Section 9.1.1.10.1, “System Info”](#).
- Report a Bug: Opens a form to submit a bug to bugs.mysql.com, and optionally attaches the log file to the report. Additional information such as the MySQL Workbench version, configuration and data directory paths, operating system, and more, are appended to the report but is made private so only those with proper permissions (such as MySQL developers) can view this helpful debugging information.
- View Reported Bugs: Opens your default browser to see a list of current bugs.
- Locate log file: Opens up the directory that contains the MySQL Workbench log files.
- Check For Updates: Opens the MySQL Workbench website using your default browser, and checks for a newer version.
- About Workbench: Displays the MySQL Workbench [About](#) window.

9.1.1.10.1. System Info

Use the System Info menu item to display information about your system. This item is especially useful for determining your rendering mode. Sample output follows.

```
read_mysql_cfg_file  C:\Program Files\MySQL\MySQL Server 5.1\my.ini
[('tmp_table_size', '9M'),
('myisam_sort_buffer_size', '18M'),
('table_cache', '256'),
('read_rnd_buffer_size', '256K'),
('port', '3306'), ('max_connections', '100'),
('innodb_buffer_pool_size', '18M'),
('myisam_max_sort_file_size', '100G'),
('sql-mode', '"STRICT_TRANS_TABLES,NO_AUTO_CREATE_USER,NO_ENGINE_SUBSTITUTION"'),
('basedir', '"C:/Program Files/MySQL/MySQL Server 5.1/"'),
('default-character-set', 'latin1'),
('datadir', '"C:/ProgramData/MySQL/MySQL Server 5.1/Data/"),
('innodb_log_buffer_size', '1M'),
```

```
('innodb_log_file_size', '10M'),
('innodb_thread_concurrency', '8'),
('read_buffer_size', '64K'),
('innodb_additional_mem_pool_size', '2M'),
('thread_cache_size', '8'),
('innodb_flush_log_at_trx_commit', '1'),
('query_cache_size', '0'),
('sort_buffer_size', '256K'),
('default-storage-engine', 'INNODB'),
('key_buffer_size', '11M'])
MySQL Workbench OSS for Windows version 5.2.8
Cairo Version: 1.8.6
Rendering Mode: GDI requested (create a diagram to confirm)
OpenGL Driver Version: Not Detected
OS: unknown
CPU: Intel(R) Core(TM)2 Duo CPU      T9300 @ 2.50GHz, 1.0 GB RAM
Video adapter info:
Adapter type: VirtualBox Graphics Adapter
Chip Type: VBOX
BIOS String: Version 0xB0C2 or later
Video Memory: 12288 KB
```

9.1.2. The Toolbar

The MySQL Workbench toolbar is located immediately below the menu bar. Click the tools in the toolbar to perform the following actions:

- The new document icon: Creates a new document
- The folder icon: Opens a MySQL Workbench file ([mwb](#) extension)
- The save icon: Saves the current MySQL Workbench project
- The right and left arrows: The left arrow performs an “Undo” operation. The right arrow performs a “Redo” operation.

Other tools appear on the toolbar depending upon the context.

9.1.2.1. Tool-Specific Toolbar Items

When an EER diagram canvas is selected, the following icons appear to the right of the arrow icons:

- The toggle grid icon: Turns the grid on and off
- The grid icon: Aligns objects on the canvas with the grid
- The new EER diagram icon: Creates a new EER diagram tab.

The toolbar also changes depending upon which tool from the vertical toolbar is active. For discussion of these tools, see [Section 9.2.1, “The Vertical Toolbar”](#).

If the [Table](#) tool is active, schemata lists, engine types, and collations appear on the toolbar. The table properties can be modified using the Properties Editor.

When an object is selected, the object's properties, such as color, can be changed in the Properties Editor.

9.1.3. EER Diagrams

Use the [Add new Diagram](#) icon in the [MySQL Model](#) area to create EER diagrams. When you add an EER diagram, a new tab appears below the toolbar. Use this tab to navigate to the newly created EER diagram. For further discussion of EER Diagrams, see [Section 9.2, “EER Diagram Editor”](#).

9.1.4. The Physical Schemata Panel

The [Physical Schemata](#) panel of the [MySQL Model](#) page shows the active schemata and the objects that they contain.

Expand and contract the [Physical Schemata](#) section by double-clicking the arrow on the left of the [Physical Schemata](#) title bar. When the [Physical Schemata](#) section is expanded, it displays all currently loaded schemata.

Each schema shows as a tab. To select a specific schema, click its tab. When MySQL Workbench is first opened, a default schema, `mydb`, is selected. You can start working with this schema or you can load a new MySQL Workbench Model file (models use the `.mwb` extension.)

There are a variety of ways to add schema to the [Physical Schemata](#) panel. You can open an MWB file, reverse engineer a MySQL create script, or, if you are using a commercial version of MySQL Workbench, you can reverse engineer a database by connecting to a MySQL server.

You can also add a new schema by clicking the `+` button on the top right of the [Physical Schemata](#) panel. To remove a schema, click its tab and use the `-` button found to the immediate left of the `+` button. To the left of these buttons are three buttons that control how database object icons are displayed:

- The left button displays database objects as large icons.
- The middle button displays small icons in multiple rows.
- The right button displays small icons in a single list.

9.1.4.1. The Schema Objects Panel

The [Physical Schemata](#) panel has the following sections:

- Tables
- Views
- Routines
- Routine Groups

Each section contains the specified database objects and an icon used for creating additional objects.

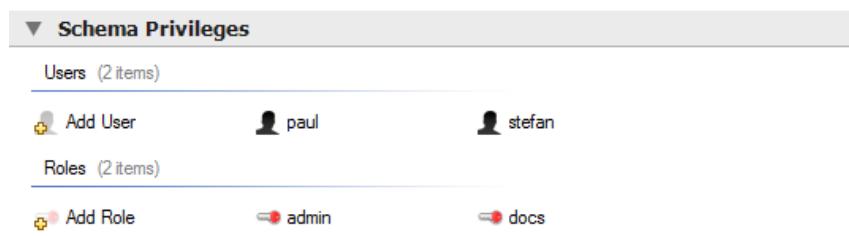
Any database objects added to an EER diagram canvas also show up in the [Physical Schemata](#) section. For information about adding objects to an EER diagram canvas, see [Section 9.2, “EER Diagram Editor”](#).

9.1.5. The Schema Privileges Panel

The [Schema Privileges](#) panel has the following sections, used to create users for your schemata and to define roles —:

- Users
- Roles

The following image displays the [Schema Privileges](#) section of the [MySQL Model](#) tab.

Figure 9.4. Roles and Privileges

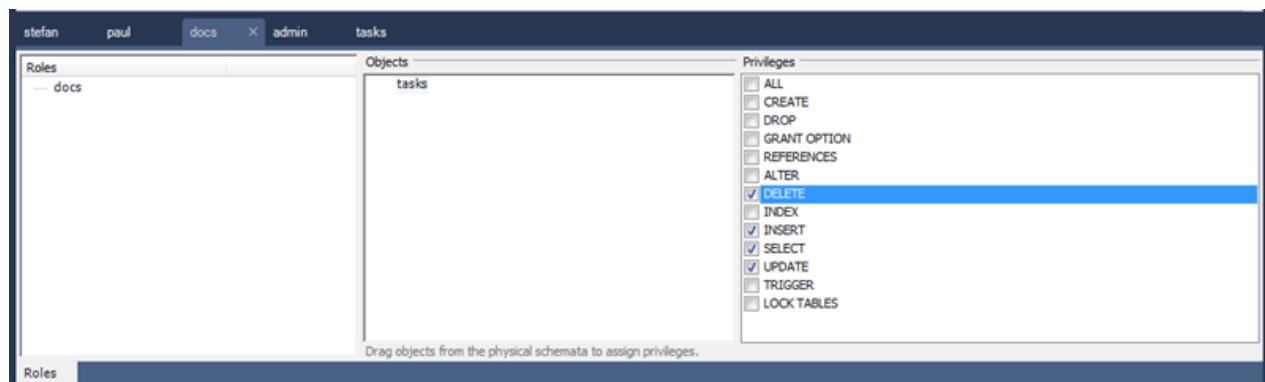
9.1.5.1. Adding Roles

To add a role, double-click the [Add Role](#) icon. This creates a role with the default name `role1`. Right-clicking a role opens a pop-up menu with the following items:

- Cut '`role_name`': Cuts the role
- Copy '`role_name`': Copies the role
- Edit Role...: Opens the role editor
- Edit in New Window...: Opens the role editor in a new editor window
- Delete '`role_name`': Removes the role
- Copy SQL to Clipboard: Currently not implemented

To rename a role, click the role name. Then you will be able to edit the text.

All roles that have been defined are listed under [Roles](#) on the left side of the role editor. Double-clicking a role object opens the role editor docked at the bottom of the page.

Figure 9.5. Role Editor

Select the role to which you wish to add objects. You may drag and drop objects from the [Physical Schemata](#) to the [Objects](#) section of the role editor. To assign privileges to a role, select it from the [Roles](#) section, then select an object in the [Objects](#) section. In the [Privileges](#) section, check the rights you wish to assign to this role. For example, a `web_user` role might have only `SELECT` privileges and only for database objects exposed through a web interface. Creating roles can make the process of assigning rights to new users much easier.

9.1.5.2. Adding Users

To add a user, double-click the [Add User](#) icon. This creates a user with the default name `user1`. Double-clicking this user opens the user editor docked at the bottom of the application.

In the [User Editor](#), set the user's name and password using the **Name** and **Password** fields. Assign one role or a number of roles to the user by selecting the desired roles from the field on the right and then clicking the < button. Roles may be revoked by moving them in the opposite direction.

Right-clicking a user opens a pop-up menu. The items in the menu function as described in [Section 9.1.5.1, "Adding Roles"](#).

9.1.6. The SQL Scripts Panel

Use the [SQL Scripts](#) panel to load and modify SQL scripts. If you created your project from an SQL script and plan to create an [ALTER](#) script, you may want to add the original script here, since it will be needed to create an [ALTER](#) script. For more information, see [Section 9.3.10.1.2, "Altering a Schema"](#).

9.1.7. The Model Notes Panel

Use the [Model Notes](#) panel to write project notes. Any scripts or notes added will be saved with your project.

9.1.8. The History Palette

Use the [History](#) palette to review the actions that you have taken. Left-clicking an entry opens a pop-up menu with the item, Copy History Entries to Clipboard. Choose this item to select a single entry. You can select multiple contiguous entries by pressing the **Shift** key and clicking the entries you wish to copy. Select noncontiguous entries by using the **Control** key.

Only actions that alter the MySQL model or change an EER diagram are captured by the [History](#) palette.

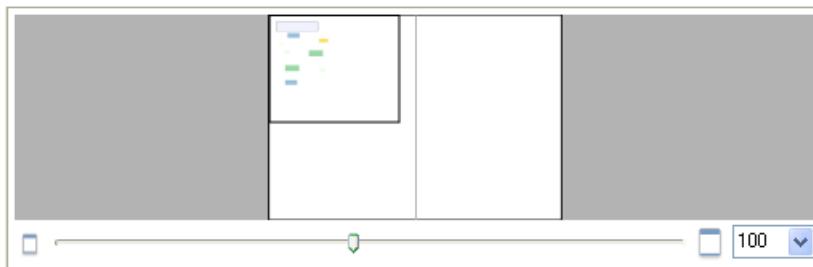
9.1.9. The Model Navigator Panel

Docked at the top left of the application is the **Model Navigator**, or **Bird's Eye** panel. This panel provides an overview of the objects placed on an EER diagram canvas and for this reason it is most useful when an EER diagram is active. Any objects that you have placed on the canvas should be visible in the navigator.

The Model Navigator shows the total area of an EER diagram. A black rectangular outline indicates the view port onto the visible area of the canvas. To change the view port of an EER diagram, left-click this black outline and drag it to the desired location. You can zoom in on selected areas of an EER diagram by using the slider tool at the bottom of this window. The dimensions of the view port change as you zoom in and out. If the slider tool has the focus, you can also zoom using the arrow keys.

The default size of the [Model Navigator](#) is two pages. To change this, use the [Model](#) menu, Diagram Size menu item.

Figure 9.6. The Model Navigator Palette



9.1.10. The Catalog Tree Palette

The [Catalog Tree](#) palette shows all the schemata that are present in the [Physical Schemata](#) section of the [MySQL Model](#) page. Expand the view of the objects contained in a specific schema by clicking the **+** button to the left of the schema name. This displays the following folder icons:

- Tables
- Views
- Routine Groups

Expand each of these in turn by clicking the **+** button to the left of the folder icon.

Selecting an object in this palette displays its properties in the [Properties](#) palette, which can be found in the lower left corner of the page.

The Catalog Tree palette is primarily used to drag and drop objects onto an EER diagram canvas.



Note

On Linux, there is a quirk in the GTK tree control, where a simple click always generates a new selection. To drag multiple objects from the Catalog Tree to the EER diagram canvas, you must perform the operation as follows:

1. Click the first item in the tree.
2. Hold the **Shift** key, click the last item, and *do not release the Shift key*.
3. Keep the **Shift** key depressed and commence the dragging operation.
4. Release the **Shift** key before you release the mouse button to drop selected objects onto the canvas.

This procedure also applies to use of the **Control** key when selecting multiple nonadjacent elements in the Catalog Tree.

You can toggle the sidebar on and off using the **Toggle Sidebar** button, which is located in the top right of the application.

9.1.11. The Layers Palette

This palette shows all the layers and figures that have been placed on an EER diagram. If a layer or figure is currently selected, an **X** appears beside the name of the object and its properties are displayed in the [Properties](#) palette. This can be especially useful in determining which objects are selected when you have selected multiple objects using the various options under the Select menu item. For more information on this topic, see [Section 9.1.1.2, “The Edit Menu”](#).

Selecting an object in the [Layers](#) palette also adjusts the view port to the area of the canvas where the object is located.

9.1.11.1. Finding Invisible Objects Using the Layers Palette

In some circumstances, you may want to make an object on an EER diagram invisible. Select the object and, in the [Properties](#) palette, set the **visible** property to **False**.

The [Layer](#) palette provides an easy way to locate an object, such as a relationship, that has been set to **hidden**. Open the [Layers](#) palette and select the object by double-clicking it. You can then edit the object and change its visibility setting to **Fully Visible**.

9.1.12. The Properties Palette

The [Properties](#) palette is used to display and edit the properties of objects on an EER diagram. It is especially useful for editing display objects such as layers and notes.

All objects except connections have the following properties except as noted:

- [color](#): The color accent of the object, displayed as a hexadecimal value. Change the color of the object by changing this value. Only characters that are legal for hexadecimal values may be entered. You can also change the color by clicking the  button to open a color changing dialog box.
- [description](#): Applicable to layers only. A means of documenting the purpose of a layer.
- [expanded](#): This attribute applies to objects such as tables that can be expanded to show columns, indexes, and triggers.
- [height](#): The height of the object. Depending upon the object, this property may be read only or read/write.
- [left](#): The number of pixels from the object to the left side of the canvas.
- [locked](#): Whether the object is locked. The value for this attribute is either `true` or `false`.
- [manualSizing](#): Whether the object has been manually sized. The value for this attribute is either `true` or `false`.
- [name](#): The name of the object.
- [top](#): The number of pixels from the object to the top of the canvas.
- [visible](#): Whether the object shows up on the canvas. Use `'1'` for true and `'0'` for false. It is currently used only for relationships.
- [width](#): The width of the object. Depending upon the object, this property may be read only or read/write.

Tables have the following additional properties:

- [indexesExpanded](#): Whether indexes are displayed when a table is placed on the canvas. Use `'1'` for true and `'0'` for false.
- [triggersExpanded](#): Whether triggers are displayed when a table is placed on the canvas. Use `'1'` for true and `'0'` for false.

For a discussion of connection properties, see [Section 9.3.2.3, “Connection Properties”](#).

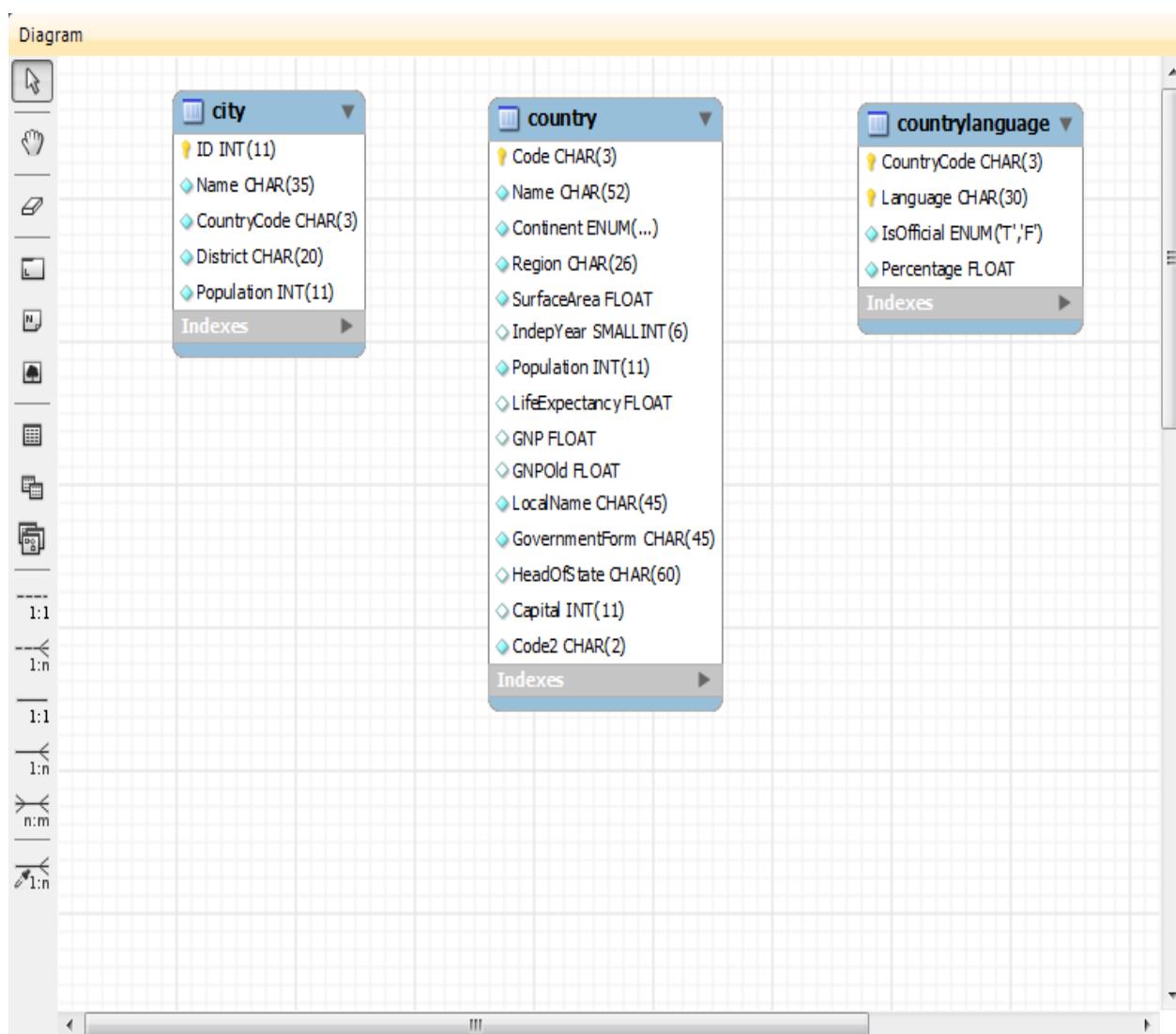
9.2. EER Diagram Editor

EER diagrams are created by double-clicking the [Add Diagram](#) icon. You may create any number of EER diagrams just as you may create any number of physical schemata. Each EER diagram shows as a tab below the toolbar; a specific EER diagram is selected by clicking its tab.

Clicking an EER diagram tab navigates to the canvas used for graphically manipulating database objects. The [Vertical Toolbar](#) is on the left side of this page.

9.2.1. The Vertical Toolbar

The vertical toolbar shows on the left sidebar when an EER diagram tab is selected. The tools on this toolbar assist in creating EER diagrams.

Figure 9.7. The Vertical Toolbar

Clicking a tool changes the mouse pointer to a pointer that resembles the tool icon, indicating which tool is active. These tools can also be activated from the keyboard by pressing the key associated with the tool. Hover the mouse pointer over a toolbar icon to display a description of the tool and its shortcut key.

A more detailed description of each of these tools follows.

9.2.1.1. The Standard Mouse Pointer

The standard mouse pointer, located at the top of the vertical toolbar, is the default mouse pointer for your operating system. Use this tool to revert to the standard mouse pointer after using other tools.

To revert to the default pointer from the keyboard, use the **Esc** key.

9.2.1.2. The Hand Tool

The hand tool is used to move the entire EER diagram. Left-click on this tool and then left-click anywhere on the EER diagram canvas. Moving the mouse while holding down the mouse button changes the view port of the canvas.

To determine your position on the canvas, look at the [Model Navigator](#) panel on the upper right. If the [Model Navigator](#) panel is not open, use [View](#), Windows, Model Navigator to open it.

To activate the hand tool from the keyboard, use the **H** key.

You can also change the view port of an EER diagram using the [Model Navigator](#) panel. See [Section 9.1.9, “The Model Navigator Panel”](#).

9.2.1.3. The Eraser Tool

Use the eraser tool to delete objects from the EER Diagram canvas. Change the mouse pointer to the eraser tool, then click the object you wish to delete. Depending upon your settings, the delete dialog box should open, asking you to confirm the type of deletion.



Note

The delete action of the [eraser](#) tool is controlled by the general option setting for deletion. Before using the eraser tool, be sure that you understand the available options described in [Section 7.3.5, “The Model Tab”](#).

To activate the eraser tool from the keyboard, use the **D** key.

You can also delete an object by selecting it and pressing **Control+Delete** or by right-clicking it and choosing Delete from the pop up menu.

9.2.1.4. The Layer Tool

The layer tool is the rectangular icon with a capital [L](#) in the lower left corner. Use the layer tool to organize the objects on an EER Diagram canvas. It is useful for grouping similar objects. For example, you may use it to group all your views.

Click the layer tool and use it to draw a rectangle on the canvas. Change to the standard mouse pointer tool and pick up any objects you would like to place on the newly created layer.

To change the size of a layer, first select it by clicking it. When a layer is selected, small rectangles appear at each corner and in the middle of each side. Adjust the size by dragging any of these rectangles.

You can also make changes to a layer by selecting the layer and changing properties in the [Properties](#) panel. Using the [Properties](#) panel is the only way to change the name of a layer.

To activate the layer tool from the keyboard, use the **L** key. For more information about layers, see [Section 9.3.5, “Creating Layers”](#).

9.2.1.5. The Text Tool

The text tool is the square icon with a capital [N](#) in the top left corner. Use this tool to place text objects on the EER diagram canvas. Click the tool, then click the desired location on the canvas. After a text object has been dropped on the canvas, the mouse pointer reverts to its default.

To add text to a text object, right-click the text object and choose [Edit Note...](#) or [Edit in New Window...](#) from the pop-up menu.

You can manipulate the properties of a text object by selecting it and then changing its properties in the [Properties](#) panel.

To activate the text tool from the keyboard, use the **N** key. For more information about text objects, see [Section 9.3.7, “Creating Text Objects”](#).

9.2.1.6. The Image Tool

Use the image tool to place an image on the canvas. When this tool is selected and you click the canvas, a dialog box opens enabling you to select the desired graphic file.

To activate the image tool from the keyboard, use the **I** key. For more information about images, see [Section 9.3.8, “Creating Images”](#).

9.2.1.7. The Table Tool

Use this tool to create a table on the EER Diagram canvas.

Clicking the canvas creates a table. To edit the table with MySQL Table Editor, right-click it and choose Edit Table... or Edit in New Window... from the pop-up menu. You can also double-click the table to load it into the table editor.

To activate the table tool from the keyboard, use the **T** key.

For more information about creating and editing tables, see [Section 9.3.1.3, “The MySQL Table Editor”](#).

9.2.1.8. The View Tool

Use this tool to create a view on an EER Diagram canvas. When the table tool is activated, a schema list appears on the toolbar below the main menu, enabling you to associate the new view with a specific schema. You can also select a color for the object by choosing from the color list to the right of the schema list.

After selecting this tool, clicking the canvas creates a new view. To edit this view, right-click it and choose Edit View... or Edit in New Window... from the pop-up menu.

To activate the view tool from the keyboard, use the **V** key.

For more information about creating and editing views, see [Section 9.3.3, “Creating Views”](#).

9.2.1.9. The Routine Group Tool

Use this tool to create a routine group on the EER Diagram canvas. When this tool is activated, a schema list appears on the toolbar below the main menu, enabling you to associate the routine group with a specific schema. You can also select a color for the routine group by choosing from the color list to the right of the schema list.

After selecting this tool, clicking the canvas creates a new group. To edit this view, right-click it and choose Edit Routine Group... or Edit in New Window... from the pop-up menu.

To activate the routine group tool from the keyboard, use the **G** key.

For more information about creating and editing routine groups, see [Section 9.3.4.2, “Routine Groups”](#).

9.2.1.10. The Relationship Tools

The five relationship tools are used to represent the following relationships:

- One-to-many nonidentifying relationships
- One-to-one nonidentifying relationships
- One-to-many identifying relationships
- One-to-one identifying relationships
- Many-to-many identifying relationships

These tools appear at the bottom of the vertical tool bar. Hover the mouse pointer over each tool to see a text hint that describes its function.

For more information about relationships, see [Section 9.3.2, “Creating Foreign Key Relationships”](#).

9.3. Working with Models

9.3.1. Creating Tables

9.3.1.1. Adding Tables to the Physical Schemata

Double-clicking the `Add table` icon in the `Physical Schemata` section of the `MySQL Model` page adds a table with the default name of `table1`. If a table with this name already exists, the new table is named `table2`.

Adding a new table automatically opens the table editor docked at the bottom of the application. For information about using the table editor, see [Section 9.3.1.3, “The MySQL Table Editor”](#).

Right-clicking a table opens a pop-up menu with the following items:

- Cut '`table_name`'
- Copy '`table_name`'
- Edit Table...
- Edit in New Window...
- Copy SQL to Clipboard
- Copy Insert to Clipboard: Copies `INSERT` statements based on the model's inserts. Nothing is copied to the clipboard if the table has no inserts defined.
- Copy Insert Template to Clipboard: Copies a generic `INSERT` statement that is based on the model.
- Delete '`table_name`'

If the table editor is not open, the Edit Table... item opens it. If it is already open, the selected table replaces the previous one. Edit in New Window... opens a new table editor tab.

The cut and copy items are useful for copying tables between different schemata.



Warning

Use the Delete '`table_name`' item to remove a table from the database. There will be **no** confirmation dialog box.

Any tables added to the `Physical Schemata` section also show up in the `Catalog` palette on the right side of the application. They may be added to an EER Diagram by dragging and dropping them from this palette.

9.3.1.2. Adding Tables to an EER Diagram

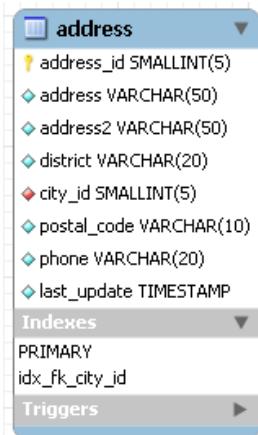
Tables can also be added to an EER Diagram using the `table` tool on the vertical toolbar. Make sure that the `EER Diagram` tab is selected, then right-click the table icon on the vertical toolbar. The table icon is the rectangular tabular icon.

Clicking the mouse on this icon changes the mouse pointer to a table pointer. You can also change the mouse pointer to a table pointer by pressing the **T** key.

Choosing the `table` tool changes the contents of the toolbar that appears immediately below the menu bar. When the `Tables` pointer is active, this toolbar contains a schemata list, an engines list, a collations list, and a color chart list. Use these lists to select the appropriate schema, engine, collation, and color accent for the new table. Make sure that you associate the new table with a database. The engine and collation of a table can be changed using the table editor. The color of your table can be changed using the `Properties` palette. The `Default Engine` and `Default Collation` values refer to the database defaults.

Create a table by clicking anywhere on the EER Diagram canvas. This creates a new table with the default name `table1`. To revert to the default mouse pointer, click the arrow icon at the top of the vertical toolbar.

Figure 9.8. A Table on an EER Diagram



As shown in the preceding diagram, the primary key is indicated by a key icon and indexed fields are indicated by a different colored diamond icon. Click the arrow to the right of the table name to toggle the display of the fields. Toggle the display of indexes and triggers in the same way.

Right-clicking a table opens a pop-up menu with the following items:

- Cut '`table_name`'
- Copy '`table_name`'
- Edit Table...
- Edit in New Window...
- Copy SQL to Clipboard
- Copy Insert to Clipboard
- Delete '`table_name`'

With the exception of the deletion item, these menu items function as described in [Section 9.3.1.1, “Adding Tables to the Physical Schemata”](#). The behavior of the delete option is determined by your MySQL Workbench options settings. For more information, see [Section 7.3.5, “The Model Tab”](#).

9.3.1.3. The MySQL Table Editor

The MySQL Table Editor is a component that enables the creation and modification of tables. You can add or modify a table's columns or indexes, change the engine, add foreign keys, or alter the table's name.

The MySQL Table Editor can be accessed in several ways, and most commonly by right-clicking on a table name within the `Object Viewer` and choosing `ALTER TABLE`. This will open a new tab within the

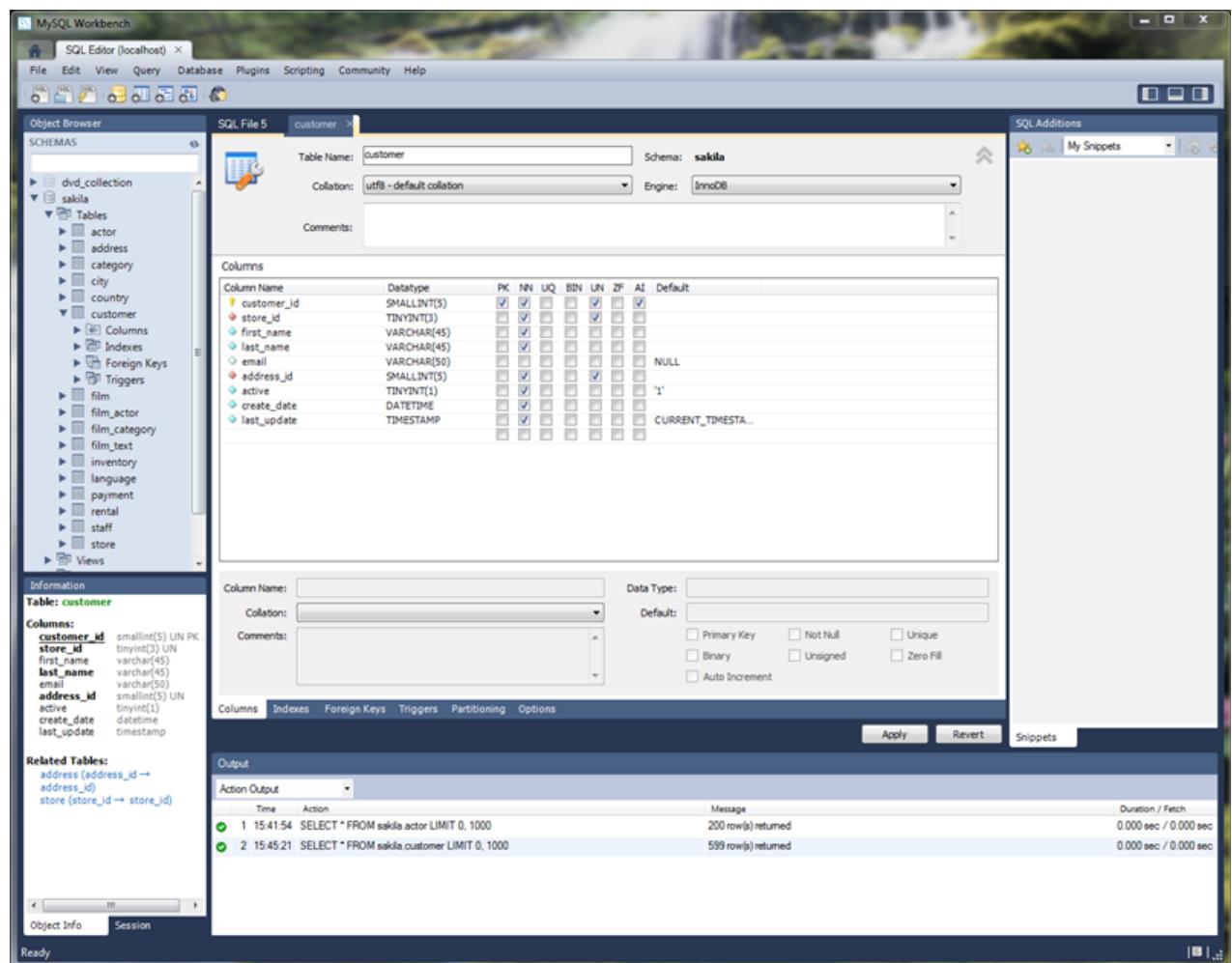
main **SQL Editor** window. You can also access the MySQL Table Editor from an EER Diagram by double-clicking a table object.

9.3.1.3.1. The Main Editor Window

Any number of tables may be edited in the MySQL Table Editor at any one time. Adding another table creates a new tab at the top of the editor. By default, the MySQL Table Editor appears docked at the top of the table editor tab, within the SQL editor..

The MySQL Table Editor is shown on top of the following figure.

Figure 9.9. The Table Editor



The MySQL Table Editor provides a work space that has tabs used to perform these actions:

- **Columns:** Add or modify columns
- **Indexes:** Add or modify indexes
- **Foreign Keys:** Add or modify foreign keys
- **Triggers:** Add or modify triggers
- **Partitioning:** Manage partitioning
- **Options:** Add or modify various general, table, and row options

The following sections discuss these tabs in further detail.

9.3.1.3.2. The Columns Tab

Use the **Columns** tab to display and edit all the column information for a table. With this tab, you can add, drop, and alter columns.

You can also use the **Columns** tab to change column properties such as name, data type, and default value.

Figure 9.10. The Columns Tab

Column Name	Datatype	PK	NN	UQ	BIN	UN	ZF	AI	Default
customer_id	SMALLINT(5)	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	
store_id	TINYINT(3)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
first_name	VARCHAR(45)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
last_name	VARCHAR(45)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
email	VARCHAR(50)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
address_id	SMALLINT(5)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
active	TINYINT(1)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
create_date	DATETIME	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
last_update	TIMESTAMP	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	CURRENT_TIMESTAMP...

Right-click a row under the **Column Name** column to open a pop-up menu with the following items:

- Move Up: Move the selected column up.
- Move Down: Move the selected column down.
- Copy: Copies the column for a model. Added in MySQL Workbench 5.2.45.
- Cut: Copies and then deletes the column for a model. Added in MySQL Workbench 5.2.45.
- Paste: Pastes the column. If a column with the same name already exists, then `_copy1` is appended to the column name. Added in MySQL Workbench 5.2.45.
- Delete Selected Columns: Select multiple contiguous columns by right-clicking and pressing the **Shift** key. Use the **Control** key to select noncontiguous columns.
- Refresh: Update all information in the **Columns** tab.
- Clear Default: Clear the assigned default value.
- Default NULL: Set the column default value to `NULL`.

- Default 0: Set the column default value to `0`.
- Default CURRENT_TIMESTAMP: Available for `TIMESTAMP` data types.
- Default CURRENT_TIMESTAMP ON UPDATE CURRENT_TIMESTAMP: Available for `TIMESTAMP` data types.

To add a column, click the `Column Name` field in an empty row and enter an appropriate value. Select a data type from the **Datatype** list. Select the column property check boxes as required according to the list of column properties below, and also read the [CREATE TABLE](#) documentation for information about what these options mean.

- **PK:** PRIMARY KEY
- **NN:** NOT NULL
- **UQ:** UNIQUE INDEX
- **BIN:** BINARY
- **UN:** UNSIGNED
- **ZF:** ZEROFILL
- **AI:** AUTO_INCREMENT

To change the name, data type, default value, or comment of a column, double-click the value you wish to change. The content then becomes editable.

You can also add column comments to the `Column Comment` field. It is also possible to set the column collation, using the list in the **Column Details** panel.

To the left of the column name is an icon that indicates whether the column is a member of the primary key. If the icon is a small key, that column belongs to the primary key, otherwise the icon is a blue diamond or a white diamond. A blue diamond indicates the column has **NN** set. To add or remove a column from the primary key, double-click the icon. You can also add a primary key by checking the **PRIMARY KEY** check box in the `Column Details` section of the table editor.

If you wish to create a composite primary key you can select multiple columns and check the **PK** check box. However, there is an additional step that is required, you must click the Indexes tab, then in the Index Columns panel you must set the desired order of the primary keys.



Note

When entering default values, in the case of `CHAR` and `VARCHAR` data types MySQL Workbench will attempt to automatically add quotation marks, if the user does not start their entry with one. For other data types the user must manage quoting if required, as it will not be handled automatically by MySQL Workbench.



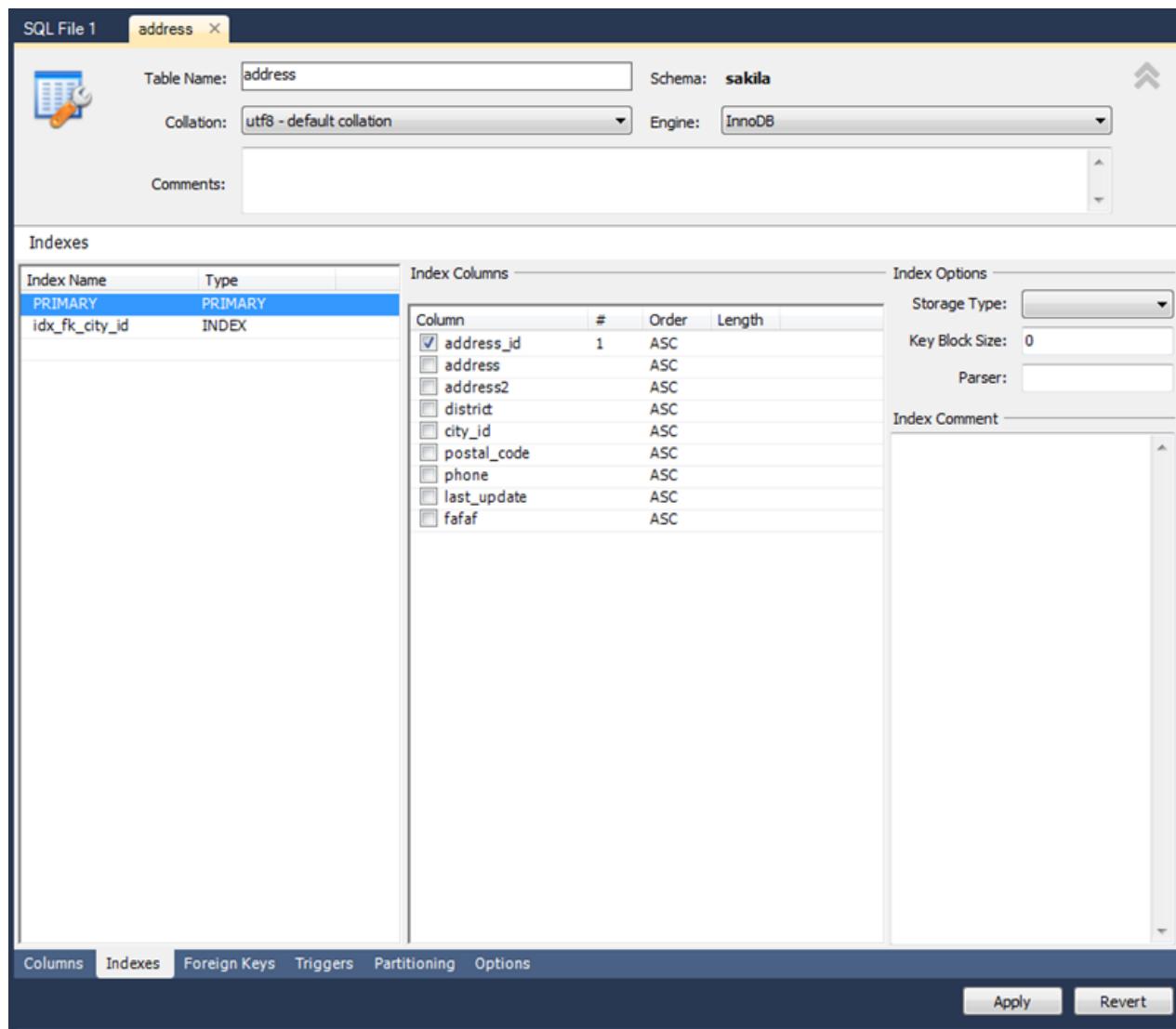
Caution

Care must be taken when entering a default value for `ENUM` columns because a nonnumeric default will not be automatically quoted. You must manually add single quote characters for the default value. Note that MySQL Workbench will **not** prevent you from entering the default value without the single quotation marks. If a nonnumeric default value is entered without quotation marks, this will lead to errors. For example, if the model is reverse engineered, the script will contain unquoted default values for `ENUM` columns and will fail if an attempt is made to run the script on MySQL Server.

9.3.1.3.3. The Indexes Tab

The **Indexes** tab holds all index information for your table. Use this tab to add, drop, and modify indexes.

Figure 9.11. The Indexes Tab



Select an index by right-clicking it. The **Index Columns** section displays information about the selected index.

To add an index, click the last row in the index list. Enter a name for the index and select the index type from the list. Select the column or columns that you wish to index by checking the column name in the **Index Columns** list. You can remove a column from the index by removing the check mark from the appropriate column.

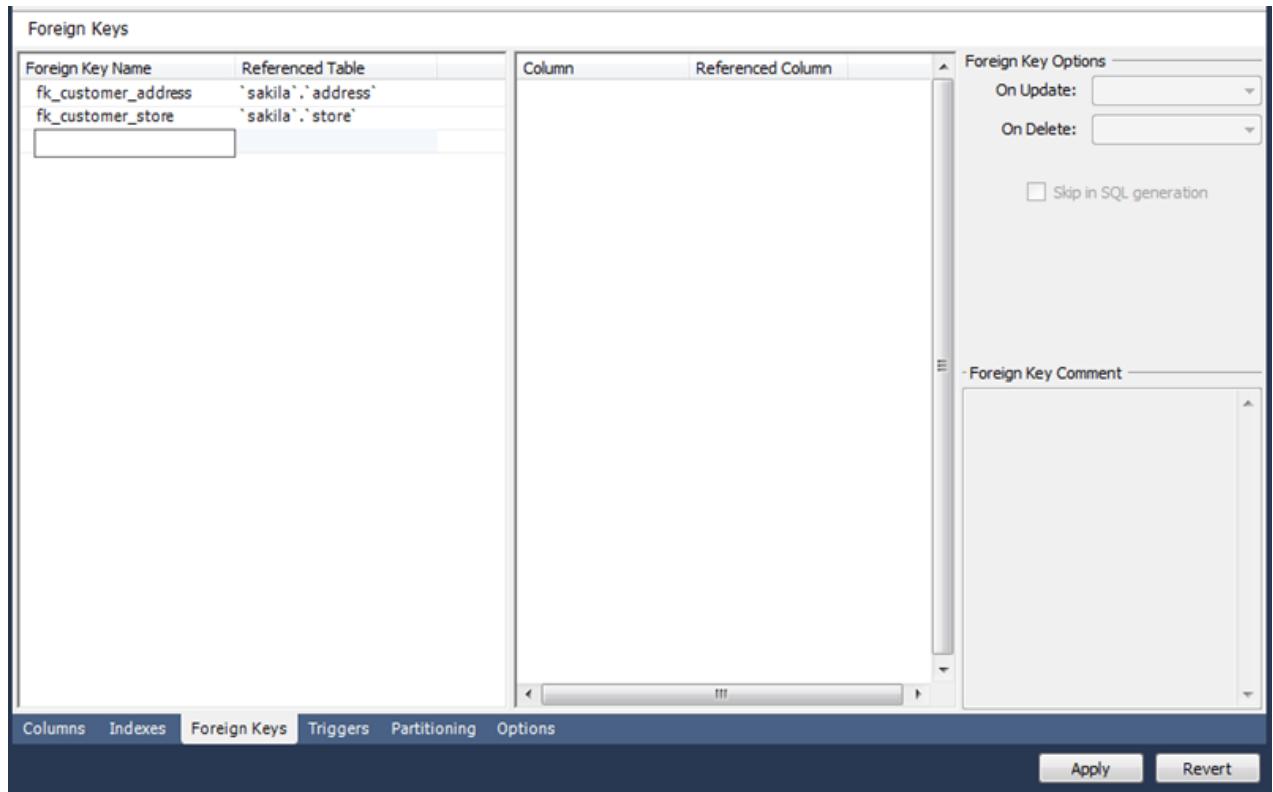
You can also specify the order of an index by choosing **ASC** or **DESC** under the **Order** column. Create an index prefix by specifying a numeric value under the **Length** column. You cannot enter a prefix value for fields that have a data type that does not support prefixing.

To drop an index, right-click the row of the index you wish to delete, then select the **Delete Selected Indexes** menu item.

9.3.1.3.4. The Foreign Keys Tab

The **Foreign Keys** tab is organized in much the same fashion as the **Indexes** tab and adding or editing a foreign key is similar to adding or editing an index.

Figure 9.12. The Foreign Keys Tab



To add a foreign key, click the last row in the **Foreign Key Name** list. Enter a name for the foreign key and select the column or columns that you wish to index by checking the column name in the **Column** list. You can remove a column from the index by removing the check mark from the appropriate column.

Under **Foreign Key Options**, choose an action for the update and delete events. The options are:

- RESTRICT
- CASCADE
- SET NULL
- NO ACTION

To drop a foreign key, right-click the row you wish to delete, then select the Delete Selected FKs menu item.

To modify properties of a foreign key, select it and make the desired changes.

9.3.1.3.5. The Triggers Tab

The **Triggers** tab opens a field for editing an existing trigger or creating a new trigger. Create a trigger as you would from the command line.

Figure 9.13. The Triggers Tab

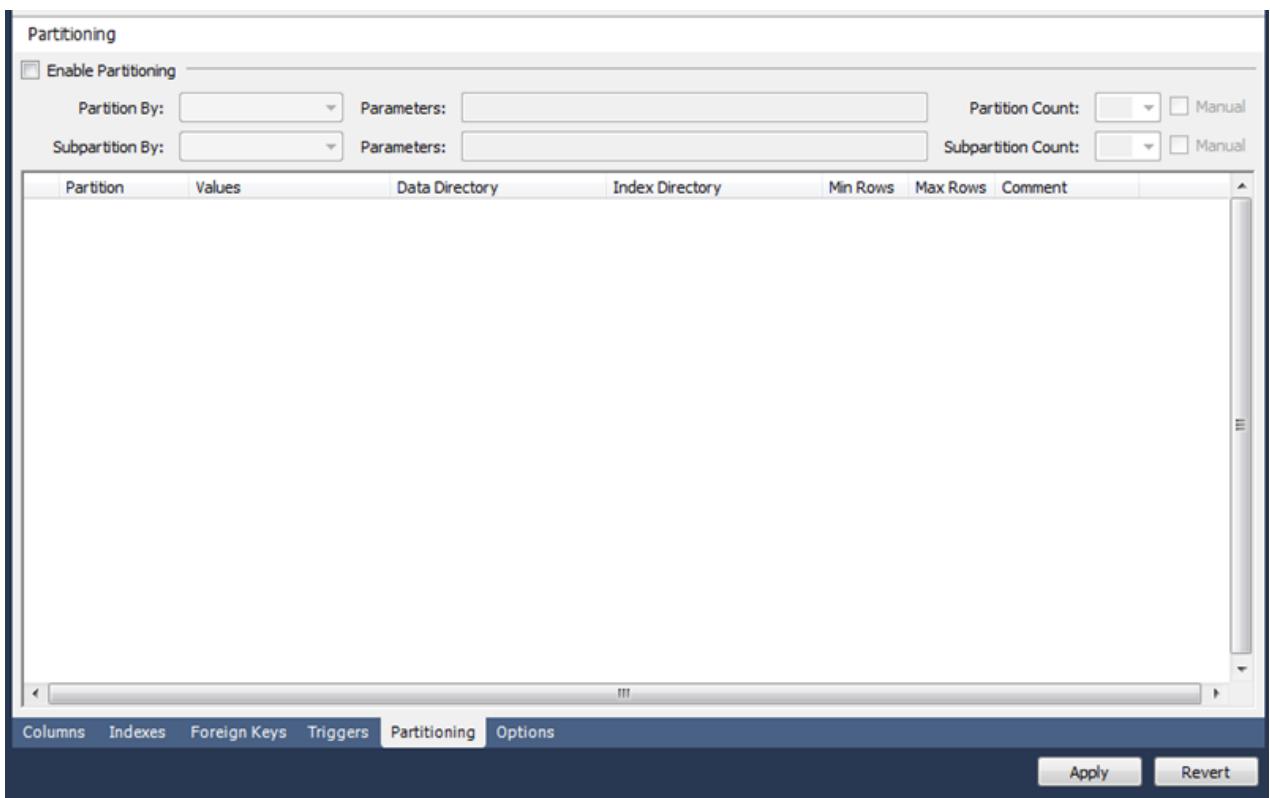
The screenshot shows the MySQL Workbench interface with the 'Triggers' tab selected. The main pane displays the following MySQL trigger code:

```
1 -- Trigger DDL Statements
2 DELIMITER $$ 
3 
4 • USE `sakila`$$
5 
6 • CREATE
7 DEFINER=`root`@`localhost`
8 TRIGGER `sakila`.`customer_create_date`
9 BEFORE INSERT ON `sakila`.`customer`
10 FOR EACH ROW
11 SET NEW.create_date = NOW()$$
12 
13
```

The tabs at the bottom are: Columns, Indexes, Foreign Keys, Triggers (selected), Partitioning, Options. There are 'Apply' and 'Revert' buttons at the bottom right.

9.3.1.3.6. The Partitioning Tab

To enable partitioning for your table, check the **Enable Partitioning** check box. This enables the partitioning options.

Figure 9.14. The Partitioning Tab

The **Partition By** pop-up menu displays the types of partitions you can create:

- HASH
- LINEAR HASH
- KEY
- LINEAR KEY
- RANGE
- LIST

Use the **Parameters** field to define any parameters to be supplied to the partitioning function, such as an integer column value.

Choose the number of partitions from the **Partition Count** list. To manually configure your partitions, check the **Manual** check box. This enables entry of values into the partition configuration table. The entries in this table are:

- **Partition**
- **Values**
- **Data Directory**
- **Index Directory**
- **Min Rows**

- [Max Rows](#)
- [Comment](#)

Subpartitioning is also available. For more information about partitioning, see [Partitioning](#).

9.3.1.3.7. The Options Tab

The **Options** tab enables you to set several types of options.

Figure 9.15. The Options Tab

The screenshot shows the 'Options' tab in MySQL Workbench. The interface is divided into several sections:

- General Options:** Contains fields for 'Pack Keys' (set to 'Don't use'), 'Table Password' (empty), 'Auto Increment' (set to 600), and a checked checkbox for 'Delay Key Updates'. A note explains that this option delays key updates until the table is closed, working only for MyISAM.
- Row Options:** Contains fields for 'Row Format' (set to 'Don't Use'), 'Avg. Row Length' (empty), 'Min. Rows' (empty), 'Max. Rows' (empty), and an unchecked checkbox for 'Use Checksum'. A note explains that this option activates a live checksum for all rows, making the table slower to update but easier to find corrupted tables.
- Storage Options:** Contains fields for 'Data Directory' (empty) and 'Index Directory' (empty). Notes explain that these work only for MyISAM tables on Windows.
- Merge Table Options:** Contains fields for 'Union Tables' (empty) and 'Merge Method' (set to 'Don't Use'). A note explains that the union table is used for inserts.

At the bottom, there are tabs for 'Columns', 'Indexes', 'Foreign Keys', 'Triggers', 'Partitioning', and 'Options' (selected). To the right of the tabs are 'Apply' and 'Revert' buttons.

which are grouped into the following sections:

- General Options
- Row Options
- Storage Options
- Merge Table options

The following discussion describes these options in more detail.

General Options Section

In the **General Options** section, choose a pack keys option. The options are [Default](#), [Pack None](#), and [Pack All](#). You may also encrypt the definition of a table. The [AUTO_INCREMENT](#) and delayed key update behaviors apply only to [MyISAM](#) tables.

Row Options Section

To set the row format, choose the desired row format from the list. For more information about the different row formats that are available, see [MyISAM Table Storage Formats](#).

These options are:

- Default
- Dynamic
- Fixed
- Compressed
- Redundant
- Compact

When you expect a table to be particularly large, use the **Avg. Row, Min. Rows**, and **Max. Rows** options to enable the MySQL server to better accommodate your data. See [CREATE TABLE Syntax](#) for more information on how to use these options.

Storage Options Section

The **Storage Options** section is available only for MyISAM tables. Use it to configure a custom path to the table storage and data files. This can help improve server performance by locating different tables on different hard drives.

Merge Table Options Section

Use the **Merge Table** Options section to configure MERGE tables. To create a MERGE table, select **MERGE** as your storage engine and then specify the MyISAM tables you wish to merge in the **Union Tables** dialog.

You may specify the action the server should take when users attempt to perform **INSERT** statements on the merge table. You may also select the **Merge Method** by selecting from the list. For more information about MERGE tables, see [The MERGE Storage Engine](#).

9.3.1.3.8. The Inserts Tab

Use the **Inserts** tab to insert rows into the table.

To edit a row, click the field you wish to change and enter the new data. Right-clicking a row displays a menu with the following items:

- Set Field(s) to NULL: Set the column value to **NULL**.
- Delete Row(s): Delete the selected row or rows.
- Copy Row Content: Copies the row to the clipboard. Strings are copied quoted, and **NULL** values are preserved.
- Copy Row Content (unquoted): Copies the row to the clipboard. Strings are not quoted and **NULL** are copied as a space.
- Copy Field Content: Copies the value of the selected field to the clipboard. Strings are quoted.
- Copy Field Content (unquoted): Copies the value of the selected field to the clipboard. Strings are not quoted.

Note that the insert editor features a toolbar. This has the same functionality as explained in [Section 8.2.3, “SQL Query Panel”](#). You can also hover the cursor over the toolbar to display tooltips.

Any rows you add will be inserted when you forward engineer the database (if you choose the [Generate INSERT statements for tables](#) option).



Note

When entering string values that there is slightly different behavior between the 5.0, 5.1, and 5.2 versions of MySQL Workbench.

For 5.0 and 5.1, if a string is entered without leading and trailing quotation marks, the Inserts Editor adds quoting and escapes characters that require it. However, if quoted text is entered, the Inserts Editor performs no further checks and assumes that a correctly escaped and quoted sequence has been entered.

5.2 features a new Inserts Editor. In this case, the user enters the string without quoting or escaping and the Inserts Editor takes care of all quoting and escaping as required.



Note

It is possible to enter a function, or other expression, into a field. Use the prefix `\func` to prevent MySQL Workbench from escaping quotation marks. For example, for the expression `md5('fred')`, MySQL Workbench normally would generate the code `md5('fred\')`. To prevent this, enter the expression as `\func md5('fred')` to ensure that the quoting is not escaped.

9.3.1.3.9. The Privileges Tab

Use the **Privileges** tab to assign specific roles and privileges to a table. You may also assign privileges to a role using the role editor. For a discussion of this topic, see [Section 9.1.5.1, “Adding Roles”](#).

When this tab is first opened, all roles that have been created are displayed in the list on the right. Move the roles you wish to associate with this table to the **Roles** list on the left. Do this by selecting a role and then clicking the `<` button. Use the **Shift** key to select multiple contiguous roles and the **Control** key to select noncontiguous roles.

To assign privileges to a role, click the role in the **Roles** list. This displays all available privileges in the **Assigned Privileges** list. The privileges that display are:

- [ALL](#)
- [CREATE](#)
- [DROP](#)
- [GRANT OPTION](#)
- [REFERENCES](#)
- [ALTER](#)
- [DELETE](#)
- [INDEX](#)
- [INSERT](#)

- [SELECT](#)
- [UPDATE](#)
- [TRIGGER](#)

You can choose to assign all privileges to a specific user or any other privilege as listed previously. Privileges irrelevant to a specific table, such as the [FILE](#) privilege, are not shown.

If a role has already been granted privileges on a specific table, those privileges show as already checked in the **Assigned Privileges** list.

9.3.2. Creating Foreign Key Relationships

Foreign key constraints are supported for the [InnoDB](#) storage engine only. For other storage engines, the foreign key syntax is correctly parsed but not implemented. For more information, see [Foreign Key Differences](#).

Using MySQL Workbench you may add a foreign key from within the table editor or by using the relationship tools on the vertical toolbar of an EER Diagram. This section deals with adding a foreign key using the foreign key tools. To add a foreign key using the table editor, see [Section 9.3.1.3.4, “The Foreign Keys Tab”](#).

The graphical tools for adding foreign keys are most effective when you are building tables from the ground up. If you have imported a database using an SQL script and need not add columns to your tables, you may find it more effective to define foreign keys using the table editor.

9.3.2.1. Adding Foreign Key Relationships Using an EER Diagram

The vertical toolbar on the left side of an EER Diagram has six foreign key tools:

- [one-to-one non-identifying relationship](#)
- [one-to-many non-identifying relationship](#)
- [one-to-one identifying relationship](#)
- [one-to-many identifying relationship](#)
- [many-to-many identifying relationship](#)
- [Place a Relationship Using Existing Columns](#)

An identifying relationship is one where the child table cannot be uniquely identified without its parent. Typically this occurs where an intermediary table is created to resolve a many-to-many relationship. In such cases, the primary key is usually a composite key made up of the primary keys from the two original tables. An identifying relationship is indicated by a solid line between the tables and a nonidentifying relationship is indicated by a broken line.

Create or drag and drop the tables that you wish to connect. Ensure that there is a primary key in the table that will be on the “one” side of the relationship. Click on the appropriate tool for the type of relationship you wish to create. If you are creating a one-to-many relationship, first click the table that is on the “many” side of the relationship, then on the table containing the referenced key. This creates a column in the table on the many side of the relationship. The default name of this column is [table_name_key_name](#) where the table name and the key name both refer to the table containing the referenced key.

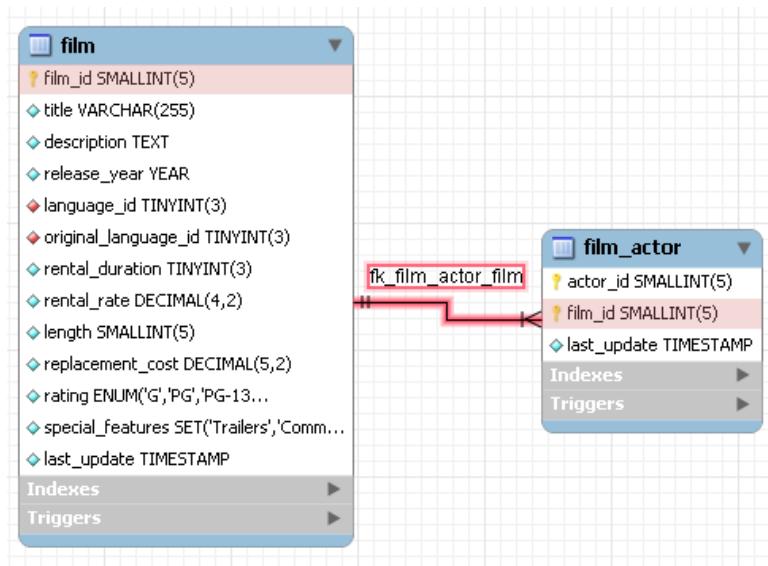
When the many-to-many tool is active, double-clicking a table creates an associative table with a many-to-many relationship. For this tool to function there must be a primary key defined in the initial table.

Use the [Model](#) menu, Menu Options menu item to set a project-specific default name for the foreign key column (see [Section 9.1.1.5.4, "The Relationship Notation Submenu"](#)). To change the global default, see [Section 7.3.5, "The Model Tab"](#).

To edit the properties of a foreign key, double-click anywhere on the connection line that joins the two tables. This opens the relationship editor.

Mousing over a relationship connector highlights the connector and the related keys as shown in the following figure. The `film` and the `film_actor` tables are related on the `film_id` field and these fields are highlighted in both tables. Since the `film_id` field is part of the primary key in the `film_actor` table, a solid line is used for the connector between the two tables.

Figure 9.16. The Relationship Connector



If the placement of a connection's caption is not suitable, you can change its position by dragging it to a different location. If you have set a secondary caption, its position can also be changed. For more information about secondary captions, see [Section 9.3.2.3, "Connection Properties"](#). Where the notation style permits, [Classic](#) for example, the cardinality indicators can also be repositioned.

The relationship notation style in [Figure 9.16, "The Relationship Connector"](#) is the default, crow's foot. You can change this if you are using a commercial version of MySQL Workbench. For more information, see [Section 9.1.1.5.4, "The Relationship Notation Submenu"](#).

You can select multiple connections by holding down the **Control** key as you click a connection. This can be useful for highlighting specific relationships on an EER diagram.

9.3.2.2. The Relationship Editor

Double-clicking a relationship on the EER diagram canvas opens the relationship editor. This has two tabs: **Relationship**, and **Foreign Key**.

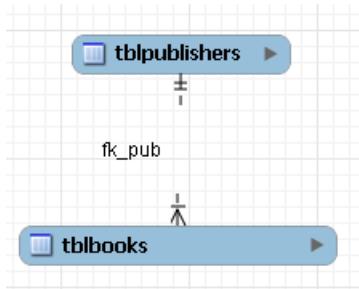
The Relationship Tab

In the **Relationship** tab, you can set the caption of a relationship using the **Caption** field. This name displays on the canvas and is also the name used for the constraint itself. The default value for this name is `fk_source_table_destination_table`. Use the [Model](#) menu, Menu Options menu item to set a project-specific default name for foreign keys. To change the global default, see [Section 7.3.5, "The Model Tab"](#).

You can also add a secondary caption and a caption to a relationship.

The **Visibility Settings** section is used to determine how the relationship is displayed on the EER Diagram canvas. **Fully Visible** is the default but you can also choose to hide relationship lines or to use split lines. The split line style is pictured in the following figure.

Figure 9.17. The Split Connector



Note

A broken line connector indicates a nonidentifying relationship. The split line style can be used with either an identifying relationship or a nonidentifying relationship. It is used for display purposes only and does not indicate anything about the nature of a relationship.

To set the notation of a relationship use the **Model** menu, **Relationship Notation** menu item. For more information, see [Section 9.1.1.5.4, "The Relationship Notation Submenu"](#).

The Foreign Key Tab

The **Foreign Key** tab contains several sections: **Referencing Table**, **Cardinality** and **Referenced Table**.

The **Mandatory** check boxes are used to select whether the referencing table and the referenced table are mandatory. By default, both of these constraints are **true** (indicated by the check boxes being checked).

The **Cardinality** section has a set of radio buttons that enable you to choose whether the relationship is one-to-one or one-to-many. There is also a check box that enables you to specify whether the relationship is an identifying relationship.

9.3.2.3. Connection Properties

Right-click a connection to select it. When a connection is selected, it is highlighted and its properties are displayed in the properties palette. Connection properties are different from the properties of other objects. The following list describes them:

- **caption**: The name of the connection. By default, the name is the name of the foreign key and the property is centered above the connection line.
- **captionXOffs**: The X offset of the caption.
- **captionYOffs**: The Y offset of the caption.
- **comment**: The comment associated with the relationship.
- **drawSplit**: Whether to show the relationship as a continuous line.
- **endCaptionXOffs**: The X termination point of the caption offset.
- **endCaptionYOffs**: The Y termination point of the caption offset.

- `extraCaption`: A secondary caption. By default, this extra caption is centered beneath the connection line.
- `extraCaptionXOffs`: The X offset of the secondary caption.
- `extraCaptionYOffs`: The Y offset of the secondary caption.
- `mandatory`: Whether the entities are mandatory. For more information, see [Section 9.3.2.2, “The Relationship Editor”](#).
- `many`: False if the relationship is a one-to-one relationship.
- `middleSegmentOffset`: The offset of the middle section of the connector.
- `modelOnly`: Set when the connection will not be propagated to the DDL. It is just a logical connection drawn on a diagram. This is used, for example, when drawing `MyISAM` tables with a visual relationship, but with no foreign keys.
- `name`: The name used to identify the connection on the EER Diagram canvas. Note that this is **not** the name of the foreign key.
- `referredMandatory`: Whether the referred entity is mandatory.
- `startCaptionXOffs`: The start of the X offset of the caption.
- `startCaptionYOffs`: The start of the Y offset of the caption.

In most cases, you can change the properties of a relationship using the relationship editor rather than the [Properties](#) palette.

If you make a relationship invisible by hiding it using the relationship editor's **Visibility Settings**, and then close the relationship editor, you will no longer be able to select the relationship to bring up its relationship editor. To make the relationship visible again, you must expand the table object relating to the relationship in the [Layers](#) palette and select the relationship object. To edit the selected object, right-click it, then select Edit Object. You can then set the **Visibility Settings** to **Fully Visible**. The relationship will then be visible in the [EER Diagram](#) window.

9.3.3. Creating Views

You can add views to a database either from the [Physical Schemata](#) section of the [MySQL Model](#) page or from the EER Diagram.

9.3.3.1. Adding Views to the Physical Schemata

To add a view, double-clicking the [Add View](#) icon in the [Physical Schemata](#) section of the [MySQL Model](#) page. The default name of the view is `view1`. If a view with this name already exists, the new view is named `view2`.

Adding a new view automatically opens the view editor docked at the bottom of the application. For information about using the view editor, see [Section 9.3.3.3, “The View Editor”](#).

Right-clicking a view opens a pop-up menu with the following items:

- Cut '`view_name`'

As of MySQL Workbench 5.2.45, the '`view_name`' is only cut from the EER canvas. Before, it was also removed from the schema.

- Copy '`view_name`'

- Paste
- Edit View...
- Edit in New Window...
- Copy SQL to Clipboard
- Delete '*view_name*': deletes from both the EER canvas and schema.
- Remove '*view_name*': deletes from the EER canvas, but not the schema.

This option exists as of MySQL Workbench 5.2.45.

If the table editor is not open, the Edit View... item opens it. If it is already open, the selected table replaces the previous one. Edit in New Window... opens a new view editor tab.

The cut and copy items are useful for copying views between different schemata. Copy SQL to Clipboard copies the `CREATE VIEW` statement to the clipboard.



Warning

Use the Delete '*view_name*' item to remove a view from the database. There will be **no** confirmation dialog box.

Any views added to the `Physical Schemata` section also show up in the `Catalog` palette on the left side of the application. They may be added to an EER Diagram, when in the EER Diagram tab, by dragging and dropping them from this palette.

9.3.3.2. Adding Views to an EER Diagram

Views can also be added to an EER Diagram using the `View` tool on the vertical toolbar. Make sure that the **EER Diagram** tab is selected, then left-click the view icon on the vertical toolbar. The view icon is the two overlapping rectangles found below the table icon.

Clicking this icon changes the mouse pointer to a view pointer. To change the mouse pointer to a view pointer from the keyboard, use the **V** key.

Choosing the `View` tool changes the contents of the toolbar that appears immediately below the main menu bar. When the `Views` pointer is active, this toolbar contains a schemata list and a color chart list. Use these lists to select the appropriate schema and color accent for the new view. Make sure that you associate the new view with a database. The color of your view can be changed using the `Properties` palette.

Create a view by clicking anywhere on the EER Diagram canvas. This creates a new view with the default name `view1`. To revert to the default mouse pointer, click the arrow icon at the top of the vertical toolbar.

Right-clicking a view opens a pop-up menu. With the exception of the delete item, these menu items function as described in [Section 9.3.3.1, “Adding Views to the Physical Schemata”](#). The behavior of the delete option is determined by your MySQL Workbench options settings. For more information, see [Section 7.3.5, “The Model Tab”](#).

9.3.3.3. The View Editor

To invoke the view editor, double-click a view object on the EER Diagram canvas or double-click a view in the `Physical Schemata` section on the `MySQL Model` page. This opens the view editor docked at the bottom of the application. Double-clicking the title bar undocks the editor. Do the same to redock it. Any number of views may be open at the same time. Each additional view appears as a tab at the top of the view editor.

There are three tabs at the bottom of the view editor: **View**, **Comments**, and **Privileges**. Navigate between different tabs using the mouse or from the keyboard by pressing **Control+Alt+Tab**.

The View Tab

Use the **View** tab to perform the following tasks:

- Rename the view using the **Name** text box.
- Enter the SQL to create a view using the **SQL** field.
- Comment a view using the **Comments** text area.

The Comments Tab

This tab enables you to enter comments for a particular view.

The Privileges Tab

The **Privileges** tab of the view editor functions in exactly the same way as the **Privileges** tab of the table editor. For more information, see [Section 9.3.1.3.9, “The Privileges Tab”](#).

9.3.3.4. Modifying a View Using the Properties Palette

When you select a view on the EER Diagram canvas, its properties are displayed in the **Properties** palette. Most of the properties accessible from the **Properties** palette apply to the appearance of a view on the EER Diagram canvas.

For a list of properties accessible through the **Properties** palette, see [Section 9.1.12, “The Properties Palette”](#).

9.3.4. Creating Routines and Routine Groups

You can add Routine Groups to a database either from the **Physical Schemata** section of the **MySQL Model** page or from an EER Diagram. Routines may be added only from the **Physical Schemata** section of the **MySQL Model** page.

To view an existing schema, along with its Routines and Routine Groups, choose Database, Reverse Engineer... from the main menu. After the schema has been added to the current model, you can see the schema objects on the **Physical Schemata** panel on the **MySQL Model** page. The Routines and Routine Groups are listed there.

MySQL Workbench unifies both stored procedures and stored functions into one logical object called a Routine. Routine Groups are used to group routines that are related. You can decide how many Routine Groups you want to create and you can use the **Routine Group Editor** to assign specific routines to a group, using a drag and drop interface.

When designing an EER Diagram, you can place the Routine Groups on the canvas by dragging them from the **Catalog Palette**. Placing individual routines on the diagram is not permitted, as it would clutter the canvas.

9.3.4.1. Routines

9.3.4.1.1. Adding Routines to the Physical Schemata

To add a routine, double-click the **Add Routine** icon in the **Physical Schemata** section of the **MySQL Model** page. The default name of the routine is `routine1`. If a routine with this name already exists, the new routine is named `routine2`.

Adding a new routine automatically opens the routine editor docked at the bottom of the application. For information about using the routine editor, see [Section 9.3.4.1.2, “The Routine Editor”](#).

Right-clicking a routine opens a pop-up menu with the following items:

- Rename
- Cut '*routine_name*'
- Copy '*routine_name*'
- Paste
- Edit Routine...
- Edit in New Window...
- Copy SQL to Clipboard
- Delete '*routine_name*'

The Edit Routine... item opens the routine editor.

The cut and paste items are useful for copying routines between different schemata.



Note

Deleting the code for a routine from the **Routines** tab of the Routine Group Editor results in removal of the routine object from the model.



Note

To remove a routine from a routine group, use the controls on the **Routine Group** tab of the Routine Group Editor.

The action of the delete option varies depending upon how you have configured MySQL Workbench. For more information, see [Section 7.3.5, “The Model Tab”](#).

9.3.4.1.2. The Routine Editor

To invoke the routine editor, double-click a routine in the **Physical Schemata** section on the [MySQL Model](#) page. This opens the routine editor docked at the bottom of the application. Any number of routines may be open at the same time. Each additional routine appears as a tab at the top of the routine editor.

Routine and **Privileges** tabs appear at the bottom of the routine editor. Navigate between different tabs using the mouse or from the keyboard by pressing **Control+Alt+Tab**.

9.3.4.1.2.1. The Routine Tab

Use the **Routine** tab of the routine editor to perform the following tasks:

- Rename the routine using the **Name** field.
- Enter the SQL to create a routine using the **SQL** field.

9.3.4.1.2.2. The Privileges Tab

The **Privileges** tab of the routine editor functions in exactly the same way as the **Privileges** tab of the table editor. For more information, see [Section 9.3.1.3.9, “The Privileges Tab”](#).

**Note**

Privileges are available only in the Standard Edition of MySQL Workbench.

9.3.4.2. Routine Groups

9.3.4.2.1. Adding Routine Groups to the Physical Schemata

Double-clicking the [Add Routine Group](#) icon in the [Physical Schemata](#) section of the [MySQL Model](#) page adds a routine group with the default name of `routines1`. If a routine group with this name already exists, the new routine group is named `routines2`.

Adding a new routine group automatically opens the routine groups editor docked at the bottom of the application. For information about using the routine groups editor, see [Section 9.3.4.2.3, “The Routine Group Editor”](#).

Right-clicking a routine group opens a pop-up menu with the following items:

- Rename
- Cut '`routine_group_name`'
- Copy '`routine_group_name`'
- Edit Routine...
- Edit in New Window...
- Copy SQL to Clipboard
- Delete '`routine_group_name`'

The [Edit Routine Group...](#) item opens the routine group editor, which is described in [Section 9.3.4.2.3, “The Routine Group Editor”](#).

The cut and paste items are useful for copying routine groups between different schemata.

Deleting a routine group from the [MySQL Model](#) page removes the group but does not remove any routines contained in that group.

Any routine groups added to the [Physical Schemata](#) also show up in the [Catalog](#) palette on the right side of the application. They may be added to an EER Diagram by dragging and dropping them from this palette.

9.3.4.2.2. Adding Routine Groups to an EER Diagram

To add routine groups to an EER Diagram, use the [Routine Groups](#) tool on the vertical toolbar. Make sure that the [EER Diagram](#) tab is selected, then right-click the routine groups icon on the vertical toolbar. The routine groups icon is immediately above the lowest toolbar separator.

Clicking the mouse on this icon changes the mouse pointer to a routine group pointer. You can also change the mouse pointer to a routine pointer by pressing the **G** key.

Choosing the [Routine Group](#) tool changes the contents of the toolbar that appears immediately below the menu bar. When the [Routine Groups](#) pointer is active, this toolbar contains a schemata list and a color chart list. Use these lists to select the appropriate schema and color accent for the new routine group. Make sure that you associate the new routine group with a database. The color of your routine group can be changed later using the [Properties](#) palette.

Create a routine group by clicking anywhere on the EER Diagram canvas. This creates a new routine group with the default name `routines1`. To revert to the default mouse pointer, click the arrow icon at the top of the vertical toolbar.

Right-clicking a routine group opens a pop-up menu. With the exception of the delete option and rename options, these menu options function as described in [Section 9.3.4.2.1, “Adding Routine Groups to the Physical Schemata”](#). There is no rename option, and the behavior of the delete option is determined by your MySQL Workbench options settings. For more information, see [Section 7.3.5, “The Model Tab”](#).

9.3.4.2.3. The Routine Group Editor

To invoke the routine group editor, double-click a routine group object on the EER Diagram canvas or double-click a routine group in the [Physical Schemata](#) section on the [MySQL Model](#) page. This opens the routine group editor docked at the bottom of the application. Double-clicking the title bar undocks the editor. Do the same to redock it. Any number of routine groups may be open at the same time. Each additional routine group appears as a tab at the top of the routine editor,

Routine group and **Privileges** tabs appear at the bottom of the routine editor. Navigate between different tabs using the mouse or from the keyboard by pressing **Control+Alt+Tab**.

9.3.4.2.3.1. The Routine Groups Tab

Use the **Routine Groups** tab of the routine groups editor to perform the following tasks:

- Rename the routine group using the **Name** field.
- Add routines to the group by dragging and dropping them.
- Add comments to the routine group.

9.3.4.2.3.2. The Privileges Tab

The **Privileges** tab of the routine group editor functions in exactly the same way as the **Privileges** tab of the table editor. For more information, see [Section 9.3.1.3.9, “The Privileges Tab”](#).



Note

Privileges are available only in the Standard Edition of MySQL Workbench.

9.3.4.2.3.3. Modifying a Routine Group Using the Properties Palette

When you select a routine group on the EER Diagram canvas, its properties are displayed in the [Properties](#) palette. All of the properties accessible from the [Properties](#) palette apply to the appearance of a routine group on the EER Diagram canvas.

For a list of properties accessible through the [Properties](#) palette, see [Section 9.1.12, “The Properties Palette”](#).

9.3.5. Creating Layers

You can add layers to a database only from an EER Diagram. Layers are used to help organize objects on the canvas. Typically, related objects are added to the same layer; for example, you may choose to add all your views to one layer.

9.3.5.1. Adding Layers to an EER Diagram

To add layers to an EER Diagram, use the [Layer](#) tool on the vertical toolbar. Select an **EER Diagram** tab and right-click the layer icon on the vertical toolbar. The layer icon is the rectangle with an ‘L’ in the lower left corner and it is found below the eraser icon.

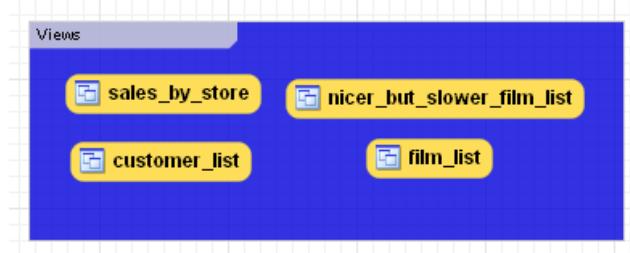
Clicking the mouse on this icon changes the mouse pointer to a layer pointer. You can also change the mouse pointer to a layer pointer by pressing the **L** key.

Choosing the [Layer](#) tool changes the contents of the toolbar that appears immediately below the menu bar. When the [Layers](#) pointer is active, this toolbar contains a color chart list. Use this list to select the color accent for the new layer. The color of your layer can be changed later using the [Properties](#) palette.

Create a layer by clicking anywhere on the EER Diagram canvas and, while holding the left mouse button down, draw a rectangle of a suitable size. This creates a new layer with the default name `layer1`. To revert to the default mouse pointer, click the arrow icon at the top of the vertical toolbar.

The following image shows a layer containing a number of views.

Figure 9.18. The Layer Object



To change the name of a layer, use the `name` property of the [Properties](#) palette.

Right-clicking a layer opens a pop-up menu with the following items:

- Cut '`layer_name`'
- Copy '`layer_name`'
- Delete '`layer_name`'

The cut and copy items are useful for copying layers between different schemata.

Since layers are not schema objects, no confirmation dialog box opens when you delete a layer regardless of how you have configured MySQL Workbench. Deleting a layer does **not** delete schema objects from the catalog.

9.3.5.1.1. Adding Objects to a Layer

To add an object to a layer, drag and drop it directly from the [Catalog](#) palette onto a layer. If you pick up an object from an EER diagram, you must press **Control** as you drag it onto the layer, otherwise it will not be “locked” inside the layer.

Locking objects to a layer prevents their accidental removal. You cannot remove them by clicking and dragging; to remove an object, you also must press the **Control** key while dragging it.

As a visual cue that the object is being “locked”, the outline of the layer is highlighted as the object is dragged over it.

If you drag a layer over a table object, the table object will automatically be added to the layer. This also works for multiple table objects.

Layers cannot be nested. That is, a layer cannot contain another layer object.

9.3.5.2. Modifying a Layer Using the Properties Palette

When you select a layer on the EER Diagram canvas, its properties are displayed in the [Properties](#) palette. The properties accessible from the [Properties](#) palette apply to the appearance of a layer on the EER Diagram canvas.

In some circumstances, you may want to make a layer invisible. Select the layer and, in the [Properties](#) palette, set the `visible` property to `False`. To locate an invisible object, open the [Layers](#) palette and select the object by double-clicking it. After an object is selected, you can reset the `visible` property from the [Properties](#) palette.

For a list of properties accessible through the [Properties](#) palette, see [Section 9.1.12, “The Properties Palette”](#). In addition to the properties listed there, a layer also has a `description` property. Use this property to document the purpose of the layer.

9.3.6. Creating Notes

You can add notes to a database only from the [Model Notes](#) section of the [MySQL Model](#) page. Notes are typically used to help document the design process.

9.3.6.1. Adding Notes

Double-clicking the [Add Note](#) icon in the [Model Notes](#) section of the [MySQL Model](#) page adds a note with the default name of `note1`. If a note with this name already exists, the new note is named `note2`.

Adding a new note automatically opens the note editor docked at the bottom of the application. For information about using the note editor, see [Section 9.3.6.2, “The Note Editor”](#).

Right-clicking a note opens a pop-up menu with the following items:

- Rename
- Cut '`note_name`'
- Copy '`note_name`'
- Delete '`note_name`'

The [Edit Note...](#) item opens the note editor. For information about using the note editor, see [Section 9.3.6.2, “The Note Editor”](#).

The cut and copy items are useful for copying notes between different schemata.

Notes can be added only on the [MySQL Model](#) page.

9.3.6.2. The Note Editor

To invoke the note editor, double-click a note object in the [Model Note](#) section on the [MySQL Model](#) page. This opens the note editor docked at the bottom of the application. Double-clicking the note tab undocks the editor. Double-click the title bar to redock it. Any number of notes may be open at the same time. Each additional note appears as a tab at the top of the note editor.

Use the editor to change the name of a note or its contents.

9.3.7. Creating Text Objects

Text objects are applicable only to an EER diagram. They can be used for documentation purposes; for example, to explain a grouping of schema objects. They are also useful for creating titles for an EER diagram should you decide to export a diagram as a PDF or PNG file.

9.3.7.1. Adding Text Objects to an EER Diagram

To add text objects to an EER Diagram, use the [Text Object](#) tool on the vertical toolbar. Make sure that the **EER Diagram** tab is selected, then right-click the text object icon on the vertical toolbar. The text object icon is the rectangular icon found below the label icon.

Clicking the mouse on this icon changes the mouse pointer to a text object pointer. You can also change the mouse pointer to a text object pointer by pressing the **N** key.

Choosing the [Text Object](#) tool changes the contents of the toolbar that appears immediately below the menu bar. When the [Text Object](#) pointer is active, this toolbar contains a color chart list. Use this list to select the color accent for the new text object. The color of your text object can be changed later using the [Properties](#) palette.

Create a text object by clicking anywhere on the EER Diagram canvas. This creates a new text object with the default name `text1`. To revert to the default mouse pointer, click the arrow icon at the top of the vertical toolbar.

Right-clicking a text object opens a pop-up menu. These menu options are identical to the options for other objects. However, since a text object is not a database object, there is no confirmation dialog box when you delete a text object.

9.3.7.2. The Text Object Editor

To invoke the text object editor, double-click a text object on the EER Diagram canvas. This opens the editor docked at the bottom of the application. Double-clicking the text object table undocks the editor. Double-click the title bar to redock it. Any number of text objects may be open at the same time. Each additional text objects appears as a tab at the top of the text editor.

Use the editor to change the name of a text object or its contents.

9.3.7.2.1. Modifying a Text Object Using the [Properties](#) Palette

When you select a text object on the EER Diagram canvas, its properties are displayed in the [Properties](#) palette. Most of the properties accessible from the [Properties](#) palette apply to the appearance of a view on the EER Diagram canvas.

For a list of properties accessible through the [Properties](#) palette, see [Section 9.1.12, “The Properties Palette”](#).

There is no property in the [Properties](#) palette for changing the font used by a text object. To do so, choose the **Appearance** tab of the Workbench Preferences dialog. For more information, see [Section 7.3.8, “The Appearance Tab”](#).

9.3.8. Creating Images

Images exist only on the EER Diagram canvas; you can add them only from the EER Diagram window.

9.3.8.1. Adding Images to an EER Diagram

To add images to an EER Diagram, use the [Image](#) tool on the vertical toolbar. Make sure that the **EER Diagram** tab is selected, then right-click the image icon on the vertical toolbar. The image icon is the icon just above the table icon.

Clicking the mouse on this icon changes the mouse pointer to an image pointer. You can also change the mouse pointer to an image pointer by pressing the **I** key.

Create an image by clicking anywhere on the EER Diagram canvas. This opens a file open dialog box. Select the desired image, then close the dialog box to create an image on the canvas. To revert to the default mouse pointer, click the arrow icon at the top of the vertical toolbar.

Right-clicking this object opens a pop-up menu with the following items:

- Cut '[Image](#)'
- Copy '[Image](#)'
- Edit Image...
- Edit in New Window...
- Delete '[Image](#)'

These menu items function in exactly the same way as they do for other objects on an EER diagram. However, images are not database objects so there is no confirmation dialog box when they are deleted.

9.3.8.2. The Image Editor

To invoke the image editor, double-click an image object on an EER Diagram canvas. This opens the image editor docked at the bottom of the application. Double-clicking the image editor tab undocks the editor. Double-click the title bar to redock it. Any number of images may be open at the same time. Each additional image appears as a tab at the top of the image editor.

9.3.8.2.1. The Image Tab

Use the **Image** tab of the image editor to perform the following tasks:

- Rename the image using the **Name** text box.
- Browse for an image using the **Browse** button.

9.3.9. Reverse Engineering

With MySQL Workbench, you can reverse engineer a database using a MySQL create script or you can connect to a live MySQL server and import a single database or a number of databases. All versions of MySQL Workbench can reverse engineer using a MySQL DDL script. Only commercial versions of MySQL Workbench can reverse engineer a database directly from a MySQL server.

9.3.9.1. Reverse Engineering Using a Create Script

To reverse engineer using a create script, choose the **File**, Import, Reverse Engineer MySQL Create Script... menu items. This opens a file open dialog box with the default file type set to an SQL script file, a file with the extension `.sql`.

You can create a data definition (DDL) script by executing the `mysqldump db_name --no-data > script_file.sql` command. Using the `--no-data` option ensures that the script contains only DDL statements. However, if you are working with a script that also contains DML statements you need not remove them; they will be ignored.



Note

If you plan to redesign a database within MySQL Workbench and then export the changes, be sure to retain a copy of the original DDL script. You will need the original script to create an [ALTER](#) script. For more information, see [Section 9.3.10.1.2, “Altering a Schema”](#).

Use the `--databases` option with `mysqldump` if you wish to create the database as well as all its objects. If there is no `CREATE DATABASE db_name` statement in your script file, you must import the database objects into an existing schema or, if there is no schema, a new unnamed schema is created.

If your script creates a database, MySQL Workbench creates a new physical schemata tab on the [MySQL Model](#) page.

Any database objects may be imported from a script file in this fashion: tables, views, routines, and routine groups. Any indexes, keys, and constraints are also imported. Objects imported using an SQL script can be manipulated within MySQL Workbench the same as other objects.

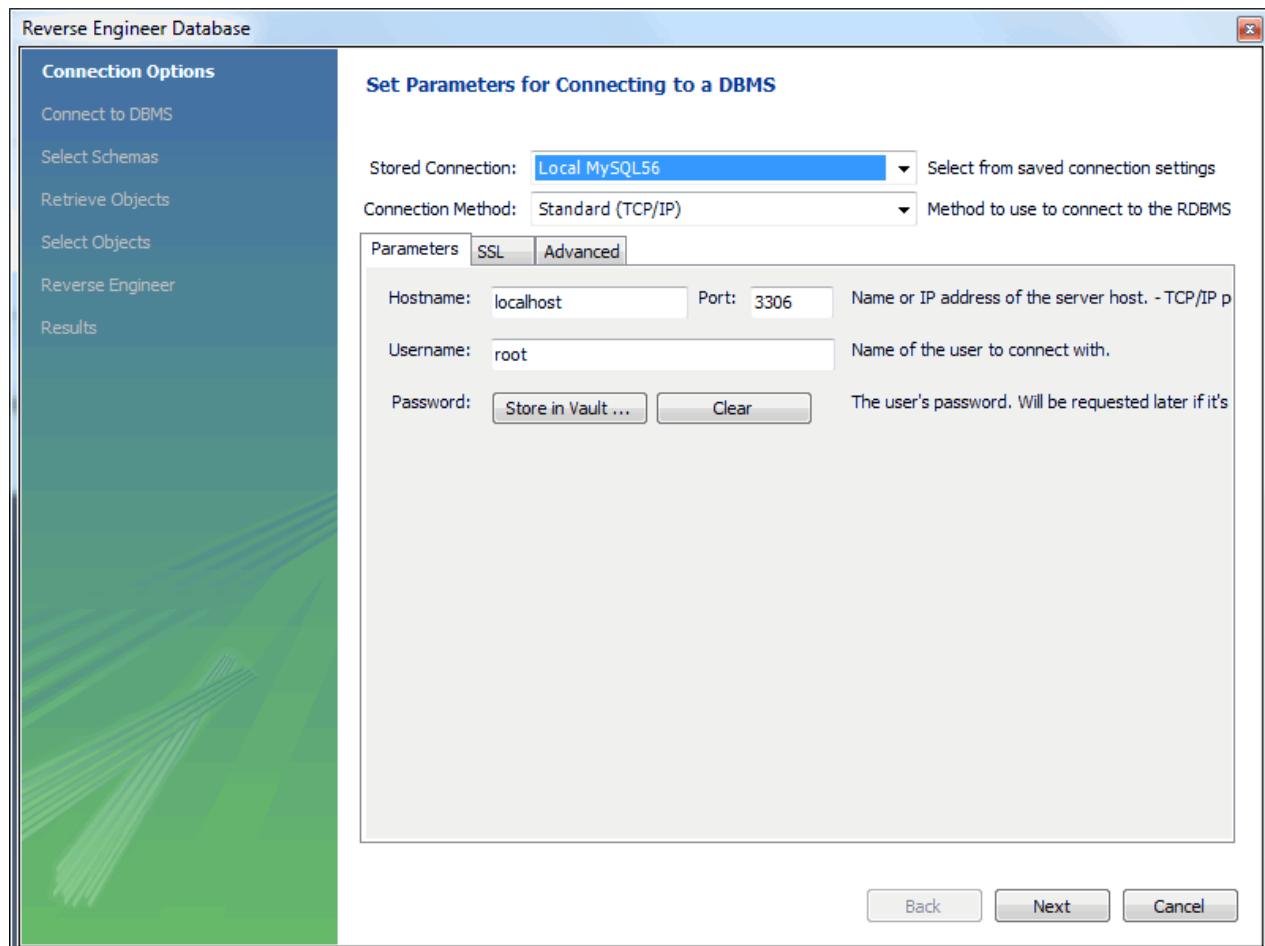
Before exiting, be sure to save the schema. Choose the [File](#), Save menu item and the reverse-engineered database will be saved as a MySQL Workbench file with the extension `mwb`.

See [Section 9.4.1, “Importing a Data Definition SQL Script”](#), for a tutorial on reverse engineering the `sakila` database.

9.3.9.2. Reverse Engineering a Live Database

To reverse engineer a live database, choose the [Database](#), Reverse Engineer... menu item from the main menu. This opens the Reverse Engineer Database wizard.

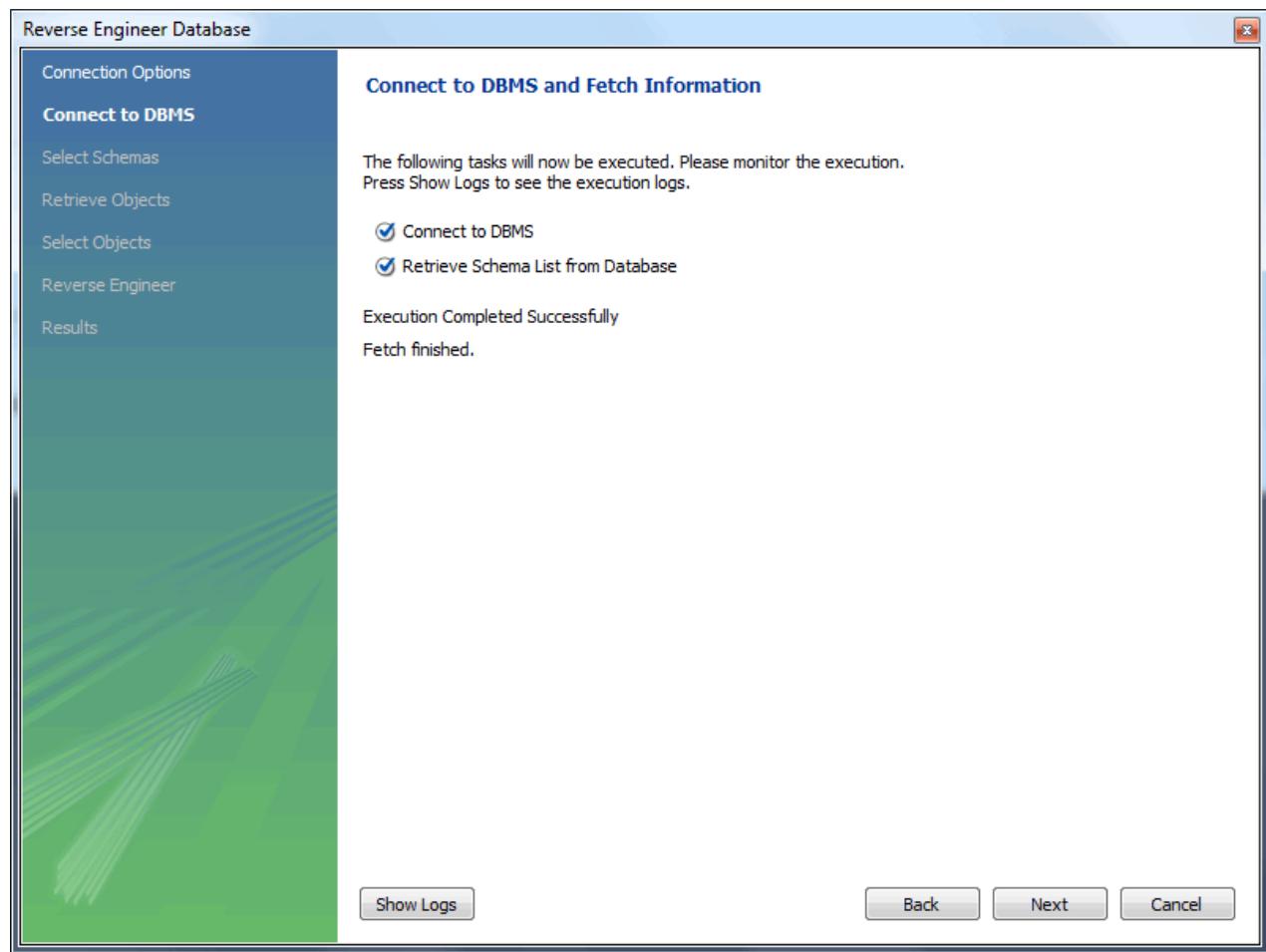
Figure 9.19. Reverse Engineer Database Wizard



The first page of the wizard enables you to set up a connection to the live database you wish to reverse engineer. You can set up a new connection or select a previously created stored connection. Typical information required for the connection includes host name, user name and password.

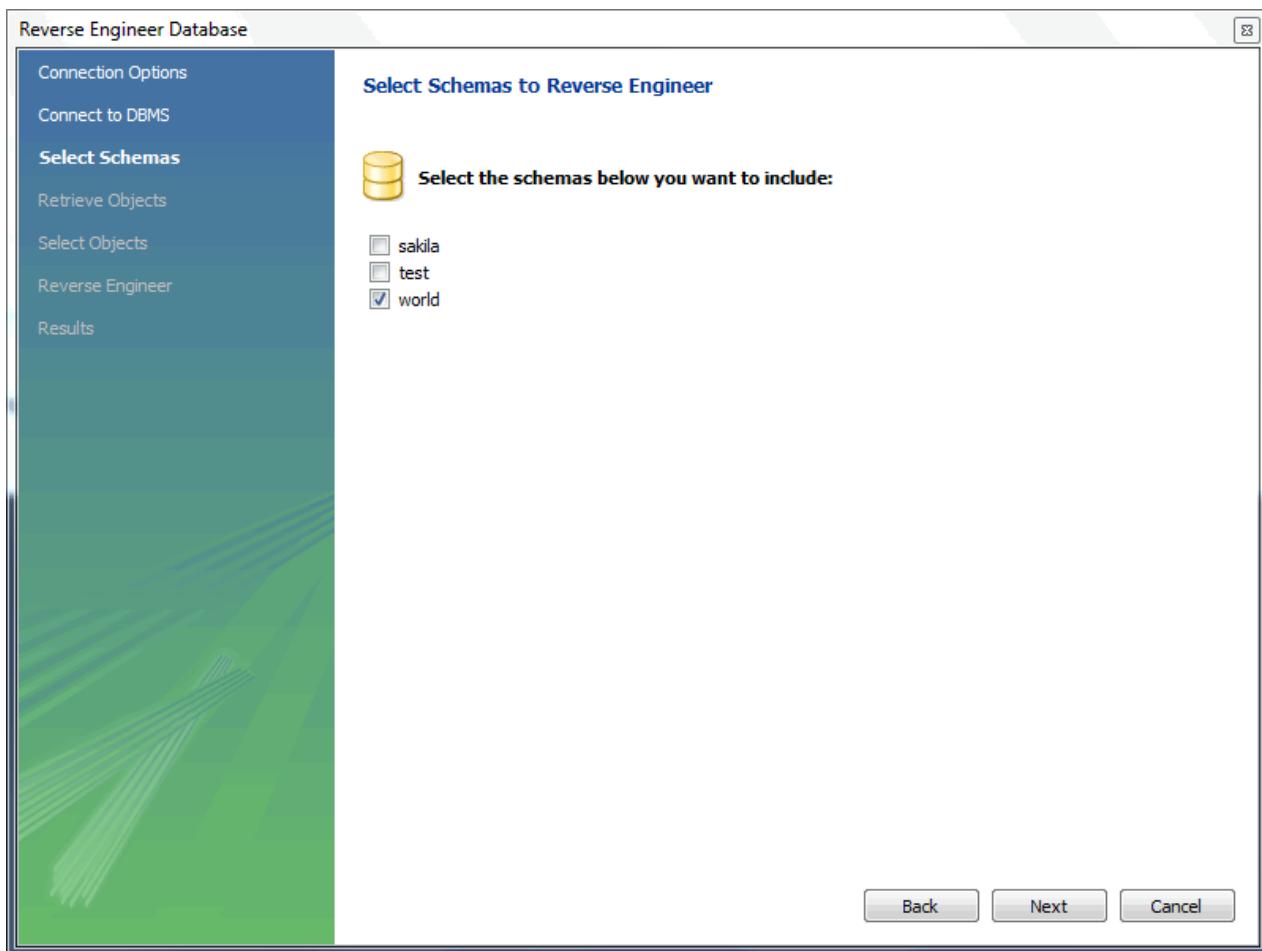
After this information has been entered, or you have selected a stored connection, click the Next button to proceed to the next page.

Figure 9.20. Connect to DBMS



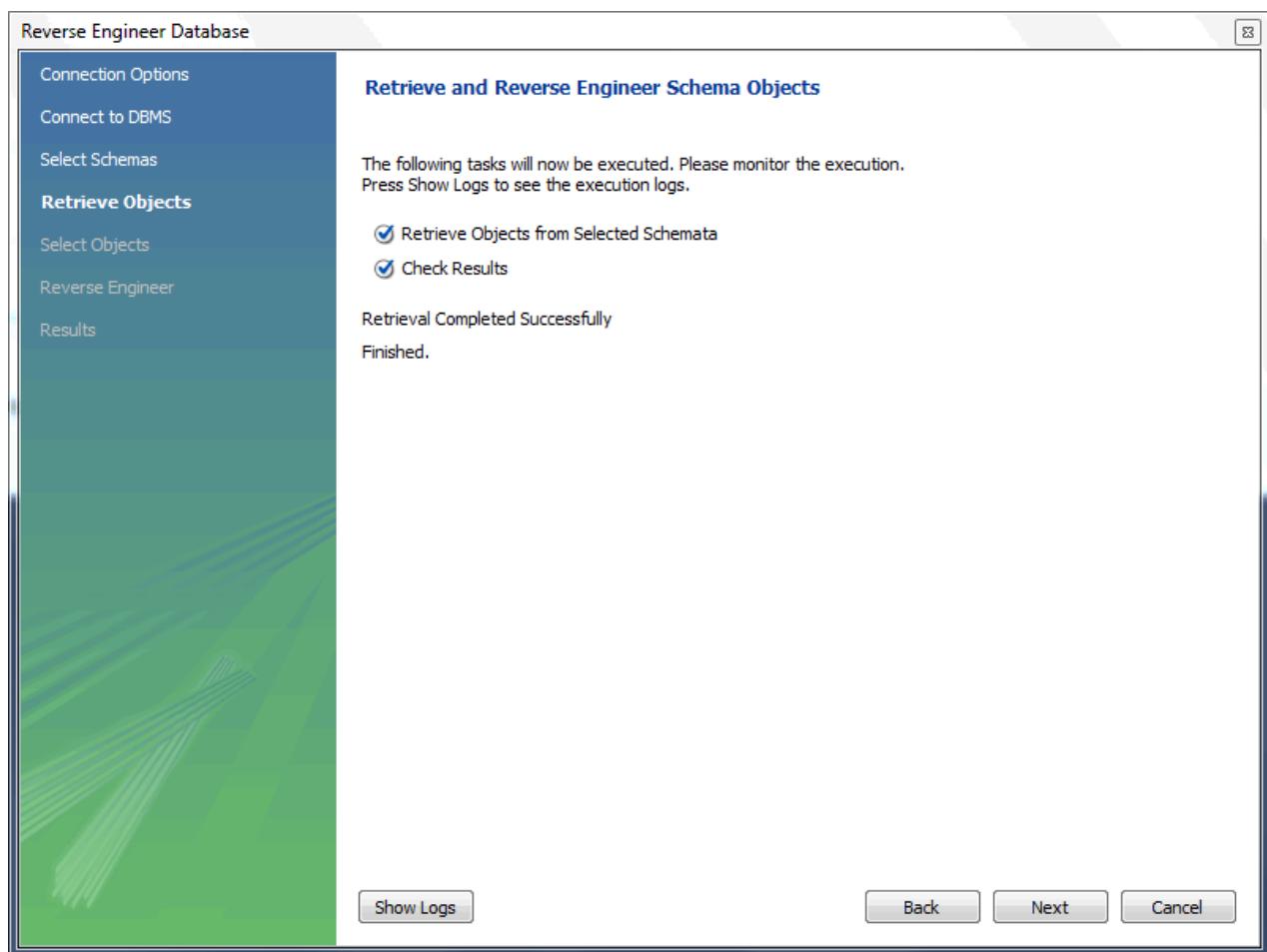
Review the displayed information to make sure that the connection did not generate errors, then click **Next**.

The next page displays the schemata available on the server. Click the check box or check boxes for any schemata you wish to process.

Figure 9.21. Select Schemas

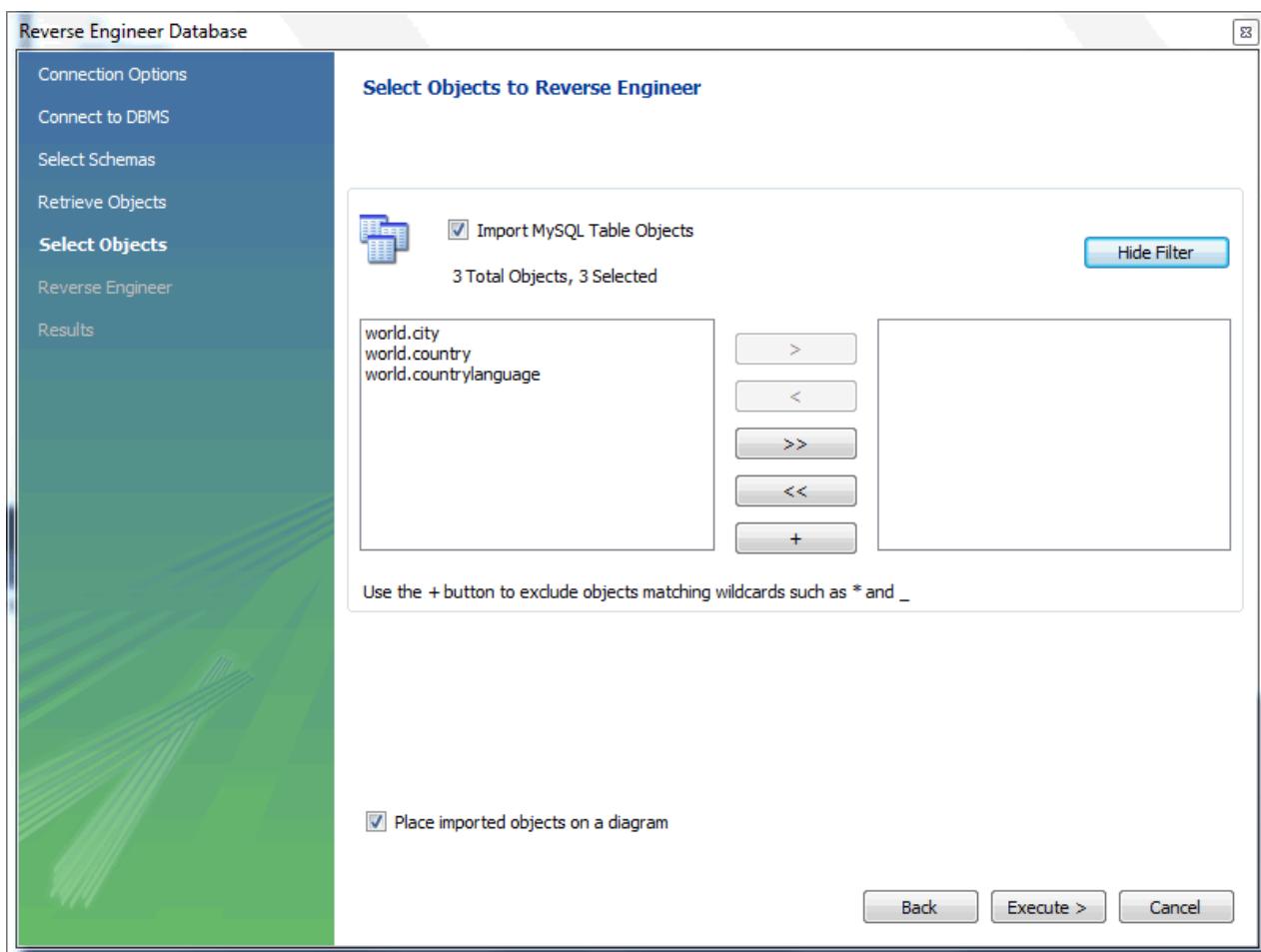
After you have selected the desired schemas, click the Next button to continue.

The wizard then displays the tasks it carried out and summarizes the results of the operation.

Figure 9.22. Retrieve Objects

Review the results before clicking Next to continue.

The next page is the [Select Objects](#) page. It has a section for each object type present in the schema (tables, views, routines, and so forth). This page is of special interest if you do not wish to import all the objects from the existing database. It gives you the option of filtering which objects are imported. Each section has a [Show Filter](#) button. Click this button if you do not want to import all the objects of a specific type. Here is this page with the filter open:

Figure 9.23. Select Objects

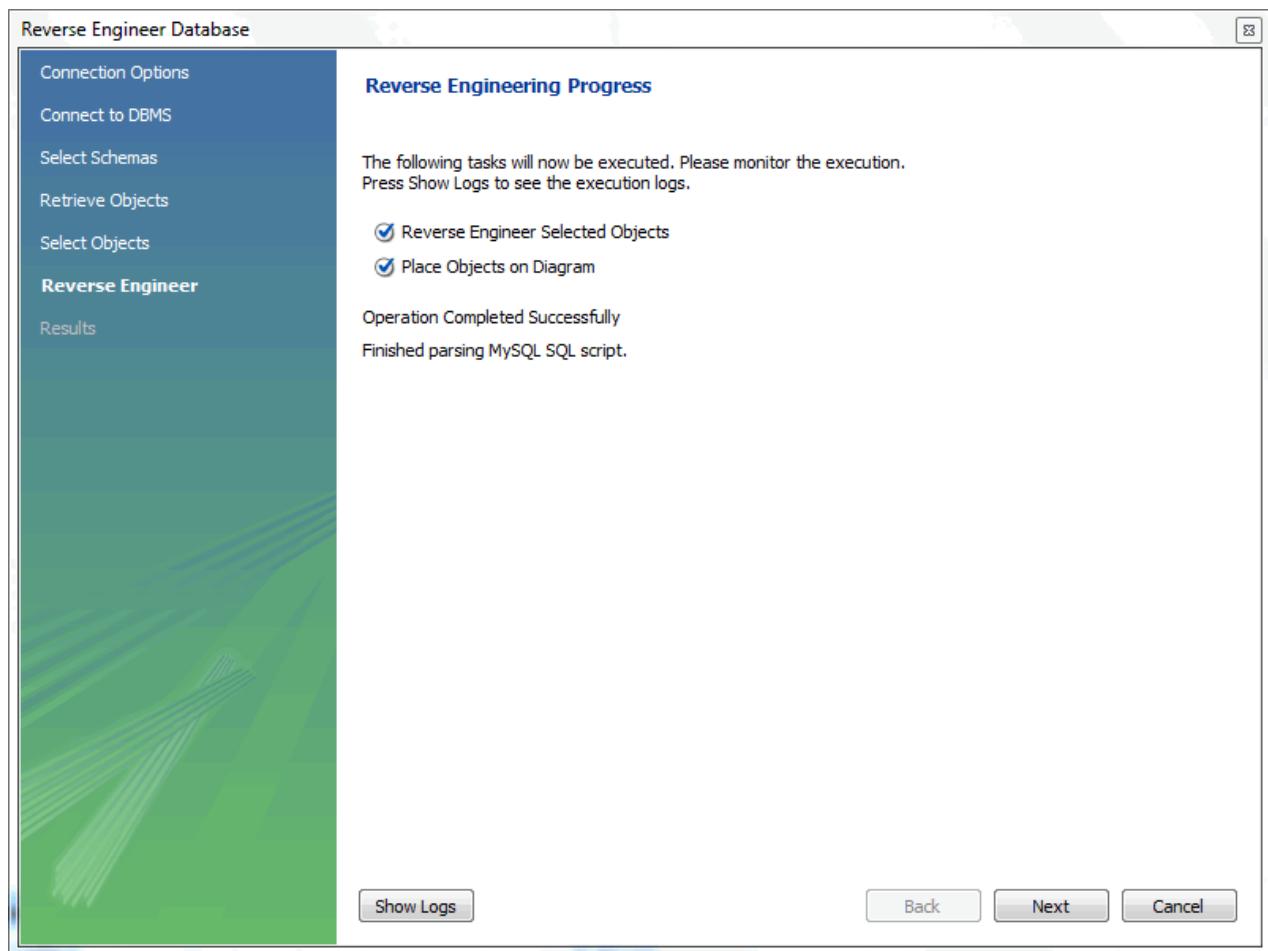
This page enables you to select specific tables for import. Having selected the desired tables, you can optionally hide the filter by clicking the **Hide Filter** button.

The other sections, such as **MySQL Routine Objects**, have similar filters available.

Click **Execute** to continue to the next page.

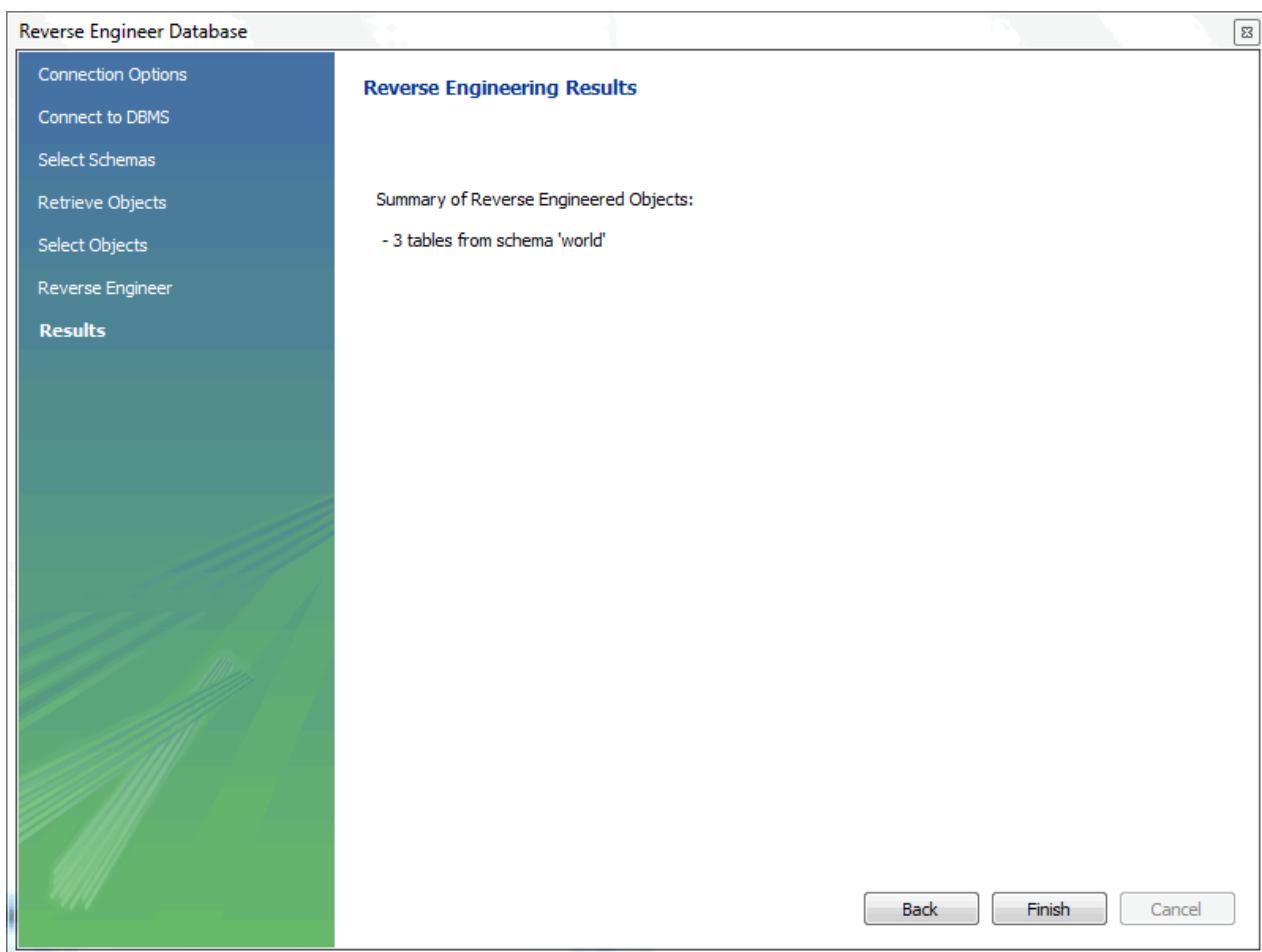
The wizard then imports objects, displaying the tasks that have been carried out and whether the operation was successful. If errors were generated, you can click the **Show Logs** button to see the nature of the errors.

Figure 9.24. Reverse Engineer Progress



Click Next to continue to the next page.

The final page of the wizard provides a summary of the reverse engineered objects.

Figure 9.25. Results

Click **Finish** to exit the wizard.

Before exiting MySQL Workbench be sure to save the schema. Choose the [File, Save](#) menu item to save the reverse-engineered database as a MySQL Workbench file with the extension [mwb](#).

9.3.9.2.1. Errors During Reverse Engineering

During reverse engineering, the application checks for tables and views that duplicate existing names and disallows duplicate names if necessary. If you attempt to import an object that duplicates the name of an existing object you will be notified with an error message. To see any errors that have occurred during reverse engineering, you can click the button [Show Logs](#). This will create a panel containing a list of messages, including any error messages than may have been generated. Click the [Hide Logs](#) button to close the panel.

If you wish to import an object with the same name as an existing object, rename the existing object before reverse engineering.

If you import objects from more than one schema, there will be a tab in the [Physical Schemata](#) section of the [MySQL Model](#) page for each schema imported.

You cannot reverse engineer a live database that has the same name as an existing schema. If you wish to do this, first rename the existing schema.

9.3.10. Forward Engineering

It is possible to forward engineer a database using an SQL script or by connecting to a live database.

9.3.10.1. Forward Engineering Using an SQL Script

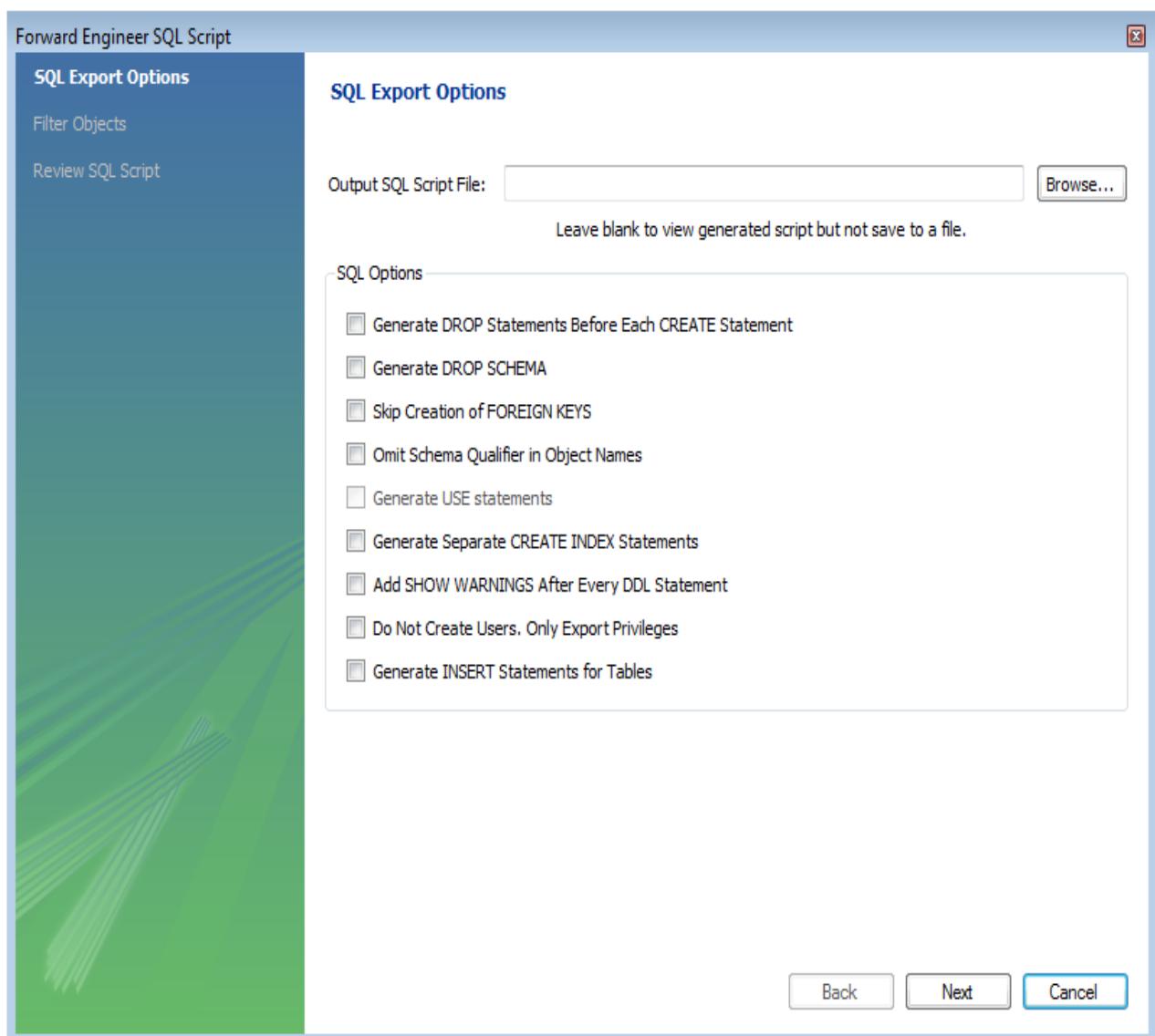
To create a script of your database model, choose the Export item from the File menu. You may export a script to alter an existing database or create a new database. The script to create a database is similar to the one created using the `mysqldump db_name` command.

If you choose to create a database, there are several export options available.

9.3.10.1.1. Creating a Schema

Select the File, Export, Forward Engineer SQL CREATE Script menu item to start the Forward Engineer SQL Script wizard. The following figure shows the first page of the wizard.

Figure 9.26. SQL Export Options



The SQL Export Options displays the following facilities:

- [Output SQL Script File](#)

To specify the output file name, enter it into the **Output SQL Script File** field, or use the **Browse** button to select a file. If this field is left blank, you will be able to view the generated script, but it will not be saved to a file.

- [Generate DROP Statements Before Each CREATE Statement](#)

Select this option to generate a statement to drop each object before the statement that creates it. This ensures that any existing instance of each object is removed when the output is executed.

- [Omit Schema Qualifier in Object Names](#)

Select this option to generate unqualified object names in SQL statements.

- [Generate Separate CREATE INDEX Statements](#)

Select this option to create separate statements for index creation instead of including index definitions in `CREATE TABLE` statements.

- [Add SHOW WARNINGS after every DDL statement](#)

Select this option to add `SHOW WARNINGS` statements to the output. This causes display of any warnings generated when the output is executed, which can be useful for debugging.

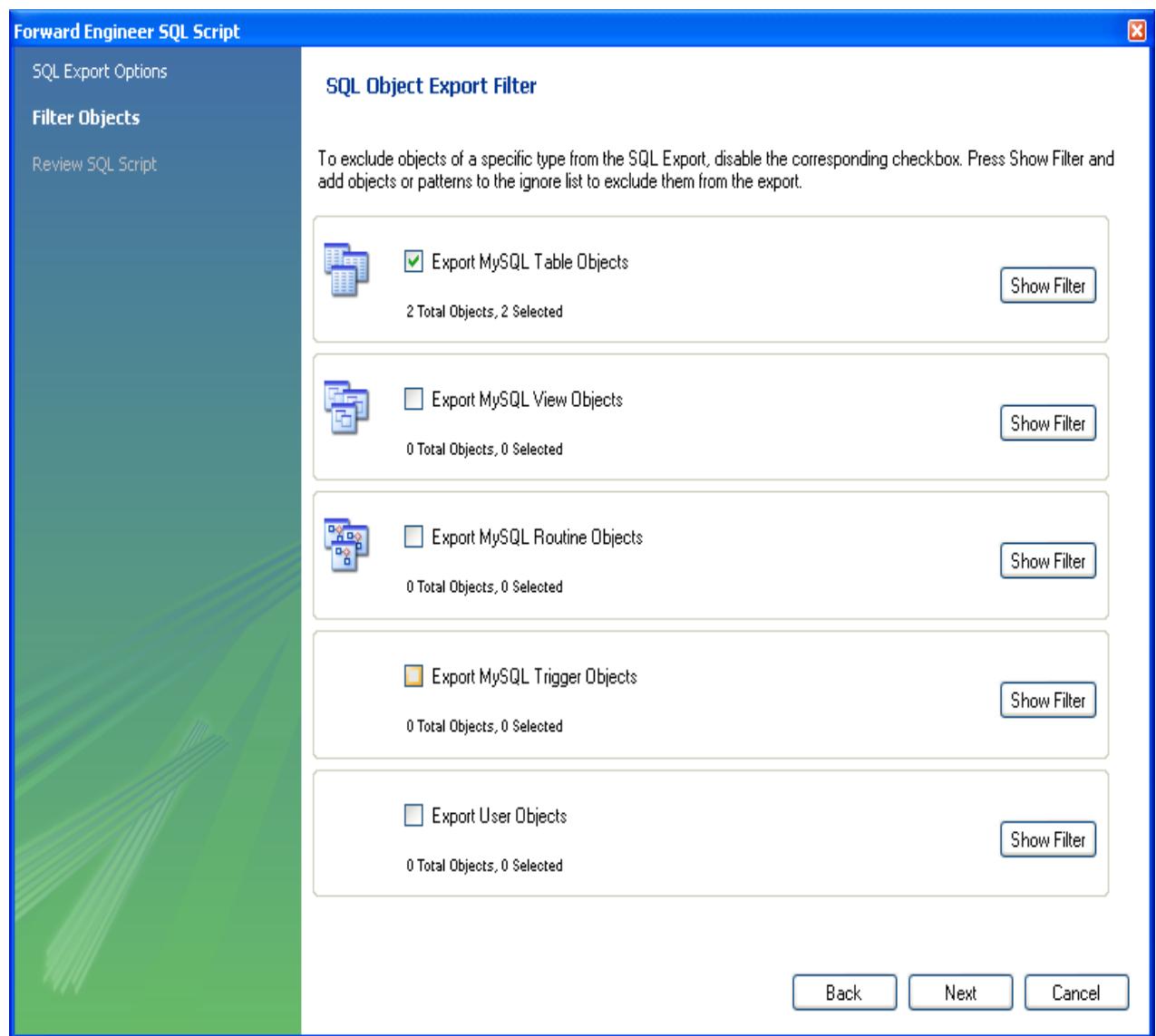
- [Do Not Create Users. Only Export Privileges](#)

Select this option to update the privileges of existing users, as opposed to creating new users. Exporting privileges for nonexistent users will result in errors when you execute the `CREATE` script. Exporting users that already exist will also result in an error.

- [Generate INSERT Statements for Tables](#)

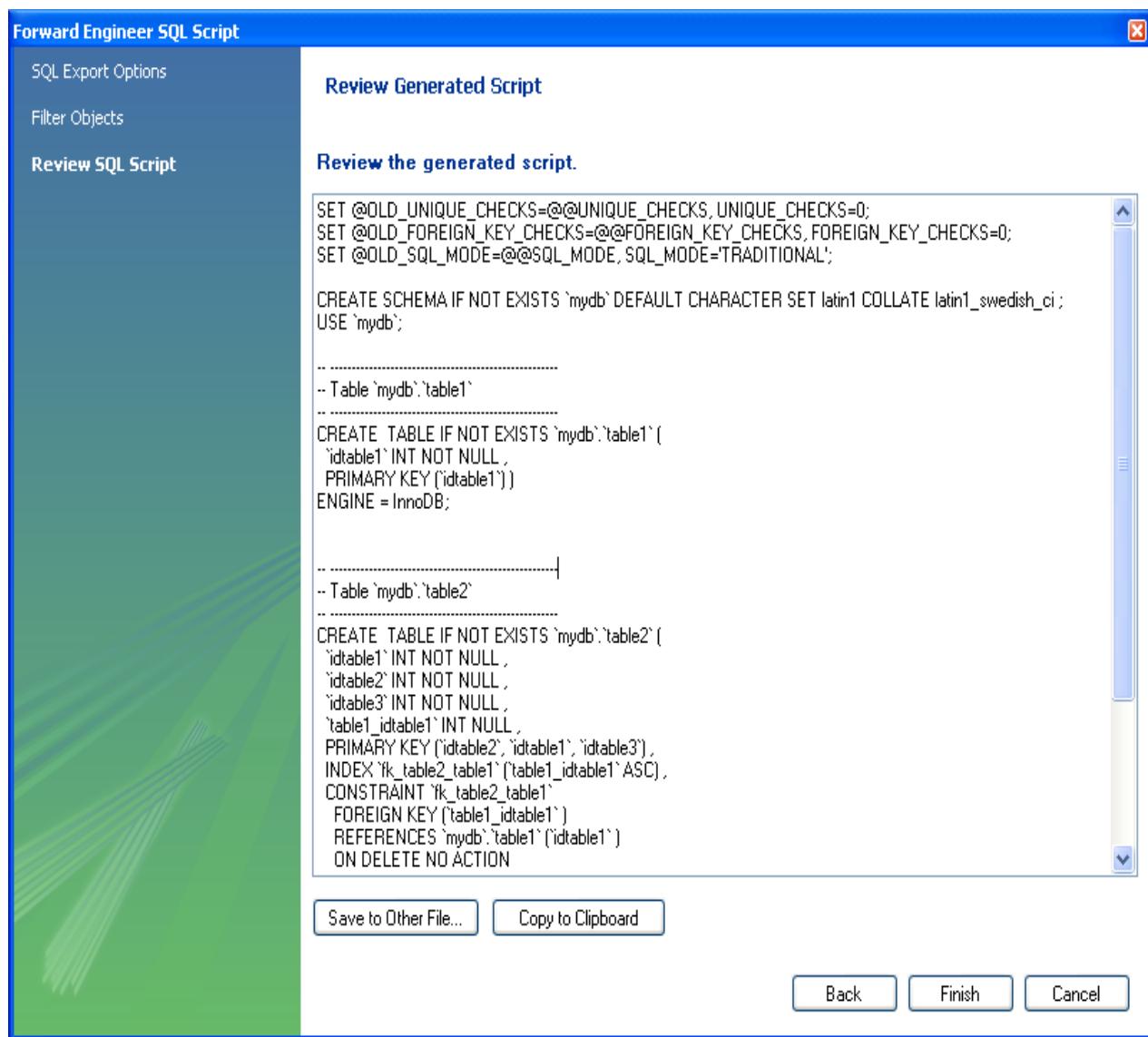
Select this option if you have added any rows to a table. For more information about inserting rows, see [Section 9.3.1.3.8, "The Inserts Tab"](#).

Clicking **Next** takes you to the **SQL Object Export Filter** page where you select the objects you wish to export.

Figure 9.27. SQL Object Export Filter

Precise control over the objects to export can be fine tuned by clicking the **Show Filter** button. After the objects to export have been selected, it is possible to reduce the expanded panel by clicking the same button, now labeled **Hide Filter**.

After selecting the objects to export, click the **Next** button to review the script that has been generated.

Figure 9.28. Review Generated Script

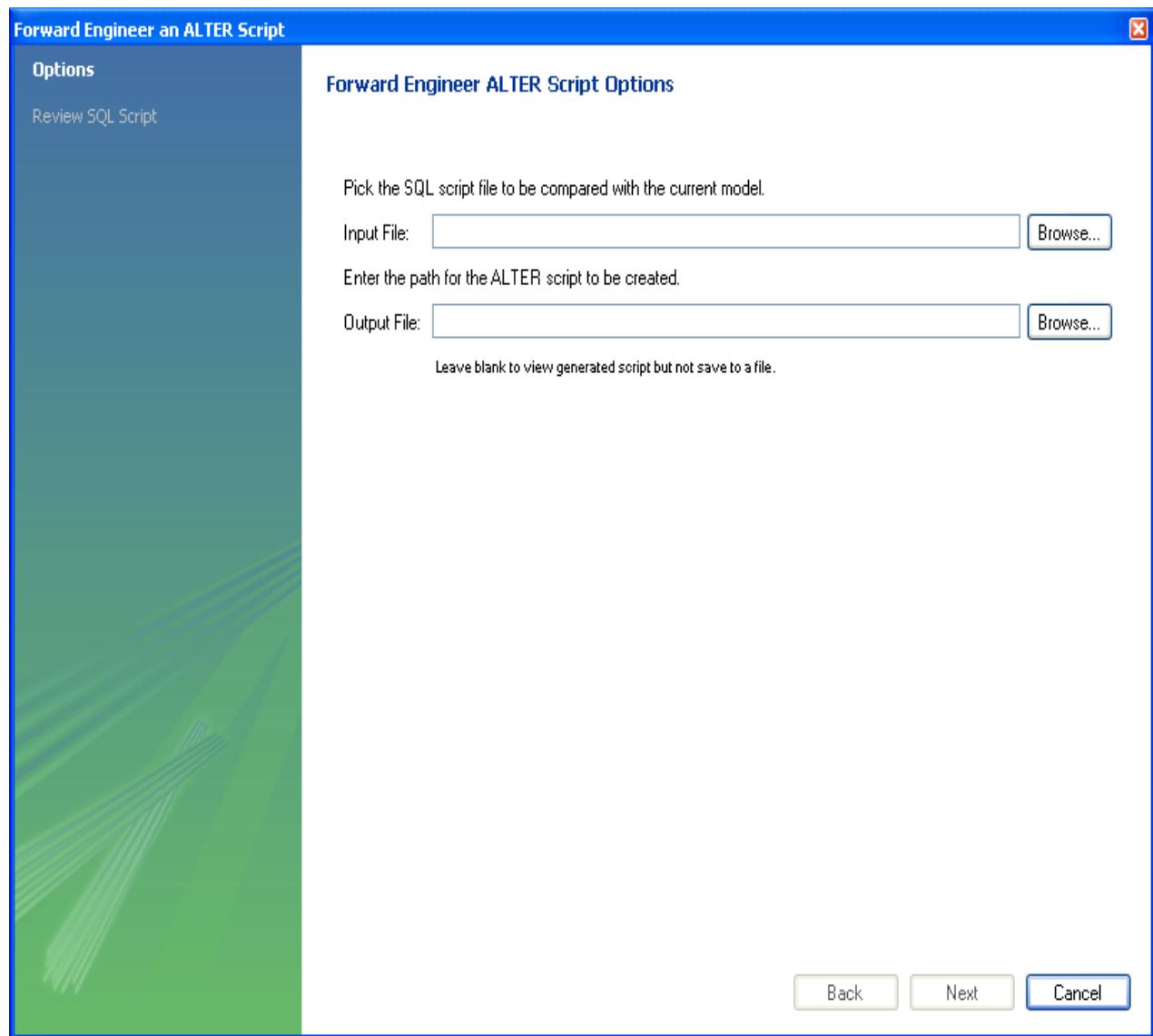
You may return to the previous page using the **Back** button.

The **Finish** button saves the script file and exits. You can then use the saved script to create a database.

9.3.10.1.2. Altering a Schema

The menu item for altering a schema, Forward Engineer SQL ALTER Script..., is used for updating a database that has been redesigned within MySQL Workbench. Typically, this option is used when the SQL script of a database has been imported into MySQL Workbench and changed, and then you want to create a script that can be run against the database to alter it to reflect the adjusted model. For instructions on importing a DDL script, see [Section 9.3.9.1, “Reverse Engineering Using a Create Script”](#).

Select the **File**, Export, Forward Engineer SQL ALTER Script menu item to start the Forward Engineer an ALTER Script wizard. You will be presented with the first page showing the available options.

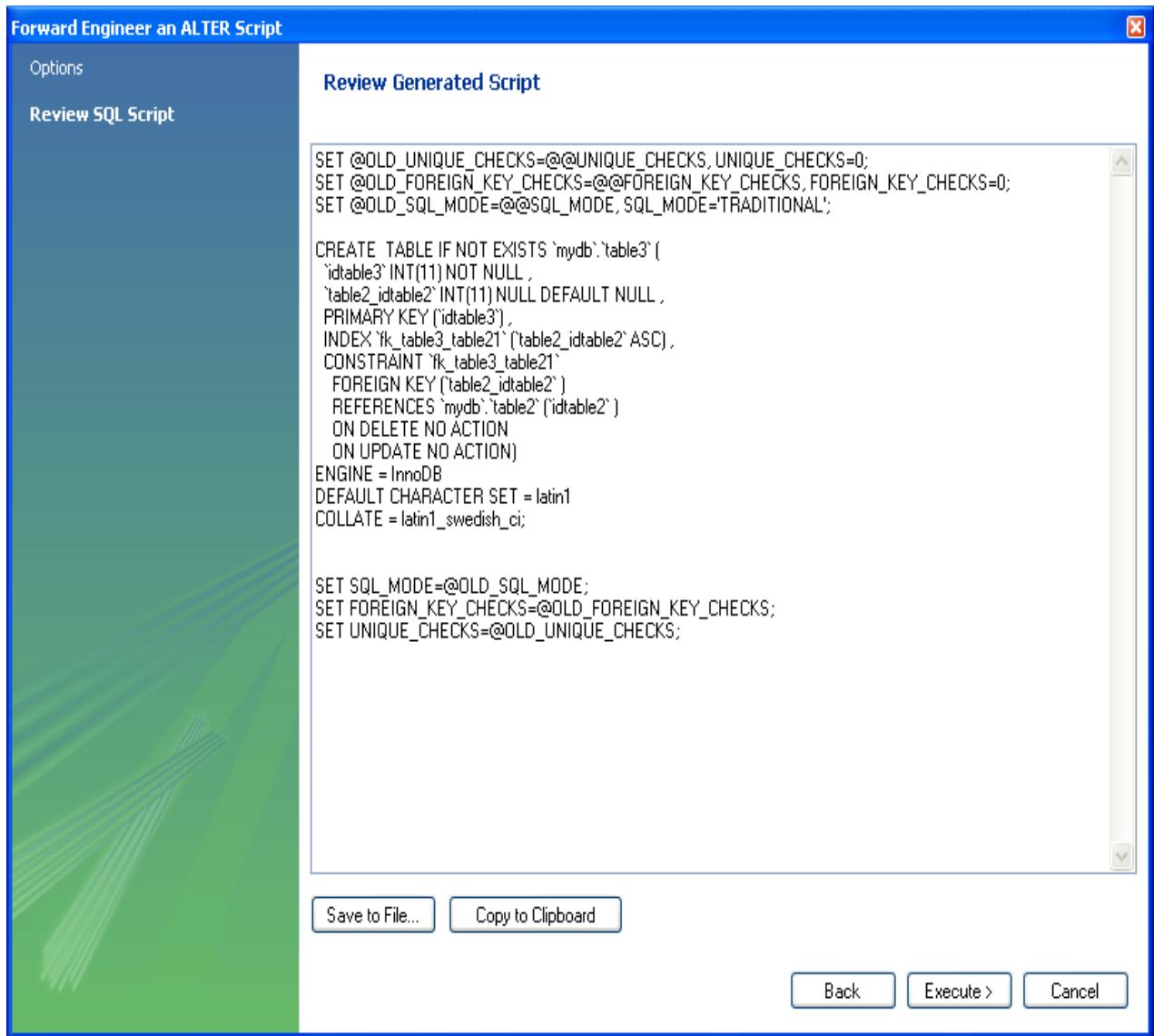
Figure 9.29. Options

This first page enables you to select an SQL script and compare it with the model currently in MySQL Workbench. The difference between the two models will be used to create an alter script that can be used to modify the target schema to match the model held in MySQL Workbench. To view the script generated, rather than saving it to a file, leave the **Output File** field empty.

**Note**

The script selected as the Input File must use full schema qualifiers, such as `schema_name.table_name`. Otherwise, MySQL Workbench cannot generate a useable alter script.

Clicking Next brings you to the **Review SQL Script** page.

Figure 9.30. Script

Here you can review and change the alter script that will be generated. Make any changes you wish and, if you are happy with the changes, save the [ALTER](#) script to file using the [Save to File...](#) button. You can also click the [Execute](#) button to tell MySQL Workbench to write the script to the previously specified output file.

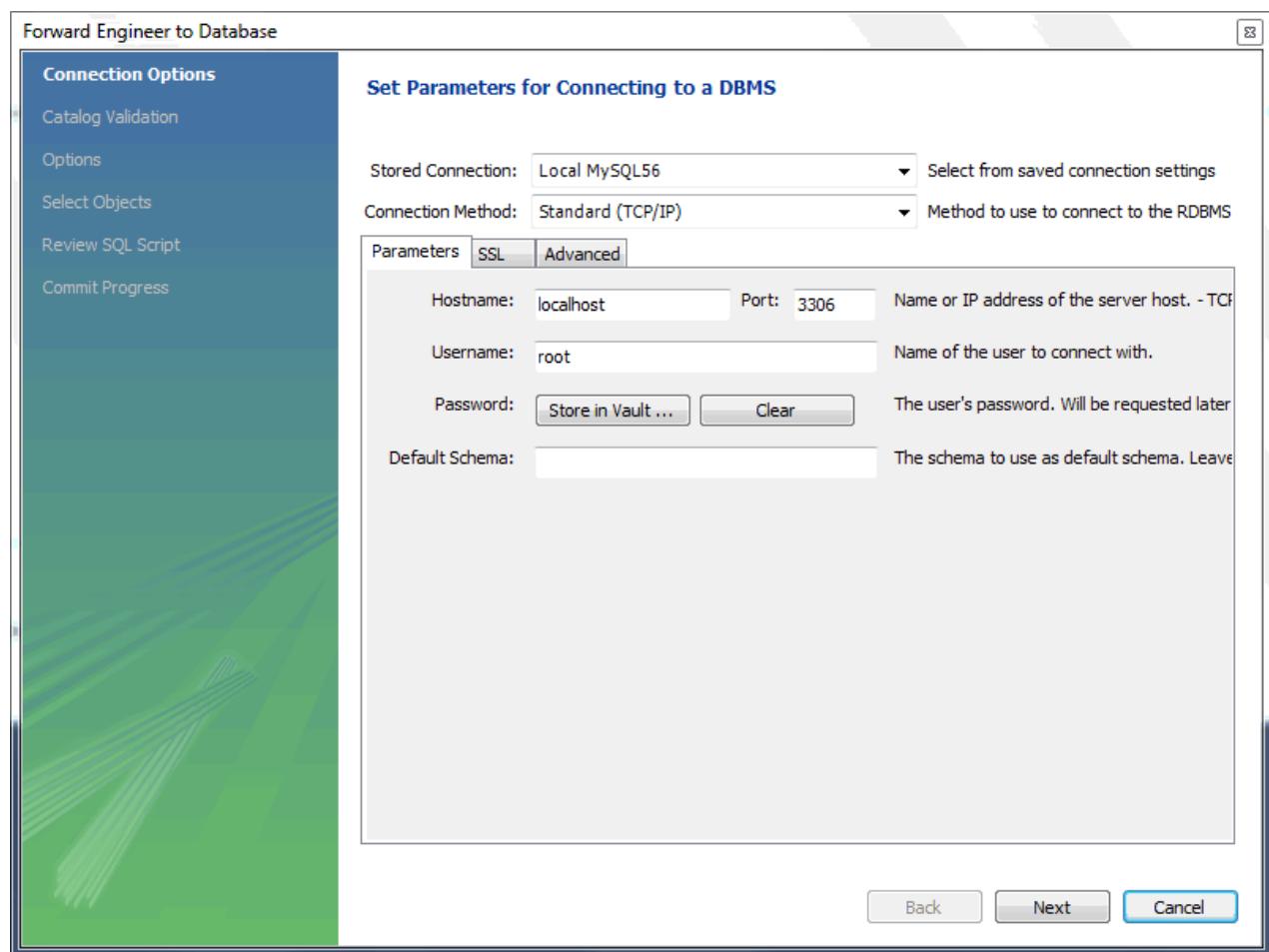
The generated script can then be used to update the database.

9.3.10.2. Forward Engineering to a Live Server

Use forward engineering to export your schema design to a MySQL server.

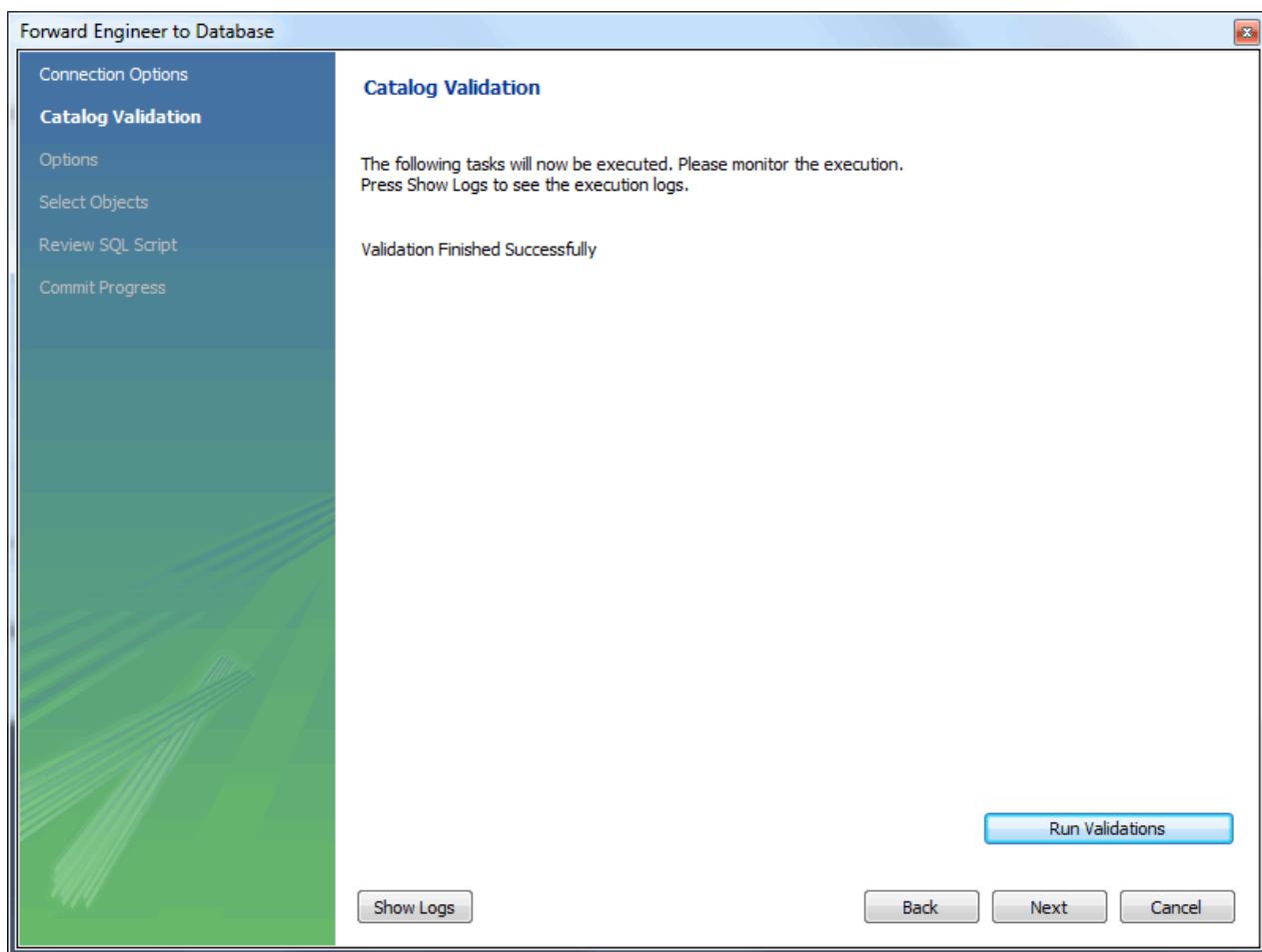
Select the schema that you wish to forward engineer and then choose the [Database](#), [Forward Engineer...](#) menu item from the main menu.

The first step of the process is to connect to a MySQL server to create the new database schema. This page enables you to use a previously stored connection, or enter the connection parameters.

Figure 9.31. Set Parameters for Connecting to a DBMS

After the connection parameters have been set, click Next. The next page of the wizard displays is Catalog Validation (validation is available only in the Commercial Edition).

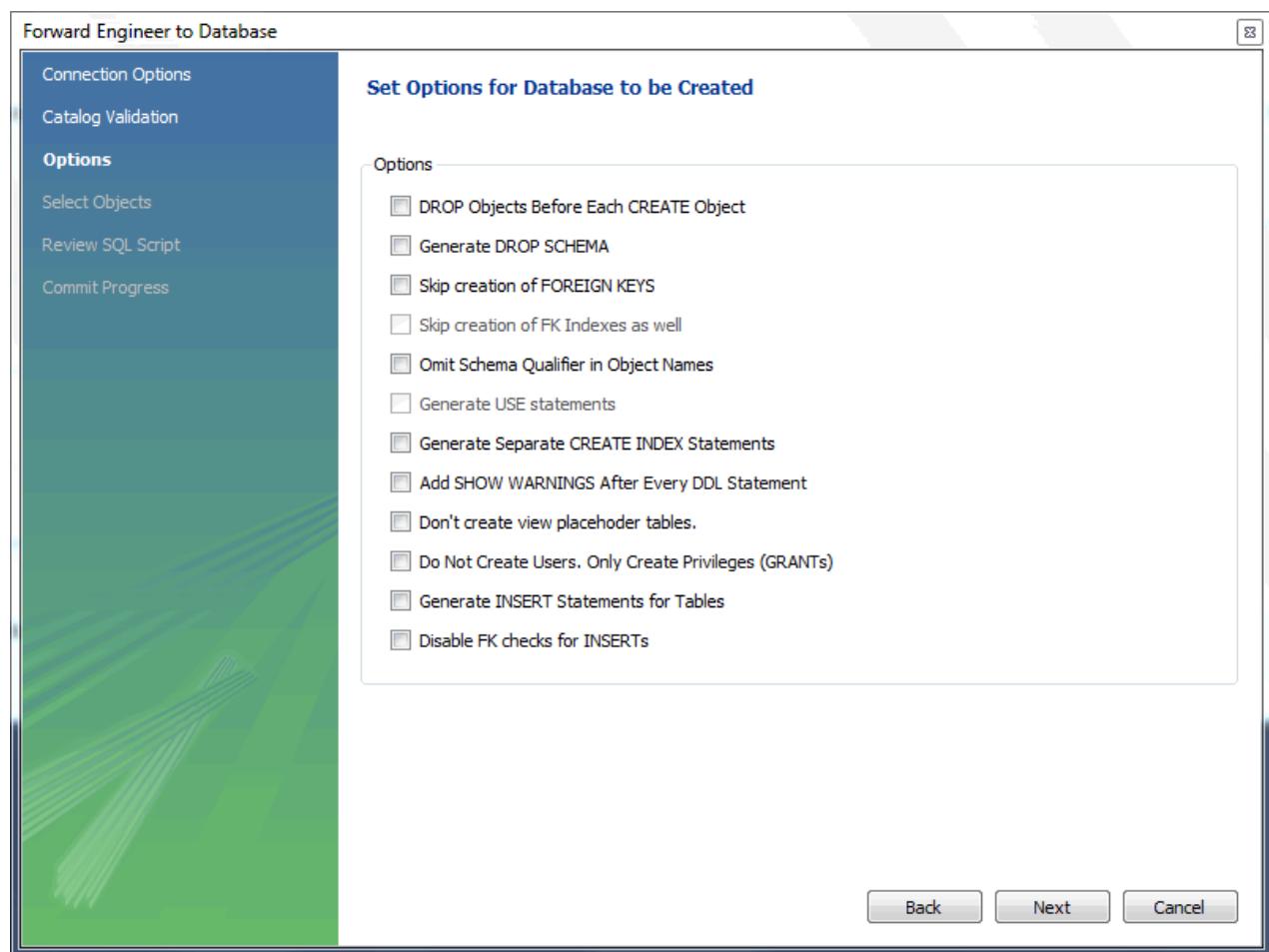
Figure 9.32. Catalog Validation



Click Run Validations to validate the catalog.

Click Next to continue.

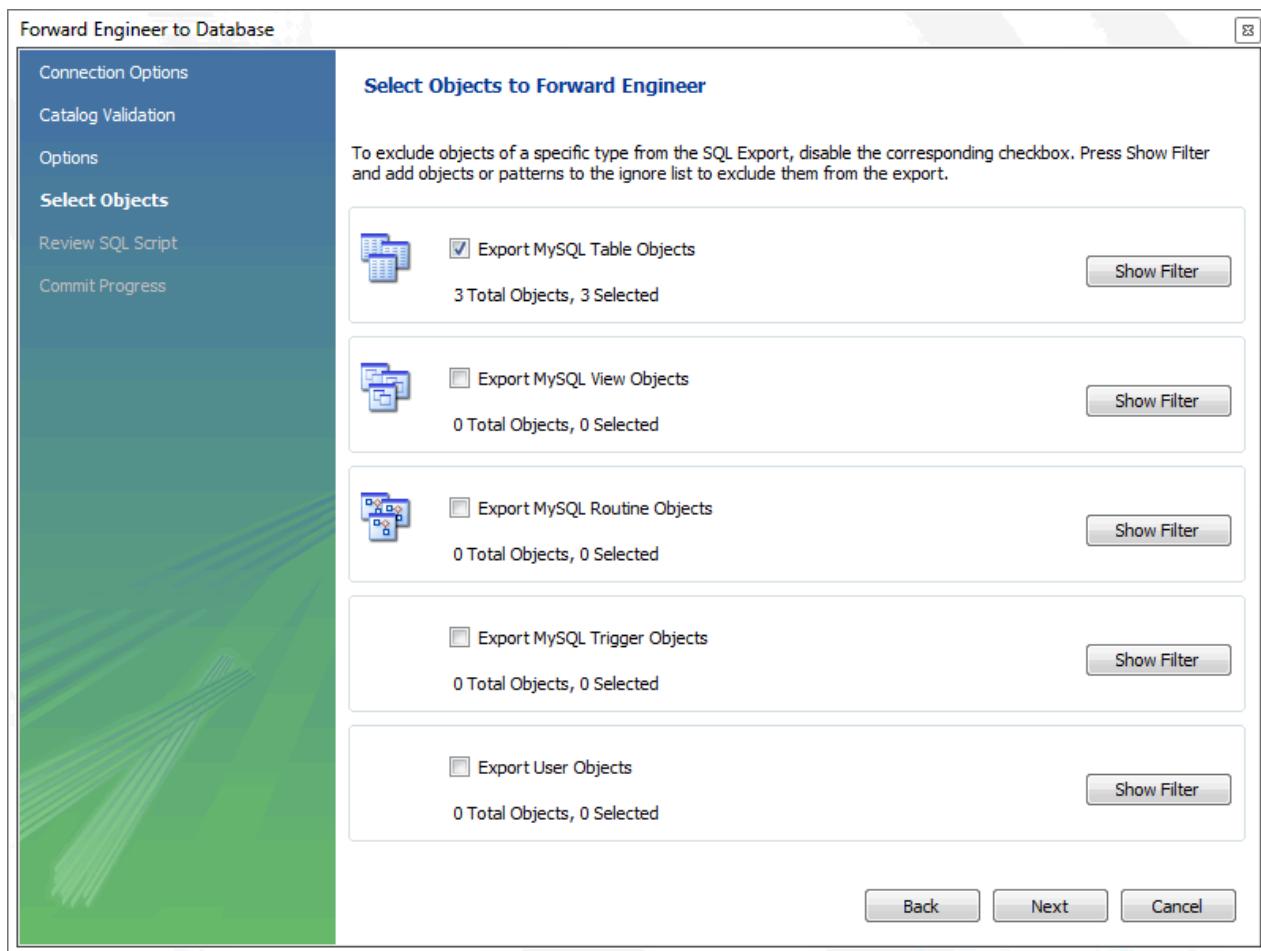
The next page enables you to set options for the database to be created. These options are as described in [Section 9.3.10.1.1, “Creating a Schema”](#).

Figure 9.33. Options

Select the required options and then click Next.

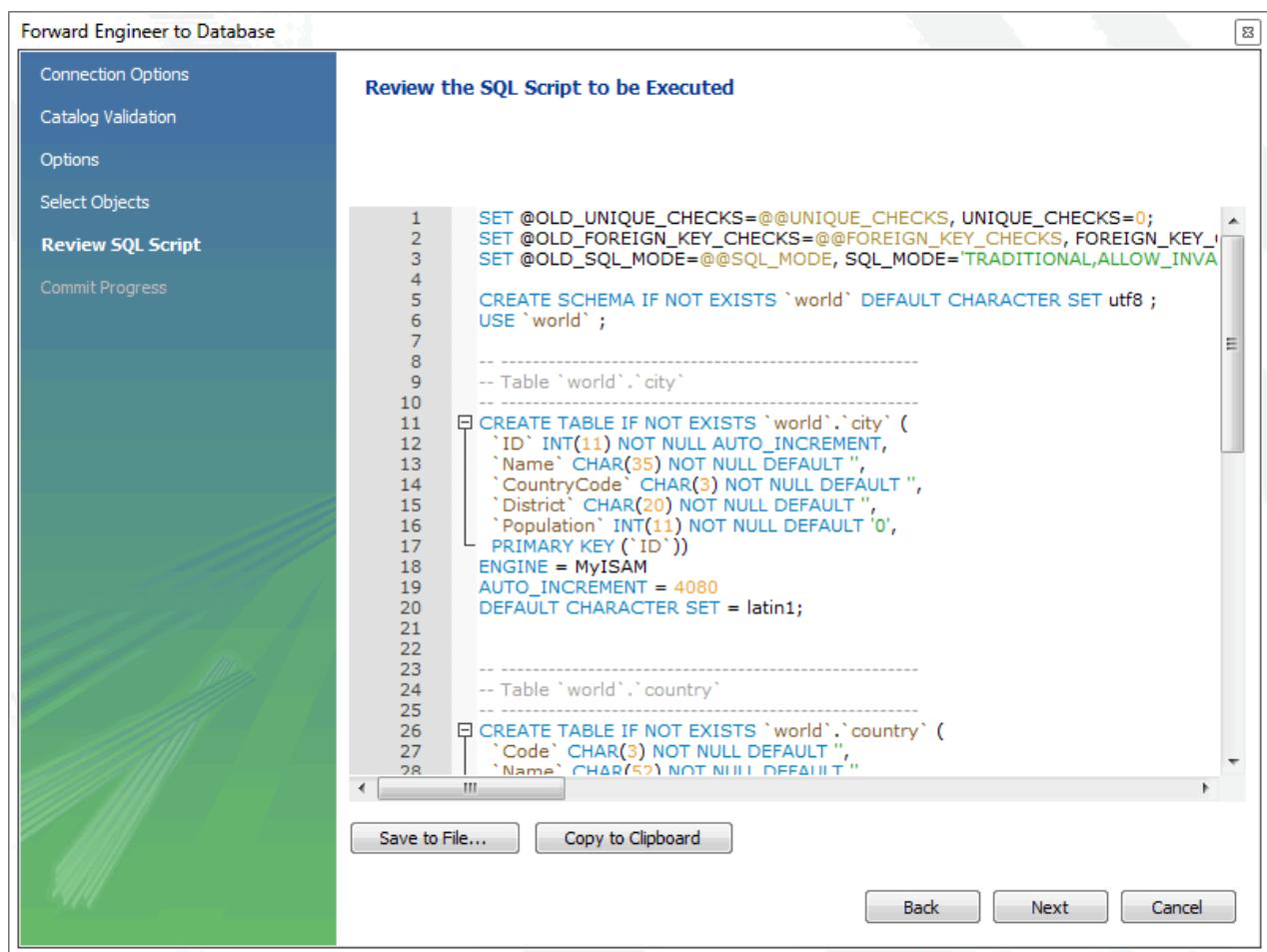
The next page enables you to select the objects to forward engineer.

Figure 9.34. Select Objects to Forward Engineer



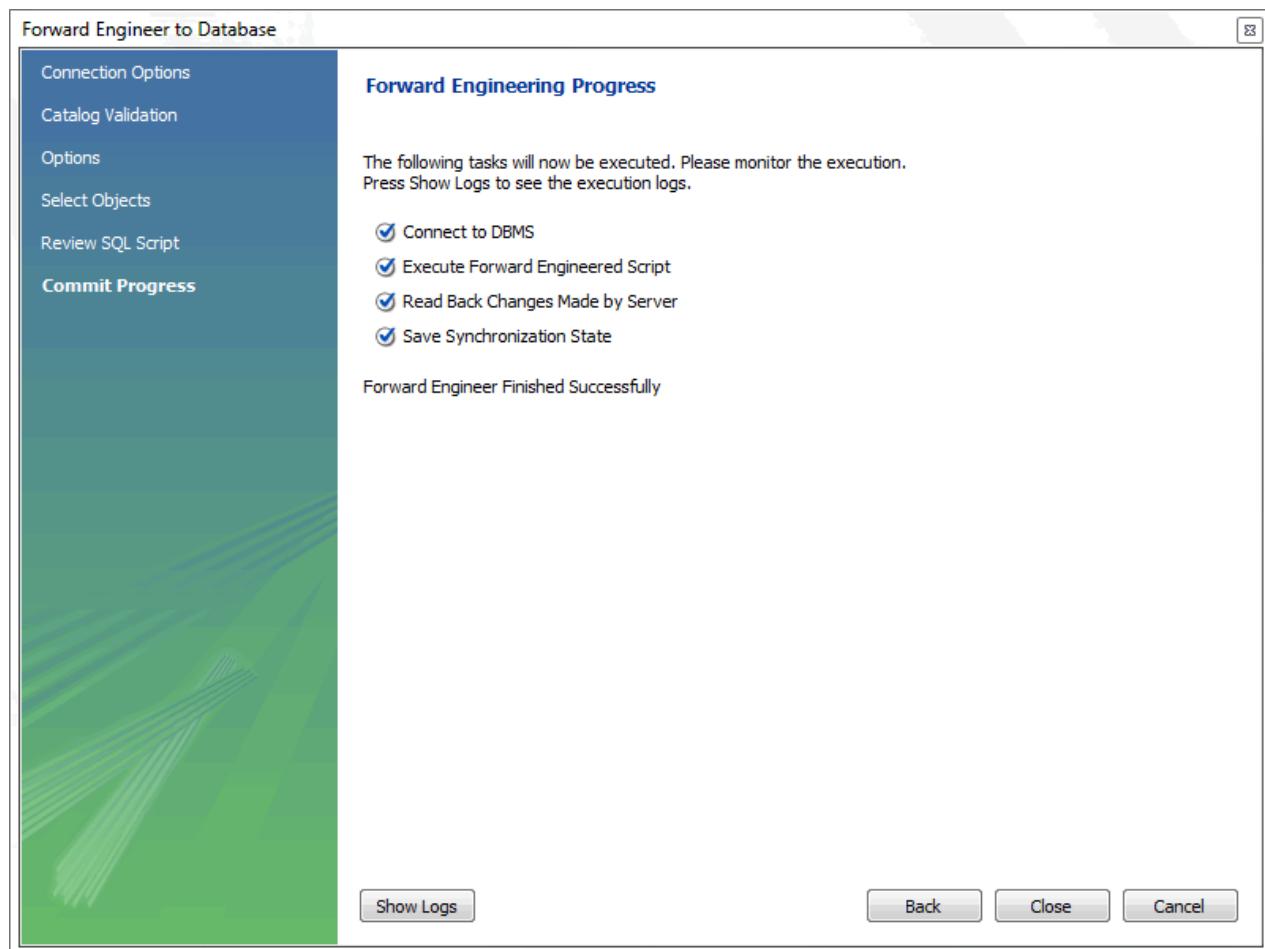
To select a subset of objects to forward engineer, use the Show Filter/Hide Filter button, then select specific objects. After you have selected your objects, click **Next** to continue.

On the **Review Script** page you may review and edit the SQL script that will be executed.

Figure 9.35. Review Script

Click Next to continue if you are satisfied with the generated script.

The next page of the wizard displays the results of the forward engineering process.

Figure 9.36. Forward Engineering Progress

You can confirm that the script created the schema by connecting to the target MySQL server and issuing a `SHOW DATABASES` statement.

9.3.10.3. Database Synchronization

It is possible to synchronize a model in MySQL Workbench with a live database. By default, the synchronization process will change the live database to be the same as the model, but this is configurable during the synchronization process.



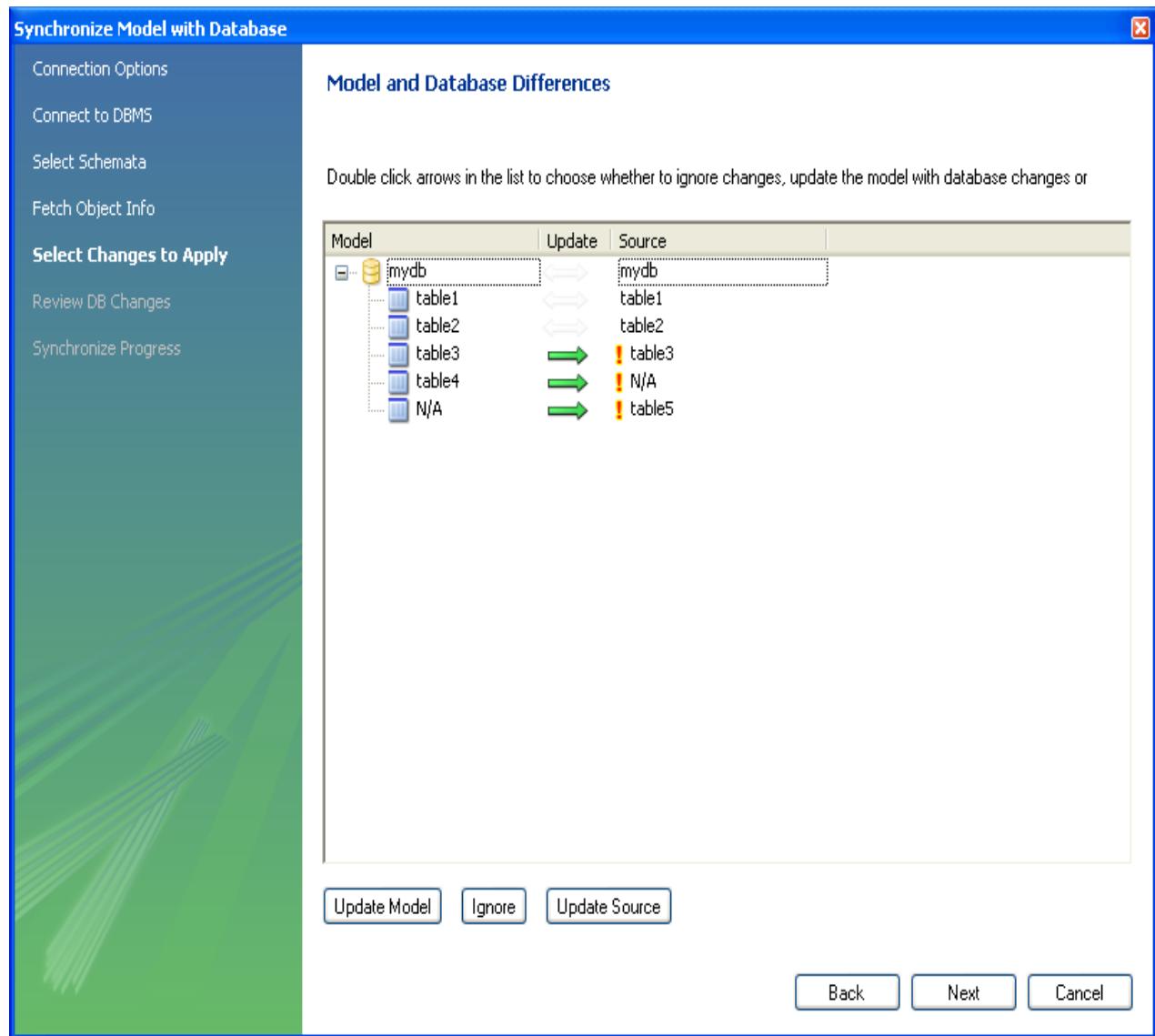
Caution

Because MySQL databases correspond to directories within the data directory, you must consider case sensitivity for database, table, and trigger names, which follow the case sensitivity rules of the underlying file system for your operating system. Synchronizing models with objects that differ in case may lead to MySQL Workbench producing a `DROP` statement for that object, before recreating it as lowercase. For more information, see [Identifier Case Sensitivity](#)

Workarounds include using a consistent convention, where the most portable code uses lower case database and table names. Or a temporary workaround is to delete the `DROP SCHEMA IF EXISTS` line from the generated query.

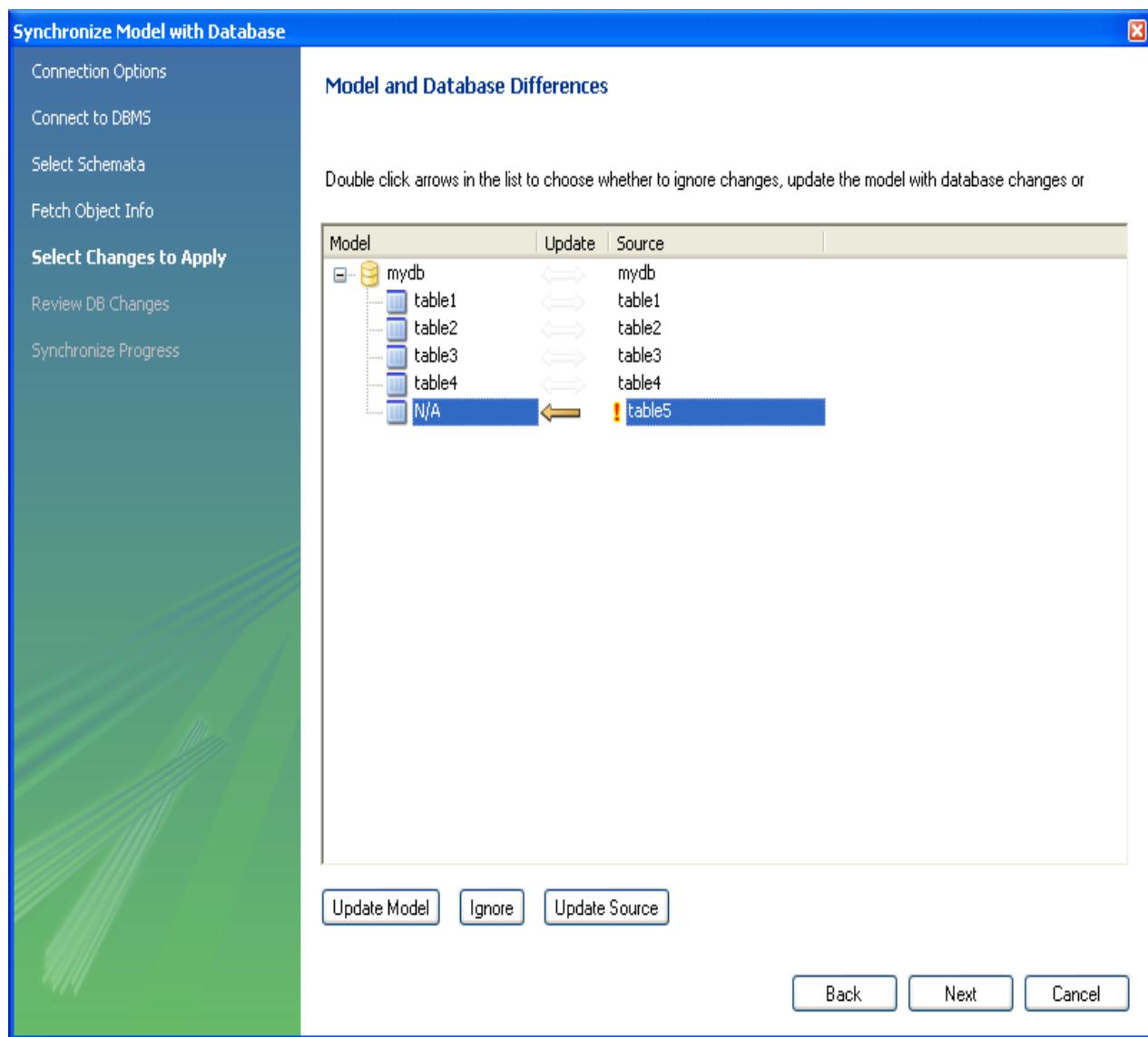
MySQL Workbench enables control over the direction of synchronization, and which objects to synchronize, in a completely flexible way. You can choose to synchronize only certain tables, enable synchronization to the live database only, enable synchronization from the live database to the model only, or a combination of directions. In effect you have complete control as to whether the synchronization is unidirectional or bidirectional, and which objects exactly are subject to synchronization. This is all controlled in the **Select Changes to Apply** page of the synchronization wizard.

Figure 9.37. Model and Database Differences



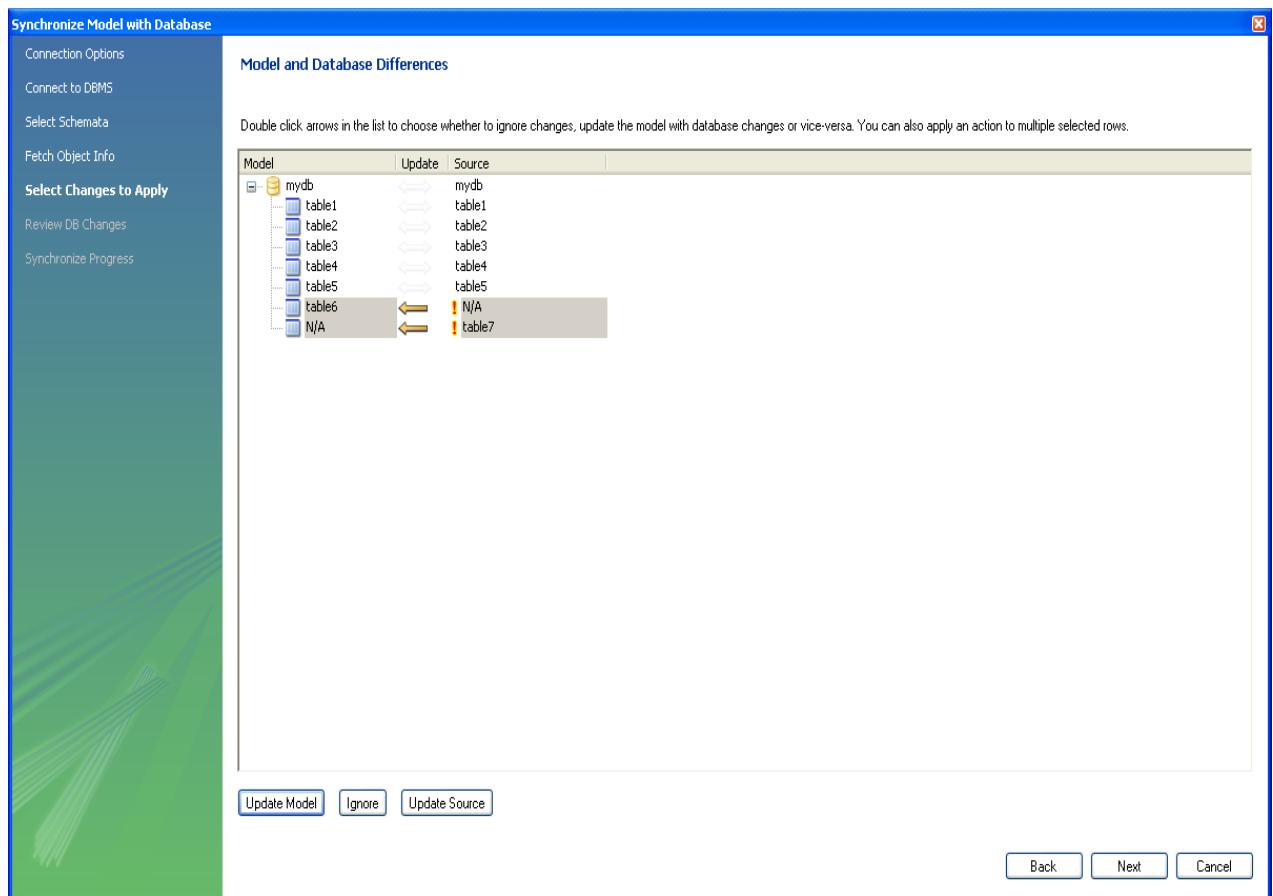
In the preceding example, the live database consists of `table1`, `table2` and `table3`. In MySQL Workbench an additional table, `table4`, has been created, along with a relationship between it and `table3`. Further, `table5` exists in the live database, but not in the model. The actions that are configured to occur would result in `table3` being altered (to include the relationship with `table4`), `table4` being created and `table5` being dropped, in the live database. It is possible to reconfigure this, though.

The next example shows how the direction of synchronization can be changed.

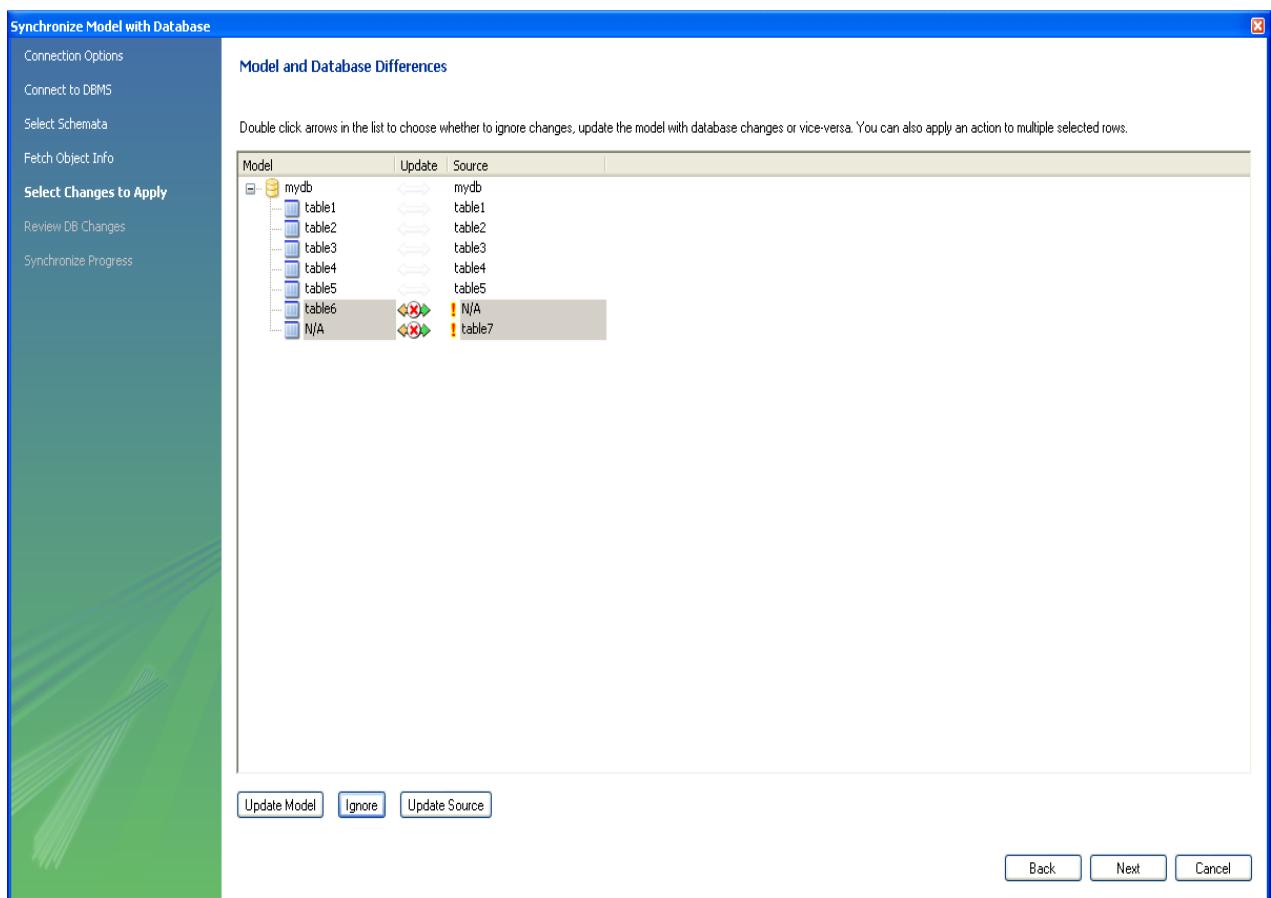
Figure 9.38. Controlling Synchronization Direction

In this case, the synchronization direction has been changed so that rather than the default action of `table5` being dropped from the live database, it will be incorporated into the MySQL Workbench model.

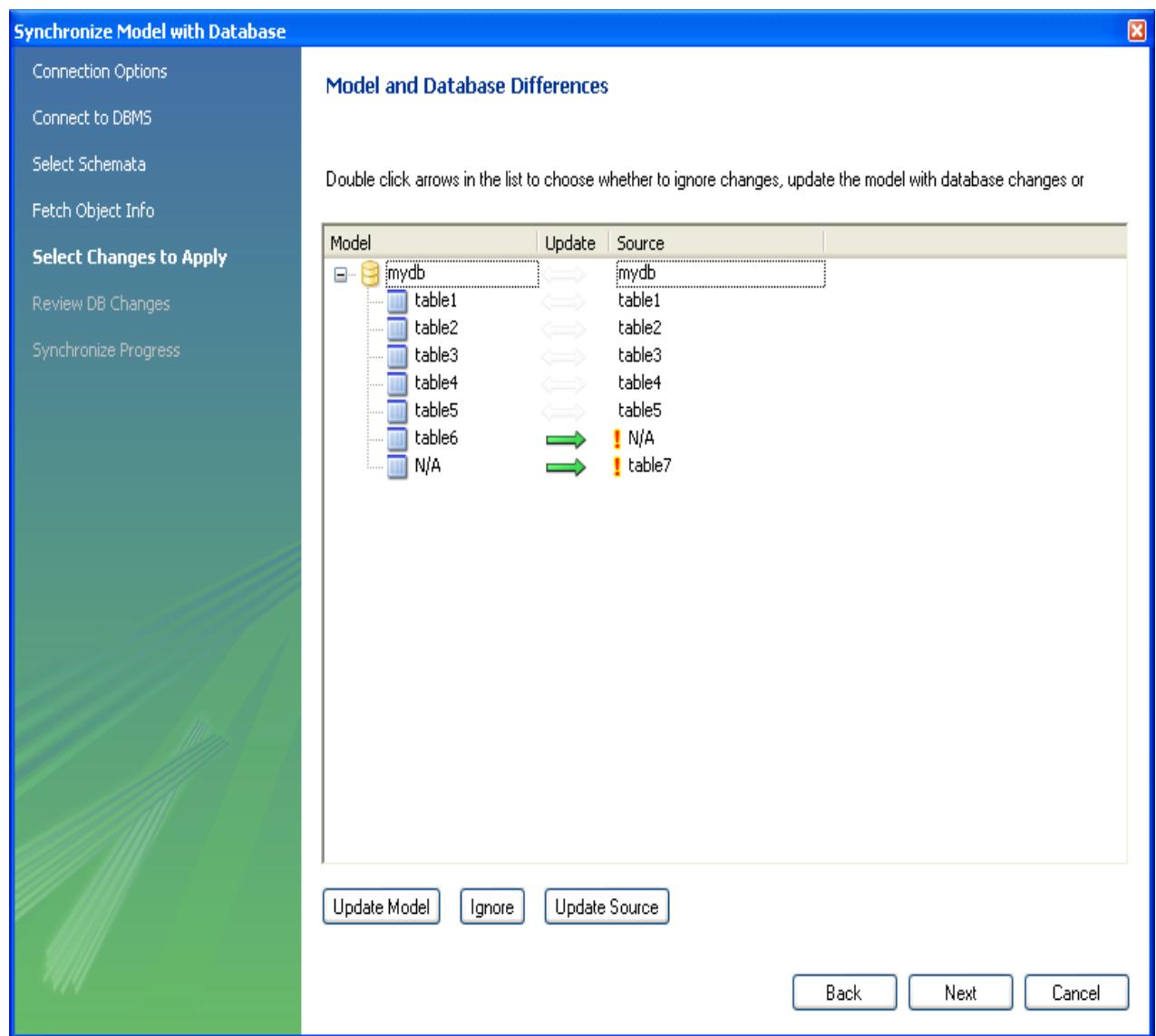
For convenience, the wizard provides three additional buttons to enable synchronization directions to be applied to a group of selected changes. The `Update Model` button causes the selected changes to be applied only to the model itself. In the following example, `table7` would be added to the model.

Figure 9.39. Update Model Button

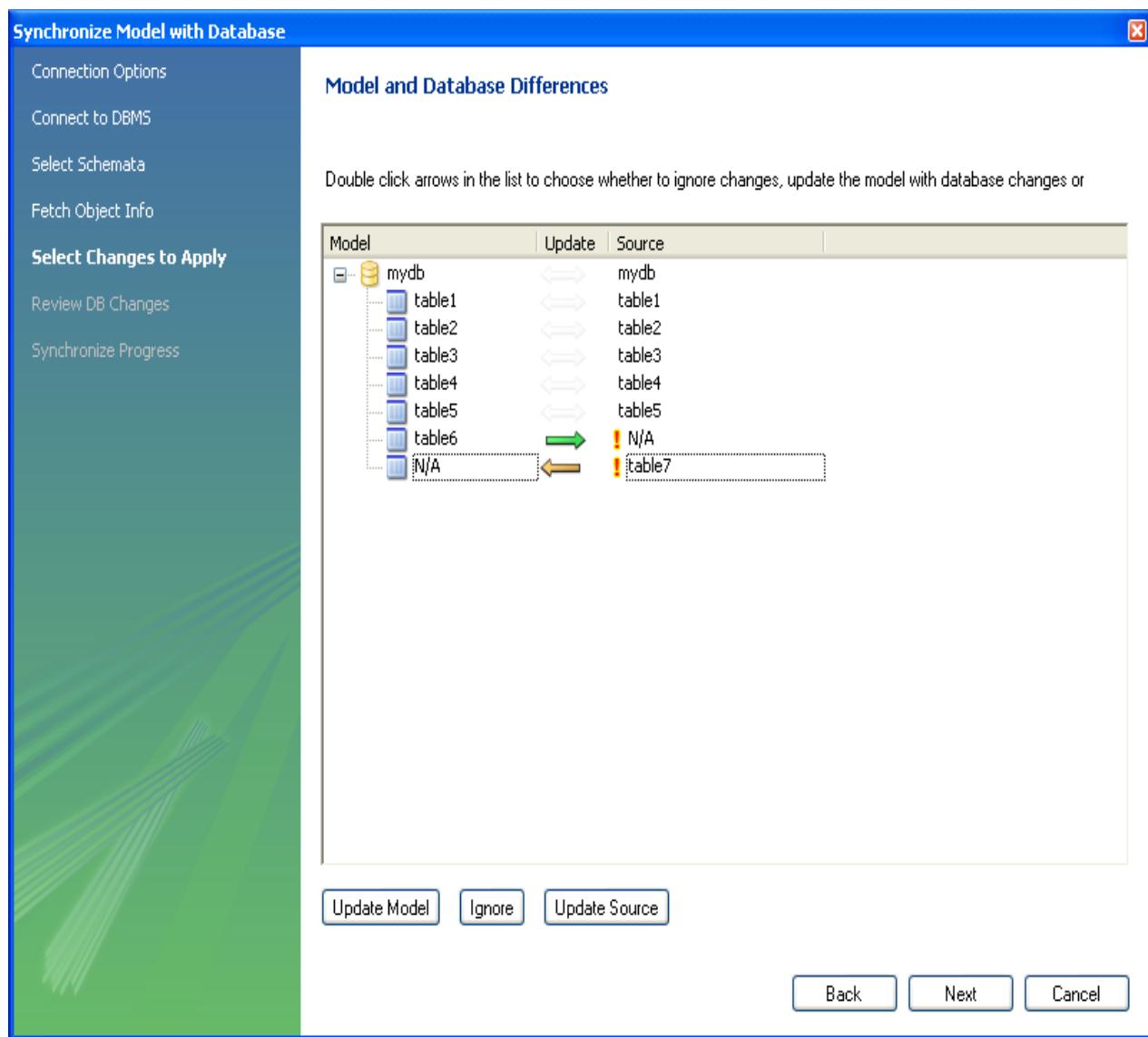
The Ignore button causes the selected changes to be ignored. No synchronization will take place for those changes. In the following example, no changes would take place.

Figure 9.40. Ignore Button

The Update Source button causes the selected changes to be applied only to the live database. In the following example, `table6` would be added to the live database and `table7` would be dropped from the live database.

Figure 9.41. Update Source Button

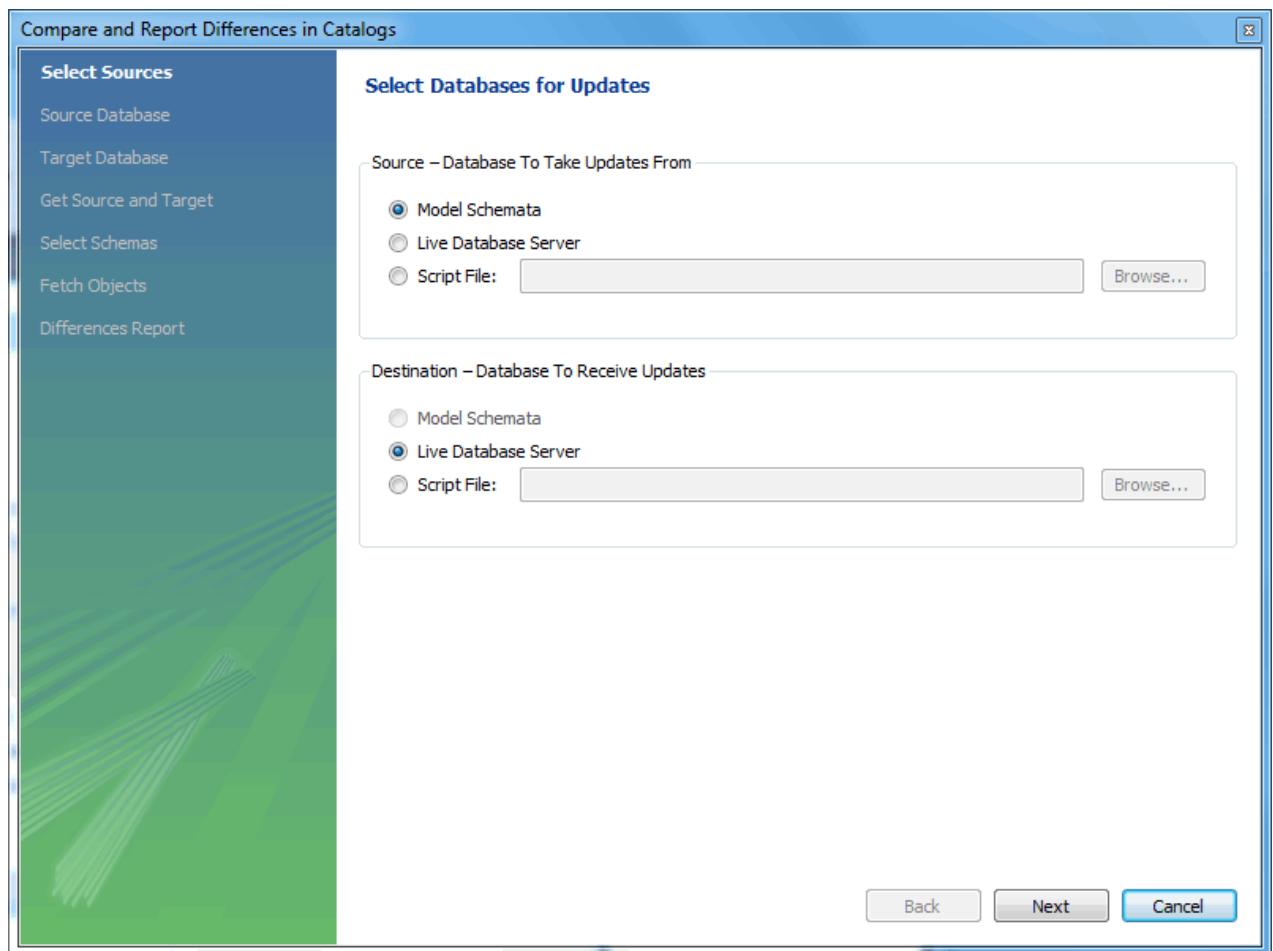
It is also possible to control individual changes by clicking the arrows. Clicking an arrow causes it to change between the three available synchronization directions: from model to source, from source to model, or bidirectionally. In the following example, `table6` will be created in the live database, and `table7` will be created in the model.

Figure 9.42. Click Arrows to Change Direction of Synchronization

9.3.10.4. Compare and Report Differences in Catalogs

This facility enables you to create a report detailing the differences between your MySQL Workbench model, and a live database or script. Choose Database, Compare Schemas from the main menu to run the **Compare and Report Differences in Catalogs** wizard.

The first step in the wizard is to specify which catalogs to compare. For example, you may wish to compare your live database against your current MySQL Workbench model.

Figure 9.43. Catalog Sources Selection

You then proceed through the wizard, providing connection information if accessing a live database. The wizard then produces a catalog diff report showing the differences between the compared catalogs.

Figure 9.44. Catalog Differences Report

```
1 +-----+
2 | Catalog Diff Report |
3 +-----+
4 Table `world`.`city` was modified
5   columns:
6     - modified column ID
7
8   attributes:
9     - default character set: latin1 --> latin1
10
11 Table `world`.`country` was modified
12   columns:
13     - modified column Region
14
15   attributes:
16     - default character set: latin1 --> latin1
17
18 -----
19
20 End of MySQL Workbench Report
```

9.4. Modeling Tutorials

This chapter contains three short tutorials intended to familiarize you with the basics of MySQL Workbench. These tutorials show how MySQL Workbench can be used both to design and to document databases.

Creating a database from scratch is the focus of [Section 9.4.2, “Using the Default Schema”](#) and exploring the graphic design capabilities of MySQL Workbench is touched upon in [Section 9.4.3, “Basic Modeling”](#). Both these tutorials show the database design capabilities of MySQL Workbench.

Importing an SQL data definition script is probably the quickest way to familiarize yourself with MySQL Workbench—this tutorial makes use of the `sakila` database and emphasizes the use of MySQL Workbench as a documentation tool. Examples taken from the `sakila` database are used throughout the documentation, so doing this tutorial can be very helpful in understanding MySQL Workbench.

9.4.1. Importing a Data Definition SQL Script

For this tutorial, use the `sakila` database script, which you can find by visiting the <http://dev.mysql.com/doc/> page, selecting the `Other Docs` tab, and looking in the `Example Databases` section

After downloading the file, extract it to a convenient location. Open MySQL Workbench and find the Reverse Engineer MySQL Create Script menu item by first choosing `File` and then `Import`. Find and

import the `sakila-schema.sql` file. This is the script that contains the data definition statements for the `sakila` database. The file filter for the file open dialog window defaults to `*.sql` so you should be able to view only files with the `sql` extension.

If the file was successfully imported, the application's status bar reads, `Import MySQL Create Script done`. To view the newly imported script, expand the `Physical Schemata` section by double-clicking the arrow on the left of the `Physical Schemata` title bar. Select the tab labeled `sakila`.

You may also wish to remove the default schema tab, `mydb`. Select this tab, then click the `-` button on the upper right in the `Physical Schemata` panel.

To view all the objects in the `sakila` schema, you may need to expand the `Physical Schemata` window. Move the mouse pointer anywhere over the gray area that defines the lower edge of the `Physical Schemata` window. Hold down the right mouse button and move the mouse to adjust the size of the window.

After you have expanded the window, all the objects in the `sakila` database should be visible. Tables appear at the top followed by views and then routines. There are no routine groups in this schema, but you should see the `Routine Groups` section and an `Add Group` icon.

For a complete description of importing a MySQL create script, see [Section 9.3.9.1, “Reverse Engineering Using a Create Script”](#).

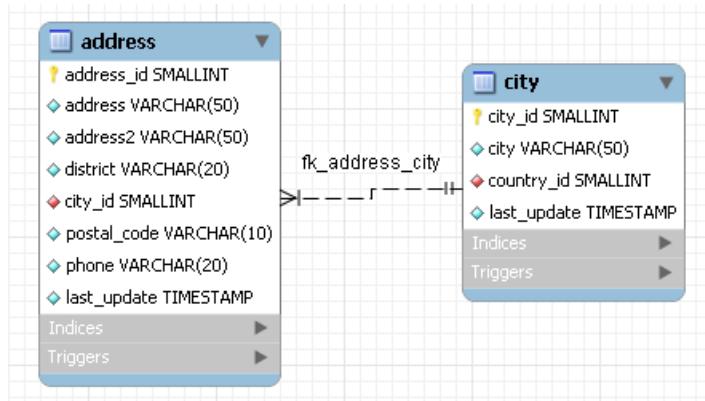
9.4.1.1. Adding an EER Diagram

To create an EER diagram for the `sakila` database, first add an EER diagram by double-clicking the `Add Diagram` icon in the `EER Diagrams` panel to create and open a new `EER Diagram` editor.

The `EER Diagram` canvas is where object modeling takes place. To add a table to the canvas, select the `Catalog` tab in the middle panel on the right side of the application to display any schemata that appear in the `MySQL Model` tab. Find the `sakila` schema and expand the view of its objects by clicking the `+` button to the left of the schema name. Expand the tables list in the same way.

You can add tables to the EER canvas by dragging them from the `Catalog` panel dropping them onto the canvas. Drop the `address` table and the `city` table onto the canvas.

Figure 9.45. Adding Tables to the Canvas



MySQL Workbench automatically discovers that `address.city_id` has been defined as a foreign key referencing the `city.city_id` field. Drop the `country` table onto the canvas and immediately you

should see the relationship between the `country` table and the `city` table. (To view all the relationships in the `sakila` database, see [Figure 9.48, “The sakila Database EER Diagram”](#).)

Click the **Properties** tab of the panel on the lower right, then click one of the tables on the canvas. This displays the properties of the table in the **Properties** window. While a table is selected, you can use the **Properties** window to change a table's properties. For example, entering `#FF0000` for the color value will change the color accent to red.

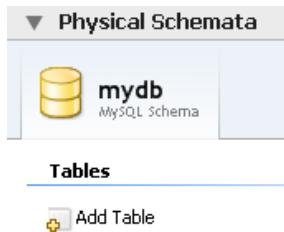
Changing the color of a table is a good way to identify a table quickly—something that becomes more important as the number of tables increases. Changing the color of a table is also an easy way to identify a table in the **Model Navigator** panel. This panel, the uppermost panel on the left side of the page, gives a bird's eye view of the entire EER canvas.

Save your changes to a [MySQL Workbench Models](#) file (`mwb` extension) by choosing Save from the **File** menu or by using the keyboard command **Control+S**.

9.4.2. Using the Default Schema

When you first open MySQL Workbench a default schema, `mydb` appears as the leftmost tab of the **Physical Schemata** section of MySQL Workbench. You can begin designing a database by using this default schema.

Figure 9.46. The Default Schema



To change the name of the default schema, double-click the schema tab. This opens a schema editor window docked at the bottom of the application. To undock or redock this window, double-click anywhere in the editor title bar.

To rename the schema, use the field labeled **Name**. After you have renamed the schema, a lightning bolt icon appears right aligned in the **Name** field, indicating that other changes are pending. Click the **Comments** field and a dialog box opens asking if you wish to rename all schema occurrences. Clicking **Yes** ensures that your changes are propagated throughout the application. Add comments to the database and change the collation if you wish. Close the schema editor by clicking the x button.

9.4.2.1. Creating a New Table

Create a new table by double-clicking the **Add Table** icon in the **Physical Schemata** panel. This opens the table editor docked at the bottom of the application. If you wish, you can undock or dock this editor in exactly the same way as the schema editor window.

Use the first tab of the table editor to change the name, collation, and engine. You may also add a comment.

Add columns to the new table by selecting the **Columns** tab. Use the default column name or enter a new name of your choosing. Use the **Tab** key to move to the next column and set the column's data type.

Altering the table by adding indexes or other features is also easily done using the table editor.

9.4.2.2. Creating Other Schema Objects

Additional objects such as views or routines can be added in the same way as tables.

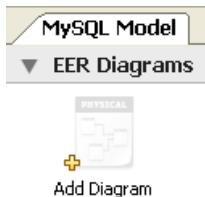
Any objects you have created can be found in the **Catalog** palette on the right. To view these schema objects, select the **Catalog** tab in the middle palette on the right. View all the objects by clicking the + button to the left of the schema name.

Save your changes to a **MySQL Workbench Models** file (**mwb** extension) by choosing Save from the **File** menu or by using the keyboard command **Control+S**.

9.4.3. Basic Modeling

On the **MySQL Model** page, double-click the **Add Diagram** icon. This creates and opens a new **EER Diagram** canvas.

Figure 9.47. Adding an EER Diagram



From an EER diagram page you can graphically design a database.

9.4.3.1. Adding a Table

The tools in the vertical toolbar on the left of the **EER Diagram** tab are used for designing an EER diagram. Start by creating a table using the table tool. The table tool is the rectangular grid in the middle of the vertical toolbar. Mousing over it shows the message, **Place a New Table (T)**.

Clicking this tool changes the mouse pointer to a hand with a rectangular grid. Create a table on the canvas by clicking anywhere on the **EER Diagram** grid.

Right-click the table and choose **Edit in New Window** from the pop-up menu. This opens the table editor, docked at the bottom of the application.

The table name defaults to **table1**. Change the name by entering **invoice** into the **Name:** field. Changes here affect the name of the tab in the table editor and the name of the table on the canvas.

Pressing **Tab** or **Enter** while the cursor is in the table name field selects the **Columns** tab of the table editor and creates a default column named **idinvoice**.

Pressing **Tab** or **Enter** again sets the focus on the **Datatype** list with **INT** selected. Notice that a field has been added to the table on the EER canvas.

Pressing **Tab** yet again and the focus shifts to adding a second column. Add a **Description** and a **Customer_id** column. When you are finished, close the table editor, by clicking the x button on the top left of the table editor.

9.4.3.2. Creating a Foreign Key

Select the table tool again and place another table on the canvas. Name this table `invoice_item`. Next click the `1:n Non-Identifying Relationship` tool.

First, click the `invoice_item` table; notice that a red border indicates that this table is selected. Next, click the `invoice` table. This creates a foreign key in the `invoice_item` table, the table on the “many” side of the relationship. This relationship between the two tables is shown graphically in crow’s foot notation.

Revert to the default mouse pointer by clicking the arrow at the top of the vertical toolbar. Click on the `invoice_item` table and select the **Foreign keys** tab.

Click the **Foreign key Name** field. The referenced table should show in the **Referenced Table** column and the appropriate column in the **Referenced Column** column.

To delete the relationship between two tables, click the line joining the tables and then press **Control +Delete**.

Experiment with the other tools on the vertical toolbar. Delete a relationship by selecting the eraser tool and clicking the line joining two tables. Create a view, add a text object, or add a layer.

Save your changes to a `MySQL Workbench Models` file (`mwb` extension) by choosing Save from the `File` menu or by using the keyboard command **Control+S**.

9.4.4. Documenting the `sakila` Database

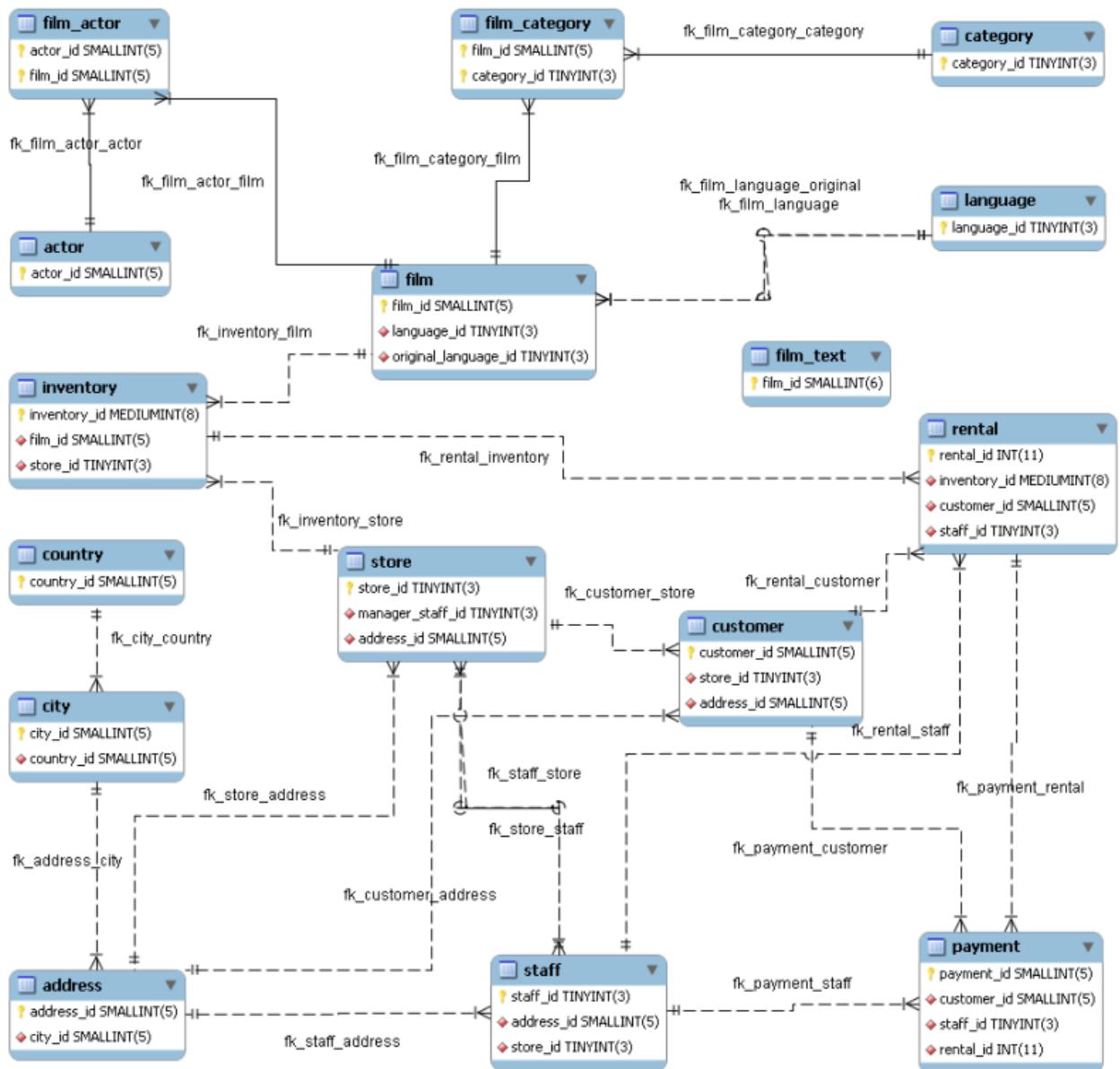
This chapter highlights the capabilities of MySQL Workbench as a documentation tool using the `sakila` database as an example. This is a sample database provided by MySQL that you can find by visiting the <http://dev.mysql.com/doc/> page, selecting the `Other Docs` tab, and looking in the `Example Databases` section.

An EER diagram is an invaluable aid to a quick understanding of any database. There is no need to read through table definition statements; glancing at an EER diagram can immediately indicate that various tables are related.

You can also see how tables are related; what the foreign keys are and what the nature of the relationship is.

9.4.4.1. A PNG File of the `sakila` Database

Find following an EER diagram showing all the tables in the `sakila` database. This image was created using the `File`, `Export`, `Export as PNG...` menu item.

Figure 9.48. The `sakila` Database EER Diagram

The object notation style used in [Figure 9.48, “The `sakila` Database EER Diagram”](#) is [Workbench \(PKs only\)](#). This notation shows only primary keys and no other columns, which is especially useful where space is at a premium. The relationship notation is the default, Crow’s Foot.

As the connection lines show, each table is related to at least one other table in the database (with the exception of the `film_text` table). Some tables have two foreign keys that relate to the same table. For example the `film` table has two foreign keys that relate to the `language` table, namely `fk_film_language_original` and `fk_film_language`. Where more than one relationship exists between two tables, the connection lines run concurrently.

Identifying and nonidentifying relationships are indicated by solid and broken lines respectively. For example, the foreign key `category_id` is part of the primary key in the `film_category` table so its relationship to the `category` table is drawn with a solid line. On the other hand, in the `city` table, the foreign key, `country_id`, is not part of the primary key so the connection uses a broken line.

9.5. Printing

The printing options used to create printouts of your EER Diagrams are found under the [File](#) menu. To create *documentation* of your models, see [Section 9.1.1.5.1, “The DBDoc Model Reporting Dialog Window \(Commercial Version\)”](#).

9.5.1. Printing Options

The printing menu items not enabled unless an EER Diagram is active. These items are available:

- [Page Setup...](#)

Enables you to choose the paper size, orientation, and margins.

- [Print](#)

Sends your EER Diagram directly to the printer. This option generates a preview before printing. From the preview you can adjust the scale of the view and also choose a multi-page view. Clicking the printer icon at the top left of this window, prints the currently selected EER Diagram. Close the print preview window if you need to adjust the placement of objects on the EER Diagram canvas.

- [Print to PDF...](#)

Creates a PDF file of your EER Diagram.

- [Print to PS...](#)

Creates a PostScript file of your EER Diagram.

9.6. MySQL Workbench Schema Validation Plugins (Commercial Version)

MySQL Workbench provides validation modules so that you can test your models before implementing them.

The validation plugins are accessed from the [Model](#) menu. One plugin performs general validation for any Relational Database Management System (RDMS) and the other is MySQL specific. Beneath these menu items are a number of specific validation tests. Running any one of these tests opens an output window docked at the bottom of the application. Warning messages are displayed on the left side of this window and the tests performed are displayed on the right.

The following sections outline the tasks performed by the validation modules.

9.6.1. General Validation

The following list names the general validation types and gives examples of specific violations:

- **Empty Content Validation**

- A table with no columns
- A routine or view with no SQL code defined
- A routine group containing no routines

- A table, view, or routine not referenced by at least one role
 - A user with no privileges
 - Objects such as tables that do not appear on at least one EER Diagram
- **Table Efficiency Validation**
 - A table with no primary key
 - A primary key that does not use an integer-based data type
 - A foreign key that refers to a column with a different data type
 - **Duplicated Identifiers Validation**
 - Duplicate object names
 - Duplicate role or user names
 - Duplicate index or routine names
 - **Consistency Validation**
 - Use of the same column with columns of differing data types
 - **Logic Validation**
 - A foreign key that refers to a column other than the primary key in the source table
 - Any object that is object is either read only or write only by role definition
 - Placeholder objects left over from reverse engineering

9.6.2. MySQL-Specific Validation

The following list names the MySQL-specific validation types and gives examples of specific violations:

- **Integrity Violation**
 - An object name longer than the maximum permitted
 - A foreign key defined for an engine type that does not support foreign keys (not yet implemented)
 - A view or routine that references a nonexistent table (not yet implemented)
 - A default value that does not match a column's data type
 - An invalid partitioning scheme
- **Syntax Violation**
 - A routine, trigger, or view with incorrect SQL syntax
 - A reserved keyword used as an identifier
 - Use of an invalid character

9.7. The DBDoc Model Reporting Dialog Window (Commercial Version)

This dialog window is found by navigating to the Model menu and choosing the DBDoc - Model Reporting... item.

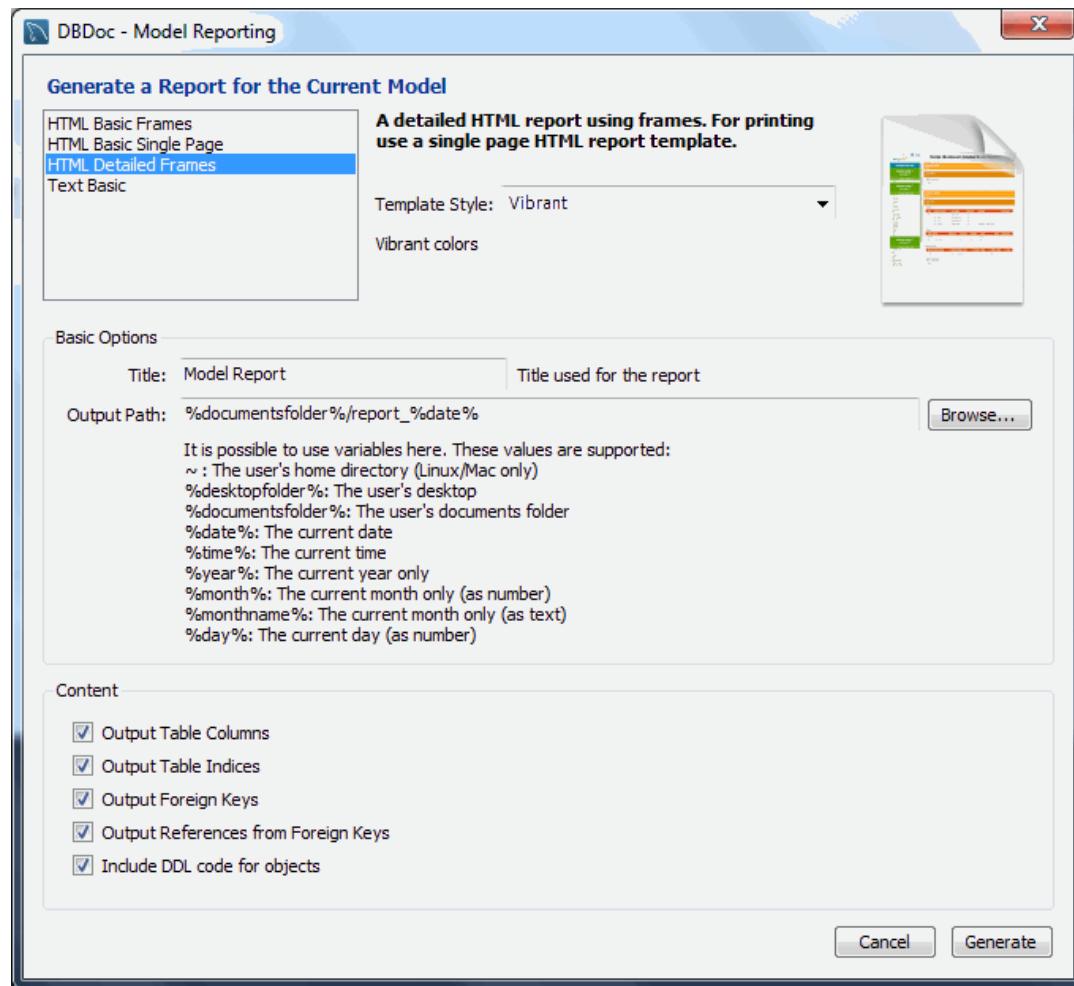


Note

The DBDoc - Model Reporting... item is not available in the MySQL Workbench Community version.

Use this dialog window to set the options for creating documentation of your database models.

Figure 9.49. The DBDoc Model Reporting Main Wizard



You can choose from four available templates:

- **HTML Basic Frames:** Model documentation in HTML format that makes use of frames
- **HTML Basic Single Page:** Single Page HTML documentation, not using frames
- **HTML Detailed Frames:** Detailed HTML documentation, using frames

- **Text Basic**: Text file documentation

When you click a template, a preview image displays on the right side of the page. For the **HTML Basic Frames** template, you can select either the **Colorful** or the **Restrained Colors** option from the **Style** list. The **HTML Basic Single Page** template offers only the **Colorful** style. The **HTML Detailed Frames** template offers the **Vibrant** style, and also the more subdued **Coated** style. The **Text Basic** template offers only the **Fixed Size Font** style.

From the **Base Options** frame choose the report title and the output directory for the report files.

The following variables may be used to configure the output path:

- **~**: The user's home directory. Available on Linux and Mac OS X versions only.
- **%desktopfolder%**: The user's desktop.
- **%documentsfolder%**: The user's Documents folders. The following table shows typical values for various platforms.

Platform	Typical Default Documents Folder
Windows	C:\Documents and Settings \user_name\My Documents
Linux	~/Documents
Mac OS X	Users/user_name/Documents

- **%date%**: The date in the format YYYY-MM-DD.
- **%time%**: The time in the format HHMM.
- **%year%**: The year in the format YYYY.
- **%month%**: The month in the format MM. January is 01 and December is 12.
- **%monthname%**: The name of the month, rather than the number.
- **%day%**: The day number in the format DD. For example, the 12th would be 12.

Content options can also be set:

- **Render Table Columns**: Display all the columns.
- **Render Table Indices**: Display all the indexes.
- **Render Foreign Keys**: Display all the foreign keys.
- **List Foreign Keys that refer to that table**: Display the tables that foreign keys reference.
- **Include DDL code for objects**: Generates DDL code.

Clicking the **Generate** button creates the directory defined in the **Output directory** text box. If you chose to create **HTML Basic Frames**, you will find the following files in this directory:

- **basic.css**: The style sheet for the **overview.html** page.
- **index.html**: The main page.
- **overview.html**: The model overview, the navigation links shown in the sidebar.

- `restrained.css`: The CSS file used if the `Restrained Colors` style option was chosen.
- `table_details.html`: The main frame of the model report.

Choosing the `HTML Basic Single Page` option creates a style sheet and an `index.html` file.

Choosing the `HTML Detailed Frames` option creates the following files:

- `basic.css`: The style sheet for the `overview.html` page. This is used if the `vibrant` style is chosen.
- `coated.css`: The CSS file used if the `Coated` style option was chosen.
- `index.html`: The main page.
- `overview.html`: Overview information for the report such as report title, project name and author.
- `overview_list.html`: A summary of schema in the model along with a list of tables contained in each schema.
- `routine_details.html`: List of all routines for the schema.
- `table_details.html`: The main report details.
- `table_details_list.html`: A Schema overview along with details of columns, indexes and foreign keys for each schema.
- `table_element_details.html`: The details for every element of the table.
- `top.html`: The top frame of the report.
- `view_details.html`: List of all columns and indexes for the schema.

Choosing the `Text Basic` option creates a directory containing one text file.

You can click `index.html` to view a report. The following screenshot shows the `HTML Detailed Frames` report being displayed:

Figure 9.50. The DBDoc Model Report

The screenshot shows the MySQL Model Report interface in Mozilla Firefox. The title bar says "Model Report - Mozilla Firefox". The address bar shows "file:///C:/Users/philip/Documents/report_2013-08-08/index.html". The main content area is titled "MySQL Model Report". On the left, there's a sidebar with "Schema Overview" and a list of schema elements: "Schema world" (with a "Back to overview" link), "Columns (24)", and "Indices (3)". Below these are "city.PRIMARY", "country.PRIMARY", and "countrylanguage.PRIMARY". There are also sections for "Foreign Keys ()" and "Triggers (0)". The main panel is titled "Schema world (1/1)" and contains a "DDL script" box with the code: "CREATE SCHEMA IF NOT EXISTS 'world' DEFAULT CHARACTER SET utf8". Below this is a section titled "Table city (1/3)" which includes a "Table Properties" table and a "Columns" table. The "Table Properties" table has columns for Average Row Length, Connection String, Default Collation, Minimal Row Count, Union Tables, Pack Keys, Data Directory, and Engine (MyISAM). The "Columns" table has columns for Key, Column Name, Datatype, Not Null, Default, and Comment. The "Indices" section at the bottom shows an empty table with columns for Index Name, Columns, Primary, Unique, Type, Kind, and Comment.

If you wish to create custom templates please refer to [Section 9.8, “Customizing DBDoc Model Reporting Templates”](#).

9.8. Customizing DBDoc Model Reporting Templates

This section provides an overview of creating and modifying DBDoc Model Reporting templates, as used by MySQL Workbench.

The MySQL Workbench DBDoc Model Reporting system is based on the [Google Template System](#). This discussion does not attempt to explain the Google Template System in detail. For a useful overview of how the Google Template System works, see the Google document, [How To Use the Google Template System](#).

The templates employed by the DBDoc Model Reporting system are text files that contain *Markers*. These text files are processed by the template system built into MySQL Workbench, and the markers replaced by

actual data. The output files are then generated. It is these output files, typically HTML or text, that are then viewed by the user.

Markers can be of six types:

- Template Include
- Comment
- Set delimiter
- Pragma
- Variable
- Section start and Section end

The last two are the most commonly used in MySQL Workbench templates and these important markers are briefly described in the following sections.

- **Variables**

The use of variables in the templates is straightforward. Any variables denoted by markers in the template file are replaced by their corresponding data prior to the output file being generated. The mapping between variables and their corresponding data is stored by MySQL Workbench in what is known as a data dictionary. In the data dictionary, the variable name is the *key* and the variable's corresponding data is the *value*. The data dictionaries are built by MySQL Workbench and filled with the data contained in the model being processed.

By way of example, the following code snippet shows part of a template file:

```
Total number of Schemata: {{SCHEMA_COUNT}}
```

In the generated output file, the variable `{{SCHEMA_COUNT}}` is replaced by the number of schemata in the model:

```
Total number of Schemata: 2
```

A variable can appear as many times as required in the template file.

- **Sections**

Sections are used to perform iteration in the templates. When MySQL Workbench exchanges the variables in a section for data, it does so iteratively, using all data in the data dictionary in which the variable is defined. MySQL Workbench builds the data dictionaries according to the model currently being processed.

Consider the following code snippet:

```
 {{#SCHEMATA}}
Schema: {{SCHEMA_NAME}}
{{/SCHEMATA}}
```

In the preceding snippet, the section start and end are indicated by the `{{#SCHEMATA}}` and `{{/SCHEMATA}}` markers. When MySQL Workbench processes the template, it notes the section and iterates it until the variable data for `{{SCHEMA_NAME}}` in the corresponding data dictionary is

exhausted. For example, if the model being processed contains two schemata, the output for the section might resemble the following:

```
Schema: Airlines
Schema: Airports
```

Data Dictionaries

It is important to understand the relationship between sections and data dictionaries in more detail. In a data dictionary the *key* for a variable is the variable name, a marker. The variable *value* is the variable's data. The entry for a section in a data dictionary is different. For a section entry in a data dictionary, the key is the section name, the marker. However, the value associated with the key is a list of data dictionaries. In MySQL Workbench each section is usually associated with a data dictionary. You can think of a section as *activating* its associated dictionary (or dictionaries).

When a template is processed, data dictionaries are loaded in a hierarchical pattern, forming a tree of data dictionaries. This is illustrated by the following table.

Table 9.1. Data Dictionaries Tree

Data Dictionary	Loads Data Dictionary
MAIN	SCHEMATA
SCHEMATA	TABLES, COLUMNS (Detailed is true), FOREIGN_KEYS (Detailed is true), INDICES (Detailed is true)
TABLES	REL_LISTING, INDICES_LISTING, COLUMNS_LISTING, TABLE_COMMENT_LISTING, DDL_LISTING
COLUMNS_LISTING	COLUMNS (Detailed is false)
REL_LISTING	REL (Detailed is false)
INDICES_LISTING	INDICES (Detailed is false)

The root of the tree is the *main* dictionary. Additional dictionaries are loaded from the root to form the dictionary tree.



Note

If a template has no sections, any variables used in the template are looked up in the main dictionary. If a variable is not found in the main dictionary (which can be thought of as associated with the default, or main, section), no data is generated in the output file for that marker.

Evaluation of variables

The tree structure of the data dictionaries is important with respect to variable evaluation. As variables are defined in data dictionaries, their associated values have meaning only when that particular data dictionary is active, and that means when the section associated with that data dictionary is active. When a variable lookup occurs, the system checks the data dictionary associated with the current section. If the variable value can be found there, the replacement is made. However, if the variable's value is not found in the current data dictionary, the parent data dictionary is checked for the variable's value, and so on up the tree until the main data dictionary, or root, is reached.

Suppose that we want to display the names of all columns in a model. Consider the following template as an attempt to achieve this:

```
Report
```

```
-----  
Column Name: {{COLUMN_NAME}}
```

This template produces no output, even for a model that contains many columns. In this example, the only data dictionary active is the main dictionary. However, `COLUMN_NAME` is stored in the `COLUMNS` data dictionary, which is associated with the `COLUMNS` section.

With this knowledge, the template can be improved as follows:

```
Report  
-----  
{#{COLUMNS}}  
Column Name: {{COLUMN_NAME}}  
{//COLUMNS}
```

This still does not produce output. To see why, see [Table 9.1, “Data Dictionaries Tree”](#). The `COLUMNS` data dictionary has the parent dictionary `COLUMNS_LISTING`. `COLUMNS_LISTING` has the parent `TABLES`, which has the parent `SCHEMATA`, whose parent is the main dictionary. Remember that for a dictionary to be involved in variable lookup, its associated section must currently be active.

To achieve the desired output, the template must be something like the following:

```
Report  
-----  
{#{SCHEMATA}}  
{#{TABLES}}  
{#{COLUMNS_LISTING}}  
{#{COLUMNS}}  
Column Name: {{COLUMN_NAME}}  
{//COLUMNS}  
{//COLUMNS_LISTING}  
{//TABLES}  
{//SCHEMATA}
```

The following template is the same, but with explanatory comments added:

```
Report  
-----  
{#! Main dictionary active}  
{#{SCHEMATA}} {#! SCHEMATA dictionary active}  
{#{TABLES}} {#! TABLES dictionary active}  
{#{COLUMNS_LISTING}} {#! COLUMNS_LISTING dictionary active}  
{#{COLUMNS}} {#! COLUMNS dictionary active}  
Column Name: {{COLUMN_NAME}} {#! COLUMN_NAME variable is looked-up,  
and found, in COLUMNS data dictionary}  
{//COLUMNS}  
{//COLUMNS_LISTING}  
{//TABLES}  
{//SCHEMATA}
```

Imagine now that for each column name displayed you also wanted to display its corresponding schema name, the template would look like this:

```
Report  
-----  
{#{SCHEMATA}}  
{#{TABLES}}  
{#{COLUMNS_LISTING}}
```

```

{{#COLUMNS}}
Schema Name: {{SCHEMA_NAME}} Column Name: {{COLUMN_NAME}}
{{/COLUMNS}}
{{/COLUMNS_LISTING}}
{{/TABLES}}
{{/SCHEMATA}}

```

When variable lookup is performed for `SCHEMA_NAME`, the `COLUMNS` dictionary is checked. As the variable is not found there the parent dictionary will be checked, `COLUMNS_LISTING`, and so on, until the variable is eventually found where it is held, in the `SCHEMATA` dictionary.

If there are multiple schemata in the model, the outer section is iterated over a matching number of times, and `SCHEMA_NAME` accordingly has the correct value on each iteration.

It's important to always consider which dictionary must be active (and which parents) for a variable to be evaluated correctly. The following section has a table that helps you identify section requirements.

9.8.1. Supported Template Markers

The following table shows the supported markers. These markers can be used in any template, including custom templates.

Using the table

The table shows which variables are defined in which sections. The variable should be used in its correct section or its value will not be displayed. If a variable `type` is a variable, then the table describes its data dictionary, and a parent dictionary if `type` is a section. Also remember that the data dictionaries used to perform variable lookups form a hierarchical tree, so it is possible to use a variable in a child section that is defined in a parent section.

Table 9.2. Supported Template Markers

Marker text	Type	Data Dictionary or Parent Dictionary	Corresponding data
TITLE	Variable	MAIN	Title of the report
GENERATED	Variable	MAIN	Date and time when the report was generated
STYLE_NAME	Variable	MAIN	The name of the style selected in MySQL Workbench, this is typically used to load the corresponding CSS file, depending on the name of the style selected in MySQL Workbench
SCHEMA_COUNT	Variable	MAIN	The number of schemata in the model
PROJECT_TITLE	Variable	MAIN	Project title as set for the model in Document Properties
PROJECT_NAME	Variable	MAIN	Project name as set for the model in Document Properties
PROJECT_AUTHOR	Variable	MAIN	Project author as set for the model in Document Properties
PROJECT_VERSION	Variable	MAIN	Project version as set for the model in Document Properties

Marker text	Type	Data Dictionary or Parent Dictionary	Corresponding data
PROJECT_DESCRIPTION	Variable	MAIN	Project description as set for the model in Document Properties
PROJECT_CREATED	Variable	MAIN	Automatically set for the model project, but as displayed in Document Properties
PROJECT_CHANGED	Variable	MAIN	Automatically set for the model project, but as displayed in Document Properties
TOTAL_TABLE_COUNT	Variable	MAIN	The number of tables in all schemata in the model
TOTAL_COLUMN_COUNT	Variable	MAIN	The number of columns in all tables in all schemata in the model
TOTAL_INDEX_COUNT	Variable	MAIN	The number of indexes in the model
TOTAL_FK_COUNT	Variable	MAIN	The number of foreign keys in the model
SCHEMATA	Section	MAIN	Used to mark the start and end of a SCHEMATA section; the SCHEMATA data dictionary becomes active in this section
SCHEMA_NAME	Variable	SCHEMATA	The schema name
SCHEMA_ID	Variable	SCHEMATA	The schema ID
TABLE_COUNT	Variable	SCHEMATA	The number of tables in the current schema
COLUMN_COUNT	Variable	SCHEMATA	The number of columns in the current schema
INDICES_COUNT	Variable	SCHEMATA	The number of indexes in the current schema
FOREIGN_KEYS_COUNT	Variable	SCHEMATA	The number of foreign keys in the current schema
TABLES	Section	SCHEMATA	Marks the start and end of a TABLES section; the TABLES data dictionary becomes active in this section
TABLE_NAME	Variable	TABLES	The table name
TABLE_ID	Variable	TABLES	The table ID
COLUMNS_LISTING	Section	TABLES	Marks the start and end of a COLUMNS_LISTING section; the COLUMNS_LISTING data dictionary becomes active in this section

Marker text	Type	Data Dictionary or Parent Dictionary	Corresponding data
COLUMNS	Section	COLUMNS_LISTING	Marks the start and end of a COLUMNS section; the COLUMNS data dictionary becomes active in this section
COLUMN_KEY	Variable	COLUMNS	Whether the column is a primary key
COLUMN_NAME	Variable	COLUMNS	The column name
COLUMN_DATATYPE	Variable	COLUMNS	The column data type
COLUMN_NOTNULL	Variable	COLUMNS	Whether the column permits NULL values
COLUMN_DEFAULTVALUE	Variable	COLUMNS	The column default value
COLUMN_COMMENT	Variable	COLUMNS	The column comment
COLUMN_ID	Variable	COLUMNS	The column ID
COLUMN_KEY_PART	Variable	COLUMNS (if detailed)	The column key type
COLUMN_NULLABLE	Variable	COLUMNS (if detailed)	Can the column contain NULL values
COLUMN_AUTO_INC	Variable	COLUMNS (if detailed)	Does the column auto-increment
COLUMN_CHARSET	Variable	COLUMNS (if detailed)	The column character set
COLUMN_COLLATION	Variable	COLUMNS (if detailed)	The column collation
COLUMN_IS_USERTYPE	Variable	COLUMNS (if detailed)	Whether the column is a user type
INDICES_LISTING	Section	TABLES	Marks the start and end of an INDICES_LISTING section; the INDICES_LISTING data dictionary becomes active in this section
INDICES	Section	INDICES_LISTING	Marks the start and end of an INDICES section; the INDICES data dictionary becomes active in this section
INDEX_NAME	Variable	INDICES	The index name
INDEX_PRIMARY	Variable	INDICES	Whether this is a primary key
INDEX_UNIQUE	Variable	INDICES	Whether this is a unique index
INDEX_TYPE	Variable	INDICES	The index type; for example, PRIMARY
INDEX_KIND	Variable	INDICES	The index kind
INDEX_COMMENT	Variable	INDICES	The index comment
INDEX_ID	Variable	INDICES	The index ID
INDEX_COLUMNS	Section	INDICES	Marks the start and end of an INDEX_COLUMNS section; the INDEX_COLUMNS data

Marker text	Type	Data Dictionary or Parent Dictionary	Corresponding data
			dictionary becomes active in this section
INDEX_COLUMN_NAME	Variable	INDEX_COLUMNS	The index column name
INDEX_COLUMN_ORDER	Variable	INDEX_COLUMNS	The index column order; for example, ascending, descending
INDEX_COLUMN_COMMENT	Variable	INDEX_COLUMNS	The index comment
INDEX_KEY_BLOCK_SIZE	Variable	INDEX_COLUMNS (if detailed)	The index key-block size
REL_LISTING	Section	TABLES	Marks the start and end of a REL_LISTING section; the REL_LISTING data dictionary becomes active in this section
REL	Section	REL_LISTING	Marks the start and end of a REL section; the REL data dictionary becomes active in this section
REL_NAME	Variable	REL, FOREIGN_KEYS	The relationship name
REL_TYPE	Variable	REL, FOREIGN_KEYS	The relationship type
REL_PARENTTABLE	Variable	REL, FOREIGN_KEYS	The relationship parent table
REL_CHILDTABLE	Variable	REL, FOREIGN_KEYS	The relationship child table
REL_CARD	Variable	REL, FOREIGN_KEYS	The relationship cardinality
FOREIGN_KEY_ID	Variable	REL	Foreign key ID
FOREIGN_KEYS	Section	SCHEMATA	Marks the start and end of a FOREIGN_KEYS section; the FOREIGN_KEYS data dictionary becomes active in this section
FK_DELETE_RULE	Variable	FOREIGN_KEYS	The foreign key delete rule
FK_UPDATE_RULE	Variable	FOREIGN_KEYS	The foreign key update rule
FK_MANDATORY	Variable	FOREIGN_KEYS	Whether the foreign key is mandatory
TABLE_COMMENT_LISTING	Section	TABLES	Marks the start and end of a TABLE_COMMENT_LISTING section; the TABLE_COMMENT_LISTING data dictionary becomes active in this section
TABLE_COMMENT	Variable	TABLE_COMMENT_LISTING	The table comment
DDL_LISTING	Section	TABLES	Marks the start and end of a DDL_LISTING section; the DDL_LISTING data dictionary becomes active in this section

Marker text	Type	Data Dictionary or Parent Dictionary	Corresponding data
DDL_SCRIPT	Variable	DDL_LISTING	Display the DDL script of the currently active entity; for example, SCHEMATA, TABLES

9.8.2. Creating a Custom Template

In the simplest case, a template consists of two files: a template file, which has a `.tpl` extension, and a special file `info.xml`. The `info.xml` file has important metadata about the template. A third file is optional, which is the preview image file. This preview file provides a thumbnail image illustrating the appearance of the generated report.

One of the easiest ways to create a custom template is to make a copy of any existing template.

For example, the following procedure describes how to make a custom template based on the `Text_Basic` template.

1. Navigate to the folder where the templates are stored. Assuming that MySQL Workbench has been installed into the default location on Windows, this would be `C:\Program Files\MySQL\MySQL Workbench 5.0 SE\modules\data\wb_model_reporting`.
2. Copy the `Text_Basic.tpl` folder. The copy can be given any suitable name; for example, `Custom_Basic.tpl`.
3. Edit the `info.xml` file to reflect your custom template. The unedited file in this case is shown here:

```
<?xml version="1.0"?>
<data>
  <value type="object" struct-name="workbench.model.reporting.TemplateInfo"
    id="{BD6879ED-814C-4CA3-A869-9864F83B88DF}" struct-checksum="0xb46b524d">
    <value type="string" key="description">
      A basic TEXT report listing schemata and objects.
    </value>
    <value type="string" key="name">HTML Basic Frame Report</value>
    <value type="list" content-type="object"
      content-struct-name="workbench.model.reporting.TemplateStyleInfo"
      key="styles">
      <value type="object" struct-name="workbench.model.reporting.TemplateStyleInfo"
        id="{7550655C-CD4B-4EB1-8FAB-AEEE49B2261E}" struct-checksum="0xab08451b">
        <value type="string" key="description">
          Designed to be viewed with a fixed sized font.
        </value>
        <value type="string" key="name">Fixed Size Font</value>
        <value type="string" key="previewImageFileName">
          preview_basic.png
        </value>
        <value type="string" key="styleTagValue">fixed</value>
      </value>
    </value>
    <value type="string" key="mainFileName">report.txt</value>
  </value>
</data>
```

The file defines two objects: the `TemplateInfo` object and the `TemplateStyleInfo` object. These objects contain information about the template that will be displayed in the DBDoc Model Reporting wizard main page.

4. Change the object GUIDs that are used in the file. In this example, there are two that need replacing:

```
id="{BD6879ED-814C-4CA3-A869-9864F83B88DF}"
...
id="{7550655C-CD4B-4EB1-8FAB-AAEE49B2261E}"
```

Generate two new GUIDS. This can be done using any suitable command-line tool. There are also free online tools that can be used to generate GUIDs. Another way to generate GUIDs is by using the MySQL [UUID\(\)](#) function:

```
mysql> SELECT UUID();
+-----+
| UUID() |
+-----+
| 648f4240-7d7a-11e0-870b-89c43de3bd0a |
+-----+
```

Once you have the new GUID values, edit the `info.xml` file accordingly.

5. Edit the textual information for the `TemplateInfo` and `TemplateStyleInfo` objects to reflect the purpose of the custom template.
6. The modified file will now look something like the following:

```
<?xml version="1.0"?>
<data>
  <value type="object" struct-name="workbench.model.reporting.TemplateInfo"
    id="{cac9ba3f-ee2a-49f0-b5f6-32580fab1640}" struct-checksum="0xb46b524d">
    <value type="string"
      key="description">
      Custom basic TEXT report listing schemata and objects.
    </value>
    <value type="string" key="name">Custom Basic text report</value>
    <value type="list" content-type="object"
      content-struct-name="workbench.model.reporting.TemplateStyleInfo" key="styles">
      <value type="object"
        struct-name="workbench.model.reporting.TemplateStyleInfo"
        id="{39e3b767-a832-4016-8753-b4cb93aa2dd6}" struct-checksum="0xab08451b">
        <value type="string" key="description">
          Designed to be viewed with a fixed sized font.
        </value>
        <value type="string" key="name">Fixed Size Font</value>
        <value type="string" key="previewImageFileName">preview_basic.png</value>
        <value type="string" key="styleTagValue">fixed</value>
      </value>
    </value>
    <value type="string" key="mainFileName">custom_report.txt</value>
  </value>
</data>
```

7. Create the new template file. This too may best be achieved, depending on your requirements, by editing an existing template. In this example the template file `report.txt.tpl` is shown here:

```
+-----+
| MySQL Workbench Report           |
+-----+
Total number of Schemata: {{SCHEMA_COUNT}}
=====
{{#SCHEMATA}}
{{SCHEMA_NR}}. Schema: {{SCHEMA_NAME}}
```

```

## Tables ({{TABLE_COUNT}}) ##
{{#TABLES}}{{{TABLE_NR_FMT}}}. Table: {{TABLE_NAME}}
{{#COLUMNS_LISTING}}## Columns ##
Key Column Name Datatype Not Null Default Comment
{{#COLUMNS}}{{{COLUMN_KEY}}}{{{COLUMN_NAME}}}{{{COLUMN_DATATYPE}}} »
{{{COLUMN_NONNULL}}}{{{COLUMN_DEFAULTVALUE}}}{{{COLUMN_COMMENT}}}
{{/COLUMNS}}{{/COLUMNS_LISTING}}
{{#INDICES_LISTING}}## Indices ##
Index Name Columns Primary Unique Type Kind Comment
{{#INDICES}}{{{INDEX_NAME}}}{{{#INDICES_COLUMNS}}}{{{INDEX_COLUMN_NAME}}} »
{{{INDEX_COLUMN_ORDER}}}{{{INDEX_COLUMN_COMMENT}}}{{{/INDICES_COLUMNS}}} »
{{{INDEX_PRIMARY}}}{{{INDEX_UNIQUE}}}{{{INDEX_TYPE}}}{{{INDEX_KIND}}}{{{INDEX_COMMENT}}}
{{/INDICES}}{{/INDICES_LISTING}}
{{#REL_LISTING}}## Relationships ##
Relationship Name Relationship Type Parent Table Child Table Cardinality
{{#REL}}{{{REL_NAME}}}{{{REL_TYPE}}}{{{REL_PARENTTABLE}}}{{{REL_CHILDTABLE}}}{{{REL_CARD}}}
{{/REL}}{{/REL_LISTING}}
-----
{{/TABLES}}
{{/SCHEMATA}}
=====
End of MySQL Workbench Report

```

This template shows details for all schemata in the model.

- The preceding template file can be edited in any way you like, with new markers being added, and existing markers being removed as required. For the custom template example, you might want to create a much simpler template, such as the one following:

```

+-----+
| MySQL Workbench Custom Report          |
+-----+
Total number of Schemata: {{SCHEMA_COUNT}}
=====
{{#SCHEMATA}}
Schema Name: {{SCHEMA_NAME}}
-----
## Tables ({{TABLE_COUNT}}) ##

{{#TABLES}}
Table Name: {{TABLE_NAME}}
{{/TABLES}}
{{/SCHEMATA}}

Report Generated On: {{GENERATED}}
=====
End of MySQL Workbench Custom Report

```

This simplified report just lists the schemata and the tables in a model. The date and time the report was generated is also displayed as a result of the use of the `{{GENERATED}}` variable.

- The custom template can then be tested. Start MySQL Workbench, load the model to generate the report for, select the Model, DBDOC - Model Reporting menu item. Then select the new custom template from the list of available templates, select an output directory, and click Finish to generate the report. Finally, navigate to the output directory to view the finished report.

Chapter 10. Code Generation Overview

Table of Contents

10.1. Generating SQL Statements	207
10.2. Generating PHP Code	207

This document provides a quick hands-on introduction to using MySQL Workbench to generate code for later use, for either in or outside of MySQL Workbench.

10.1. Generating SQL Statements

MySQL Workbench can be used to generate SQL, most typically as either `INSERT` statements or `SELECT` queries.

Below are the most common methods of generating SQL statements in MySQL Workbench.

- Right-clicking on a table or column name within the schema view will offer many different SQL generating options.

For example, right-clicking on a table name allows creating "`SELECT All`", "`INSERT`", "`UPDATE`", "`DELETE`", and "`CREATE`" statements, and with the option to either send these statements to the system's clipboard, or to the SQL Editor window.

- Right-clicking on a field within a cell in the SQL Editor offers the "Copy Row Content" and "Copy Field Content" options, and includes the option to leave the chosen values unquoted.
- All of the MySQL Workbench Export options include the option to export as SQL.

10.2. Generating PHP Code

MySQL Workbench can be used to generate PHP code with the bundled PHP plugin, by using the Tools, Utilities, Copy as PHP Code menu option.

Below is an example scenario for how to create PHP code. It is a `SELECT` statement, and optionally uses `SET` to set variables.

SQL `@variables` will generate PHP variables in the code, which will then be bounded to the statement before execution.

1. Generate or type in the desired SQL query into the SQL editor. This example will use the `sakila` database, with the query being:

```
SET @last_update = '2006-02-14';

SELECT actor_id, first_name, last_name, last_update
  FROM actor
 WHERE last_update > @last_update;
```

2. While in the SQL editor, choose Tools, Utilities, Copy as PHP Code (Iterate SELECT Results) from the main menu. This will copy PHP code to the clipboard.
3. Paste the code to the desired location.

Additionally, PHP code that connects to the MySQL database can also be generated by choosing [Tools, Utilities, Copy as PHP Code \(Connect to Server\)](#).

The generated code will look like this:

```
<?php

$host      = "localhost";
$port      = 3306;
$socket    = "";
$user      = "nobody";
$password  = "";
$dbname   = "sakila";

$con = new mysqli($host, $user, $password, $dbname, $port, $socket)
      or die ('Could not connect to the database server' . mysqli_connect_error());

// $con->close();

$query = "SELECT actor_id, first_name, last_name, last_update
          FROM actor
          WHERE last_update > ?";
$last_update = '';

$stmt->bind_param('s', $last_update);

if ($stmt = $con->prepare($query)) {

    $stmt->execute();
    $stmt->bind_result($actor_id, $first_name, $last_name, $last_update);

    while ($stmt->fetch()) {
        // printf("%s, %s, %s, %s\n",
        //       $actor_id, $first_name, $last_name, $last_update);
    }

    $stmt->close();
}

?>
```

Note that the PHP code uses the [mysqli](#) PHP extension. This extension must be enabled in your PHP distribution for this code to work. For additional details, see [MySQL Improved Extension \(Mysqli\)](#).

Chapter 11. MySQL Enterprise Features

Table of Contents

11.1. MySQL Audit Inspector Interface	209
11.2. MySQL Enterprise Backup Interface	213
11.3. MySQL Workbench Backup Recovery	216

A MySQL Enterprise subscription is the most comprehensive offering of MySQL database software, services and support; it ensures that your business achieves the highest levels of reliability, security, and uptime.

An Enterprise Subscription includes a MySQL Workbench GUI for the following features:

- The MySQL Enterprise Server: The most reliable, secure, and up-to-date version of the world's most popular open source database
- **MySQL Enterprise Backup**: Performs backup and restore operations for MySQL data, and MySQL Workbench offers a GUI for these operations
- **MySQL Enterprise Audit**: An easy to use auditing and compliance solution for applications that are governed by both internal and external regulatory guidelines
- MySQL Production Support: Technical and consultative support when you need it, along with regularly scheduled service packs, and hot-fixes, and MySQL Workbench links to your My Oracle Support (MOS) service

For more information, visit <http://www.mysql.com/enterprise>

Now, let us further explore a few of the Enterprise features:

11.1. MySQL Audit Inspector Interface

MySQL Workbench offers a GUI interface to the Audit Inspector.

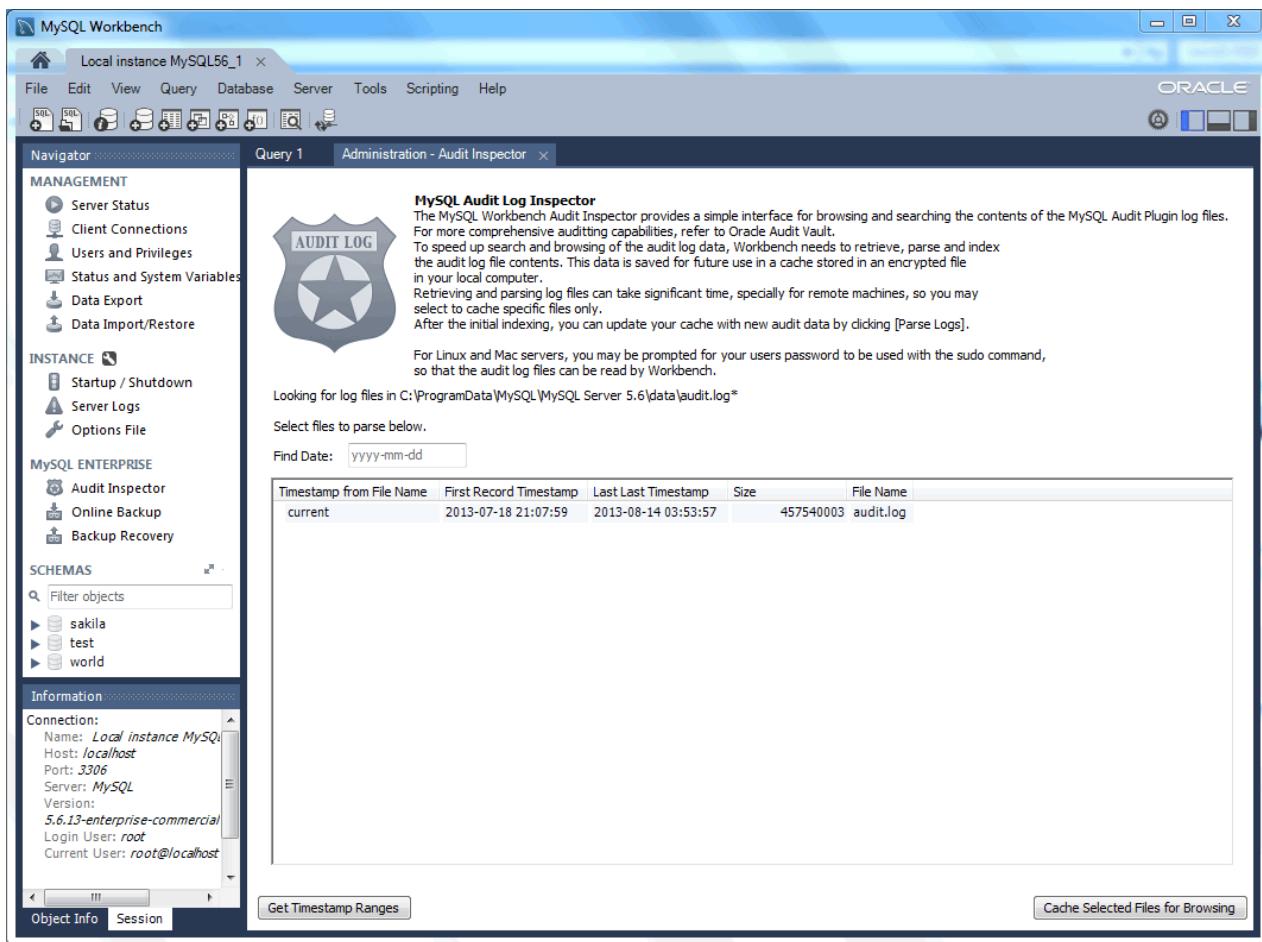


Note

The Audit Inspector interface was added in MySQL Workbench 6.0.0.

Initially, when you first load the **Audit Inspector**, you must use MySQL Workbench to cache the audit log for performance reasons. MySQL Workbench will then parse, index, and retrieve values from the encrypted cached file on your local computer. At this stage, you also set a password for the encrypted file that will be used when viewing this file. The initial screen looks similar to:

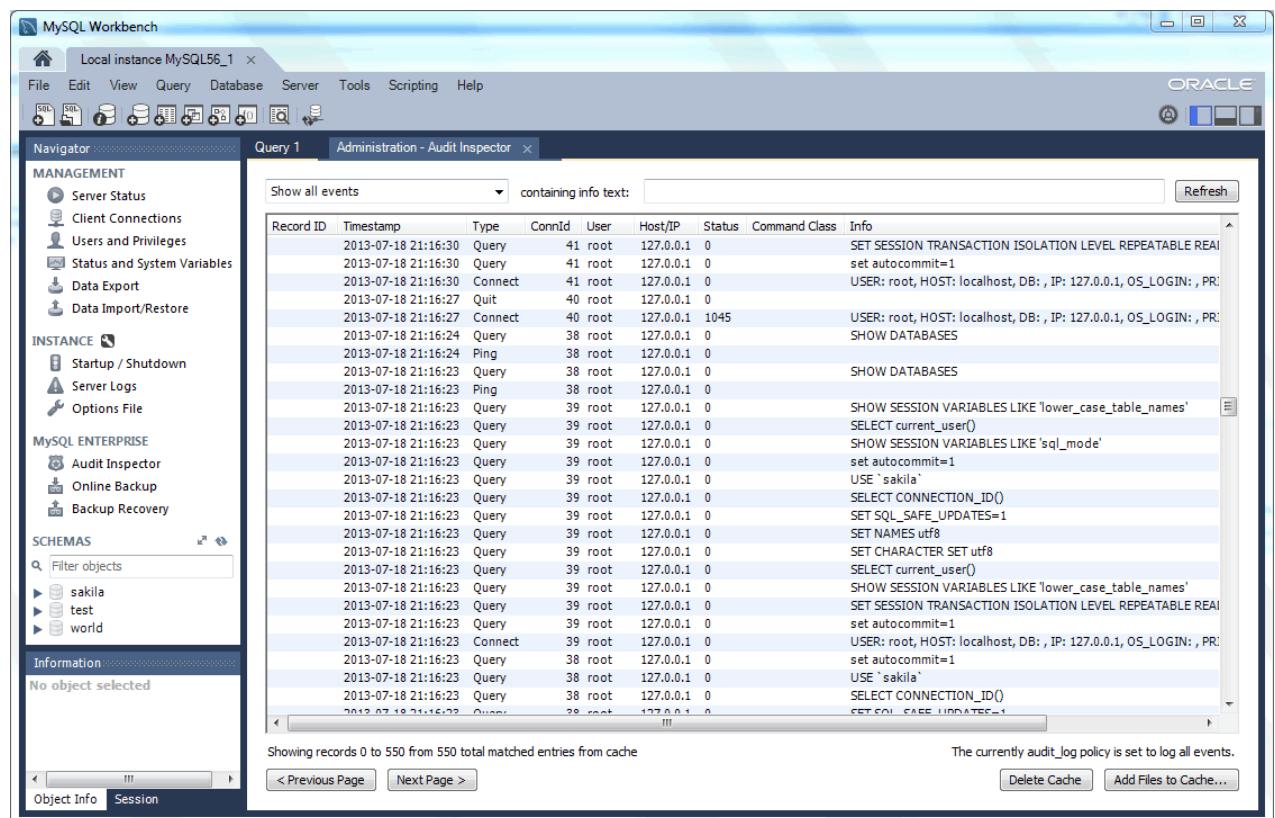
Figure 11.1. Workbench: Audit Inspector: Initializing



Note

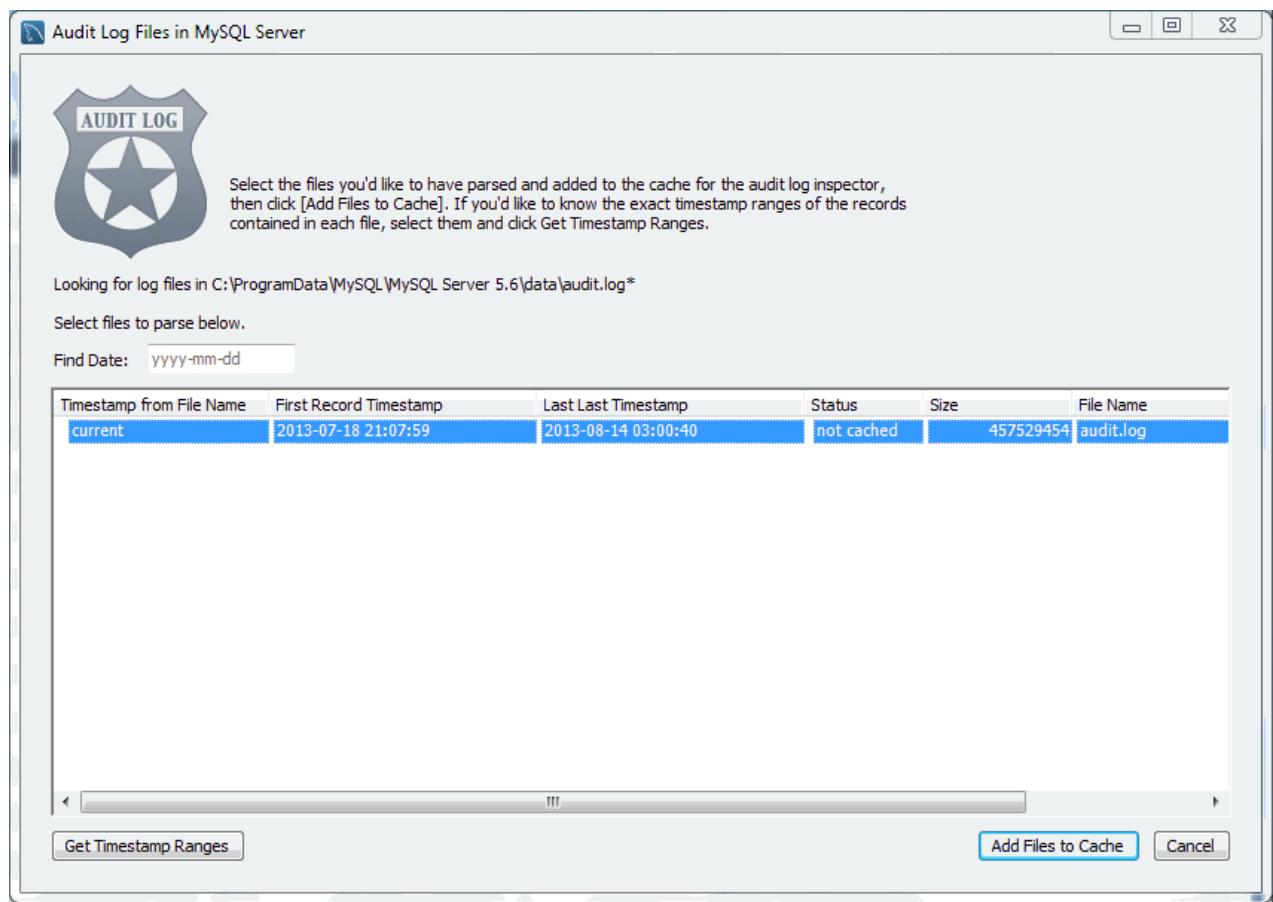
Generating the cache file can take a long time. If you press Abort during the caching process, MySQL Workbench will save the results that were cached at the point you pressed Abort.

After caching an audit log, the **Audit Inspector** page will display the results:

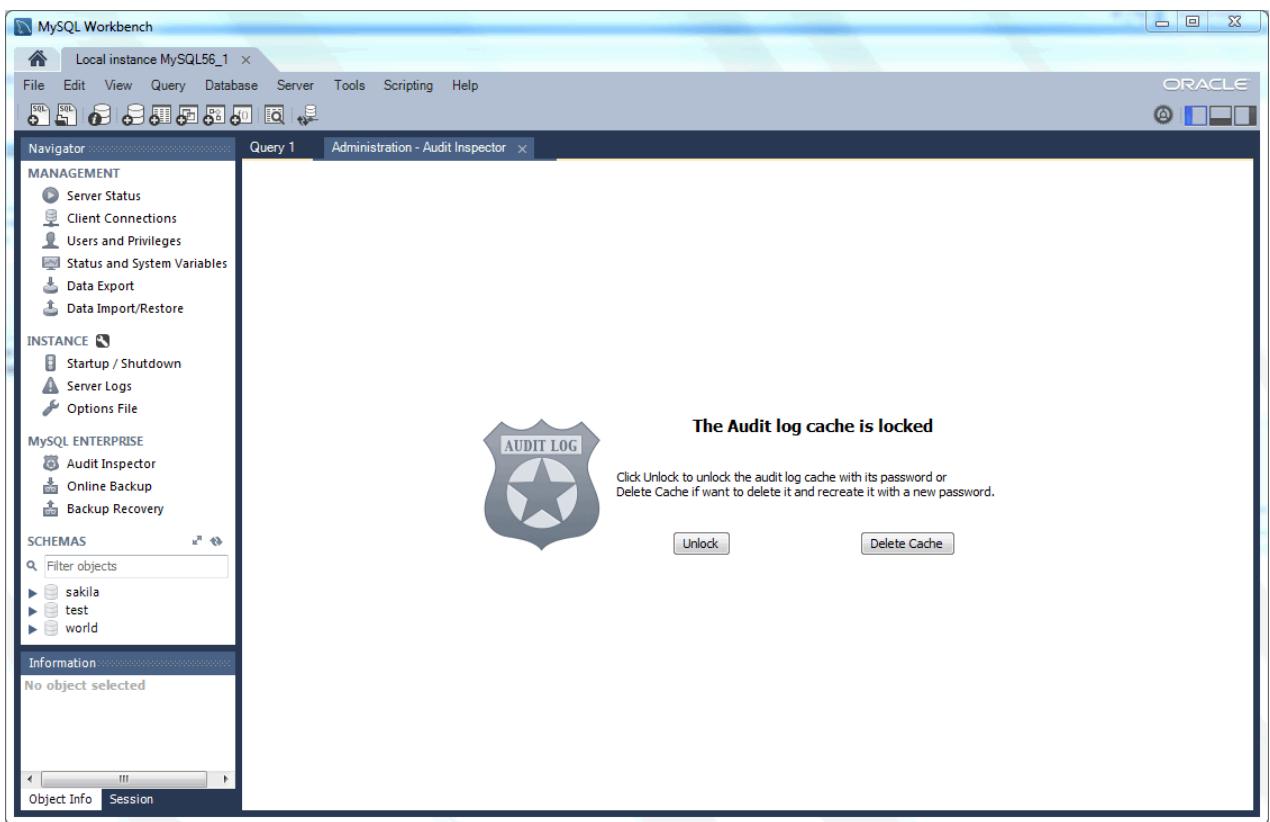
Figure 11.2. Workbench: Audit Inspector

The search field offers criteria for narrowing the displayed events, including Show events of type Fetch and Show events of type Query, and defaults to Show all events. Custom filters are also available.

You can Add Files to Cache from the main Audit Inspector page:

Figure 11.3. Workbench: Audit Inspector: Add Files to Cache

Future uses of the Audit Inspector will require the password that you set during the initial step. The login page:

Figure 11.4. Workbench: Audit Inspector: Unlock

11.2. MySQL Enterprise Backup Interface

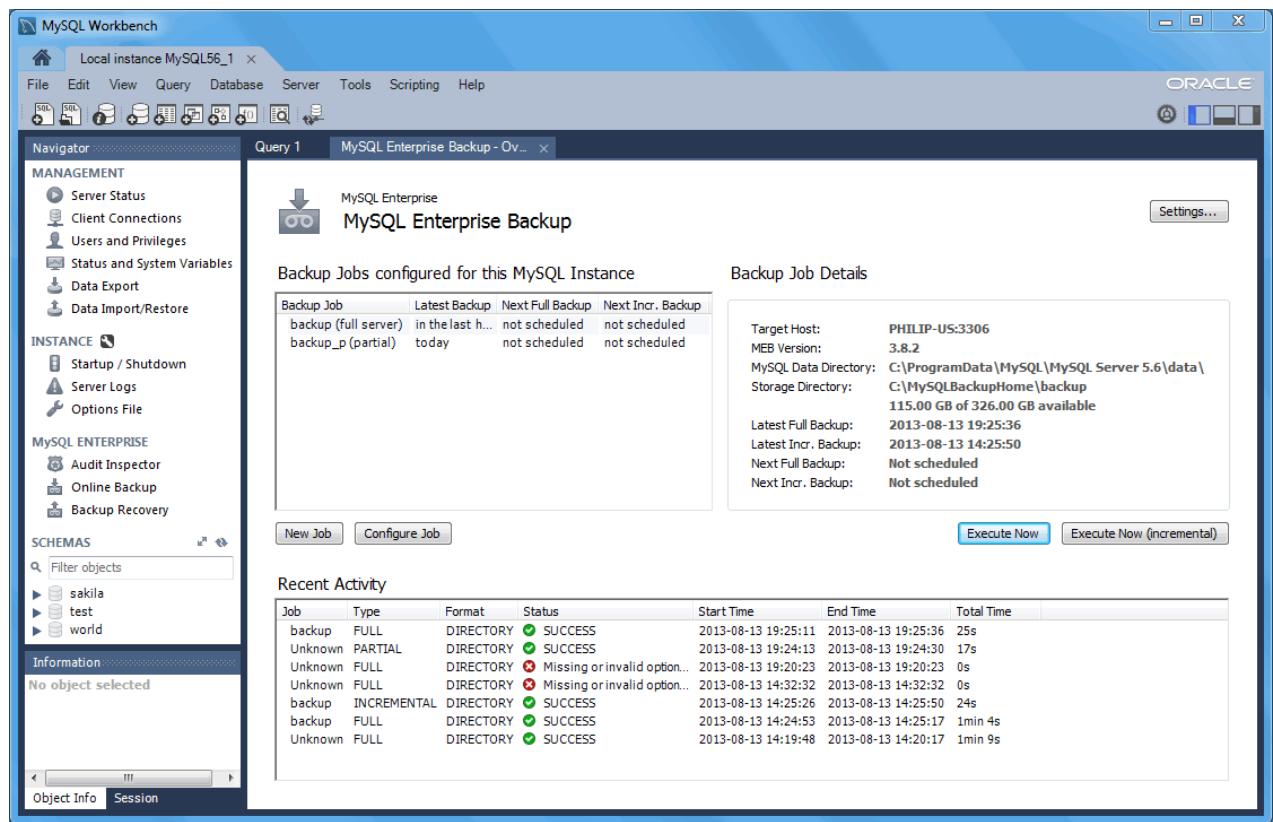
MySQL Workbench offers a MySQL Enterprise Backup GUI interface, and supports MySQL Enterprise Backup 3.6.0 and above.



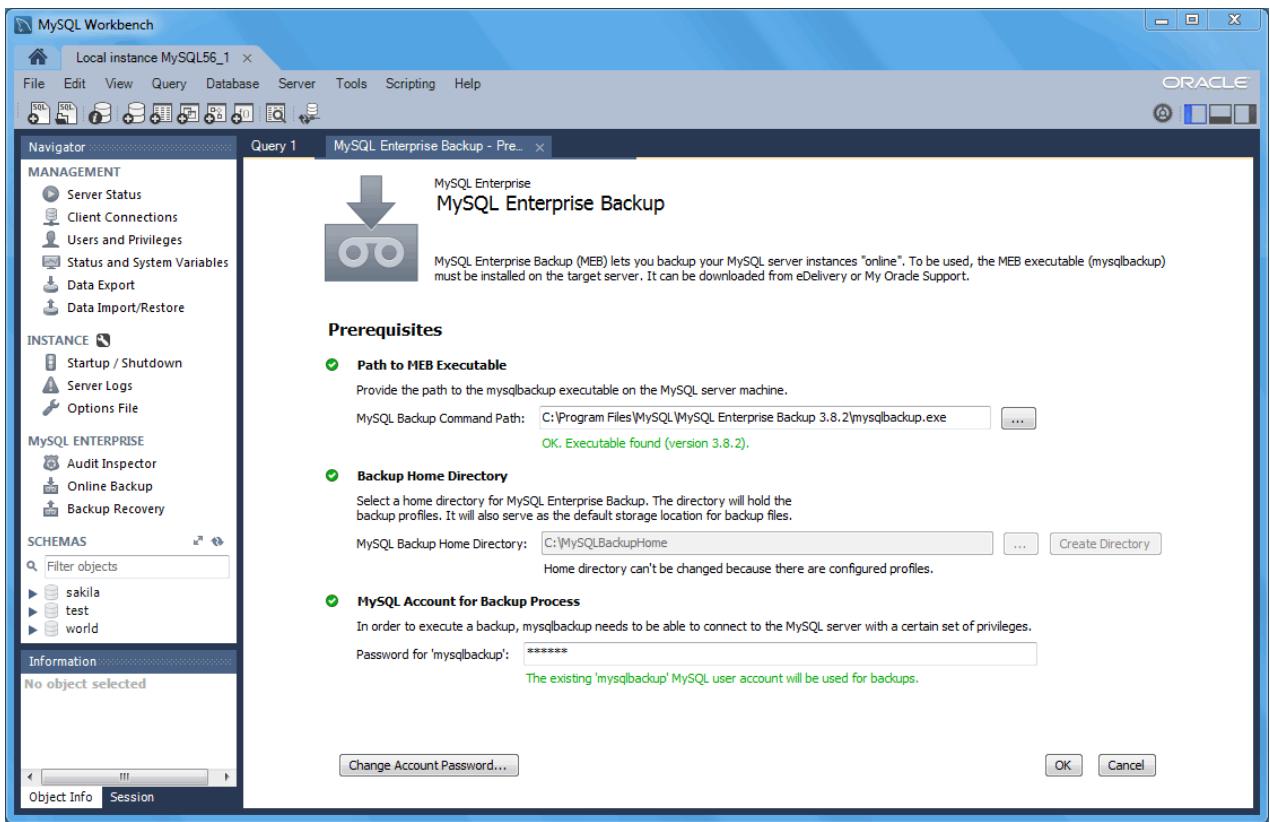
Note

The MEB interface was added in MySQL Workbench 6.0.0.

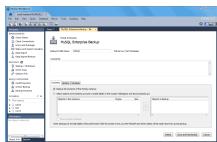
The MySQL Enterprise Backup interface offers the ability to create, schedule, and restore both full and incremental backups.

Figure 11.5. Workbench: MySQL Enterprise Backup

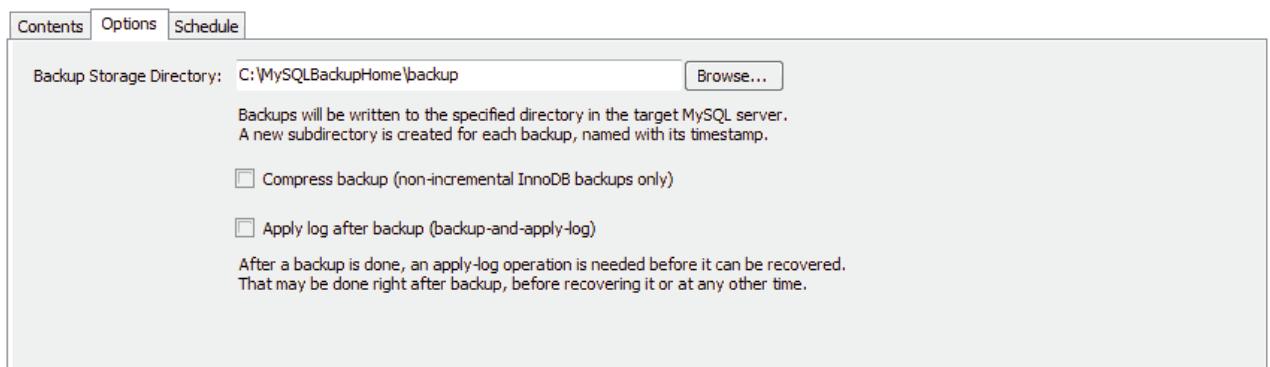
The **Settings** include options to set a path to the MySQL Enterprise Backup executable, the backup home directory, and MySQL user account used to execute the backup process.

Figure 11.6. Workbench: MySQL Enterprise Backup Settings

A job can be scheduled, and the configuration options are separated using three tabs. The **Contents** tab includes the profile name, comments that describe the job, and whether the job is a full or partial backup.

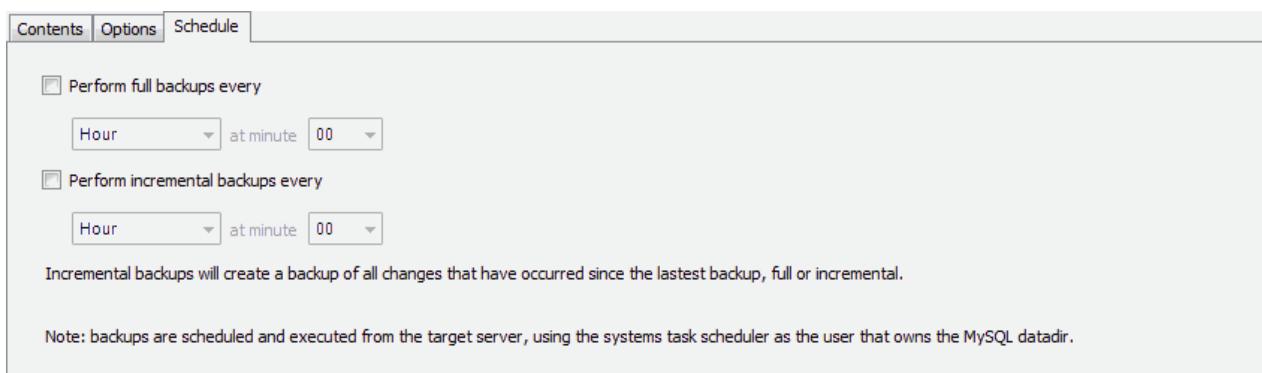
Figure 11.7. Workbench: MySQL Enterprise Backup Configuration: Contents

The **Options** tab includes setting the storage directory, compression options, and whether an [apply-log](#) operation is performed after the backup.

Figure 11.8. Workbench: MySQL Enterprise Backup Configuration: Options

The **Schedule** tab sets the backup schedule for both full and incremental backups.

Figure 11.9. Workbench: MySQL Enterprise Backup Configuration: Schedule



To recover backups, see [Section 11.3, “MySQL Workbench Backup Recovery”](#).

11.3. MySQL Workbench Backup Recovery

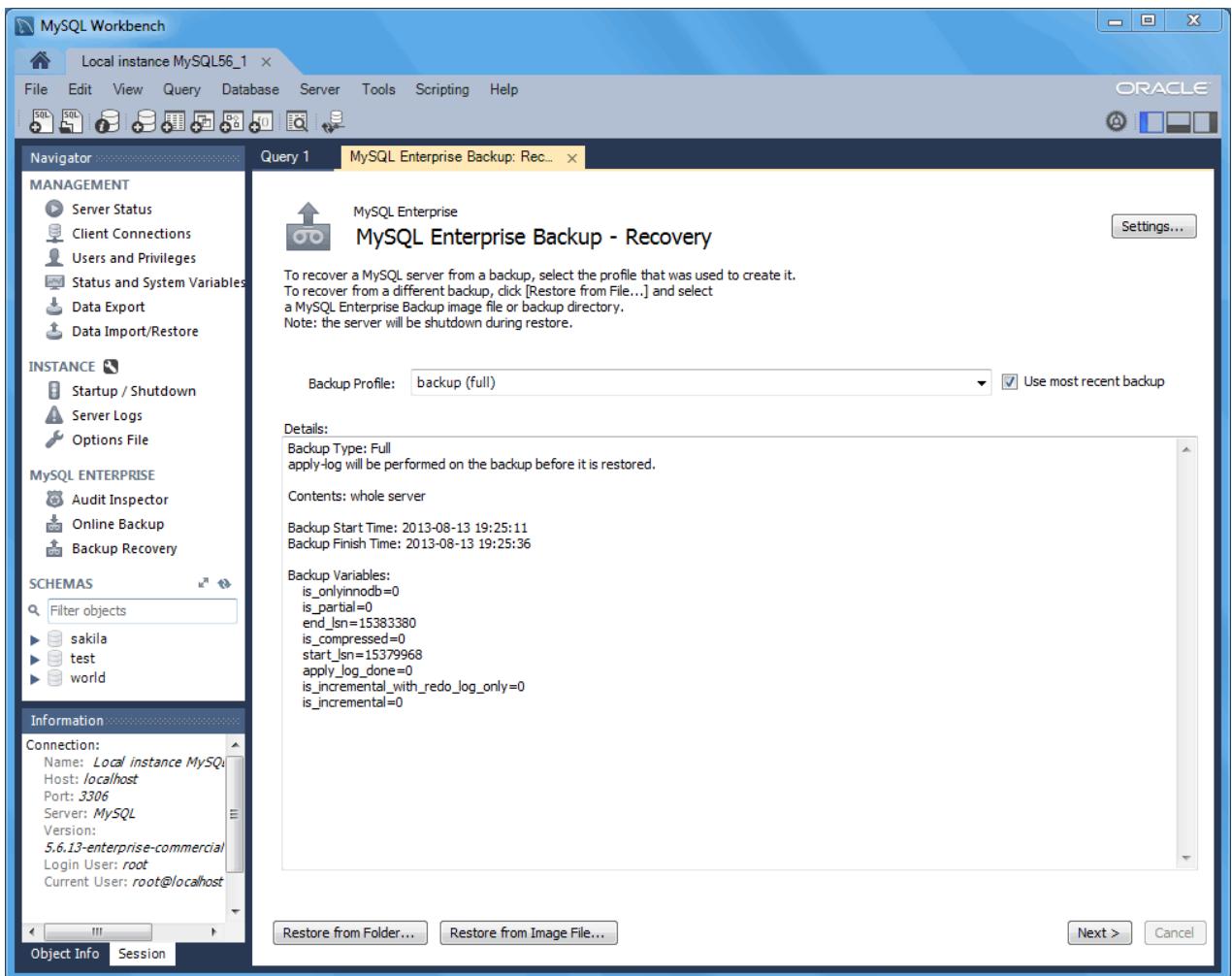
The Backup Recovery wizard is used to recover MySQL Enterprise Backup backups. For more information about creating MySQL Enterprise Backup backups using MySQL Workbench, see [Section 11.2, “MySQL Enterprise Backup Interface”](#)



Note

This interface was added in MySQL Workbench 6.0.0.

The Backup Recovery wizard allows you to restore backups from folders, image files, and backup profiles created by [Section 11.2, “MySQL Enterprise Backup Interface”](#).

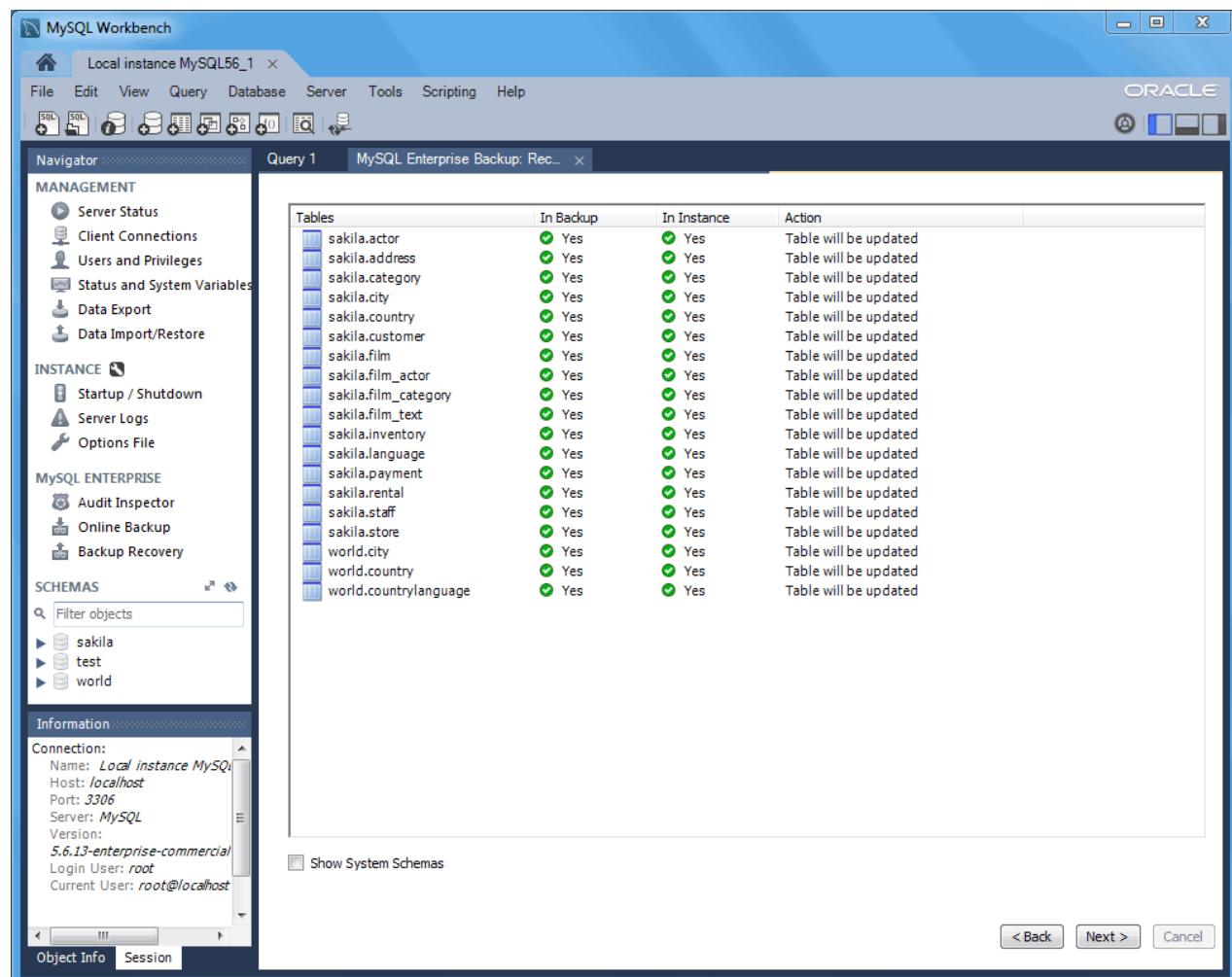
Figure 11.10. Workbench: Backup Recovery: Main page

In our example, we will restore a full backup that was created by MySQL Workbench Online Backup. After choosing the "backup (full)" profile that we created earlier, the next page lists each table that will be updated:

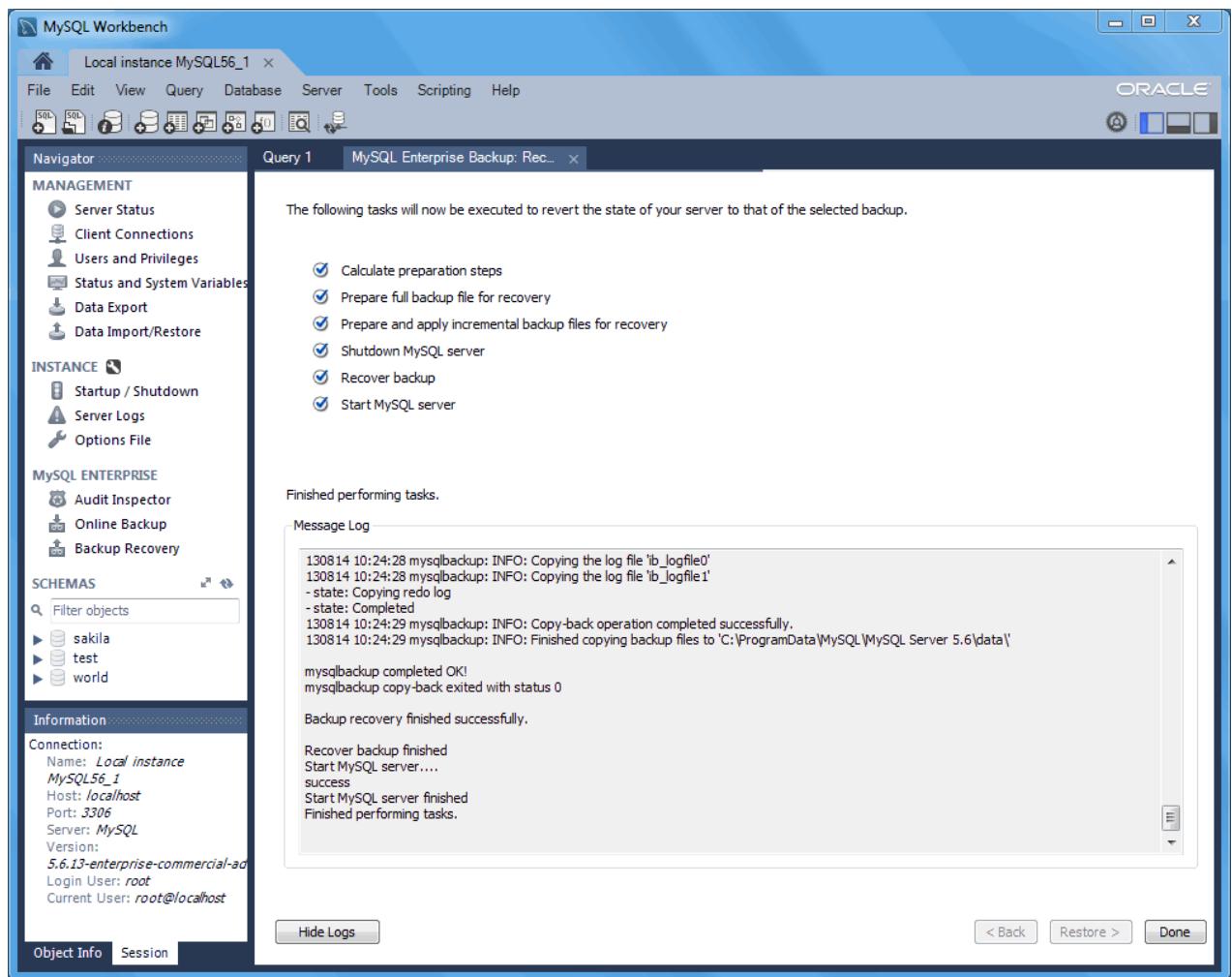


Note

We unchecked **Show System Schemas** the option, which is enabled by default.

Figure 11.11. Workbench: Backup Recovery: Table View

Clicking **Next >** will open the **Restore** wizard. We then clicked **Restore >** to execute the restoration process, and toggled the message logs in our example below:

Figure 11.12. Workbench: Backup Recovery: Restore

Chapter 12. Database Migration Wizard

Table of Contents

12.1. General installation requirements	222
12.1.1. ODBC Libraries	222
12.1.2. ODBC Drivers	223
12.2. Migration Overview	223
12.2.1. A visual guide to performing a database migration	224
12.2.2. Migrating from supported databases	238
12.2.3. Migrating from unsupported (generic) databases	239
12.3. Conceptual DBMS equivalents	239
12.4. Microsoft SQL Server migration	241
12.4.1. Preparations	241
12.4.2. Drivers	241
12.4.3. Connection Setup	243
12.4.4. Microsoft SQL Server Type Mapping	243
12.5. PostgreSQL migration	244
12.5.1. Preparations	244
12.5.2. Drivers	247
12.5.3. Connection Setup	247
12.5.4. PostgreSQL Type Mapping	247
12.6. MySQL migration	249
12.7. Using the MySQL Workbench Migration Wizard	249
12.7.1. Connecting to the databases	249
12.7.2. Schemata Retrieval and Selection	249
12.7.3. Reverse Engineering	250
12.7.4. Object Selection	250
12.7.5. Migration	250
12.7.6. Manual Editing	250
12.7.7. Target Creation Options	250
12.7.8. Schema Creation	251
12.7.9. Create Target Results	251
12.7.10. Data Migration Setup	251
12.7.11. Bulk Data Transfer	251
12.7.12. Migration Report	251
12.8. MySQL Workbench Migration Wizard FAQ	251

MySQL Workbench provides the ability to migrate ODBC compliant databases to MySQL.

The MySQL Workbench Migration Wizard was added in MySQL Workbench 5.2.41.

- Convert (migrate) different database types, including MySQL, across servers
- Convert tables and copy data, but will not convert stored procedures, views, or triggers
- Allows customization and editing during the migration process
- Works on Linux, Mac OS X, and Microsoft Windows

This is not an exhaustive list. The following sections discuss these and additional migration capabilities.

Set up may be the most challenging aspect of using the MySQL Workbench Migration Wizard. There is the [installation section](#), which describes setting up ODBC requirements for Linux, Mac OS X, and Microsoft

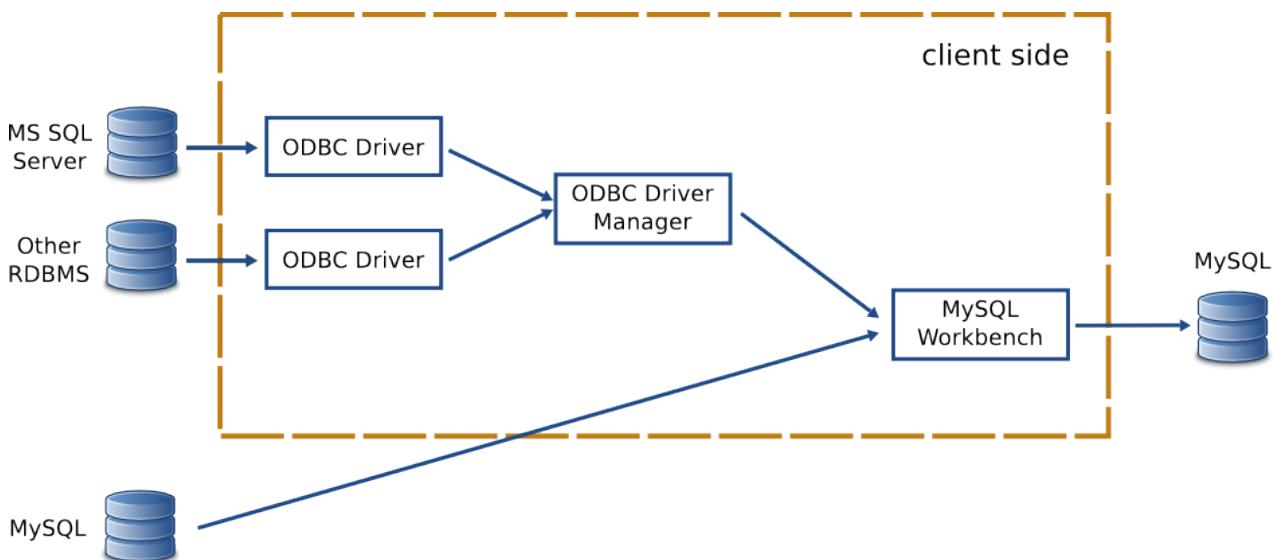
Windows, and the [Database Product Specific Notes](#) section that references setup conditions for each RDBMS.

12.1. General installation requirements

The MySQL Workbench Migration Wizard uses ODBC to connect to a source database, except for MySQL. You will need the ODBC driver installed that corresponds to the database you want to migrate from. For example, PostgreSQL can be migrated with the psqlodbc ODBC driver; Microsoft SQL Server can be migrated using the native Microsoft SQL Server driver on Windows or with FreeTDS on Linux and Mac OS X.

The following diagram shows the general components involved in an ODBC connection:

Figure 12.1. MySQL Workbench migration installation diagram



When specifying the source RDBMS, you can either use a data source configured externally, or provide the individual connection parameters to MySQL Workbench. If you already have an ODBC Data Source configured in your system, then you can use that in MySQL Workbench.

12.1.1. ODBC Libraries



Note

This section may be skipped when using a MySQL Workbench binary that is provided by Oracle.

An ODBC Driver Manager library must be present. Both Windows and Mac OS X provides one.

Linux

iODBC: MySQL Workbench binaries provided by Oracle already include iODBC and no additional action is required. If you compile it yourself, you must install iODBC or unixODBC. iODBC is recommended. You can use the iODBC library provided by your distribution.

pyodbc: is the Python module used by MySQL Workbench to interface with ODBC, and may be used to migrate ODBC compliant databases such as PostgreSQL and DB2. In Windows and Mac OS X, it is included with Workbench. In Linux, binaries provided by Oracle also include pyodbc.

If you're using a self-compiled binary, make sure you have the latest version, and that it is compiled against the ODBC manager library that you chose, whether it is iODBC or unixODBC. As of version 3.0.6, pyodbc will compile against unixODBC by default. If you are compiling against iODBC then you must perform the following steps:

1. Install the development files for iODBC. Usually you just need to install the `libiodbc-devel` or `libiodbc2-dev` package provided by your distribution.
2. In the pyodbc source directory, edit the `setup.py` file and around line 157, replace the following line: `settings['libraries'].append('odbc')` with `settings['libraries'].append('iodbc')`
3. Execute the following command as the root user: `CFLAGS=`iodbc-config --cflags` LDFLAGS=`iodbc-config --libs` python setup.py install`

12.1.2. ODBC Drivers

For each RDBMS, you need its corresponding ODBC driver, which must also be installed on the same machine that MySQL Workbench is running on. This driver is usually provided by the RDBMS manufacturer, but in some cases they can also be provided by third party vendors or open source projects.

Operating systems usually provide a graphical interface to help set up ODBC drivers and data sources. Use that to install the driver (i.e., make the ODBC Manager "see" a newly installed ODBC driver). You can also use it to create a data source for a specific database instance, to be connected using a previously configured driver. Typically you need to provide a name for the data source (the DSN), in addition to the database server IP, port, username, and sometimes the database the user has access to.

If MySQL Workbench is able to locate an ODBC manager GUI for your system, a Tools, Start ODBC - Administrator menu item be present under the Tools menu as a convenience shortcut to start it.

- **Linux:** There are a few GUI utilities, some of which are included with unixODBC. Refer to the documentation for your distribution. iODBC provides `iodbcadm-gtk`. Official binaries of MySQL Workbench include it and it can be accessed through the Tools, Start ODBC Administrator menu item.
- **Mac OS X:** You can use the ODBC Administrator tool, which is provided as a separate download from Apple. If the tool is installed in the `/Applications/Utilities` folder, you can start it through the Tools, Start ODBC Administrator menu item.
- **Microsoft Windows:** You can use the Data Sources (ODBC) tool under Administrative Tools. And it can be started through the Tools, Start ODBC Administrator menu item.



ODBC Driver architecture

Since the ODBC driver needs to be installed in the client side, you will need an ODBC driver that supports your clients operating system and architecture. For example, if you are running MySQL Workbench from Linux x64, then you need a Linux x64 ODBC driver for your RDBMS. In Mac OS X, MySQL Workbench is built as a 32-bit application, so you need the 32-bit drivers.

12.2. Migration Overview

The Migration Wizard performs the following steps when migrating a database to MySQL:

1. Connects to the source RDBMS and retrieves a list of available databases/schemas.
2. Reverse engineers selected database/schemas into a internal representation specific to the source RDBMS. This step will also perform the renaming of objects/schemas depending on the type of object name mapping method that is chosen.

3. Automatically migrates the source RDBMS objects into MySQL specific objects.
 - a. Target schema objects are created.
 - b. Target table objects are created.
 - i. Columns for each table are copied.
 - A. Datatypes are mapped to MySQL datatypes.
 - B. Default values are mapped to a MySQL supported default value, if possible.
 - ii. Indexes are converted.
 - iii. Primary Keys are converted.
 - iv. Triggers are copied, and commented out if the source is not MySQL.
 - c. Foreign Keys for all tables (of all schemas) are converted.
 - d. View objects are copied, and commented out if the source is not MySQL.
 - e. Stored Procedure and Function objects are copied, and commented out if the source is not MySQL.
 4. Provides an opportunity to review the changes, for editing and correcting errors in the migrated objects.
 5. Creates the migrated objects in the target MySQL server. If there are errors, you can return to the previous step and correct them, and retry the target creation.
 6. Copy data of the migrated tables from the source RDBMS to MySQL.

MySQL Workbench provides support for migrating from some specific RDBMS products. The Migration Wizard will provide the best results when migrating from such products. However, in some cases, other unsupported database products can also be migrated by using its Generic database support, as long as you have an ODBC driver for it. In this case, the migration will be less automatic, but should still work nonetheless.

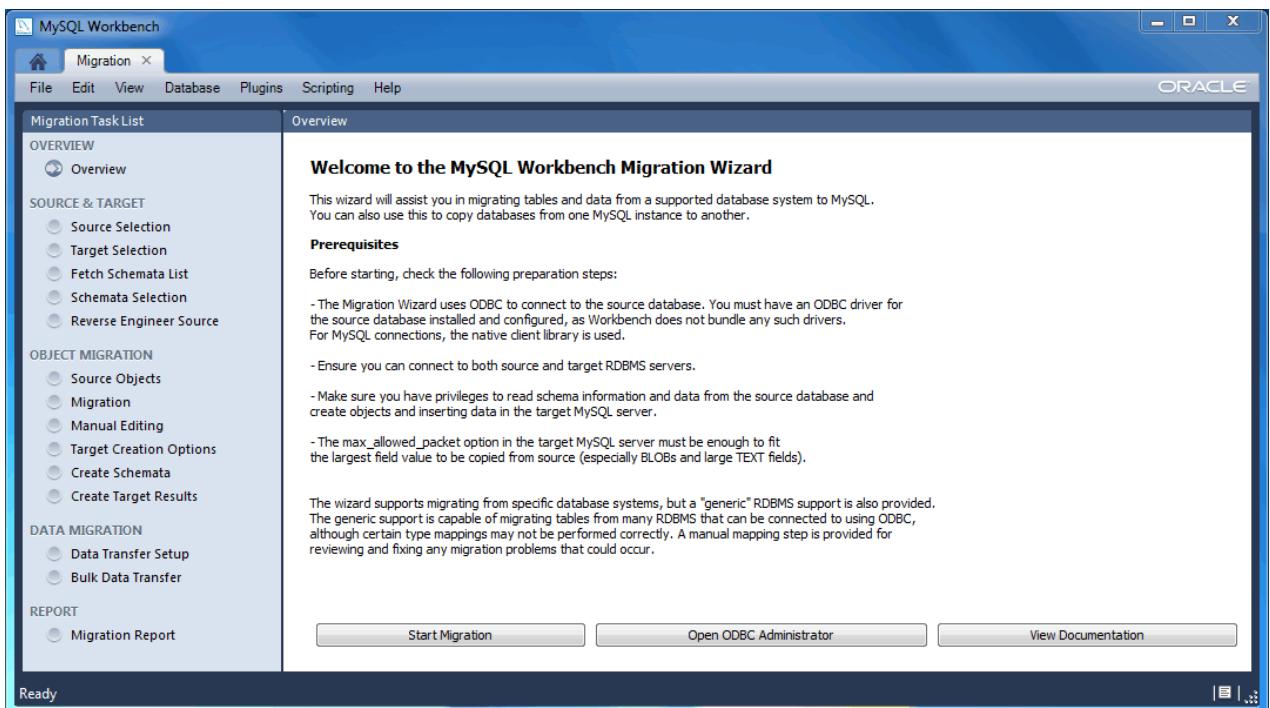
12.2.1. A visual guide to performing a database migration

This example will migrate a Microsoft SQL Server database to MySQL, and include a screenshot for each step.

From MySQL Workbench, choose Database, Migrate to open the migration wizard. The opening screen will look like this:

Overview

Figure 12.2. MySQL Workbench migration: Overview



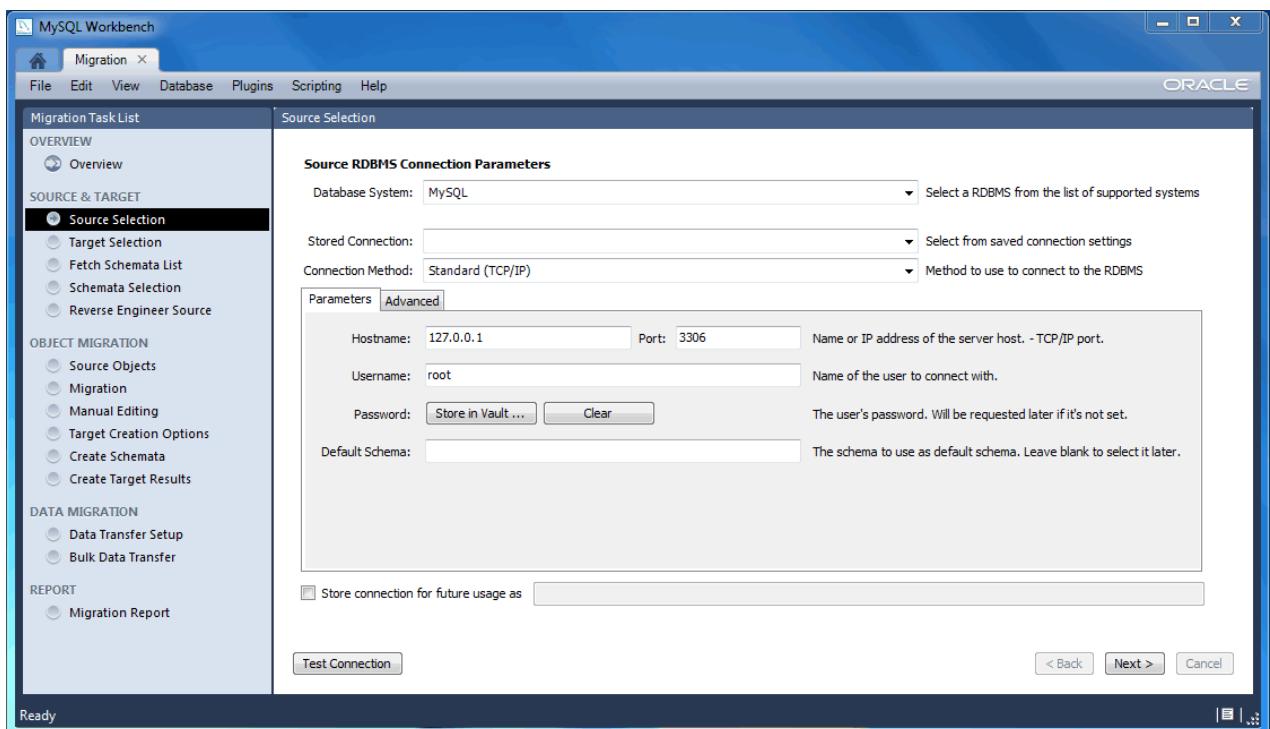
It describes the prerequisites and requirements that should be understood before proceeding further. The [Open ODBC Administrator](#) option will load [odbcad32.exe](#), and is used to confirm that the ODBC Driver for SQL Server is installed, and to make configuration changes if needed.

Click [Start Migration](#) to continue.

Source Selection

The source is the RDBMS that will be migrated to MySQL. Define the connection parameters and related information here by first choosing the **Database System**, as the other parameters will change accordingly to this choice.

Figure 12.3. MySQL Workbench migration: Source Selection (Parameters)

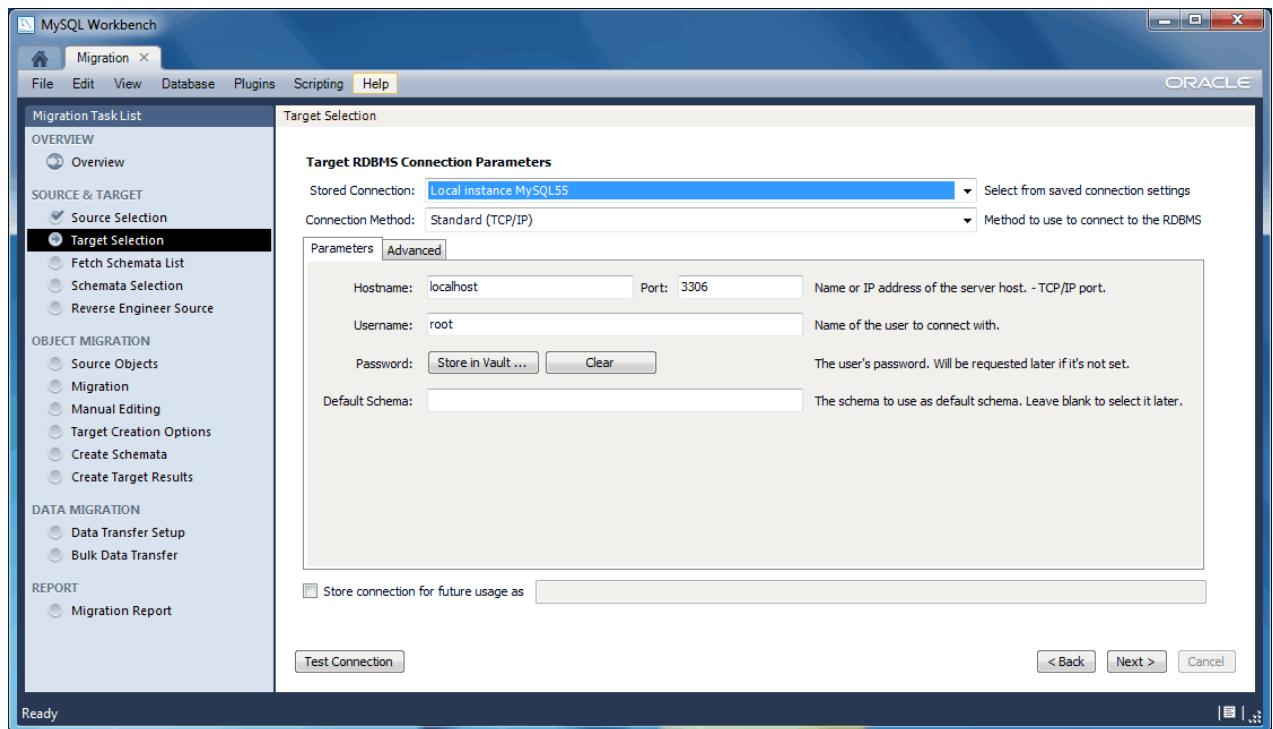


The optional **Store connection** option will save the connection details. It must be set before proceeding to the next step by clicking **Next**.

Target Selection

The target is the MySQL database that will contain the newly migrated database. The current Workbench MySQL connections will be available here, or you can choose [Manage DB Connections](#) to create a new connection.

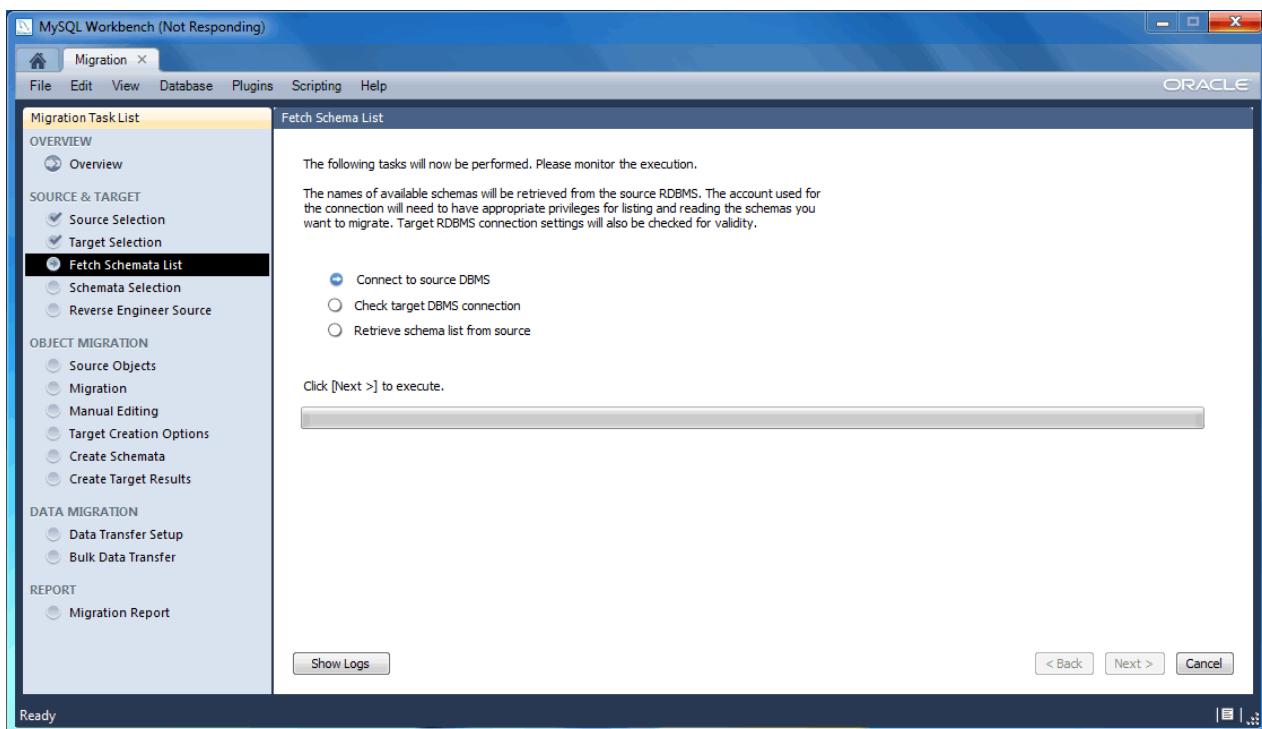
Figure 12.4. MySQL Workbench migration: Target selection



Fetch Schemata List

The Schemata list is retrieved from both the source and target RDBMS. This is an automated and informational step that reports connection related errors and/or general log information. Press **Next** to continue.

Figure 12.5. MySQL Workbench migration: Fetch Schemata List



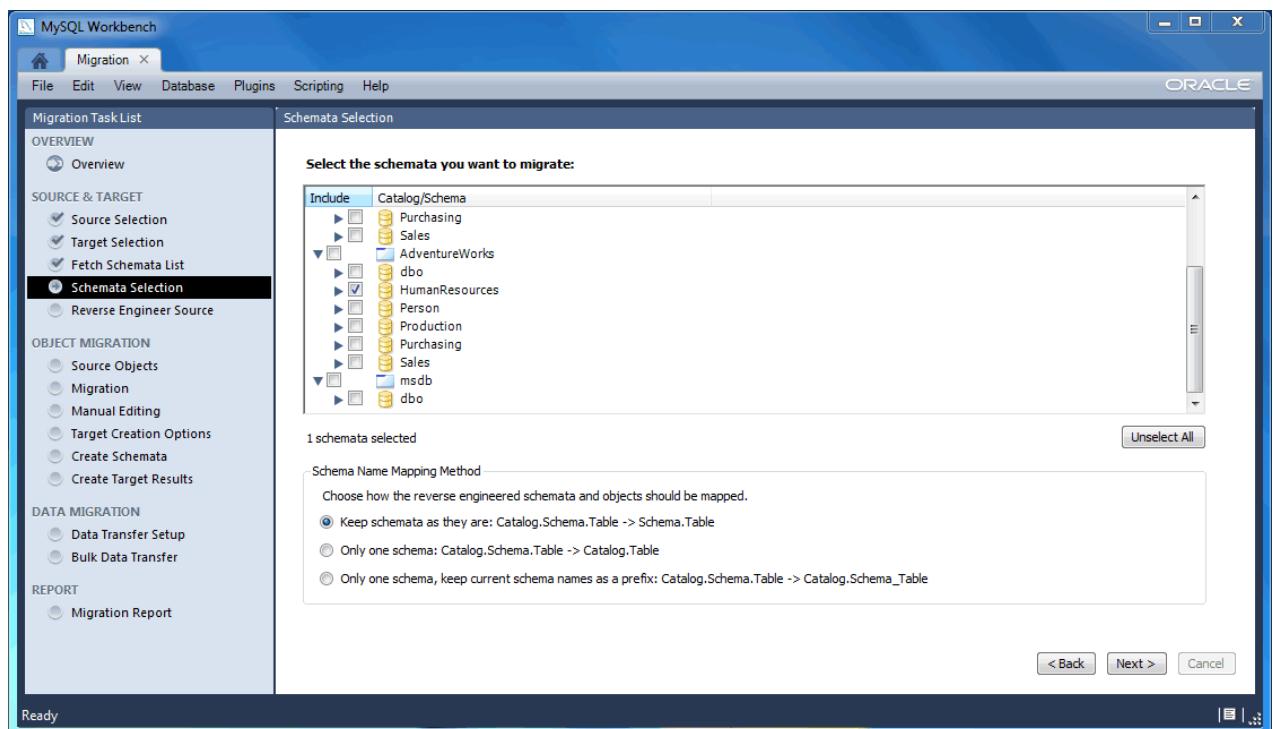
Schemata Selection

Choose the schemata you want to migrate.

"Schema Name Mapping Method" options while migrating Microsoft SQL Server:

- Keep schemata as they are: Catalog.Schema.Table -> Schema.Table: This will create multiple databases, one per schema.
- Only one schema: Catalog.Schema.Table -> Catalog.Table: Merges each schema into a single database.
- Only one schema, keep current schema names as a prefix: Catalog.Schema.Table -> Catalog.Schema_table: Preserves the schema name as a prefix.

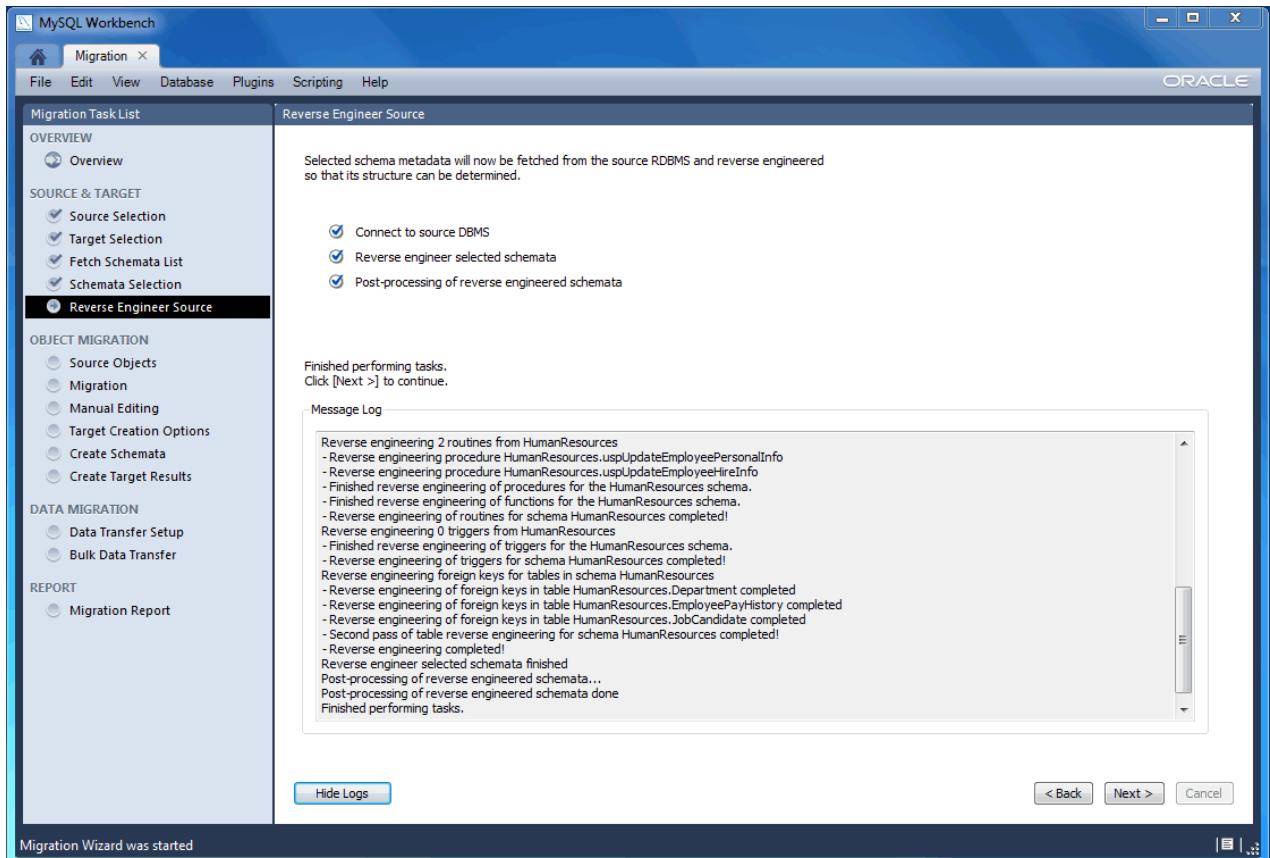
Figure 12.6. MySQL Workbench migration: Schemata Selection



Reverse Engineer Source

The source metadata is fetched from the source RDBMS, and reverse engineered. This is an automated and informational step that reports related errors and/or general log information. View the logs and then press **Next** to continue.

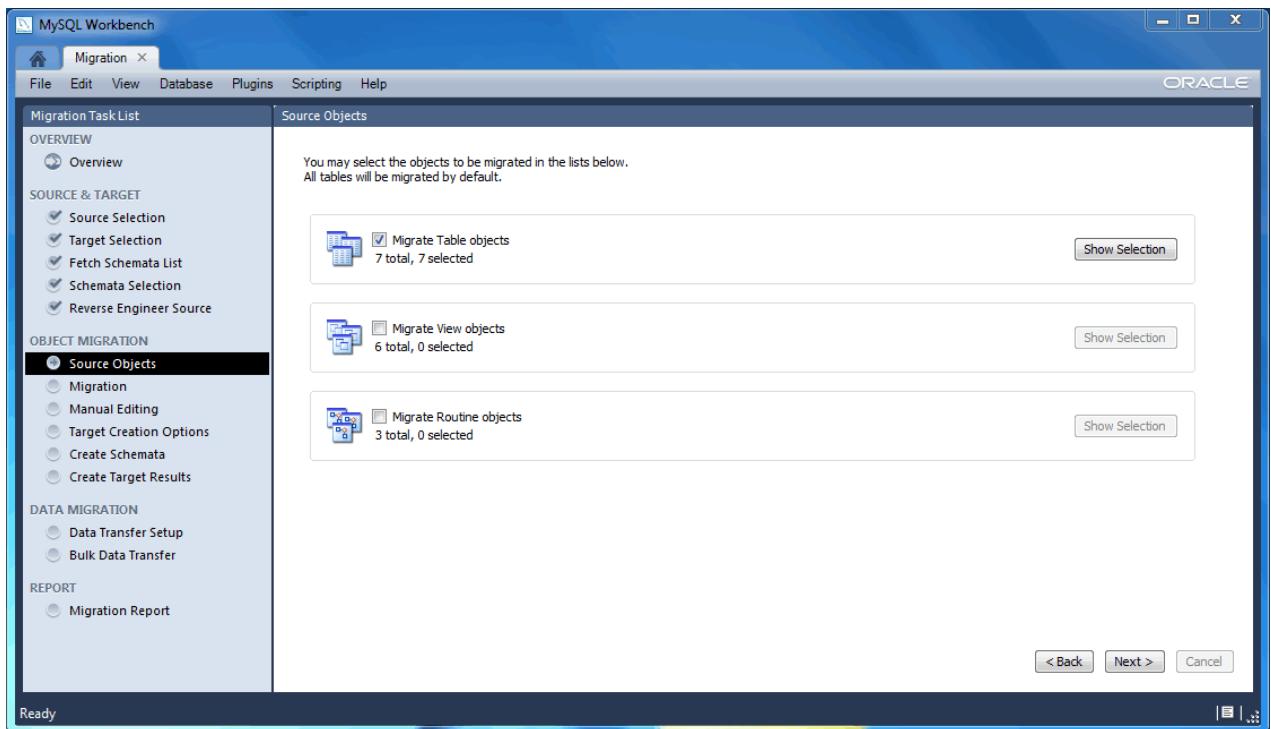
Figure 12.7. MySQL Workbench migration: Reverse Engineer Source



Source Objects

The discovered objects from the **Reverse Engineer Source** stage are revealed and made available. This includes Table, View, and Routine objects, with only the Table objects being selected by default.

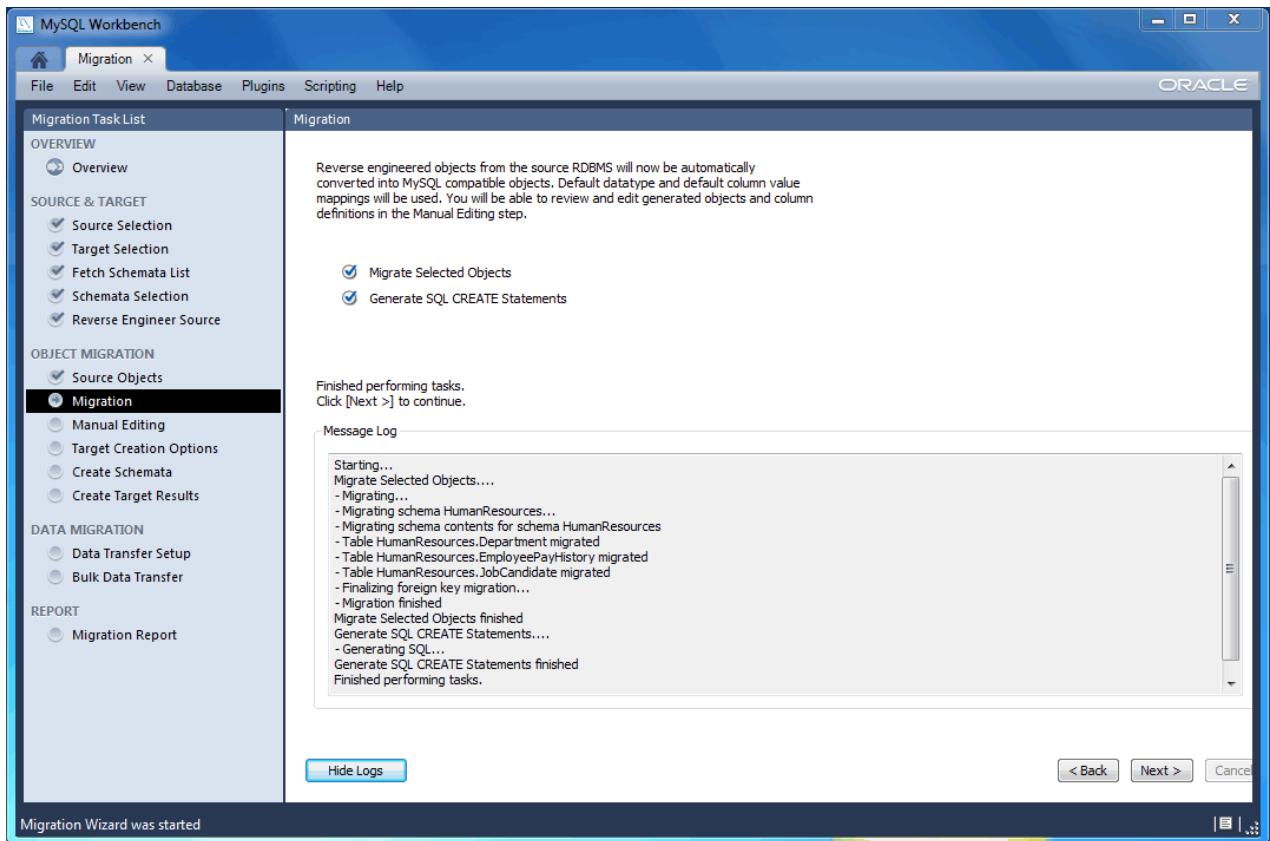
Figure 12.8. MySQL Workbench migration: Source Objects



Migration

The migration process now converts the selected objects into MySQL compatible objects. View the logs and then proceed.

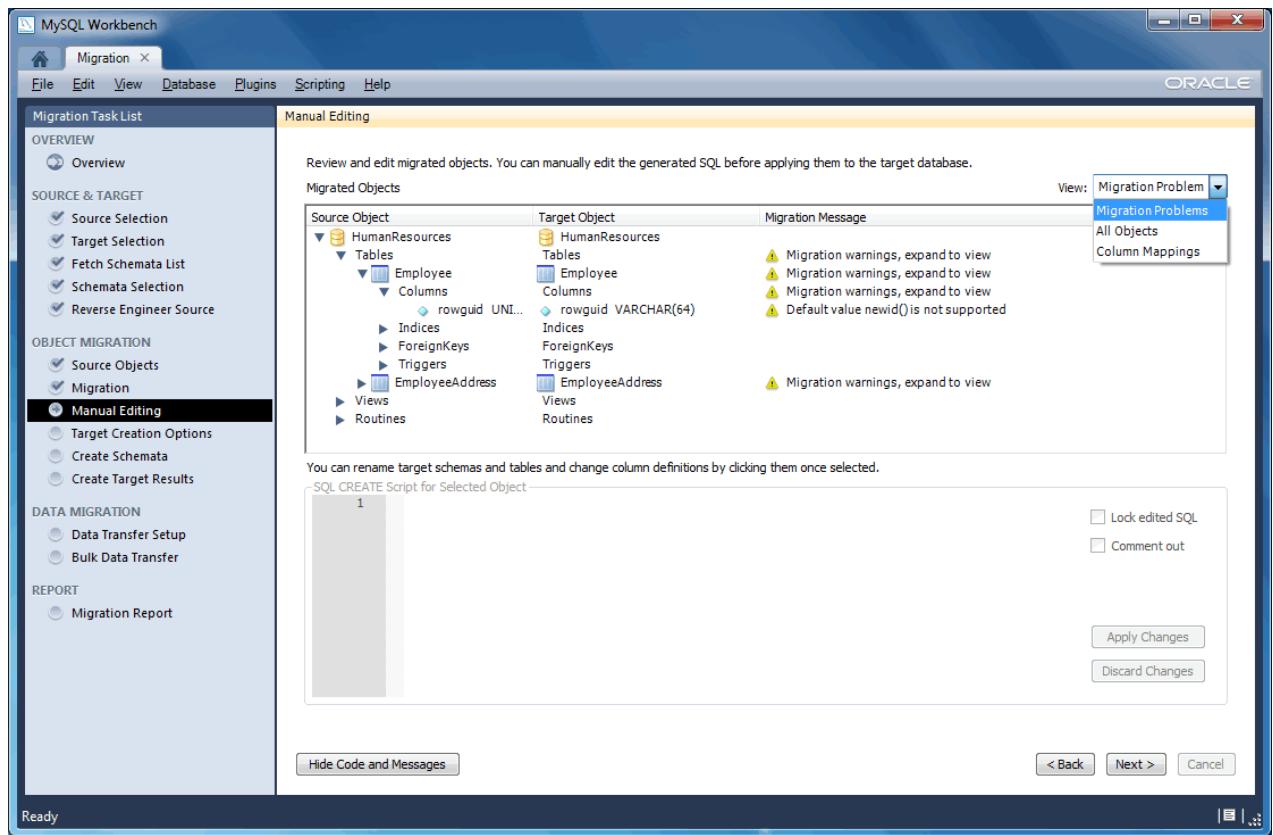
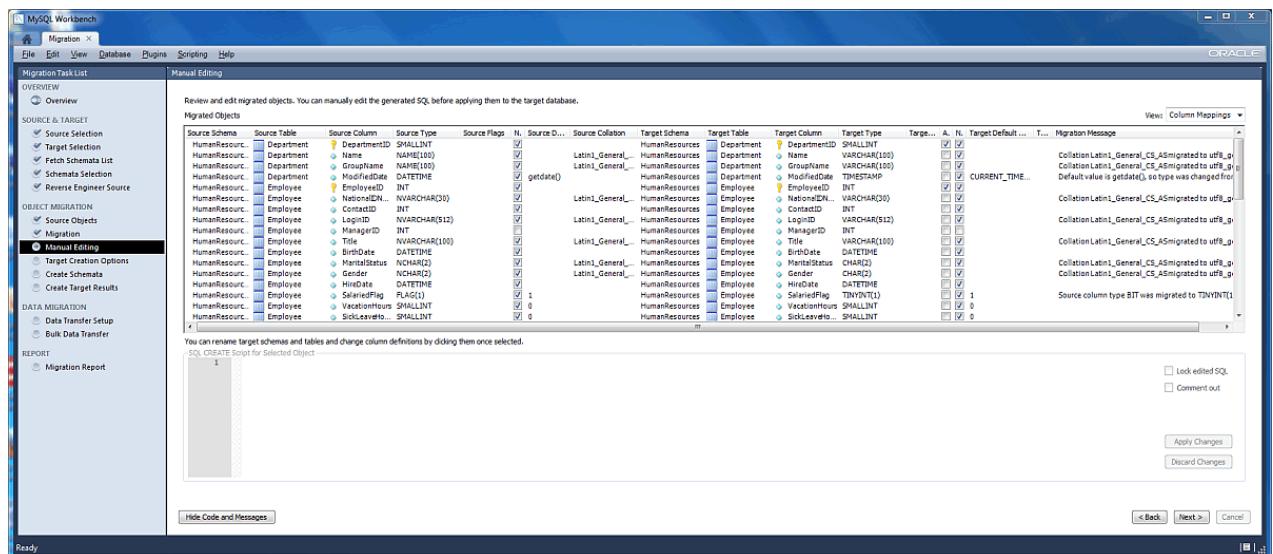
Figure 12.9. MySQL Workbench migration: Migration



Manual Editing

There are three sections to edit here, which are selected via the **View** select box on the top right. The **Show Code and Messages** button is available with every view, and it will show the generated MySQL code that corresponds to the selected object.

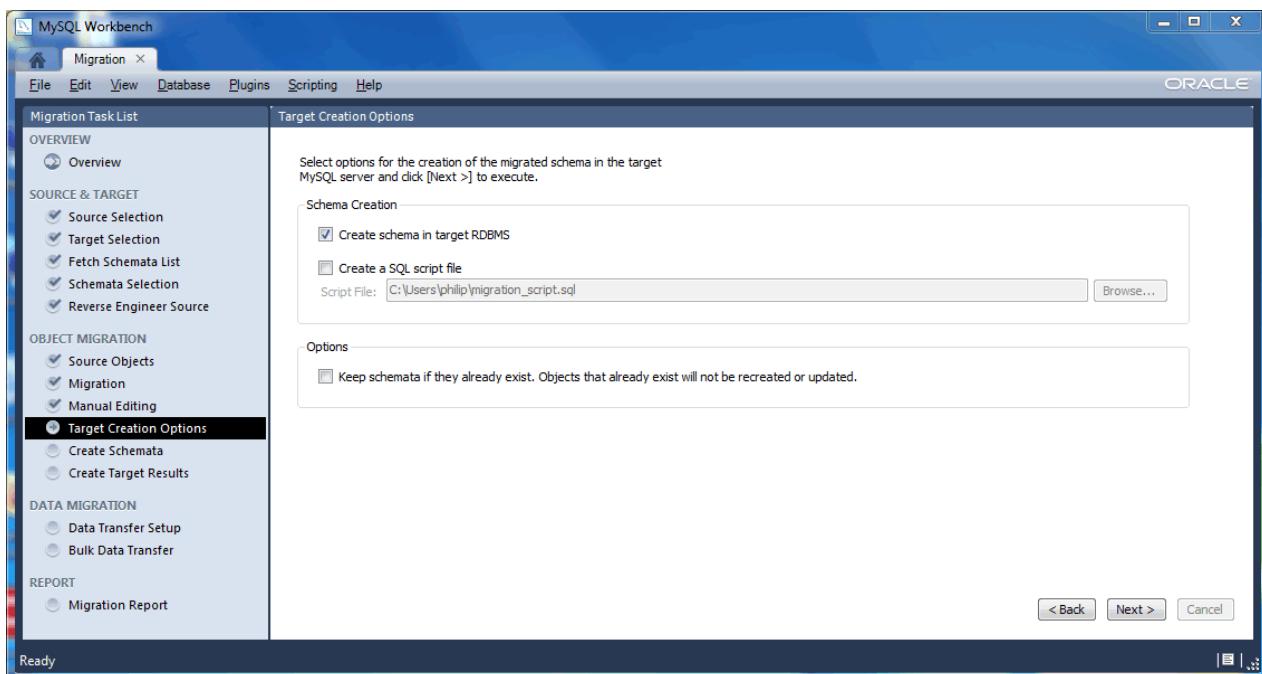
- **Migration Problems:** This will either report problems or display "No mapping problems found." It is an informational screen.
- **All Objects:** An object view that allows you to view and edit the object definitions. Double-click on a row to modify a target objects name.
- **Column Mappings:** Shows all of the table column mappings, and allows you to individually review and fix the mapping for all column types, default values, and other attributes.

Figure 12.10. MySQL Workbench migration: Manual Editing (All Objects)

Figure 12.11. MySQL Workbench migration: Manual Editing (Column Mappings)


Target Creation Options

The schema may be created by either adding it to the target RDBMS, creating an SQL script file, or both.

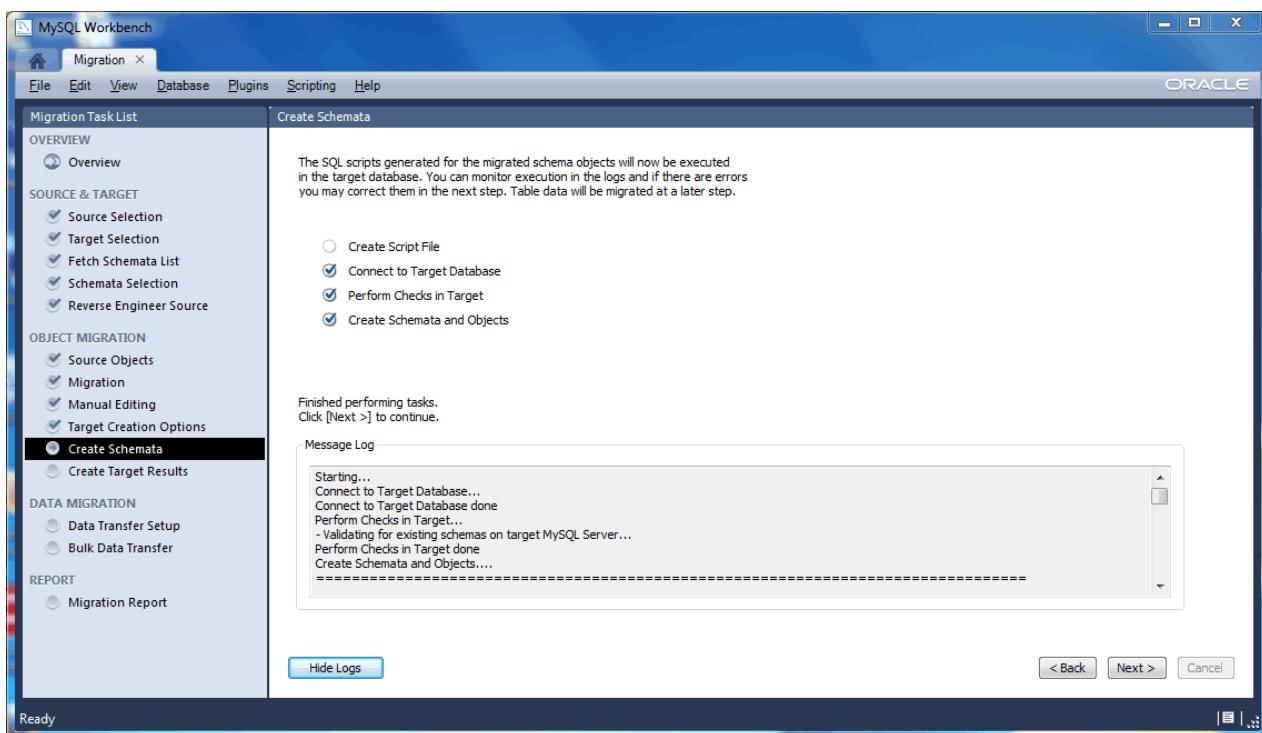
Figure 12.12. MySQL Workbench migration: Target Creation Options



Create Schemata

Now the schemata is created. The complete log is also available here.

Figure 12.13. MySQL Workbench migration: Create Schemata



Create Target Results

The generated objects are listed here, along with the error messages if any exist.

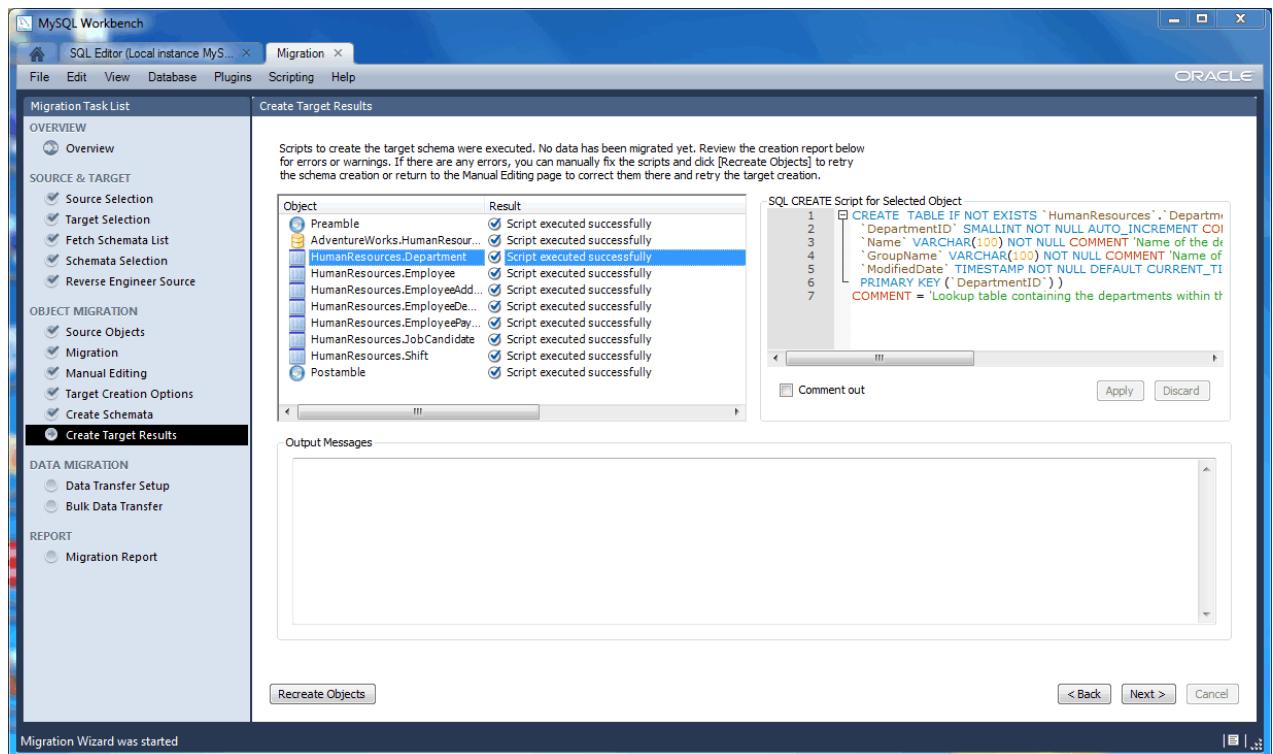
The migration code may also be viewed and edited here. To make changes, select an object, edit the query code, and press **Apply**. Repeat this process for each object that will be edited. And then, press **Recreate Objects** to save the results.



Note

The **Recreate Objects** operation is required to save any changes here. It will then execute the previous migration step (**Create Schemata**) with the modified code, and then continue the migration process. This also means that the previously saved schema will be dropped.

Figure 12.14. MySQL Workbench migration: Create Target Results



Data Transfer Setup

The next steps involve transferring data from the source RDBMS to the target MySQL database. The setup screen includes the following options:

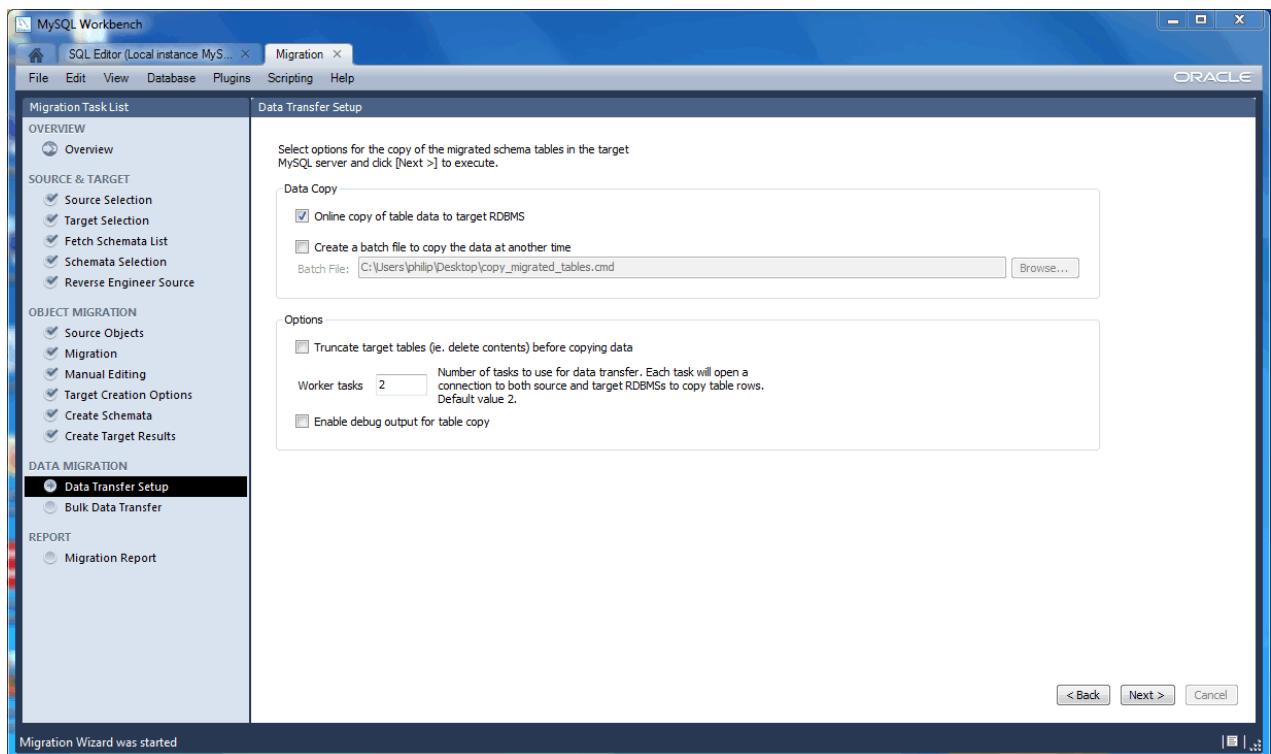
Data Copy:

- Online copy of table data to target RDBMS: This (default) will copy the data to the target RDBMS.
- Create a batch file to copy the data at another time: The data may also be dumped to a file that can be executed at a later time, or be used as a backup.

Options:

- Truncate target tables before copying data: In case the target database already exists, this will delete said data.
- Worker tasks: The default value is 2. This is the number of tasks (database connections) used while copying the data.
- Enable debug output for table copy: Shows debugging information.

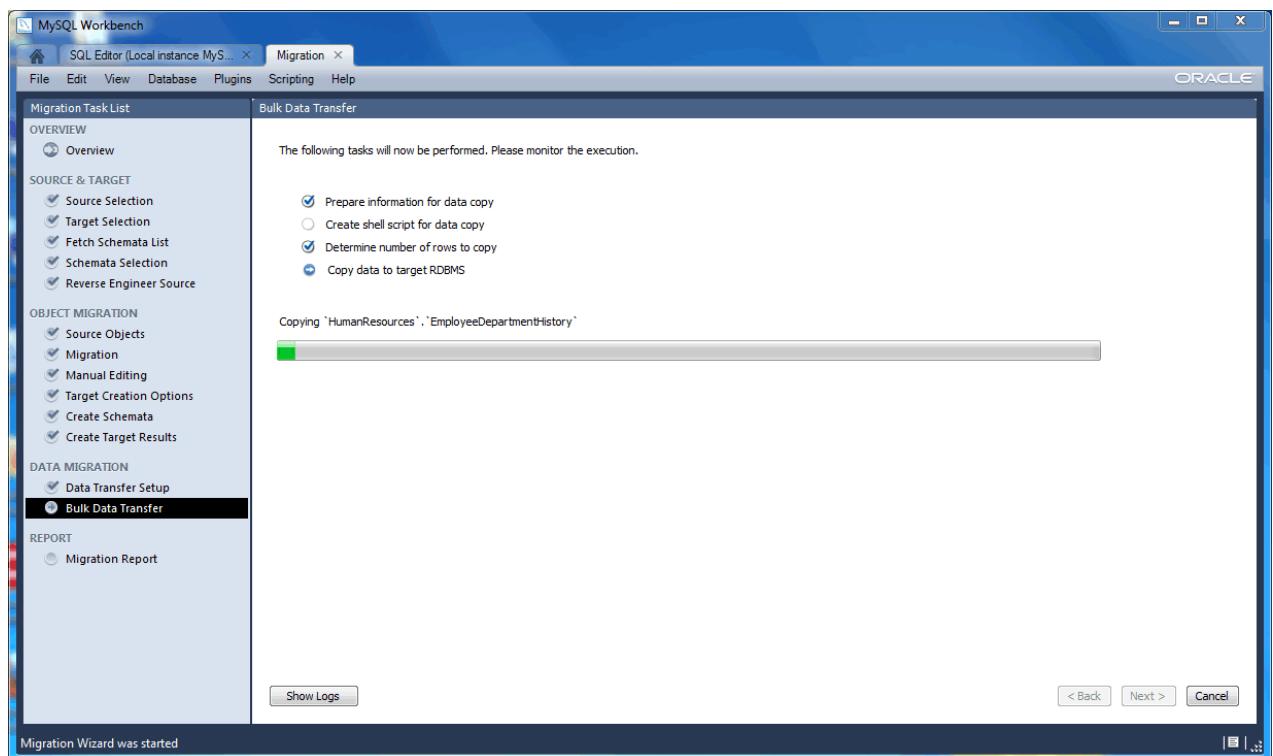
Figure 12.15. MySQL Workbench migration: Data Transfer Setup



Bulk Data Transfer

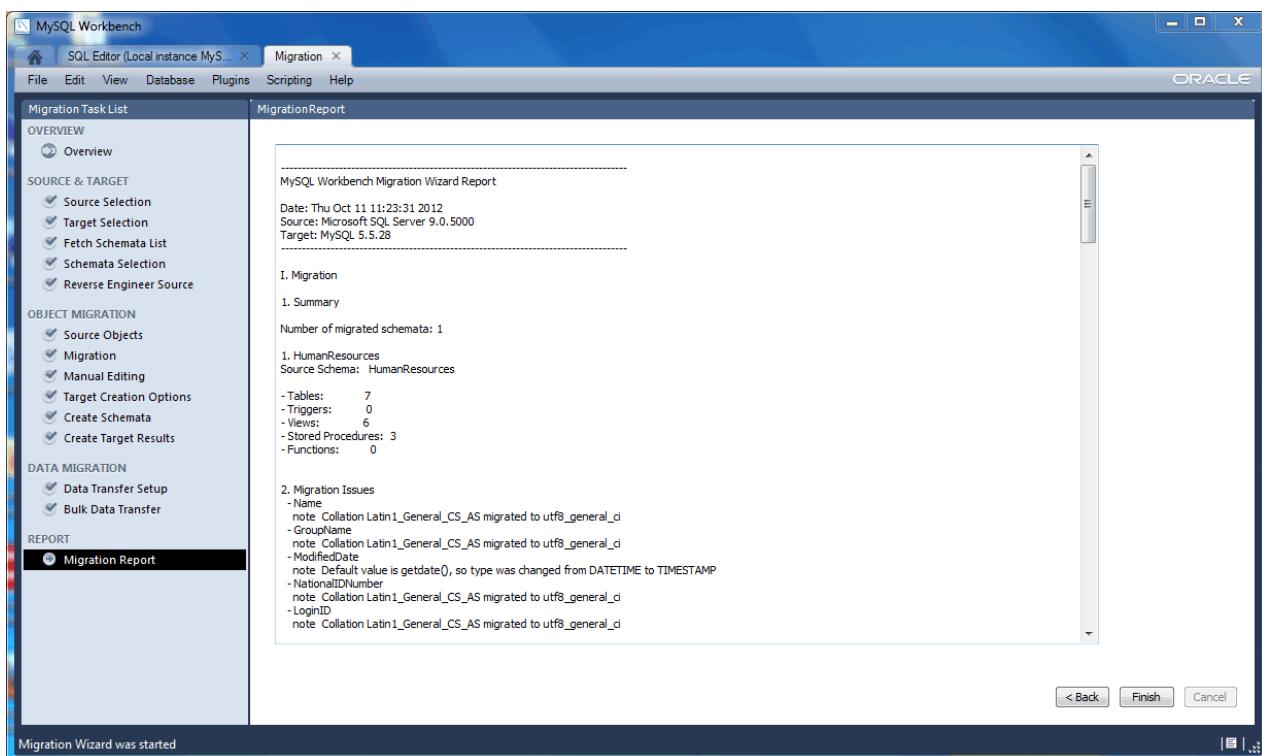
And now the data is transferred to the target RDBMS. Optionally, view the logs to confirm.

Figure 12.16. MySQL Workbench migration: Bulk Data Transfer



Migration Report

And finally, the migration report is available and summarizes the entire migration process.

Figure 12.17. MySQL Workbench migration: Migration Report

Pressing Finish will close the migration window. The database may now be viewed within the MySQL Workbench SQL editor.



Note

If a MySQL Workbench SQL Editor tab is already opened, then the schema list within the Object Browser must be refreshed in order to view the newly imported schema.

12.2.2. Migrating from supported databases

When a supported RDBMS product is being migrated, the MySQL Workbench Migration Wizard will automatically convert as much information as it can, but you may still be required to manually edit the automatically migrated schema for difficult cases, or when the default mapping is not as desired.

Generally speaking, only table information and its data are automatically converted to MySQL. Code objects such as views, stored procedures, and triggers, are not. But supported RDBMS products will be retrieved and displayed in the wizard. You can then manually convert them, or save them for converting at a later time.

The following RDBMS products and versions are currently tested and supported by the MySQL Workbench Migration Wizard, although other RDBMS products can also be migrated with [Section 12.2.3, “Migrating from unsupported \(generic\) databases”](#)

- Microsoft SQL Server 2000
- Microsoft SQL Server 2005
- Microsoft SQL Server 2008

- Microsoft SQL Server 2012
- MySQL Server 4.1 and greater as the source, and MySQL Server 5.1 and greater as the target
- PostgreSQL 8.0 and greater
- Sybase Adaptive Server Enterprise 15.x and greater

12.2.3. Migrating from unsupported (generic) databases

Most ODBC compliant databases may be migrated using the generic database support. In this case, code objects will not be retrieved from the source database; only tables and data.

When using the generic support, column datatypes are mapped using the following steps:

1. It searches for the first entry in the [Generic Datatype Mapping Table](#) for the source type name. If the length/scale ranges of the entry matches the source column, it will pick that type. Otherwise, it continues searching.
2. If no matches were found in the generic table, then it tries to directly map the source type to a MySQL type of the same name.
3. If the source type name doesn't match any of the MySQL datatypes, then it will not be converted and an error is logged. You can then manually specify the target datatype in the Manual Object Editing step of the wizard.

12.3. Conceptual DBMS equivalents

Table 12.1. Conceptual equivalents between supported DBMS products and MySQL

Concept	MS SQL Server	Sybase ASE	PostgreSQL	MySQL	Note
Authentication	Yes	Yes	Yes	Yes	
Auto_Increment	Yes	Yes	Yes	Yes	PostgreSQL uses sequences for Auto_Increment.
Backup	Yes	Yes	Yes	Yes	See MySQL Enterprise Backup
Catalog	Yes	Yes	Yes	N/A	You can map a catalog into a schema and drop the , use the owner as the schema name or merge the owner and object name together. ownerobject
Constraints	Yes	Yes	Yes	Yes	
Data Dictionary				N/A	
Database	Yes	Yes	Yes	Yes	
Database Instance					
Dump	Yes	Yes	Yes	Yes	mysqldump
Events	Yes	Yes	Yes	Yes	
Foreign Keys	Yes	Yes	Yes	Yes	
Full Text Search	Yes	Yes	Yes	Yes	In InnoDB as of MySQL Server 5.6, and in all versions of MyISAM
Index	Yes	Yes	Yes	Yes	

Concept	MS SQL Server	Sybase ASE	PostgreSQL	MySQL	Note
Information Schema	Yes	No	Yes	Yes	
Object Names Case Sensitivity	Depends on collation	Depends on collation	Mixed	Mixed	MySQL: sensitivity of database, table, and trigger names OS dependent; other object names are case insensitive. PostgreSQL: as specified in the SQL-99 standard, unquoted object names are treated as case insensitive while quoted object names are case sensitive. Unlike the standard, unquoted object names are converted to lowercase instead of uppercase.
Object Naming Conventions	Yes	Yes	Yes	Yes	
Packages	N/A	N/A	N/A	N/A	
Partitioning	Yes	Yes	Yes	Yes	
Performance Schema	N/A	N/A	Yes	Yes	
Permissions	Yes	Yes	Yes	Yes	
Primary Key	Yes	Yes	Yes	Yes	
Referential Integrity	Yes	Yes	Yes	Yes	Sybase ASE: referential integrity only through triggers.
Replication	Yes	Yes	Yes	Yes	
Role	Yes	Yes	Yes	N/A	Roles are not available in MySQL at the database level.
Schema	Yes	Yes*	Yes	Yes	Equivalent to database in MySQL. Sybase ASE: Schemata corresponds to user names.
Sequences	Yes*	Yes*	Yes	Yes*	Standalone sequence objects are not supported in MySQL. Similar functionality can be obtained with IDENTITY columns in MSSQL and AUTO_INCREMENT columns in MySQL
SQL Modes	Yes		Yes	Yes	SET_ANSI_* in MSSQL
Storage Engines	N/A	N/A	Yes*	Yes	PostgreSQL itself supports and uses only one storage engine (Postgresql). Other companies have added extra storage engines to PostgreSQL.
Stored Procedures	Yes	Yes	Yes	Yes	
Synonyms	N/A	N/A	N/A	N/A	
Table	Yes	Yes	Yes	Yes	
Tablespace	Yes	Yes*	Yes	N/A	MSSQL groups tables in schemata (unless referring to CREATE

Concept	MS SQL Server	Sybase ASE	PostgreSQL	MySQL	Note
					TABLESPACE). Sybase ASE: tables are grouped in schemata which are more like user names.
Temporary Tables	Yes	Yes	Yes	Yes	
Transactions	Yes	Yes	Yes	Yes	
Triggers	Yes	Yes	Yes	Yes	
UDFs	Yes	Yes	Yes	Yes	
Unicode	Yes	Yes	Yes	Yes	
Unique Key	Yes	Yes	Yes	Yes	
User	Yes	Yes	Yes	Yes	
Views	Yes	Yes	Yes	Yes	



Handling Microsoft SQL Server and MySQL structural differences

A Microsoft SQL Server database is made up of one catalog and one or more schemata. MySQL only supports one schema for each database (or rather, a MySQL database is a schema) so this difference in design must be planned for. The Migration Wizard must know how to handle the migration of schemata for the source (Microsoft SQL Server) database. It can either keep all of the schemata as they are (the Migration Wizard will create one database per schema), or merge them into a single MySQL database. Additional configure options include: either remove the schema names (the Migration Wizard will handle the possible name collisions that may appear along the way), and an option to add the schema name to the database object names as a prefix.

12.4. Microsoft SQL Server migration

Introduction.

The MySQL Workbench Migration Wizard is tested against the following Microsoft SQL Server versions: 2000, 2005, 2008, and 2012.

12.4.1. Preparations

To be able to migrate from Microsoft SQL Server, ensure the following:

- The source SQL Server instance is running, and accepts TCP connections
- You know the IP and port of the source SQL server instance. If you will be migrating using a Microsoft ODBC driver for SQL Server (the default in Windows), you will need to know the host and the name of the SQL Server instance.
- Make sure that the SQL Server is reachable from where you will be running MySQL Workbench. More specifically, check the firewall settings.
- Make sure that the account you will use has proper privileges to the database that will be migrated.

12.4.2. Drivers

General thoughts on the topic.

12.4.2.1. Windows

Microsoft Windows XP or newer includes an ODBC driver for Microsoft SQL Server, so there are no additional actions required.

12.4.2.2. Linux

Setting up drivers on Linux.

FreeTDS

FreeTDS version 0.92 or greater is required. Note that many distributions ship older versions of FreeTDS, so it may need to be installed separately. Additionally, the FreeTDS version provided by distributions may also be compiled for the wrong ODBC library (usually to unixODBC instead of iODBC, which MySQL Workbench uses). Because of that you will probably need to build this library yourself.



Important: using FreeTDS with iODBC

When compiling FreeTDS for use with iODBC (the default with the official binaries), you must compile it with the `--enable-odbc-wide` command line option for the `configure` script. Failing to do so will result in crashes and other unpredictable errors.

A script is provided to compile FreeTDS using the options required for MySQL Workbench. You can find it in the `/usr/share/mysql-workbench/extras/build_freetds.sh` directory in Linux or `MySQLWorkbench.app/Contents/SharedSupport/build_freetds.sh` folder in the Mac. To use it, follow these steps:

1. Make sure you have the iODBC headers installed. In Linux, install the `libiodbc-devel` or `libiodbc2-dev` package from your distribution. In Mac OS X, the headers come with the system and no additional action is required for this step.
2. `mkdir ~/freetds` to create a directory - within the users home directory.
3. Copy the `build_freetds.sh` script to `~/freetds`
4. Get the latest FreeTDS sources from <ftp://ftp.freetds.org/pub/freetds/> and place it in the `~/freetds` directory. Make sure to get version 0.92 or newer.
5. `cd ~/freetds`
6. Execute `build_freetds.sh`
7. After compilation is done, install it using `make install` from the path given by the script.
8. Install the driver in the ODBC Administrator, to make the ODBC subsystem to recognize it. The name of the driver file is `libtdsodbc.so` and is located in `/usr/lib` or `/usr/local/lib`

Once the driver is installed, you should be able to create data sources for it from the ODBC Administrator GUI.



Protocol version selection in FreeTDS

When using FreeTDS, `TDS_VERSION=7.0` is needed in the connection string. If you pick a FreeTDS specific connection method option in the connection dialog, that option is added to the connection string automatically.

12.4.2.3. Mac OS X

See the FreeTDS setup notes for Linux.

12.4.3. Connection Setup

Using an ODBC DataSource

Using Connection Parameters

12.4.4. Microsoft SQL Server Type Mapping

Table 12.2. Type mapping

Source Type	MySQL Type	Comment
INT	INT	
TINYINT	TINYINT	UNSIGNED flag set in MySQL
SMALLINT	SMALLINT	
BIGINT	BIGINT	
BIT	TINYINT(1)	
FLOAT	FLOAT	Precision value is used for storage size in both
REAL	FLOAT	
NUMERIC	DECIMAL	
DECIMAL	DECIMAL	
MONEY	DECIMAL	
SMALLMONEY	DECIMAL	
CHAR	CHAR/LONGTEXT	Depending on its length. MySQL Server 5.5 and above can have CHAR columns with a length up to 255 characters. Anything larger is migrated as LONGTEXT
NCHAR	CHAR/LONGTEXT	Depending on its length. MySQL Server 5.5 and above can have VARCHAR columns with a length up to 65535 characters. Anything larger is migrated to one of the TEXT blob types. In MySQL, character set of strings depend on the column character set instead of the datatype.
VARCHAR	VARCHAR/MEDIUMTEXT/LONGTEXT	Depending on its length. MySQL Server 5.5 and above can have VARCHAR columns with a length up to 65535 characters. Anything larger is migrated to one of the TEXT blob types.
NVARCHAR	VARCHAR/MEDIUMTEXT/LONGTEXT	Depending on its length. MySQL Server 5.5 and above can have VARCHAR columns with a length up to 65535 characters. Anything larger is migrated to one of the TEXT blob types. In MySQL, character set of strings depend on the column character set instead of the datatype.
DATE	DATE	
DATETIME	DATETIME	

Source Type	MySQL Type	Comment
DATETIME2	DATETIME	Date range in MySQL is '1000-01-01 00:00:00' to '9999-12-31 23:59:59'. Note: fractional second values are only stored as of MySQL Server 5.6.4
SMALLDATETIME	DATETIME	
DATETIMEOFFSET	DATETIME	
TIME	TIME	
TIMESTAMP	TIMESTAMP	
ROWVERSION	TIMESTAMP	
BINARY	BINARY/MEDIUMBLOB/LONGBLOB	Depending on its length
VARBINARY	VARBINARY/MEDIUMBLOB/LONGBLOB	Depending on its length
TEXT	VARCHAR/MEDIUMTEXT/LONGTEXT	Depending on its length
NTEXT	VARCHAR/MEDIUMTEXT/LONGTEXT	Depending on its length
IMAGE	TINYBLOB/MEDIUMBLOB/LONGBLOB	Depending on its length
SQL_VARIANT	not migrated	There is not specific support for this datatype.
TABLE	not migrated	There is not specific support for this datatype.
HIERARCHYID	not migrated	There is not specific support for this datatype.
UNIQUEIDENTIFIER	VARCHAR(64)	A unique flag set in MySQL. There is not specific support for inserting unique identifier values.
SYSNAME	VARCHAR(160)	
XML	TEXT	

12.5. PostgreSQL migration

Native support for PostgreSQL 8.x and 9.x was added in MySQL Workbench 5.2.44. MySQL Workbench versions prior to this would migrate PostgreSQL using the generic migration support.

12.5.1. Preparations

Before proceeding, you will need the following:

- Follow the installation guide for installing iODBC on your system. For more information, see [Section 12.1, “General installation requirements”](#).
- Access to a running PostgreSQL instance with privileges to the database you want to migrate, otherwise known as the “source database.” The Migration Wizard officially supports PostgreSQL 8.0 and above, although older versions may work.

- Access to a running MySQL Server instance with privileges to the database you want to migrate. The Migration Wizard officially supports MySQL 5.0 and above.
- MySQL Workbench 5.2.44 or newer.

12.5.1.1. Microsoft Windows

Download and install the MSI package for psqlODBC. Choose the newest file from <http://www.postgresql.org/ftp/odbc/versions/msi/>, which will be at the bottom of the downloads page. This will install psqlODBC on your system and allow you to migrate from Postgresql to MySQL using MySQL Workbench.

12.5.1.2. Linux

After installing iODBC, proceed to install the PostgreSQL ODBC drivers.

Download the psqlODBC source tarball file from <http://www.postgresql.org/ftp/odbc/versions/src/>. Use the latest version available for download, which will be at the bottom of the downloads page. The file will look similar to `psqlodbc-09.01.0200.tar.gz`. Extract this tarball to a temporary location, open a terminal, and cd into that directory. The installation process is:

```
shell> cd the/src/directory
shell> ./configure --with-iodbc --enable-pthreads
shell> make
shell> sudo make install
```

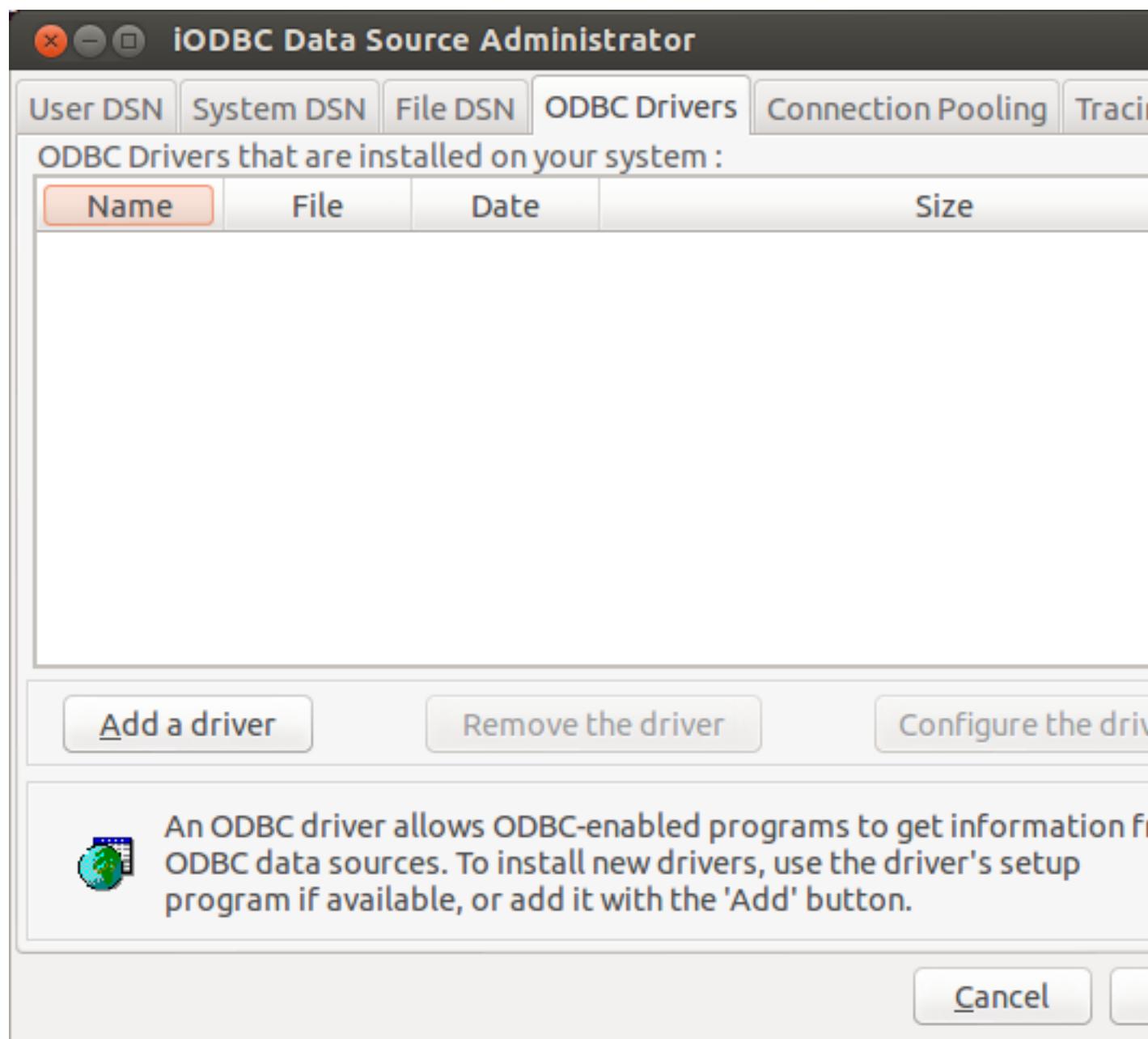
Verify the installation by confirming that the file `psqlodbcw.so` is in the `/usr/local/lib` directory.

Next, you must register your new ODBC Driver.

Open the iODBC Data Source Administrator application by either executing `iodbcadm-gtk` in the command-line, or by launching it from the **Overview** page of the MySQL Workbench Migration Wizard by clicking the Open ODBC Administrator button.

Go to the **ODBC Drivers** tab in the iODBC Data Source Administrator. It should look similar to:

Figure 12.18. The iODBC Data Source Administrator



Click **Add a driver** then fill out the form with the following values:

- **Description of the driver:** psqlODBC
- **Driver file name:** /usr/local/lib/psqlodbcw.so
- **Setup file name:** No value is needed here

And lastly, clicking **OK** will complete the **psqlODBC** driver registration.

12.5.1.3. Mac OS X

To compile `psqlODBC` on Mac OS X, you will need to have Xcode and its "Command Line Tools" component installed on your system, as this includes the required `gcc` compiler. Xcode is free, and available from the AppStore. And after installing Xcode, open it and go to Preferences, **Downloads**, **Components**, and then install the "Command Line Tools" component.

Download the psqlODBC source tarball file from <http://www.postgresql.org/ftp/odbc/versions/src/>. Use the latest version available for download, which will be at the bottom of the downloads page. The file will look similar to `psqlodbc-09.01.0200.tar.gz`. Extract this tarball to a temporary location, open a terminal, and cd into that directory. The installation process is:

```
shell> cd the/src/directory
shell> ./configure --with-iodbc --enable-pthreads
shell> CFLAGS="-arch i386 -arch x86_64" make
shell> sudo make install
```

12.5.2. Drivers

If you are compiling psqlodb, first configure with the `--without-libpq` option.

12.5.3. Connection Setup

After loading the Migration Wizard, click on the Start Migration button in the **Overview** page to begin the migration process. You will first connect to the source PostgreSQL database. Here you will provide the information about the PostgreSQL RDBMS that you are migrating from, the ODBC driver that will be used for the migration, and all of the parameters required for the connection. The name of the ODBC driver is the one you set up when you registered your psqlODBC driver with the driver manager.

Opening the **Database System** dropdown will reveal each RDBMS that is supported on your system. Select PostgreSQL from the list. Below that is the **Stored Connection** dropdown, which is optional. Stored connections will be listed here, which are connections saved after defining a connection with the **Store connection for future use as** checkbox enabled.

The three **Connection Method** options are:

- `ODBC (manually entered parameters)`: Each parameter, like a username, is defined separately
- `ODBC Data Source`: For pre-configured data sources (DSN)
- `ODBC (direct connection string)`: A full ODBC connection string



Note

The psqlODBC driver does not allow a connection without specifying a database name.

The migration process is similar to other databases. See [Section 12.5.4, “PostgreSQL Type Mapping”](#) for information on how the migration wizard migrates types from PostgreSQL to MySQL, and [Section 12.2.1, “A visual guide to performing a database migration”](#) for a general migration guide.

12.5.4. PostgreSQL Type Mapping

Table 12.3. Type mapping

Source Type	MySQL Type	Comment
INT	INT	

Source Type	MySQL Type	Comment
SMALLINT	SMALLINT	
BIGINT	BIGINT	
SERIAL	INT	Sets AUTO_INCREMENT in its table definition.
SMALLSERIAL	SMALLINT	Sets AUTO_INCREMENT in its table definition.
BIGSERIAL	BIGINT	Sets AUTO_INCREMENT in its table definition.
BIT	BIT	
BOOLEAN	TINYINT(1)	
REAL	FLOAT	
DOUBLE PRECISION	DOUBLE	
NUMERIC	DECIMAL	
DECIMAL	DECIMAL	
MONEY	DECIMAL(19,2)	
CHAR	CHAR/LONGTEXT	Depending on its length. MySQL Server 5.5 and above can have CHAR columns with a length up to 255 characters. Anything larger is migrated as LONGTEXT
NATIONAL CHARACTER	CHAR/LONGTEXT	Depending on its length. MySQL Server 5.5 and above can have VARCHAR columns with a length up to 65535 characters. Anything larger is migrated to one of the TEXT blob types. In MySQL, character set of strings depend on the column character set instead of the datatype.
VARCHAR	VARCHAR/ MEDIUMTEXT/ LONGTEXT	Depending on its length. MySQL Server 5.5 and above can have VARCHAR columns with a length up to 65535 characters. Anything larger is migrated to one of the TEXT blob types.
NATIONAL CHARACTER VARYING	VARCHAR/ MEDIUMTEXT/ LONGTEXT	Depending on its length. MySQL Server 5.5 and above can have VARCHAR columns with a length up to 65535 characters. Anything larger is migrated to one of the TEXT blob types. In MySQL, character set of strings depend on the column character set instead of the datatype.
DATE	DATE	
TIME	TIME	
TIMESTAMP	DATETIME	
INTERVAL	TIME	
BYTEA	LONGBLOB	
TEXT	LONGTEXT	
CIDR	VARCHAR(43)	
INET	VARCHAR(43)	
MACADDR	VARCHAR(17)	
UUID	VARCHAR(36)	

Source Type	MySQL Type	Comment
XML	LONGTEXT	
JSON	LONGTEXT	
TSVECTOR	LONGTEXT	
TSQUERY	LONGTEXT	
ARRAY	LONGTEXT	
POINT	VARCHAR	
LINE	VARCHAR	
LSEG	VARCHAR	
BOX	VARCHAR	
PATH	VARCHAR	
POLYGON	VARCHAR	
CIRCLE	VARCHAR	
TXID_SNAPSHOT	VARCHAR	

12.6. MySQL migration

Introduction

Notes about copying MySQL, and what you can do with it.

12.7. Using the MySQL Workbench Migration Wizard

Introduction, and general usage notes.

12.7.1. Connecting to the databases

A connection is made to the source and target database servers.

Source Connection Setup

The Source Connection offers the [MySQL](#), [Microsoft SQL Server](#), and [Generic RDBMS](#) database system options. This selection determines the available [Parameters](#) and [Advanced](#) configuration options.

This connection definition may be saved using the [Store connection for future use as](#) option, and there is also the [Test Connection](#) option.

Target Connection Setup

The MySQL Server that will be home to the newly migrated database.

12.7.2. Schemata Retrieval and Selection

General thoughts.

Fetch Schemata List

The names of available schemas will be retrieved from the source RDBMS. The account used for the connection will need to have appropriate privileges for listing and reading the schemas you want to migrate. Target RDBMS connection settings will also be validated.

The steps that are performed include: connects to the source DBMS, checks the connection, and retrieves the schema list from the source.

Schemata Selection

Select the schemata that you want to migrate.

12.7.3. Reverse Engineering

This is an automated step, where the actions include: Connect to the source DBMS, Reverse engineer the selected schemata, and perform post-processing if needed.

12.7.4. Object Selection

By default, all table objects will be migrated. Use the Show Selection button in order to disable individual table objects from being migrated.

12.7.5. Migration

Reverse engineered objects from the source RDBMS will be automatically converted to MySQL compatible objects. Default datatype and default column value mappings will be used. You will be able to review and edit the generated objects and column definitions in the next step, which is [Section 12.7.6, “Manual Editing”](#).

The steps performed include Migrating the selected objects, and generating the SQL CREATE statements.

12.7.6. Manual Editing

The migrated objects may be reviewed and edited here. You can manually edit the generated SQL before applying them to the target database. Target schemas and tables may be renamed, and column definitions may be changed, by double-clicking on them.

By default, the All Objects View is loaded. Other View options include Migration Problems and Column Mappings.

- All Objects: Shows all objects, which can also be edited by double-clicking.
- Migration Problem: Will list all of the migration problems, or report that no mapping problems were found.
- Column Mappings: Displays all of the schema columns, which may also be edited. There is an advanced Show Code and Messages option that displays the SQL CREATE script for the selected object.

12.7.7. Target Creation Options

Defines addition settings for the target schema.

Configuration options include:

- Create schema in target RDBMS:
- Create a SQL script file:
- An option to keep the schemata if they already exist. Objects that already exist will not be recreated or update.

12.7.8. Schema Creation

The SQL scripts generated for the migrated schema objects will now be executed in the target database. You can monitor execution in the logs, if errors exist then they will be fixed in the next step. Table data will be migrated in a later step as well.

This is an automated step, where the actions include: Create Script File, Connect to Target Database, and Create Schemata and Objects.

12.7.9. Create Target Results

Scripts to create the target schemas were executed, but the data has not yet been migrated. This step allows reviewing a creation report. If there are any errors, then you can manually fix the scripts and click **Recreate Objects** to retry the schema creation or return to the [Manual Editing](#) page to correct them there, and then retry the target creation.

To edit, first select the object, and then the SQL CREATE Script will be shown for the selected object. Edit it there, then press **Apply** to save.

12.7.10. Data Migration Setup

Provides additional options for data transfer, including the ability to set up a script to automate this transfer in the future.

12.7.11. Bulk Data Transfer

The transfer is executed here.

12.7.12. Migration Report

Displays the final report, that can be reviewed to ensure a proper migration was executed.

12.8. MySQL Workbench Migration Wizard FAQ

Frequently Asked Questions with answers.

Questions

- **12.8.1: [251]** While using the Postgresql psqlodbc driver, I see the following error: ('08001', '[08001] Already connected. (202) (SQLDriverConnect)')

Questions and Answers

12.8.1: While using the Postgresql psqlodbc driver, I see the following error: ('08001', '[08001] Already connected. (202) (SQLDriverConnect)')

This means that PostgreSQL is not configured to accept connections from the source IP.

Chapter 13. Extending Workbench

Table of Contents

13.1. GRT and Workbench Data Organization	253
13.2. Modules	254
13.3. Plugins / Tools	255
13.4. Adding a GUI to a Plugin Using MForms	256
13.5. The Workbench Scripting Shell	257
13.5.1. Exploring the Workbench Scripting Shell	257
13.5.2. The Shell Window	258
13.5.3. The Files, Globals, Classes, and Modules Tabs	259
13.6. Tutorial: Writing Plugins	260

MySQL Workbench provides an extension and scripting system that enables the developer to extend MySQL Workbench capabilities. While the core of MySQL Workbench is developed using C++, it is possible to harness this core functionality using both the Lua and Python scripting languages. MySQL Workbench also provides access to a cross-platform GUI library, MForms, which enables the creation of extensions that feature a graphical user interface.

The extension system enables the following capabilities:

- Automate common tasks
- Extend the Workbench user-interface
- Create Tools/Plugins (code which can be invoked from the Workbench menu system)
- Manipulate schemata
- Create custom Workbench features

13.1. GRT and Workbench Data Organization

The GRT, or Generic RunTime, is the internal system used by Workbench to hold model document data. It is also the mechanism by which Workbench can interact with Modules and Plugins. Workbench model data, such as diagrams, schemata, and tables, is stored in a hierarchy of objects that can be accessed by any plugin. The information is represented using standard data types: integers, doubles, strings, dicts, lists, and objects.

The GRT can be accessed using external scripting languages such as Lua and Python. Awareness is required of how the GRT data types map into the scripting language. In Python, for example, the GRT integer, double, and string data types are seen as corresponding Python data types. Lists and dicts are kept in their internal representation, but can generally be treated as Python lists and dicts, and accessed in the usual way. Objects contain data fields and methods, but the GRT recognizes only objects from a pre-registered class hierarchy.

It is possible to fully examine the classes contained within the GRT using the Workbench Scripting Shell. Dots in class names are changed to underscores in their Python counterparts. For example, `db.mysql.Table` becomes `db_mysql_Table` in Python.

The Application Objects Tree (GRT Tree)

As mentioned previously, Workbench document data is stored in an object hierarchy. This hierarchy is known as the GRT Tree. The GRT Tree can be accessed and modified from supported external scripting languages such as Python. Care should be taken when modifying the GRT Tree, to prevent a mistake from leading to corruption of the document. Backups should be made before manipulating the tree. Read-only access to the tree is the safest approach, and is sufficient in most cases.

The main nodes in the Application Object Tree

Table 13.1. The main nodes in the Application Object Tree

Node	Description
wb.registry	Application data such as plugin registry, list of editors, and options.
wb.customData	A generic dictionary for data you can use to store your own data. This dictionary is saved and reloaded with Workbench and is global (not document specific).
wb.options	Contains some default options that are used by Workbench.
wb.rdbmsMgmt	Internal registry of supported RDBMS modules, known data types.
wb.doc	The currently loaded model document.
wb.doc.physicalModels[0]	The currently loaded model object, containing the database catalog and diagrams.
wb.doc.physicalModels[0].catalog	The database catalog for the model. Contains the list of schemata.
wb.doc.physicalModels[0].catalog.schemata	List of schemata in the model. Individual schema can be accessed as a list: schemata[0], schemata[1] ...
wb.doc.physicalModels[0].catalog.schemata[0].tables (.views, .routines, ...)	Lists of tables, views, routines in the schema.
wb.doc.physicalModels[0].diagrams	List of EER diagrams in the model.
wb.doc.physicalModels[0].diagrams[0].figures (.layers, .connections, ...)	List of figures, layers, connections (relationships) in the diagram.

13.2. Modules

In the GRT Modules are libraries containing a list of functions that are exported for use by code in other modules, scripts, or Workbench itself. Modules can currently be written in C++, Lua, or Python, but the data types used for arguments and the return value must be GRT types.

GRT modules are similar to Python modules, but are imported from the built-in `grt` module, instead of directly from an external file. The list of modules loaded into the `grt` module is obtained from `grt.modules`. Modules can be imported in Python using statements such as `from grt.modules import WbModel`.

To export functions as a module from Python code, you must carry out the following steps:

1. The source file must be located in the user modules folder. This path is displayed in the Workbench Scripting Shell with the label **Looking for user plugins in...**. It is also possible to install the file using the main menu item Scripting, Install Plugin/Module File.

2. The source file name must have the extension `_grt.py`; for example, `my_module_grt.py`.
3. Some module metadata must be defined. This can be done using the `DefineModule` function from the `wb` module:

```
from wb import *
ModuleInfo = DefineModule(name='MyModule', author='Your Name', version='1.0')
```

4. Functions to be exported require their signature to be declared. This is achieved using the `export` decorator in the previously created `ModuleInfo` object:

```
@ModuleInfo.export(grt.INT, grt.STRING)
def checkString(s):
    ...
```

For the `export` statement, the return type is listed first, followed by the input parameter types, specified as GRT typenames. The following typenames can be used:

- `grt.INT`: An integer value. Also used for boolean values.
- `grt.DOUBLE`: A floating-point numeric value.
- `grt.STRING`: UTF-8 or ASCII string data.
- `grt.DICT`: A key/value dictionary item. Keys must be strings.
- `grt.LIST`: A list of other values. It is possible to specify the type of the contents as a tuple in the form (`grt.LIST, <type-or-class>`). For example, (`grt.LIST, grt.STRING`) for a list of strings. For a list of table objects, the following would be specified: (`grt.LIST, grt.classes.db_table`).
- `grt.OBJECT`: An instance of a GRT object or a GRT class object, from `grt.classes`.

Note that these types are defined in the `grt` module, which must be imported before they can be used.

The following code snippet illustrates declaring a module that exports a single function:

```
from wb import *
import grt

ModuleInfo = DefineModule(name='MyModule', author="your name", version='1.0')

@ModuleInfo.export(grt.DOUBLE, grt.STRING, (grt.LIST, grt.DOUBLE))
def printListSum(message, doubleList):
    sum = 0
    for d in doubleList:
        sum = sum + d
    print message, sum
    return sum
```

13.3. Plugins / Tools

Plugins are special Modules that are exposed to the user through the Workbench GUI. This is typically done using the main menu, or the context-sensitive menu. Much of the MySQL Workbench functionality is implemented using plugins; for example, table, view, and routine editors are native C++ plugins, as are the forward and reverse engineering wizards. The Administrator facility in MySQL Workbench is implemented entirely as a plugin in Python.

A plugin can be a simple function that performs some action on an input, and ends without further interaction with the user. Examples of this include auto-arranging a diagram, or making batch changes to objects. To create a simple plugin, the function must be located in a module and declared as a plugin using the `plugin` decorator of the `ModuleInfo` object.

Plugins can have an indefinite runtime, such as when they are driven by the user through a graphical user interface. This is the case for the object editors and wizards within MySQL Workbench. Although the wizard type of plugin must be declared in the usual way, only the entry point of the plugin will need to be executed in the plugin function, as most of the additional functionality will be invoked as a result of the user interacting with the GUI.

**Note**

Reloading a plugin requires MySQL Workbench to be restarted.

Declare a plugin using this syntax:

```
@ModuleInfo.plugin(plugin_name, caption, [input], [groups], [pluginMenu])
```

These parameters are defined as follows:

- **plugin_name**: A unique name for the plugin. It may contain only alphanumeric characters, dots, and underscores.
- **caption**: A caption to use for the plugin in menus.
- **input**: An optional list of input arguments.
- **groups**: Optional list of groups the plugin belongs to. Recognized values are:
 - `Overview/Utility`: The [Context](#) menu in the Model Overview.
 - `Model/Utility`: The menu for diagram objects.
 - `Menu/<category>`: The [Plugins](#) menu in the main menu.
- **pluginMenu**: Optional name of a submenu in the Plugins menu where the plugin should appear. For example, Catalog, Objects, Utilities. This is equivalent to adding a `Menu/<category>` in the groups list.

13.4. Adding a GUI to a Plugin Using MForms

MySQL Workbench is implemented with a C++ core back-end, and a native front-end for each supported platform. Currently the front-end is implemented with Windows Forms on Microsoft Windows, GTK+ on Linux, and Cocoa on Mac OS X. This approach permits the application to have a native look and feel, while reducing the amount of work required to maintain the project. However, the GUI functionality required by MySQL Workbench can be met by a subset of graphical operations. These are implemented in a cross-platform GUI library, MForms. This further reduces the development effort because plugin developers can use MForms rather than writing front-end specific code for each supported platform. This also helps consistency of operation across all platforms. MForms is coded in C++, but provides a Python interface. To use it, the Python code must import the `mforms` module.

MForms Containers

Given the problems of using an absolute coordinate system across different platforms, MForms employs containers that perform automatic layout. The basic containers that MForms provides include:

- **Form:** A top-level window which can contain a single control, usually another container. The window will be sized automatically to fit its contents, but can also be sized statically.
- **Box:** This is a container that can be filled with one or more controls in a vertical or horizontal layout. Each child control can be set to use either the minimum of required space, or fill the box in the direction of the layout. In the direction perpendicular to the layout, for example vertical in a horizontal layout, the smallest possible size that can accommodate all child controls will be employed. So, in this example, the smallest height possible to accommodate the controls would be used.
- **Table:** This is a container that can organize one or more controls in a grid. The number of rows and columns in the table, and the location of controls within the grid, can be set by the developer.
- **ScrollView:** This is a container that can contain a single child control, and will add scrollbars if the contents do not fit the available space.

13.5. The Workbench Scripting Shell

The Workbench Scripting Shell provides a means for entering and executing scripts. Through the use of the scripting shell, MySQL Workbench can support new behavior and data sources using code written in Lua and Python. The shell can also be used to explore the current Workbench GRT (Generic RunTime) facilities.

The scripting shell is not only useful for expanding MySQL Workbench. You can use a script file from the scripting shell command line to perform repetitive tasks programmatically.

The development language can be either [Python](#) or [Lua](#). The default programming language used in Workbench Scripting Shell is defined in the **General** tab of the MySQL Workbench [Preferences](#) dialog, and defaults to Python.



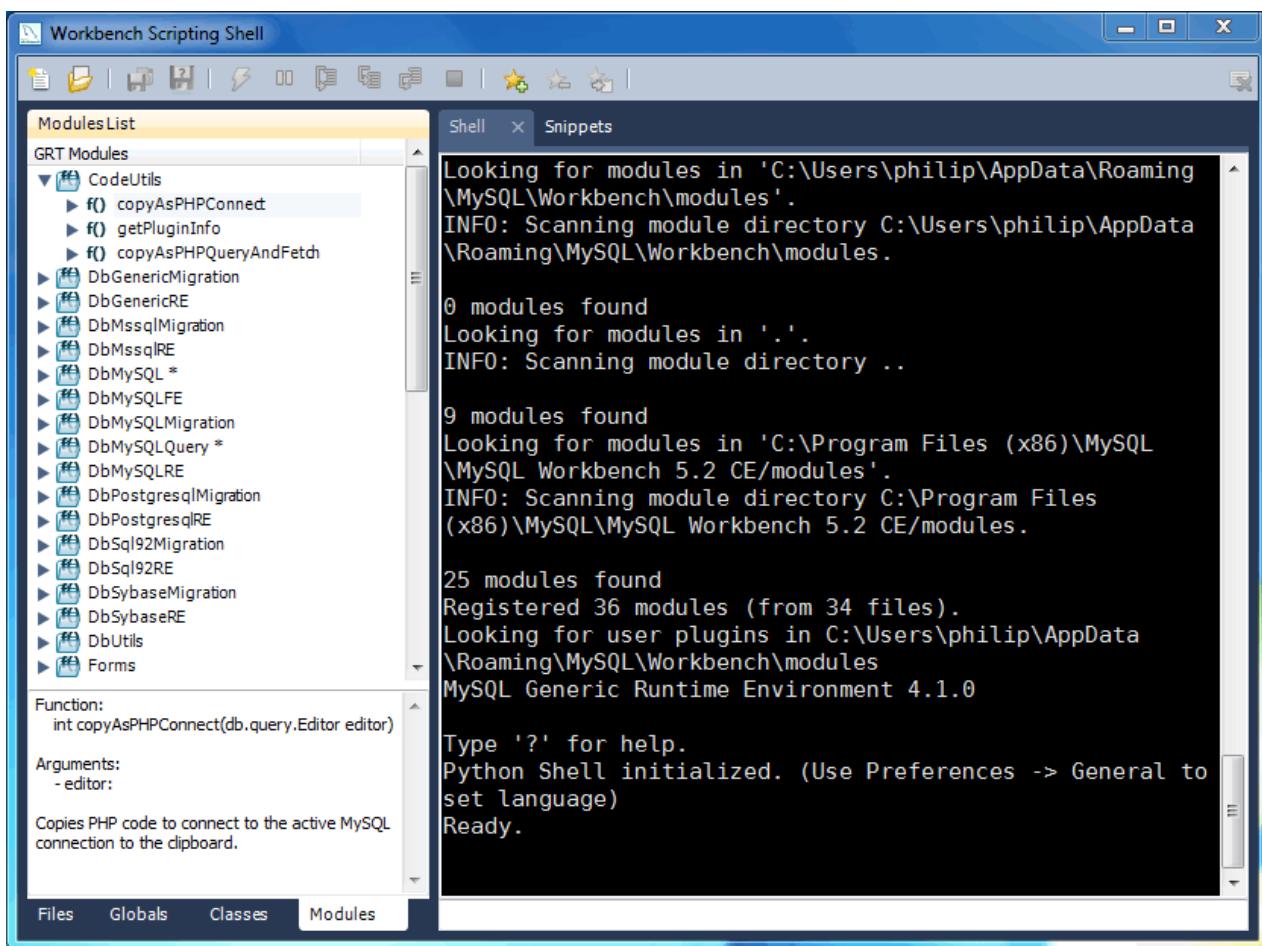
Note

Although they serve a different purpose, the MySQL Utilities are also bundled with MySQL Workbench. For more information, see [Chapter 15, MySQL Utilities](#).

13.5.1. Exploring the Workbench Scripting Shell

To open the Workbench Scripting Shell, select [Scripting](#), Scripting Shell from the main menu. You can also open the Workbench Scripting Shell using the **Control+F3** key combination on Windows and Linux, **Command+F3** on Mac OS X, or by clicking the shell button above the EER diagram navigator. The Workbench Scripting Shell will then open in a new dialog.

The following screenshot shows the Workbench Scripting Shell dialog.

Figure 13.1. The Workbench Scripting Shell

13.5.2. The Shell Window

The Workbench Scripting Shell is primarily used for running Python or Lua scripts, or typing commands in these languages directly. However, you can also use it to access the Workbench Scripting Shell Scripting Library functions and global functions and objects. To see the available commands, type "?". You can also cut and paste text to and from the shell window.

The **Snippets** tab is a scratch pad for saving code snippets. This makes it easy to reuse code and does away with the need to retype it at the command line.

If you have opened script files, each will have its own tab to the right of the **Snippets** tab. These tabs will be labeled with the names of the script files, or **Unnamed** for snippets without a name. As with the **Snippets** tab you can cut and paste to or from any of the tabs. This gives you the opportunity to test code from the command line. Right clicking on a snippet opens a dialog with options to Execute Snippet, Send to Script Editor, or Copy To Clipboard.

While individual commands can be entered into the shell, it is also possible to run a longer script, stored in an external file, using the main menu item Scripting, Run Workbench Script File. When scripts are run outside of the shell, to see the output use the main menu item View, Output.

It is also possible to run script files directly from the shell. For details on running script files, type `? run` at the Workbench Scripting Shell prompt. The following message is displayed:

```
Help Topics
-----
grt      General information about the Workbench runtime
scripting Practical information when working on scripts and modules for Workbench
wbdata   Summary about Workbench model data organization
modules   Information about Workbench module usage
plugins   Information about writing Plugins and Modules for Workbench
Type '? [topic]' to get help on the topic.

Custom Python Modules
-----
grt      Module to work with Workbench runtime (grt) objects
grt.root  The root object in the internal Workbench object hierarchy
grt.modules Location where Workbench modules are available
grt.classes List of classes known to the GRT system
mforms   A Module to access the cross-platform UI toolkit used in some Workbench features
wb       Utility module for creating Workbench plugins

Type 'help(module/object/function)' to get information about a module, object or function.
Type 'dir(object)'                  to get a quick list of methods an object has.

For an introductory tutorial on the Python language, visit http://docs.python.org/tutorial/
For general Python and library reference documentation, visit http://python.org/doc/
```

Within the Workbench Scripting Shell, there are four tabs on the top of the left side panel: **Files**, **Globals**, **Classes**, and **Modules**. Discussion of these additional tabs follows.



Note

An exception is thrown while attempting to use `input()` or read from `stdin`.

13.5.3. The Files, Globals, Classes, and Modules Tabs

The Workbench Scripting Shell features the **Files**, **Globals**, **Classes** and **Modules** tabs, in addition to the main **Shell** tab.

The Files Tab

Lists folders and files for user-defined (custom) script files. The categories are **User Scripts**, **User Modules**, and **User Libraries**.

The Globals Tab

At the top of the window is a list that is used to select the starting point, or root, of the GRT Globals tree displayed beneath it. By default, this starting point is the root of the tree, that is, '/'. You can expand or collapse the GRT Globals tree as desired. The GRT Globals tree is the structure in which MySQL Workbench stores document data. Clicking any item results in its name and value being displayed in the panel below the tree.

The Classes Tab

A **class** is a user-defined data type formed by combining primitive data types: integers, doubles, strings, dicts, lists, and objects. This tab shows the definitions of the classes used by the objects in the **Modules** tab. Clicking a class causes a brief description of the class to be displayed in a panel below the classes explorer.

When the **Classes** tab is selected, the list displays the following items:

- **Group by Name:** Group by the object name

- **Group by Hierarchy**: Group by inheritance
- **Group by Package**: Group by functionality

The default view for this tab is **Group By Name**. This view shows all the different objects arranged alphabetically. Click the + icon or double-click a package to show the properties of the struct.

If you switch to the hierarchical view, you will see [GrtObject](#): the parent object from which all other objects are derived.

The Modules Tab

The **Modules** tab enables you to browse the MySQL Workbench installed modules and their functions. Clicking a module within the explorer causes its details to be displayed in a panel below the explorer. This facility is useful for exploring the available modules, and their supported functions. It is also a way to check whether custom modules have been correctly installed.

13.6. Tutorial: Writing Plugins

This tutorial shows you how to extend MySQL Workbench by creating a plugin.

The Sample Plugin

EER Diagrams are useful for visualizing complex database schemata. They are often created for existing databases, to clarify their purpose or document them. MySQL Workbench provides facilities for reverse engineering existing databases, and then creating an EER Diagram automatically. In this case, relationship lines between foreign keys in the table will automatically be drawn. This graphical representation makes the relationships between the tables much easier to understand. However, one of the most popular storage engines for MySQL, MyISAM, does not include support for foreign keys. This means that MyISAM tables that are reverse engineered will not automatically have the relationship lines drawn between tables, making the database harder to understand. The plugin that will be created in this tutorial gets around this problem by using the fact that a naming convention is very often used for foreign keys: [tablename_primarykeyname](#). Using this convention, foreign keys can automatically be created after a database is reverse engineered, which will result in relationship lines being drawn in the EER diagram.

Algorithm

The basic algorithm for this task would be as follows:

```
for each table in the schema
    for each column in the table
        look for another table whose name and primary key name match the current column name
        if such a table is found, add a foreign key referencing it
```

As iterating the complete table list to find a match can be slow for models with a large number of tables, it is necessary to optimize by pre-computing all possible foreign key names in a given schema.

```
import grt

def auto_create_fks(schema):
    fk_name_format = "%(table)s_%(pk)s"
    possible_fks = []
    # create the list of possible foreign keys from the list of tables
    for table in schema.tables:
        if table.primaryKey:
            format_args = {'table':table.name, 'pk':table.primaryKey.name}
            fkname = fk_name_format % format_args
            possible_fks.append(fkname)
```

```
possible_fks[fkname] = table

# go through all tables in schema, this time to find columns that may be a fk
for table in schema.tables:
    for column in table.columns:
        if possible_fks.has_key(column.name):
            ref_table = possible_fks[column.name]
            if ref_table.primaryKey.formattedType != column.type:
                continue
            fk = table.createForeignKey(column.name+"_fk")
            fk.referencedTable = ref_table
            fk.columns.append(column)
            fk.referencedColumn.append(ref_table.primaryKey)
            print "Created foreign key %s from %s.%s to %s.%s" \
                  % (fk.name, table.name, column.name, ref_table.name, ref_table.primaryKey.name)

auto_create_fks(grt.root.wb.doc.physicalModels[0].catalog.schemata[0])
```

Creating a Plugin from a Script

To create a plugin from an arbitrary script, it is first necessary to make the file a module, and export the required function from it. It is then necessary to declare the module as a plugin, and specify the return type and input arguments.

```
from wb import *
import grt

ModuleInfo = DefineModule(name="AutoFK", author="John Doe", version="1.0")

@ModuleInfo.plugin("sample.createGuessedForeignKeys",
    caption="Create Foreign Keys from ColumnNames",
    input=[wbinputs.objectOfClass("db.mysql.Schema")],
    groups=["Overview/Utility"])

@ModuleInfo.export(grt.INT, grt.classes.db_mysql_Schema)
def auto_create_fks(schema):
    ...
```

With the addition of the preceding code, the `auto_create_fks()` function is exported and will be added to the schema context menu in the model overview. When invoked, it receives the currently selected schema as its input.

Chapter 14. Keyboard Shortcuts

The following tables list keyboard shortcuts for MySQL Workbench commands. **Modifier** in the tables stands for the platform-specific modifier key. This is **Command** on Mac OS X, **Control** on other platforms. On Mac OS X, the **Alt** key is **Option**.

There are keyboard shortcut tables for the [File](#), [Edit](#), [View](#), [Arrange](#), [Model](#), [Query](#), [Database](#), [Scripting](#), [Help](#), and [EER Diagram Mode](#) menus.

File Menu

Table 14.1. File menu keyboard shortcuts

Function	Keyboard Shortcut	Context
New Model	Modifier+N	All
Open Model	Modifier+O	All
Open SQL Script	Modifier+Shift+O	SQL Editor
Close Tab	Modifier+W, Modifier+F4 on Windows	All
Save Model	Modifier+S	Model
Save Script	Modifier+S	SQL Editor
Save Model As	Modifier+Shift+S	Model
Save Script As	Modifier+Shift+S	SQL Editor
Forward Engineer SQL CREATE Script	Modifier+Shift+G	Model
Forward Engineer SQL ALTER Script	Modifier+Alt+Y	Model
Synchronize With SQL CREATE Script	Modifier+Shift+Y	Model
Print	Modifier+P	EER Diagram mode only
Exit	Modifier+Q	All

Edit Menu

Table 14.2. Edit menu keyboard shortcuts

Function	Keyboard Shortcut	Context
Undo	Modifier+Z	Model, EER Diagram
Redo	Modifier+Y, Modifier+Shift+Z (Mac OS X)	Model, EER Diagram
Cut	Modifier+X	All
Copy	Modifier+C	All
Paste	Modifier+V	All
Delete	Modifier+Delete, Command+BackSpace (Mac OS X)	All
Edit Selected	Modifier+E	Model, EER Diagram
Edit Selected in New Window	Modifier+Shift+E	Model, EER Diagram
Select All	Modifier+A	EER Diagram

Function	Keyboard Shortcut	Context
Find	Modifier+F	All
Find Advanced	Modifier+Alt+F	All
Find Next	F3	All
Find Previous	Shift+F3	All
Search and Replace	Modifier+Shift+F	All
Comment/Uncomment lines of SQL	Modifier+ /	SQL Editor
Auto-Complete SQL	Modifier+Space	SQL Editor

View Menu

Table 14.3. View menu keyboard shortcuts

Function	Keyboard Shortcut	Context
Output Window	Modifier+F2, Modifier+Option+2 (Mac OS X)	All
Set Marker n	Modifier+Shift+n (n is integer 1..9)	EER Diagram
Go to Marker n	Modifier+n (n is integer 1..9)	EER Diagram

Arrange Menu

Table 14.4. Arrange menu keyboard shortcuts

Function	Keyboard Shortcut	Context
Bring to Front	Modifier+Shift+F	EER Diagram
Send to Back	Modifier+Shift+B	EER Diagram

Model Menu

Table 14.5. Model menu keyboard shortcuts

Function	Keyboard Shortcut	Context
Add Diagram	Modifier+T	Model, EER Diagram
Validate All	Modifier+Alt+V	Model, EER Diagram
Validate All (MySQL)	Modifier+Alt+B	Model, EER Diagram
Model Options	Command+Alt+, (Shortcut available only on Mac OS X)	Model, EER Diagram

Query Menu

Table 14.6. Query menu keyboard shortcuts

Function	Keyboard Shortcut	Context
Execute statement	Modifier+Return	SQL Editor
Execute statements	Modifier+Shift+Return	SQL Editor
New Tab	Modifier+T	SQL Editor

Database Menu

Table 14.7. Database menu keyboard shortcuts

Function	Keyboard Shortcut	Context
Query Database	Modifier+U	All
Reverse Engineer	Modifier+R	Model, EER Diagram
Forward Engineer	Modifier+G	Model, EER Diagram
Synchronize Model	Modifier+Y	Model, EER Diagram

Scripting Menu

Table 14.8. Scripting menu keyboard shortcuts

Function	Keyboard Shortcut	Context
Scripting Shell	Modifier+F3, Modifier+Option+3 (on Mac OS X)	All
Run Workbench Script File	Modifier+Shift+R	All

Help Menu

Table 14.9. Help menu keyboard shortcuts

Function	Keyboard Shortcut	Context
Help Index	F1, Command+Option+question (on Mac OS X)	All

EER Diagram Mode

In the EER Diagram view, a number of other keyboard shortcuts are available.

Table 14.10. EER diagram mode keyboard shortcuts

Function	Keyboard Shortcut
Selection tool	Escape
Hand tool	H
Delete tool	D
Layer tool	L
Note tool	N
Image tool	I
Table tool	T
View tool	V
Routine Group tool	G
Non-Identifying Relationship 1:1	1
Non-Identifying Relationship 1:n	2
Identifying Relationship 1:1	3
Identifying Relationship 1:n	4
Identifying Relationship n:m	5
Relationship Using Existing Columns	6

Table 14.11. Keyboard shortcut changes

MySQL Workbench version	The Change
5.2.45	The "Modifier+/" shortcut was added to comment/uncomment SQL in the SQL editor
5.2.45	On Microsoft Windows, the "Modifier+W" shortcut was changed to "Control+F4" -- this shortcut closes MySQL Workbench tabs

Chapter 15. MySQL Utilities

Abstract

This is the MySQL™ Utilities Reference Manual. It documents both the GPL and commercial editions through 1.3.5.

If you have not yet installed MySQL Utilities please download your free copy from the [download site](#). MySQL Utilities is available for Windows, Mac OS X, and Linux variants.

For release notes detailing the changes in each release, see the [MySQL Utilities Release Notes](#).

Table of Contents

15.1. The Preface	268
15.2. How to Install MySQL Utilities	268
15.2.1. Prerequisites	269
15.2.2. Source Code	269
15.2.3. Oracle and Red Hat Linux 6	269
15.2.4. Debian Linux	269
15.2.5. Microsoft Windows	270
15.3. Introduction	270
15.3.1. Introduction to MySQL Utilities	270
15.3.2. Connection Parameters	271
15.4. MySQL Utilities Administrative Tasks	271
15.4.1. Database Operations	272
15.4.2. General Operations	278
15.4.3. High Availability Operations	289
15.4.4. Server Operations	301
15.4.5. Specialized Operations	304
15.5. Overview of MySQL Utilities	309
15.5.1. Database Operations	309
15.5.2. General Operations	309
15.5.3. High Availability Operations	310
15.5.4. Server Operations	311
15.5.5. Specialized Operations	311
15.6. Manual Pages	312
15.6.1. mysqlauditadmin	312
15.6.2. mysqlauditgrep	316
15.6.3. mysqldbcompare	324
15.6.4. mysqldbcopy	331
15.6.5. mysqldbexport	336
15.6.6. mysqldbimport	343
15.6.7. mysqldiff	347
15.6.8. mysqldiskusage	351
15.6.9. mysqlfailover	354
15.6.10. mysqlfrm	364
15.6.11. mysqlindexcheck	367
15.6.12. mysqlmetagrep	370
15.6.13. mysqlprocgrep	374
15.6.14. mysqlreplicate	377
15.6.15. mysqlrpladmin	380
15.6.16. mysqlrplcheck	389
15.6.17. mysqlrplshow	392

15.6.18. mysqlserverclone	396
15.6.19. mysqlserverinfo	398
15.6.20. mysqluc	401
15.6.21. mysqluserclone	404
15.7. Extending MySQL Utilities	406
15.7.1. Introduction to extending the MySQL Utilities	406
15.7.2. MySQL Utilities copy_server.py sample	413
15.7.3. Specialized Operations	415
15.7.4. Parsers	418
15.8. MySQL Utilities Testing (MUT)	420
15.8.1. <code>mut</code> — MySQL Utilities Testing	420
15.9. Appendix	422
15.9.1. MySQL Utilities FAQ	422
15.9.2. MySQL Utilities Change History	424

15.1. The Preface

This is the User Manual for the MySQL Utilities.

MySQL Utilities is both a set of command-line utilities as well as a Python library for making the common tasks easy to accomplish. The library is written entirely in Python, meaning that it is not necessary to have any other tools or libraries installed to make it work. It is currently designed to work with Python v2.6 or later and there is no support (yet) for Python v3.1.

Layout

This manual is arranged in an order designed to provide a quick reference for how to use MySQL Utilities. It begins with a brief introduction of MySQL Utilities then presents a list of common administration tasks with examples of how utilities can be used to perform the tasks. From there, the manual begins a deeper dive into the utilities starting with overviews of each utility leading to a detailed description of each via a manual page format. Thus, the manual provides a documentation solution for several needs.

How to Use This Manual

You can use this manual to get a quick solution to an administrative task complete with explanation of how to run the utilities involved and the options and parameters needed. See the tasks chapter for this information.

You can use the manual to learn what utilities exist and how each fits into your own administrative needs. See the utility overview chapter for this information.

You can also use the manual to get more information about each utility and what each option and parameter does via the manuals section.

The manual concludes with a look at extending the MySQL Utilities library, a look at the developer testing environment, and a list of frequently asked questions.

15.2. How to Install MySQL Utilities

MySQL Utilities is available in a number of repository formats. While you may not see your specific operating system or platform listed, we provide general repository formats for most platforms. If none of the available repositories are applicable to your platform, you can use the Source Code repository and install MySQL Utilities from the command line.

You can find the latest repositories for download at [Download MySQL Utilities](#). We discuss each repository in the following sections.

15.2.1. Prerequisites

MySQL Utilities requires Python 2.6. All of the Python code is written to conform to this version of Python.

MySQL Utilities requires the MySQL Connector/Python library for connecting to MySQL. If you have not install Connector/Pyton, see the download section for Connector/Python to [download](#) the appropriate repository.

15.2.2. Source Code

This repository is the bundled source code for MySQL Utilities. It includes all of the utility code as well as the MySQL Utilities library and manual pages. It is available as either a single Windows (Architecture Independent), ZIP or Generic Linux (Architecture Independent), Compressed TAR archive.

You can use this repository to install on any platform that has Python 2.6 installed. For example, you can use the .tar version of this repository to install MySQL Utilties on Mac OS X or Ubuntu.

Once you download and unzip or untar the file, open a terminal and navigate to the directory containing the file. Then unpack the file and install MySQL Utilities using the setup.py script as shown below.

```
$ tar -xvf mysql-utilities-1.3.4a.tar.gz  
$ cd mysql-utilities-1.3.4/  
$ python ./setup.py build  
$ sudo python ./setup.py install
```



Note

Using this repository requires that you have Connector/Python installed or install it separately.

15.2.3. Oracle and Red Hat Linux 6

This repository is available as a architecture independent .rpm repository.

Once you download the file, you can install it using the following command or similar depending on your platform configuration.

```
$ sudo rpm -i mysql-utilities-1.3.4-1.el6.noarch.rpm
```

You can also use the RPM package manager that is part of your base operating system. See your operating system documentation for more details.

15.2.4. Debian Linux

The .deb repository is built for Debian 6 and is architecture independent. While built expressly for Debian 6, it can be installed on various ports such as amd64, i386, etc.



Note

The repository does not work for Debian 7 because MySQL Utilities requires Python 2.6 and Debian 7 currently ships with Python 2.7. For Debian 7, use the Source Code repository to install MySQL Utilities.

Once you download the file, you can install it using the following command or similar depending on your specific release or version of Debian.

```
$ sudo dpkg -i mysql-utilities_1.3.4-1debian6.0_all.deb
```

15.2.5. Microsoft Windows

The .msi available for Microsoft Windows is built for the x86 32-bit platform but can be installed on the 64-bit platform. You can use this repository to install MySQL Utilities separately. Note that MySQL Utilities is included in the MySQL Installer available at [MySQL on Windows](#)

Once you download the .msi, you can install MySQL Utilities by double clicking on the file (or any other valid Windows .msi installation method you prefer).

15.3. Introduction

This chapter introduces MySQL Utilities and presents information on how to access and download MySQL Utilities. It also includes the basics of how to use the account login option common to all utilities.

15.3.1. Introduction to MySQL Utilities

What are the MySQL Utilities?

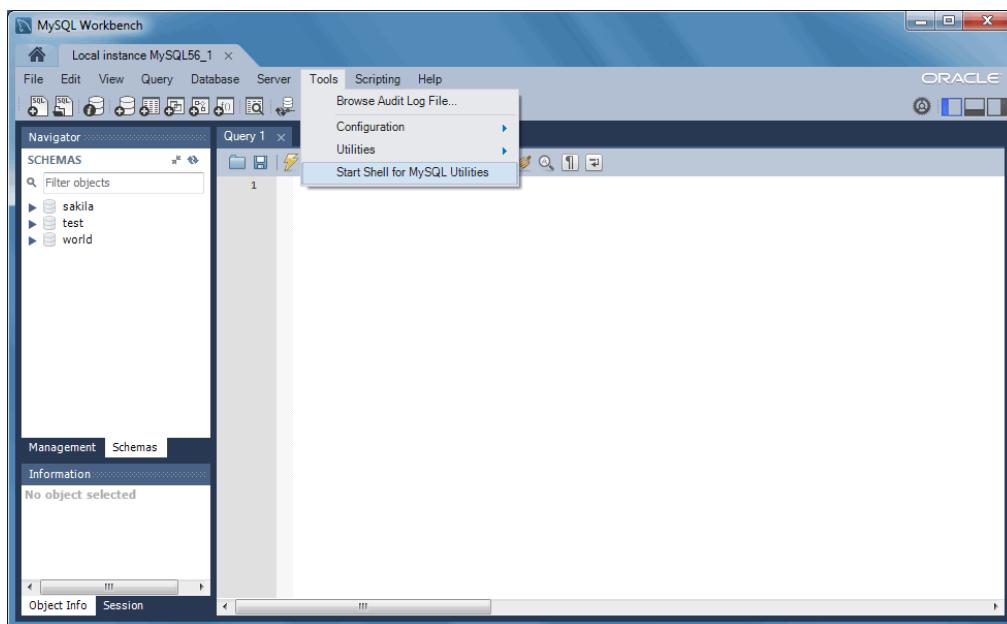
It is a package of utilities that are used for maintenance and administration of MySQL servers. These utilities encapsulate a set of primitive commands, and bundles them so they can be used to perform macro operations with a single command. They can be installed via MySQL Workbench, or as a standalone package.

The utilities are written in Python, available under the GPLv2 license, and are extendable using the supplied library. They are designed to work with Python versions 2.6 or later and there is no support (yet) for Python v3.1.

How do we access the MySQL Utilities?

There are two ways to access the utilities from within the MySQL Workbench. Either use [Tools](#), [Start Shell for MySQL Utilities](#) from the main Workbench toolbar, or click the MySQL Utilities icon from the Workbench home page. Both methods will open a terminal/shell window in the `mysqluc` utility shell. Type "help" to list the available commands.

Figure 15.1. Starting MySQL Utilities from Workbench



You can launch any of the utilities listed by typing the name of the command. To find out what options are available, use the option, or read the appropriate manual page.

The utilities are designed to work on MySQL systems with grants enabled but can also operate on servers started with the `--skip-grant-tables` startup option. However, this practice is strongly discouraged and should be used only in situations where it is appropriate or deemed a last resort.

15.3.2. Connection Parameters

To connect to a server, it is necessary to specify connection parameters such as user name, host name, password, and perhaps also port or socket.

Whenever connection parameters are required, they can be specified three different ways:

- As a dictionary containing the connection parameters.
- As a connection specification string containing the connection parameters.
- As a Server instance.

When providing the connection parameters as a dictionary, the parameters are passed unchanged to the connector's `connect` function. This enables you to pass parameters not supported through the other interfaces, but at least these parameters are supported:

- `user`

The name of the user to connect as. The default if no user is supplied is login name of the user, as returned by `getpass.getuser`.

- `passwd`

The password to use when connecting. The default if no password is supplied is the empty password.

- `host`

The domain name of the host or the IP address. The default if no host name is provided is 'localhost'. This field accepts host names, and IPv4 and IPv6 addresses. It also accepts quoted values which are not validated and passed directly to the calling methods. This enables users to specify host names and IP addresses that are outside of the supported validation mechanisms.

- `port`

The port to use when connecting to the server. The default if no port is supplied is 3306 (which is the default port for the MySQL server as well).

- `unix_socket`

The socket to connect to (instead of using the host and port parameters).

Providing the connection parameters as a string requires the string to have the format `user[:passwd]@host[:port][:socket]`, where some values are optional. If a connection specification string is provided, it is parsed using the `options.parse_connection` function.

15.4. MySQL Utilities Administrative Tasks

MySQL Utilities provides a command-line set of tools for working with MySQL Servers and databases. MySQL Utilities fully supports MySQL Server versions 5.1 and above. It is also compatible with MySQL Server 5.0, but not every feature of 5.0 may be supported. It does not support MySQL Server versions 4.x.

In this section, we present a number of example administrative tasks. Included in each is a description of the need, goals, example execution, and a discussion about the specific options and techniques illustrated. Also included is a description of the specific permissions required to execute the utilities demonstrated.

15.4.1. Database Operations

The tasks described in this section relate to those that are performed on or with one or more databases.

15.4.1.1. How Do I Provision a Slave?

When working with replication, one of the common tasks is adding a new slave for scale out. While adding a new slave has been simplified with utilities like `mysqlreplicate`, provisioning the slave (copying data and getting replication started properly) can be a challenge if done manually.

Fortunately, we have two utilities - `mysqldbexport` and `mysqldbimport` - that have been designed to work with replication so that when the export is generated, you can include the proper replication control statements in the output stream.

15.4.1.1.1. Objectives

Perform slave provisioning using `mysqldbexport` and `mysqldbimport`.

15.4.1.1.2. Example Execution

```
$ mysqldbexport --server=root:root@localhost:13001 --all --export=both --rpl=master --rpl-user=rpl:rpl > data.sql
$ mysqldbimport --server=root:root@localhost:13002 data.sql
# Source on localhost: ... connected.
# Importing definitions from data.sql.
ERROR: The import operation contains GTID statements that require the global gtid_executed system variable on
$ mysql -uroot -proot -h 127.0.0.1 --port=13002 -e "RESET MASTER"
$ mysqldbimport --server=root:root@localhost:13002 data.sql
# Source on localhost: ... connected.
# Importing definitions from data.sql.
CAUTION: The following 1 warning messages were included in the import file:
# WARNING: A partial export from a server that has GTIDs enabled will by default include the GTIDs of all transactions.
#...done.
```

15.4.1.1.3. Discussion

There are several operations listed here. The first one we see is the execution of the `mysqldbexport` utility to create a file that includes an export of all databases as designated with the `--all` [339] option. We add the '`--export=both`' option to ensure we include the definitions as well as the data.

We also add the `--rpl=master` [339] option which instructs `mysqldbexport` to generate the replication commands with respect to the source server being the master. Lastly, we include the replication user and password to be included in the CHANGE MASTER command.

Next, we see an attempt to run the import using `mysqldbimport` but we see there is an error. The reason for the error is the `mysqldbimport` utility detected a possible problem on the slave whereby there were global transaction identifiers (GTIDs) recorded from the master. You can see this situation if you setup replication prior to running the import. The way to resolve the problem is to run the RESET MASTER command on the slave as shown in the next operation.

The next operation is a rerun of the import and in this case it succeeds. We see a warning that is issued any time there are replication commands detected in the input stream whenever GTIDs are enabled.

15.4.1.1.4. Permissions Required

The user used to read data from the master must have the SELECT privilege on all databases exported. The user on the slave must have the SUPER privilege to start replication.

15.4.1.1.5. Tips and Tricks

The warning issued during the import concerning GTIDs is to ensure you are aware that the process for gathering the proper GTIDs to execute on the slave include transactions from all databases. Thus, if you ran a partial export that includes the replication commands and you have GTIDs enabled, you should use the `--skip-rpl` [346] option to skip the replication commands and restart replication manually.

Should your data be large enough to make the use of `mysqldbexport` impractical, you can use `mysqldbexport` to generate the correct replication commands anyway by using the `--export=definitions` [338] option. This will generate the SQL statements for the objects but not the data. You can then use the replication commands generated with your own backup and restore tools.

You can use the option `--rpl=slave` [339] to generate a output stream that considers the source server a slave and uses the source servers master settings for generating the CHANGE MASTER command.

15.4.1.2. How Do I Make a Copy of a Database on the Same Server?

If you are working with a database and want to experiment with changes to objects or data either from direct manipulation (SQL commands) or as a result of interaction with an application, it is prudent to always have a copy to fall back to if something should go wrong.

Naturally, a full backup is key for any production server but what if you just want to do something as a test or as a prototype? Sure, you can restore from your backup when the test is complete but who has the time for that? Why not just make a copy of the database in question and use it?

15.4.1.2.1. Objectives

The goal is to make a copy of a database and rename it to another name. We want to do this on a single database server without resorting to messy file copies and/or stopping the server.

In this case, we want to copy the world_innodb database in its entirety and rename the copy to word_innodb_clone.

The utility of choice here is named `mysqldbcopy` and it is capable of copying databases from server to another or on the same server. Lets have a look at the execution.

15.4.1.2.2. Example Execution

```
$ mysqldbcopy --source=root:root@localhost \
--destination=root:root@localhost world_innodb:world_innodb_clone
# Source on localhost: ... connected.
# Destination on localhost: ... connected.
# Copying database world_innodb renamed as world_innodb_clone
# Copying TABLE world_innodb.City
# Copying TABLE world_innodb.Country
# Copying TABLE world_innodb.CountryLanguage
# Copying data for TABLE world_innodb.City
# Copying data for TABLE world_innodb.Country
# Copying data for TABLE world_innodb.CountryLanguage
#...done.
```

```
$ mysql -uroot -proot -e "SHOW DATABASES"
+-----+
| Database      |
+-----+
| information_schema |
| employees      |
| mysql          |
| world_innodb   |
| world_innodb_clone |
+-----+
```

15.4.1.2.3. Discussion

Notice we specified the source of the database we wanted to copy as well as the destination. In this case, they are the same server. You must specify it this way so that it is clear we are operating on the same server.

Notice how we specified the new name. We used the <old_name>:<new_name> syntax. You can do this for as many databases as you want to copy. That's right - you can copy multiple databases with a single command renaming each along the way.

To copy a database without renaming it (if the destination is a different server), you can omit the <new_name> portion.

15.4.1.2.4. Permissions Required

The user must have SELECT privileges for the database(s) on the source server and have CREATE, INSERT, UPDATE on the destination server.

15.4.1.2.5. Tips and Tricks

You can copy all of the databases on a source server to the destination by using the [--all](#) [333] option. Note that this does not permit rename. To rename, you must specify the databases one at a time.

You can specify certain objects to exclude (skip) in the copy. Use the [--skip](#) [333] option to omit the type of objects. For example, you may want to exclude copying of triggers, procedures, and functions. In this case, use the option '`--skip=TRIGGERS,PROCEDURES,FUNCTIONS`'. Note that case is not important but is used for emphasis only.

The copy is replication and GTID aware and will take actions to preserve the binary log events during the copy.

You can set the locking type with the [--locking](#) [332] option. Possible values include: no-locks = do not use any table locks, lock-all = use table locks but no transaction and no consistent read, snapshot (default): consistent read using a single transaction.

15.4.1.2.6. Risks

Should the copy fail in the middle, the destination databases may be incomplete or inconsistent. Should this occur, drop the destination database in question, repair the cause of the failure, and restart the copy.

15.4.1.3. How Can I Make a Copy of a Database and Change the Storage Engine?

Sometimes you may have need to create a copy of a database but change the storage engine of all tables to another engine.

For example, if you are migrating your database to InnoDB (a wise choice), you can copy the database to a new database on a new server and change the storage engine to InnoDB. For this, we can use the `mysqldbcopy` utility.

15.4.1.3.1. Objectives

In this example, we want to make a copy of the `world_innodb` database but change the storage engine to MyISAM and rename the database accordingly.

You can cause all tables in the destination databases to use a different storage engine with the `--new-storage-engine` [332] option.

15.4.1.3.2. Example Execution

```
$ mysqldbcopy --source=root:root@localhost:3306 \
--destination=root:root@localhost:3307 --new-storage-engine=myisam \
world_innodb:world_myisam
# Source on localhost: ... connected.
# Destination on localhost: ... connected.
# Copying database world_innodb renamed as world_myisam
# Replacing ENGINE=InnoDB with ENGINE=myisam for table `world_myisam`.City.
# Copying TABLE world_innodb.City
# WARNING: FOREIGN KEY constraints for table world_myisam.City are missing because the new storage engine ...
# Replacing ENGINE=InnoDB with ENGINE=myisam for table `world_myisam`.Country.
# Copying TABLE world_innodb.Country
# Replacing ENGINE=InnoDB with ENGINE=myisam for table `world_myisam`.CountryLanguage.
# Copying TABLE world_innodb.CountryLanguage
# WARNING: FOREIGN KEY constraints for table world_myisam.CountryLanguage are missing because the new stor...
# Copying data for TABLE world_innodb.City
# Copying data for TABLE world_innodb.Country
# Copying data for TABLE world_innodb.CountryLanguage
#...done.
$ mysql -uroot -proot -h 127.0.0.1 --port=3307 -e "SHOW DATABASES"
+-----+
| Database      |
+-----+
| information_schema |
| mysql          |
| sakila         |
| world          |
| world_myisam   |
+-----+
$ mysql -uroot -proot -h 127.0.0.1 --port=3307 -e "SHOW CREATE TABLE world_myisam.countrylanguage\G"
***** 1. row *****
      Table: countrylanguage
Create Table: CREATE TABLE `countrylanguage` (
  `CountryCode` char(3) NOT NULL DEFAULT '',
  `Language` char(30) NOT NULL DEFAULT '',
  `IsOfficial` enum('T','F') NOT NULL DEFAULT 'F',
  `Percentage` float(4,1) NOT NULL DEFAULT '0.0',
  PRIMARY KEY (`CountryCode`, `Language`),
  KEY `CountryCode` (`CountryCode`)
) ENGINE=MyISAM DEFAULT CHARSET=latin1
```

15.4.1.3.3. Discussion

Notice here we created a copy of the database and changed all tables in the destination database to use the InnoDB storage engine with the `--new-storage-engine` [332] option.

We show proof of the change by displaying the CREATE statement for one of the tables on the destination server.

Notice we also renamed the database by using the <old_name>:<new_name> syntax.

15.4.1.3.4. Permissions Required

The user must have SELECT privileges for the database(s) on the source server and have CREATE, INSERT, UPDATE on the destination server.

15.4.1.3.5. Tips and Tricks

You can exclude specific options by using the [--exclude \[332\]](#) option specifying a SQL pattern expression. For example, to exclude objects that start with xy, use '--exclude=xy%'.

You can use REGEXP patterns in the [--exclude \[332\]](#) option by specifying [--regexp \[333\]](#) in addition to the [--exclude \[332\]](#) option.

15.4.1.3.6. Risks

Should the copy fail in the middle, the destination databases may be incomplete or inconsistent. Should this occur, drop the destination database in question, repair the cause of the failure, and restart the copy.

15.4.1.4. How Can I Tell If a Table on Server X has the same Structure as the Same Table on Server Y?

Multiple database servers that are kept synchronized manually or are compartmentalized for security purposes but are by practice kept up-to-date manually are prone to unintentional (and sometimes intentional) divergence.

For example, you may maintain a production server and a development server. The development server may have the same databases and the same structures as the production server (but maybe not the same data). However, the natural course of administrative tasks and maintenance can sometimes leave the development server behind.

When this happens, you need to have a way to quickly check the schema for a table on the production server to see if the development server has the same structure. The utility of choice for this operation is [mysqldiff](#).

15.4.1.4.1. Objectives

The goal is to compare a table schema on one machine to another and show they differ.

15.4.1.4.2. Example Execution

```
$ mysqldiff --server1=root:root@localhost \
--server2=root:root@localhost:3307 world.city:world.city --changes-for=server2
# server1 on localhost: ... connected.
# server2 on localhost: ... connected.
# Comparing world.city to world.city                               [FAIL]
# Object definitions differ. (--changes-for=server2)
#
--- world.city
+++ world.city
@@ -4,6 +4,7 @@
 `CountryCode` char(3) NOT NULL DEFAULT '',
 `District` char(20) NOT NULL DEFAULT '',
 `Population` int(11) NOT NULL DEFAULT '0',
+ `Climate` enum('tropical','dry','mild','continental','polar') DEFAULT NULL,
 PRIMARY KEY (`ID`),
```

```
KEY `CountryCode` (`CountryCode`),
CONSTRAINT `city_ibfk_1` FOREIGN KEY (`CountryCode`) REFERENCES `Country` (`Code`)
Compare failed. One or more differences found.
```

15.4.1.4.3. Discussion

Notice to accomplish this task, we simply specified each server with [--server1 \[349\]](#) and [--server2 \[349\]](#) then specified the database objects to compare with the <db>.<object>:<db>.<object> syntax.

15.4.1.4.4. Permissions Required

You must have SELECT privileges for both objects on both servers as well as SELECT on the mysql database.

15.4.1.4.5. Tips and Tricks

You can set the direction of the compare by using the [--changes-for \[348\]](#) option. For example, to see the changes for server1 as the target, use '--changes-for=server1'.

15.4.1.5. How Can I Syncronize a Table on Two Servers Where Neither is Up-to-date?

When working with servers that are used in different networks or are compartmentalized, or simply intentionally manually redundant (they do not use replication), or perhaps through some crisis, you may encounter a situation where a table (or an entire database) becomes out of synch.

We don't simply want to know which rows differ, rather, we need to know the SQL statements needed to bring the tables into synch. Furthermore, we aren't sure which table is most out of date so we'd like to see the transformation statements for both directions.

In this case, it would be very helpful to know exactly how the tables differ. For this, we use the [mysqldbcompare](#) utility.

15.4.1.5.1. Objectives

The goal is to generate the SQL transformation statements to bring two tables into synch.

15.4.1.5.2. Example Execution

```
$ mysqldbcompare --server1=root:root@localhost:13001 --server2=root:root@localhost:13002 \
menagerie -a --difftype=SQL --show-reverse --quiet
# Checking databases menagerie on server1 and menagerie on server2
#
#
# Row counts are not the same among `menagerie`.`pet` and `menagerie`.`pet`.
#
# Transformation for --changes-for=server1:
#
DELETE FROM `menagerie`.`pet` WHERE `pet_num` = '10';
DELETE FROM `menagerie`.`pet` WHERE `pet_num` = '12';
INSERT INTO `menagerie`.`pet` (`pet_num`, `name`, `owner`, `species`, `sex`, `birth`, `death`) VALUES('11'
#
# Transformation for reverse changes (--changes-for=server2):
#
# DELETE FROM `menagerie`.`pet` WHERE `pet_num` = '11';
# INSERT INTO `menagerie`.`pet` (`pet_num`, `name`, `owner`, `species`, `sex`, `birth`, `death`) VALUES('12')
```

```
# INSERT INTO `menagerie`.`pet` (`pet_num`, `name`, `owner`, `species`, `sex`, `birth`, `death`) VALUES('12',  
#
```

15.4.1.5.3. Discussion

In the example above, we connected to two servers and compare the database named menagerie. We enabled the transformation statements using a combination of options as follows.

The [--diff-type=SQL](#) [327] option instructs the utility to generate the SQL statements.

The [--show-reverse](#) [327] option instructs the utility to generate the differences in both directions. That is, from the perspective of server1 as compared to server2 and server2 as compared to server1. By convention, the second set is commented out should you wish to pipe the output to a consumer.

Lastly, the [--quiet](#) [327] option simply turns off the verbosity of print statements that normally occur for communicating progress.

15.4.1.5.4. Permissions Required

The user must have the SELECT privilege for the databases on both servers.

15.4.1.5.5. Tips and Tricks

You can change the direction using the [--changes-for](#) [327] option. For example, '--changes-for=server1' is the default direction and '--changes-for=server2' is the reverse. In the second case, the [--show-reverse](#) [327] displays the perspective of server1 commented out.

15.4.2. General Operations

The tasks described in this section include general tasks such as reporting information about a server and searching for objects or processes on a server.

15.4.2.1. How Can I Find Out How Much Space My Data Uses?

When preparing to create a backup or maintenance on a server related to storage, it is often the case we need to know how much space is used by our data and the logs the server maintains. Fortunately, there is a utility for that.

15.4.2.1.1. Objectives

Show the disk space used by the databases and all logs using the `mysqldiskusage` utility.

15.4.2.1.2. Example Execution

```
$ sudo env PYTHONPATH=$PYTHONPATH mysqldiskusage \  
--server=root:root@localhost --all  
# Source on localhost: ... connected.  
# Database totals:  
+-----+-----+  
| db_name | total |  
+-----+-----+  
| oltp2 | 829,669 |  
| bvm | 15,129 |  
| dbl | 9,895 |  
| db2 | 11,035 |  
| employees | 206,117,692 |  
| griots | 14,415 |  
| mysql | 995,722 |
```

```

| oltp1           | 177,393   |
| room_temp       | 9,847     |
| sakila          | 791,727   |
| test            | 647,911   |
| test_arduino    | 9,999     |
| welford_kindle | 72,032    |
| world           | 472,785   |
| world_innodb    | 829,669   |
+-----+-----+
Total database disk usage = 210,175,251 bytes or 200.44 MB

# Log information.
+-----+-----+
| log_name        |      size   |
+-----+-----+
| host123.log     | 957,282,265 |
| host123-slow.log | 123,647   |
| host123.local.err | 321,772,803 |
+-----+-----+
Total size of logs = 1,279,178,715 bytes or 1.19 GB

# Binary log information:
Current binary log file = my_log.000287
+-----+-----+
| log_file        |      size   |
+-----+-----+
| my_log.000285   | 252208    |
| my_log.000286   | 256       |
| my_log.000287   | 3063     |
| my_log.index    | 48        |
+-----+-----+
Total size of binary logs = 255,575 bytes or 249.58 KB

# Server is not an active slave - no relay log information.
# InnoDB tablespace information.
+-----+-----+
| innodb_file     |      size   |
+-----+-----+
| ib_logfile0     | 5,242,880  |
| ib_logfile1     | 5,242,880  |
| ibdata1         | 815,792,128 |
| ibdata2         | 52,428,800  |
+-----+-----+
Total size of InnoDB files = 889,192,448 bytes or 848.00 MB
InnoDB freespace = 635,437,056 bytes or 606.00 MB

```

15.4.2.1.3. Discussion

To see all of the logs, we use the `--all` [352] option which shows all logs and the InnoDB disk usage.

Notice we used elevated privileges to allow for reading of all of the files and databases in the data directory. In this case, the data directory is owned by the mysql user and a normal user account does not have read access.

The `--all` [352] option instructs the utility to list all databases even if they contain no data.

15.4.2.1.4. Permissions Required

You must have permissions to read the data directory or use an administrator or super user (sudo) account as shown in the example.

15.4.2.1.5. Tips and Tricks

You can run `mysqldiskusage` without privileges to read the data directory but in this case you will see an estimate of the disk usage rather than actual bytes used. You may also not be able to see a list of the logs.

15.4.2.2. My Server Crashed! I Need to Know the Structure of a Table. How Can I Do That?

When things go wrong badly enough that your server is down, but you can still access the disks, you may find yourself faced with a number of complex recovery tasks.

One of those is the need to discover the structure of a particular table or set tables. Perhaps this is needed for an emergency recovery, a redeployment, or setup for a forensic investigation. Whatever the case, without a running MySQL server it is not possible to know the structure of a table unless you keep meticulous notes and/or use some form of high availability (redundancy).

Fortunately, there is a utility for situations like this. The `mysqlfrm` utility can be used to discover the structure of a table.

15.4.2.2.1. Objectives

With a downed or offline server, discover the structure of a table. More specifically, generate the CREATE TABLE SQL command.

15.4.2.2.2. Example Execution

```
$ sudo env PYTHONPATH=$PYTHONPATH mysqlfrm --basedir=/usr/local/mysql --port=3333 --user=<user> /usr/local/mys  
# Spawning server with --user=<user>.  
# Starting the spawned server on port 3333 ... done.  
# Reading .frm files  
#  
# Reading the books.frm file.  
#  
# CREATE statement for /usr/local/mysql/data/kindle/books.frm:  
#  
  
CREATE TABLE `welford_kindle`.`books` (  
  `ISBN` char(32) NOT NULL PRIMARY KEY,  
  `title` char(128) DEFAULT NULL,  
  `purchase_date` date DEFAULT NULL,  
  `cost` float(10,2) DEFAULT NULL  
) ENGINE=MyISAM DEFAULT CHARSET=latin1  
  
#...done.
```

15.4.2.2.3. Discussion

For this example, we used three required parameters; the base directory for the offline server (basedir), a new port to use for the spawned server (port), and a user name to use to run the spawned server (port). The later is necessary since we must launch the `mysqlfrm` utility as root (sudo) in order to be able to read (copy) files from the protected data directory of the host server.

The `--port` [365] option is always required for running the utility in default mode (it is not needed for diagnostic mode). You must supply a valid unused port. The utility will check to see if the port is in use and if so will produce an error.

We use the `--basedir` [364] option instead of the `--server` [365] option because we were faced with a situation where the original server was offline (down). Note that you can use the `--basedir` [364] option for a running server if you do not want the utility to connect to the original server in any way.

15.4.2.2.4. Permissions Required

The permissions for using `mysqlfrm` will vary and depend entirely on how you use it. If you use the utility to read .frm files in a protected folder like the example above (in either mode), you must have the ability to run the server as root.

If you use the utility with a server connection, the user you use to connect must have the ability to read system variables at a minimum (read access to the mysql database).

15.4.2.2.5. Tips and Tricks

The utility is designed to work on the host where the .frm files reside. It does not permit connecting to a remote host to read .frm files.

If something goes wrong during the spawning of the server, use the verbosity option three times (-vvv) to turn on maximum depth debug statements. This will ensure you will see all of the messages from the start of the spawned server from bootstrap onward. Look for errors in these statements as to why the spawned server will not start.

15.4.2.2.6. Risks

The utility performs a best effort approximation of the CREATE statement when run in diagnostic mode. As such, if you read a .frm file that uses character sets or collations other than the default and you do not use a `--server` [365] option to connect to a server to read the character sets, this can result in miscalculated column sizes.

For example, suppose your default character set is latin1 which uses 1 byte per character. Lets also suppose you are attempting to read a .frm file that uses a character set that uses 3 bytes per character. Furthermore, we have no server to connect. In this case, the column sizes may be off by a factor of 3. A case in point would be a field such as col_a char(3) would appear in the output of the `mysqlfrm` utility as col_a char(9).

To mitigate risks such as this and to produce the most accurate CREATE statement in diagnostic mode, always use the `--server` [365] option.

15.4.2.3. Creating a New User With The Same Privileges as Another User

The MySQL privilege system permits you to create a set of permissions for each user. Sometimes the set of permissions are complex and may require multiple GRANT statements to effect. Other times, the user may acquire privileges over time.

Regardless of how it came about, you may find yourself needing to create a new user that has the same privileges as another user.

15.4.2.3.1. Objectives

The goal is to create one or more users whose permissions are identical to an original user on a single server.

Rather than discover what those privileges are with a SHOW GRANTS FOR statement and copy them into a script, modify it, copy and paste again for each user, etc., etc., we can use a single command to copy one user to a list of new users. We can even set different passwords for each user as we go.

Let's assume we have a user, joe@localhost, who has a long list of permissions. We need to create a clone of his user account for two new users, sally and john. Each of these users will require a new password.

15.4.2.3.2. Example Execution

```
$ mysqluserclone --source=root@localhost \
--destination=root@localhost \
joe@localhost sally:secret1@localhost john:secret2@localhost
# Source on localhost: ... connected.
# Destination on localhost: ... connected.
# Cloning 2 users...
# Cloning joe@localhost to user sally:secret1@localhost
# Cloning joe@localhost to user john:secret2@localhost
# ...done.
```

15.4.2.3.3. Discussion

In the above example, we see the use of the `mysqluserclone` utility to clone the `joe` user to two new user accounts.

Notice we used the `--source` [405] option to connect to the original server and `--destination` [405] for the same server.

After that, we simply list the user we want to clone and the new users we want to create. In this case we use the format `username:password@host` to specify the user account name, password (optional), and host.

When the utility finishes, you will have two new user accounts that have the same privileges as the original user, `joe@localhost`.

15.4.2.3.4. Permissions Required

On the source server, the user must have the `SELECT` privilege for the `mysql` database.

On the destination server, the user must have the global `CREATE USER` privilege or the `INSERT` privilege for the `mysql` database as well as the `GRANT OPTION` privilege, and the privileges that the original user has (you grant a privilege you do not have yourself).

15.4.2.3.5. Tips and Tricks

You can use `--destination` [405] option to specify a different server to copy a user account to another server.

Use the `--dump` [405] option with only the `--source` [405] option to see all user accounts.

Use the `--include-global-privileges` [405] option to include `GRANT` statements that the `user@host` combination matches. This is useful for copying user accounts from one server to another where there are global privileges in effect.

15.4.2.4. What Options Are Used With Each Utility?

There are many utilities and it is not always easy to remember all of the options and parameters associated with each. Sometimes we need to run several utilities using nearly the same options. For example, you may want to run several utilities logging into a particular server. Rather than retype the connection information each time, you would like to save the option value some way and reuse it.

Fortunately, the `mysqluc` utility does this and more. It is named the MySQL Users' Console and provides type completion for options, utility names, and even user-defined variables for working with common option values. Not only that, it also provides the ability to get help for any utility supported.

15.4.2.4.1. Objectives

Discover what utilities exist and find the options for certain utilities.

Run several utilities with the same server using the type completion feature to make using the suite of utilities easier.

15.4.2.4.2. Example Execution



Note

In the example below, keystrokes are represented using square brackets. For example, [TAB] indicates the tab key was pressed. Similarly, portions in the commands specific with angle brackets are values you would replace with actual values. For example, <user> indicates you would place the user's login name here.

```
$ mysqluc
Launching console ...

Welcome to the MySQL Utilities Client (mysqluc) version 1.3.4
Copyright (c) 2010, 2013 Oracle and/or its affiliates. All rights reserved.
This is a release of dual licensed MySQL Utilities. For the avoidance of
doubt, this particular copy of the software is released
under the version 2 of the GNU General Public License.
MySQL Utilities is brought to you by Oracle.

Type 'help' for a list of commands or press TAB twice for list of utilities.

mysqluc> help
Command           Description
-----
help utilities   Display list of all utilities supported.
help <utility>  Display help for a specific utility.
show errors      Display errors captured during the execution of the
                  utilities.
clear errors     clear captured errors.
show last error  Display the last error captured during the
                  execution of the utilities
help | help commands Show this list.
exit | quit       Exit the console.
set <variable>=<value> Store a variable for recall in commands.
show options     Display list of options specified by the user on
                  launch.
show variables   Display list of variables.
<ENTER>          Press ENTER to execute command.
<ESCAPE>         Press ESCAPE to clear the command entry.
<DOWN>           Press DOWN to retrieve the previous command.
<UP>             Press UP to retrieve the next command in history.
<TAB>            Press TAB for type completion of utility, option,
                  or variable names.
<TAB><TAB>        Press TAB twice for list of matching type
                  completion (context sensitive).

mysqluc> help utilities
Utility           Description
-----
myslauditadmin   audit log maintenance utility
myslauditgrep    audit log search utility
mysqldbcompare   compare databases for consistency
mysqldbcopy      copy databases from one server to another
mysqldbexport    export metadata and data from databases
mysqldbimport    import metadata and data from files
mysqldiff        compare object definitions among objects where the
                  difference is how db1.obj1 differs from db2.obj2
mysqldiskusage   show disk usage for databases
```

```

mysqlfailover      automatic replication health monitoring and failover
mysqlfrm          show CREATE TABLE from .frm files
mysqlindexcheck   check for duplicate or redundant indexes
mysqlmetagrep     search metadata
mysqlprocgrep    search process information
mysqlreplicate    establish replication with a master
mysqlrpladmin     administration utility for MySQL replication
mysqlrplcheck    check replication
mysqlrplshow      show slaves attached to a master
mysqlserverclone  start another instance of a running server
mysqlserverinfo   show server information
mysqluserclone    clone a MySQL user account to one or more new users

mysqluc>
mysqluc> help mysqldb[TAB][TAB]
Utility          Description
-----
mysqldbcompare   compare databases for consistency
mysqldbcopy      copy databases from one server to another
mysqldbexport    export metadata and data from databases
mysqldbimport    import metadata and data from files

mysqluc> mysqlrplshow --m[TAB][TAB]

Option          Description
-----
--master=MASTER connection information for master server in the
                  form: <user>[:<password>]@<host>[:<port>][:<socket>]
                  or <login-path>[:<port>][:<socket>].
--max-depth=MAX_DEPTH limit the traversal to this depth. Valid only with
                  the --recurse option. Valid values are non-negative
                  integers.

mysqluc> mysqlrplshow --mast[TAB]er=<user>:<password>@localhost:13001
An error was found, while executing the command, use "show last error" command to display details

mysqluc> show last error
Execution of utility: mysqlrplshow --master=<user>:<password>@localhost:13001
returned errorcode: 2 with error message:
Usage: mysqlrplshow.py --master=root@localhost:3306

mysqlrplshow.py: error: The --discover-slaves-login is required to test slave connectivity.

mysqluc> mysqlrplshow --master=<user>:<password>@localhost:13001 --discover-slaves-login=<user>:<password>
# master on localhost: ... connected.
# Finding slaves for master: localhost:13001

# Replication Topology Graph
localhost:13001 (MASTER)
|
+--- localhost:13002 - (SLAVE)
|
+--- localhost:13003 - (SLAVE)
|
+--- localhost:13004 - (SLAVE)
|
+--- localhost:13005 - (SLAVE)

mysqluc>
```

15.4.2.4.3. Discussion

There is a lot going on here in this example! Let's look through the command entries as they occur in the text.

The first command, `mysqluc`, starts the users' console. Once the console starts, you will see a welcome banner followed by a simple prompt, `mysqluc>`. No additional options or parameters are necessary. However, it should be noted that you can pass commands to the console to execute on start. For a complete list of options, see MySQL Users' Console manual page.

The next command, `help`, shows the help for the users' console itself. As you can see, there are a number of options available. You can set user defined variables, discover the help for other utilities, display the latest error, and see the options used to start the console.

The `help utilities` command shows you a list of the available utilities and a short description of each.

Next, we decide we want to get help for one of the database utilities but we do not remember the name. We know it starts with `mysqldb` but we are not sure of the rest. In this case, if we type `mysqldb` then hit TAB twice, the users' console will show us a list of all of the utilities that begin with `mysqldb`. Very nice!

Now let's say we want to see a graph of our replication topology but we are not sure what the option for specifying the master. In this case, we type the command to launch the `mysqlrplshow` utility and type the start of the option, '`--m`', then press TAB twice. What we see is there are two options that match that prefix. Notice we also see a short description (`help`) for each. This is a real time saving feature for the users' console.

Notice in the next segment we do not have to type the entire name of the option. In this case we typed '`--mast[TAB]`' which the users' console completed with '`--master=`'. This is tab completion for option names. Also very nice.

Notice the result of the command we entered, `mysqlrplshow --master=<user>:<password>@localhost:13001`. There was an error here. We can see the error with the `show errors` command. We see in the error we failed to provide any connection information for the slaves.

Once we correct that omission, the last command shows how the users' console executes a utility and displays the results in the same stream as the console - much like the `mysql` client tool.

15.4.2.4.4. Permissions Required

There are no special permissions required to run `mysqluc` however, you must have the necessary privileges to execute the desired utilities.

15.4.2.5. I've Got Too Many Indexes! How Do I Know Which Ones to Drop?

MySQL allows its users to create several indexes that might be the same (duplicate indexes) or partially similar (redundant indexes) in its structure. While duplicate indexes have no advantages, there are some cases where redundant indexes might be helpful. However, both have disadvantages. Duplicate and redundant indexes slow down update and insert operations. As a result it is usually a good idea to find and remove them.

Doing this manually would be a time consuming task, especially for big databases and that is why there is a utility to automate this type of task: `mysqlindexcheck`.

15.4.2.5.1. Objectives

Our goal is to use the `mysqlindexcheck` utility to help us find duplicate and redundant indexes. For that we are going to use the following table as an example:

```
CREATE TABLE `test_db`.`indexcheck_test` (
```

```

`emp_id` INT(11) NOT NULL,
`fiscal_number` int(11) NOT NULL,
`name` VARCHAR(50) NOT NULL,
`surname` VARCHAR (50) NOT NULL,
`job_title` VARCHAR (20),
`hire_date` DATE default NULL,
`birthday` DATE default NULL,
PRIMARY KEY (`emp_id`),
KEY `idx_fnumber`(`fiscal_number`),
UNIQUE KEY `idx_unifnumber`(`fiscal_number`),
UNIQUE KEY `idx_uemp_id`(`emp_id`),
KEY `idx_full_name`(`name`, `surname`),
KEY `idx_full_name_dup`(`name`, `surname`),
KEY `idx_name`(`name`),
KEY `idx_surname`(`surname`),
KEY `idx_reverse_name`(`surname`, `name`),
KEY `idx_id_name`(`emp_id`, `name`),
KEY `idx_id_hdate`(`emp_id`, `hire_date`),
KEY `idx_id_bday`(`emp_id`, `birthday`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8

```

15.4.2.5.2. Example Execution

```

$ mysqlindexcheck --server=test_user@localhost:13010 test_db.indexcheck_test
# Source on localhost: ... connected.
# The following indexes are duplicates or redundant for table test_db.indexcheck_test:
#
CREATE INDEX `idx_uemp_id` ON `test_db`.`indexcheck_test` (`emp_id`) USING BTREE
#      may be redundant or duplicate of:
ALTER TABLE `test_db`.`indexcheck_test` ADD PRIMARY KEY (`emp_id`)
#
CREATE INDEX `idx_fnumber` ON `test_db`.`indexcheck_test` (`fiscal_number`) USING BTREE
#      may be redundant or duplicate of:
CREATE INDEX `idx_unifnumber` ON `test_db`.`indexcheck_test` (`fiscal_number`) USING BTREE
#
CREATE INDEX `idx_full_name_dup` ON `test_db`.`indexcheck_test` (`name`, `surname`) USING BTREE
#      may be redundant or duplicate of:
CREATE INDEX `idx_full_name` ON `test_db`.`indexcheck_test` (`name`, `surname`) USING BTREE
#
CREATE INDEX `idx_name` ON `test_db`.`indexcheck_test` (`name`) USING BTREE
#      may be redundant or duplicate of:
CREATE INDEX `idx_full_name` ON `test_db`.`indexcheck_test` (`name`, `surname`) USING BTREE
#
CREATE INDEX `idx_surname` ON `test_db`.`indexcheck_test` (`surname`) USING BTREE
#      may be redundant or duplicate of:
CREATE INDEX `idx_reverse_name` ON `test_db`.`indexcheck_test` (`surname`, `name`) USING BTREE
#
ALTER TABLE `test_db`.`indexcheck_test` ADD PRIMARY KEY (`emp_id`)
#      may be redundant or duplicate of:
CREATE INDEX `idx_id_name` ON `test_db`.`indexcheck_test` (`emp_id`, `name`) USING BTREE
#
CREATE INDEX `idx_id_hdate` ON `test_db`.`indexcheck_test` (`emp_id`, `hire_date`) USING BTREE
#      may be redundant or duplicate of:
CREATE INDEX `idx_id_name` ON `test_db`.`indexcheck_test` (`emp_id`, `name`) USING BTREE
#
CREATE INDEX `idx_id_bday` ON `test_db`.`indexcheck_test` (`emp_id`, `birthday`) USING BTREE
#      may be redundant or duplicate of:
CREATE INDEX `idx_id_name` ON `test_db`.`indexcheck_test` (`emp_id`, `name`) USING BTREE
#
CREATE INDEX `idx_uemp_id` ON `test_db`.`indexcheck_test` (`emp_id`) USING BTREE
#
CREATE INDEX `idx_id_name` ON `test_db`.`indexcheck_test` (`emp_id`, `name`) USING BTREE
#
CREATE INDEX `idx_id_hdate` ON `test_db`.`indexcheck_test` (`emp_id`, `hire_date`) USING BTREE
#
CREATE INDEX `idx_id_bday` ON `test_db`.`indexcheck_test` (`emp_id`, `birthday`) USING BTREE

```

15.4.2.5.3. Discussion

As we can see, the utility first points out that neither the idx_uemp_id index nor the idx_fnumber are necessary and it points out why. The first, idx_uemp_id, is redundant because the primary key already ensures that emp_id values have to be unique. As for idx_fnumber, it is also redundant because of idx_ufnumber, a UNIQUE type index which also works as a regular index. Then it points out that idx_full_name_dup is also not necessary. In this case it is a duplicate of the idx_full_name index since it contains the exact same columns on the same order.

Notice that it also indicates that idx_name, idx_surname and even the PRIMARY INDEX on emp_id might be redundant. This happens because we are dealing with BTREE type indexes and for this type of indexes an index X is redundant to an index Y if and only if the first n columns in X also appear in Y.

Given that we are using InnoDB engine, it also warns us that `idx_uemp_id`, `idx_id_name`, `idx_id_hdate` and `idx_id_bday` might not be needed. This happens because, in InnoDB, secondary indexes contain the primary key columns for the row that are not in the secondary index.

It must be noted however that these are just indications, they must not be followed blindly because redundant indexes can be useful depending on how you use (query) your tables.

15.4.2.5.4. Permissions Required

Regarding the privileges needed to run this utility, the test_user needs SELECT privilege on the mysql database as well as for the databases which tables are being checked.

15.4.2.5.5. Tips and Tricks

You can use the -d option to generate the SQL drop statements needed to remove the indexes.

The `--stats` [369] option can be used alone or together with either `--best` [368] or `--worst` [369] to show statistics about the indexes.

Use the `--show-indexes` [369] option to show each table together with its indexes.

15.4.2.6. I Need to Find an Object on My Server But All I Have is a Partial Name. How Do I Find All Objects with That Name Prefix?

One of the challenges for database administrators who manage servers with thousands of objects is the task of finding an object by name. Sometimes all you have to go on is the name of a table or perhaps an obscure reference to a partial name. This can come about through a diagnosis of a problem connection, application, or via an incomplete description from a defect report.

It is also possible you need to simply check to see if certain things exist. For example, suppose among your databases are parts or inventory data and you want to check to see if there are any functions or procedures that operate on a column named 'cost'. Moreover, you want to see anything related that has 'cost' as part of its name.

Whatever the case, it would be a big time saver if you could search through all of the database objects and see a list of the objects whose name matches a prefix (or pattern). Fortunately, the `mysqlmetagrep` utility can get this done.

15.4.2.6.1. Objectives

Find all objects whose name begins with a known prefix. More specifically, find any mention of the word 'cost'.

15.4.2.6.2. Example Execution

```
$ mysqlmetagrep --server=root:root@localhost --body --pattern='%cost%'
+-----+-----+-----+-----+-----+-----+
| Connection | Object Type | Object Name | Database | Field Type | Matches |
+-----+-----+-----+-----+-----+-----+
| root:@localhost:3306 | FUNCTION | adjust_cost | griots | ROUTINE | adjust_cost |
| root:@localhost:3306 | TABLE | supplies | griots | COLUMN | cost |
| root:@localhost:3306 | TABLE | film | sakila | COLUMN | replacement_cost |
+-----+-----+-----+-----+-----+-----+
$ mysql -uroot -proot -e "SHOW CREATE FUNCTION griots.adjust_cost \G"
***** 1. row *****
      Function: adjust_cost
      sql_mode:
      Create Function: CREATE DEFINER='root'@'localhost' FUNCTION `adjust_cost`(cost double) RETURNS double
      DETERMINISTIC
      return cost * 1.10
      character_set_client: latin1
      collation_connection: latin1_swedish_ci
      Database Collation: latin1_swedish_ci
```

15.4.2.6.3. Discussion

In this example, we see the use of the database pattern '%cost%' to find objects that have 'cost' anywhere in their name. We also see the use of the [--body \[372\]](#) option to instruct the utility to look inside procedures and functions. This can be very handy to locate routines that manipulate data as you can see.

Notice once we found a routine that had 'cost' mentioned, we can examine its body via the SHOW CREATE FUNCTION command to see just how it is using the column 'cost'. In this case, we see someone has written a function to adjust the cost by 10%.

Therefore, not only can you find objects that have anything named 'cost', you can also discover any hidden logic that may operate on something named 'cost'.

15.4.2.6.4. Permissions Required

The user must have the SELECT privilege on the mysql database.

15.4.2.6.5. Tips and Tricks

If you are familiar with using regular expressions, you can use the [--regexp \[372\]](#) option to use regular expressions instead of database patterns. For example, the regex for the search above would be --pattern='^.*cost.*' --basic-regex.

15.4.2.7. How Can I Run a Process Every Night To Kill Certain Connections?

Some database administrators use nightly routines to perform maintenance on their databases or servers. Sometimes these routines can be blocked by long running queries or applications that hang onto locks for longer than expected.

Naturally, priority is given to the application and maintenance routines are often cancelled rather than interfere with an application. Should it happen that you subscribe to this notion and you have a routine that is still being blocked or for some reason hasn't completed by a certain time, you need a quick way to generate an event to kill the connection involved. This is where the [mysqlprocgrep](#) utility can help.

15.4.2.7.1. Objectives

The objective is to generate an event that will kill all connections based on a user login ('msaladin') but only if that connection is trying to run a custom administration script named 'my_admin_things'.

15.4.2.7.2. Example Execution

```
$ mysqlprocgrep --sql-body \
--match-command='my_admin_thingy%' --match-user='msaladin%' --kill-connection
DECLARE kill_done INT;
DECLARE kill_cursor CURSOR FOR
    SELECT
        Id, User, Host, Db, Command, Time, State, Info
    FROM
        INFORMATION_SCHEMA.PROCESSLIST
    WHERE
        COMMAND LIKE 'my_admin_thingy%'
        AND
        USER LIKE 'msaladin%'
OPEN kill_cursor;
BEGIN
    DECLARE id BIGINT;
    DECLARE EXIT HANDLER FOR NOT FOUND SET kill_done = 1;
    kill_loop: LOOP
        FETCH kill_cursor INTO id;
        KILL CONNECTION id;
    END LOOP kill_loop;
END;
CLOSE kill_cursor;
```

15.4.2.7.3. Discussion

Notice in the example above, we did not connect to any server to get this information. That is one of the great things about this utility - you can generate all manner of SQL commands for finding processes and try them out on a test system before incorporating them into your events, triggers, and routines.

We specified the user with the [--match-user](#) [375] option using a wildcard in case the user is logged in from a different system. We also specified the name of the maintenance routine in the same manner in case it gets renamed with a version number or some such.

The output of this utility then is the SQL statement we need to use to find and kill the connections that meet these criteria. Armed with this, we can make a procedure we can call from an event and execute the SQL at a precise time every day.

15.4.2.7.4. Permissions Required

The user must have the SELECT permission on the mysql database.

15.4.2.7.5. Tips and Tricks

If you are familiar with using regular expressions, you can use the [--regexp](#) [375] option to use regular expressions instead of database patterns.

15.4.3. High Availability Operations

The tasks described in this section include those for replication and general to specific high availability tasks such as automatic failover.

15.4.3.1. How Can I Use Replication?

MySQL has built-in support for several types of replication. Replication is usually employed with the purpose of increasing the performance and/or the fault-tolerance of a service. However, setting up replication can be a somewhat complicated and error prone process. But fear not, MySQL Utilities has tools that can help simplify and even automate several replication related tasks.

Lets see a possible scenario where replication is used to obtain scalability, i.e. to increase the performance of a service. Lets imagine an online shopping service. The shop started out small so a single server was enough to handle all the requests, however now it has become quite popular and as a result that single server is no longer able to handle all the requests. Being an online store, most of the operations are read operations (checking existing products, reviews, stock availability, etc).

15.4.3.1.1. Objectives

Our goal is to use replication in order to improve the throughput of the service by adding more servers which will become replicas of the already existing server. These replicas will allow scaling out of the service by taking up all the read requests, leaving the old server (now called the master) only in charge of the writes. Rather than doing everything "by hand" with the mysql command line, we are going to setup this replication scenario using a single script, `mysqlreplicate` which will do most of the hard work for us. We then check the result using the `mysqlrpladmin` utility.

Lets assume the existing server, Server1, is running on port 13001 on the local machine with IP 192.168.1.1 and that we want to add 2 new servers, Server2 running on 192.168.1.2:13001 and Server3 running on 192.168.1.3:3306.

15.4.3.1.2. Example Execution

```
$ mysqlreplicate --master=m_account@192.168.1.1:13001 \
--slave=slave_accl@192.168.1.2:13001 --rpl-user=repl:slavepass -b
# master on 192.168.1.1: ... connected.
# slave on 192.168.1.2: ... connected.
# Checking for binary logging on master...
# Setting up replication...
# ...done.

$ mysqlreplicate --master=m_account@192.168.1.1:13001 \
--slave=slave_acc2@192.168.1.3:3306 --rpl-user=repl:slavepass -b
# master on 192.168.1.1: ... connected.
# slave on 192.168.1.3: ... connected.
# Checking for binary logging on master...
# Setting up replication...
# ...done.

$ mysqlrplcheck --master=m_account@192.168.1.1:13001 \
--slave=slave_accl@192.168.1.2:13001

# master on 192.168.1.1: ... connected.
# slave on 192.168.1.2: ... connected.
Test Description Status
-----
Checking for binary logging on master [pass]
Are there binlog exceptions? [pass]
Replication user exists? [pass]
Checking server_id values [pass]
Checking server_uuid values [pass]
Is slave connected to master? [pass]
Check master information file [pass]
Checking InnoDB compatibility [pass]
Checking storage engines compatibility [pass]
Checking lower_case_table_names settings [pass]
Checking slave delay (seconds behind master) [FAIL]

Slave is NNN seconds behind master.

# ...done.

$ mysqlrplcheck --master=m_account@192.168.1.1:13001 \
--slave=slave_acc2@192.168.1.3:3306
```

```

# master on 192.168.1.1: ... connected.
# slave on 192.168.1.3: ... connected.
Test Description                                Status
-----
Checking for binary logging on master          [pass]
Are there binlog exceptions?                  [pass]
Replication user exists?                      [pass]
Checking server_id values                    [pass]
Checking server_uuid values                 [pass]
Is slave connected to master?                [pass]
Check master information file               [pass]
Checking InnoDB compatibility              [pass]
Checking storage engines compatibility   [pass]
Checking lower_case_table_names settings [pass]
Checking slave delay (seconds behind master) [FAIL]

Slave is N seconds behind master.

# ...done.

```

15.4.3.1.3. Discussion

In the above example we made use of the `mysqlreplicate` utility to setup a two layer replication topology, where the existing server is now the master for the two new servers which will act as slaves. Notice how we used the address of the old existing server in the `--master` [390] option and in the `--slave` [390] option we used the addresses of the new servers. Also notice the use of the `-b` flag, this makes replication start from the first event recorded in the master's binary log.

Also notice how we used the `mysqlrplcheck` utility to check the health of the replication. In this case, the failing test "Check slave delay" is expected, since the slaves are catching up with the master. When the slaves have read and applied all the transactions from the master's binary log the "Check slave delay" test will pass. Also, in case the slave wasn't properly configured and pointing to the master specified the "Is slave connect to master" test would notify us of that with a FAIL or WARN status.

15.4.3.1.4. Permissions Required

As for the required privileges to run these utilities:

The `m_account` user needs the following privileges for the `mysqlreplicate`: SELECT and INSERT privileges on `mysql` database, REPLICATION SLAVE, REPLICATION CLIENT and GRANT OPTION. As for the `slave_acc` users, they need the SUPER privilege. The `repl` user, used as the argument for the `--rpl-user` [378] option, is either created automatically or if it exists, it needs the REPLICATION SLAVE privilege.

Also, when using GTIDs, the `slave_acc` users must also have SELECT privilege over the `mysql` database in order to run the `mysqlrplcheck` utility successfully.

15.4.3.1.5. Tips and Tricks

In the `mysqlreplicate` utility we could have also used the `--test-db` option which creates a dummy database to test the replication setup. However, the `mysqlrplcheck` provides more detailed information in that regard.

As previously stated, the `-b` option tells the utility to start replication from the first event recorded in the master's binary log. Omitting this flag, in turn, makes the slaves replicate only what is stored in the master's binary log from the present moment onward.

Furthermore, using the `--master-log-file` [378] and `--master-log-pos` [378] options it is possible to specify respectively the master log file and the master log position from which the slave will start its replication process.

The `-p` flag can be used to ensure that the replication setup is only executed in case the storage engines match in both the master and the slave.

Regarding the `mysqlrplcheck` utility, we can use the `-s` option to check the output of the `show slave status` command. This can be useful for instance to check what might be causing the "Is slave connected" test to fail. We can also use the `--master-log-file` [378] option to specify the name of the master information file to read.

Lastly, we can use the `--verbose` [378] option in order to get more information about what is happening "under the hood".

15.4.3.2. How Do I Add New Servers to My Topology and Change Master Role

We will examine a scenario similar to the previous one where we want to make one of the two new servers added the master server (perhaps because it has better specs and is faster).

15.4.3.2.1. Objectives

Our goal in this example is to create replication configuration with 3 servers, two new ones and an existing one, and we want to replicate all the information, but make one of the new servers the master server.

Like the previous example, let's assume that the existing server, Server1, is running on port 13001 on the local machine with IP 192.168.1.1 that the two new machines with MySQL server instances are Server2 running on 192.168.1.2:13001 and Server3 running on 192.168.1.3:3306. We want to make Server2 the new master.

15.4.3.2.2. Example Execution

```
$ mysqlreplicate --master=m_account@192.168.1.1:13001 \
--slave=slave_accl@192.168.1.2:13001 --rpl-user=repl:slavepass -b
# master on 192.168.1.1: ... connected.
# slave on 192.168.1.2: ... connected.
# Checking for binary logging on master...
# Setting up replication...
# ...done.

$ mysqlreplicate --master=m_account@192.168.1.1:13001 \
--slave=slave_acc2@192.168.1.3:3306 --rpl-user=repl:slavepass -b
# master on 192.168.1.1: ... connected.
# slave on 192.168.1.3: ... connected.
# Checking for binary logging on master...
# Setting up replication...
# ...done.

$ mysqlrpladmin --master=m_account@192.168.1.1:13001 \
--slaves=slave_accl@192.168.1.2:13001,slave_acc2@192.168.1.3:3306 health

# Checking privileges.
#
# Replication Topology Health:
+-----+-----+-----+-----+-----+
| host      | port    | role     | state   | gtid_mode | health   |
+-----+-----+-----+-----+-----+
| 192.168.1.1 | 13001  | MASTER   | UP     | ON        | OK
| 192.168.1.2 | 13001  | SLAVE    | UP     | ON        | Slave delay is NNN seconds
| 192.168.1.3 | 3306   | SLAVE    | UP     | ON        | Slave delay is NNN seconds
+-----+-----+-----+-----+-----+
# ...done.

$ mysqlrpladmin --master=m_account@192.168.1.1:13001 \
--slaves=slave_accl@192.168.1.2:13001,slave_acc2@192.168.1.3:3306 health

# Checking privileges.
```

```

#
# Replication Topology Health:
+-----+-----+-----+-----+-----+-----+
| host | port | role | state | gtid_mode | health |
+-----+-----+-----+-----+-----+-----+
| 192.168.1.1 | 13001 | MASTER | UP | ON | OK |
| 192.168.1.2 | 13001 | SLAVE | UP | ON | OK |
| 192.168.1.3 | 3306 | SLAVE | UP | ON | OK |
+-----+-----+-----+-----+-----+-----+
# ...done.

$ mysqlrpladmin --master=m_account@192.168.1.1:13001 \
--slaves=slave_accl@192.168.1.2:13001,slave_acc2@192.168.1.3:3306 \
--new-master=slave_accl@localhost:13002 --demote-master switchover
# Checking privileges.
# Performing switchover from master at 192.168.1.1:13001 to slave at 192.168.1.2:13001.
# Checking candidate slave prerequisites.
# Checking slaves configuration to master.
# Waiting for slaves to catch up to old master.
# Stopping slaves.
# Performing STOP on all slaves.
# Demoting old master to be a slave to the new master.
# Switching slaves to new master.
# Starting all slaves.
# Performing START on all slaves.
# Checking slaves for errors.
# Switchover complete.
#
# Replication Topology Health:
+-----+-----+-----+-----+-----+-----+
| host | port | role | state | gtid_mode | health |
+-----+-----+-----+-----+-----+-----+
| 192.168.1.2 | 13001 | MASTER | UP | ON | OK |
| 192.168.1.1 | 13001 | SLAVE | UP | ON | OK |
| 192.168.1.3 | 3306 | SLAVE | UP | ON | OK |
+-----+-----+-----+-----+-----+-----+

```

15.4.3.2.3. Discussion

As with our previous scenario we used the `mysqlreplicate` utility to set up a replication topology between the existing server and the two new servers. Notice the use of the `-b` flag which this replication start from the first event recorded in the master's binary log.

After setting our replication topology we made use of the `mysqlrpladmin` utility specifying both the master and slave servers and using the `health` command to check the status of the replication. Since our master server had lots of information, it is normal for the new slaves to take some time to catch up, thus the slave delay message on the `health` column of the output.

However, if all goes well, after some time the slaves will eventually catch up, and when that happens, the `heath` column will show an `OK` status.

When this happened, we used the `mysqlrpladmin` utility yet again, this time with `switchover` command. Using the `--new-master` [384] option we specify the server that will become the new master. We can't forget the `--demote-master` [383] option which turns the old master into a slave, otherwise it would still behave as a master just without any slaves, Server3 will become a slave of Server2.

After the switchover, Server2 becomes the master server for both Server1 and Server3 which are now the slaves.

15.4.3.2.4. Permissions Required

The `m_account` user needs the following privileges for the `mysqlreplicate`: `SELECT` and `INSERT` privileges on `mysql` database, `REPLICATION SLAVE`, `REPLICATION CLIENT` and `GRANT OPTION`. As

for the slave_acc users, they need the SUPER privilege. The repl user, used as the argument for the --rpl-user [384] option, is either created automatically or if it exists, it needs the REPLICATION SLAVE privilege.

To run the `mysqlrpladmin` utility with the health command, the m_account used on the master needs an extra SUPER privilege.

As for the switchover command all the users need the following privileges: SUPER, GRANT OPTION, SELECT, RELOAD, DROP, CREATE and REPLICATION SLAVE

15.4.3.2.5. Tips and Tricks

We can use the --discover-slaves-login [383] option for `mysqlrpladmin` in order to detect the slaves automatically instead of manually specifying the slaves.

The `mysqlrpladmin` utility allows users to specify a script to execute before and after the failover and switchover operations using the --exec-before [383] and --exec-after [383] options respectively. Note that the script specified using the exec-after option only runs in case the switchover/failover executes successfully.

We can use the `mysqlrpladmin` utility to start and stop all the slaves with the start/stop commands. Using the stop command only stops servers that are actually slaves of the specified master thus preventing us from stopping unwanted servers.

15.4.3.3. Setup Automatic Failover

Once your replication topology is setup, it is important to consider the possible occurrences of failures in order to maintain the high availability level of your system. Several failures independently from their cause (network connection issue, hard drive crash, cosmic lightning, etc.) can stop the replication process by making the master no longer accessible by its slaves.

In this type of situation, it is desirable to promote one of the slaves to master while the problem with the old master is being solved as it can take some considerable time. It will be even better to have an application to monitor the replicate topology and automatically perform failover, minimizing downtime and keeping replication running.

15.4.3.3.1. Objectives

The goal is to start the `mysqlfailover` utility to monitor a replication topology and perform failover automatically when required.

When the current master fails, manually promoting a slave to the new master can be a very tedious and error prone task, as all the remaining slave have to point out to the new master and the new master needs to catch up with all the slaves to make sure that no transactions are lost.

Fortunately, the `mysqlfailover` utility is capable of executing this full process automatically and in a optimized way.

Let's assume that a replication topology with one master (server1:3311) and four slaves (server2:3312, server3:3313, server4:3314, server5:3315) was previously setup.

15.4.3.3.2. Example Execution

Start the `mysqlfailover` utility (in console mode - default):

```
$ mysqlfailover --master=root@server1:3311 \
```

```
--slaves=root@server2:3312,root@server3:3313,root@server4:3314,root@server5:3315 \
--log=log.txt --rpl-user=rpl:rpl
NOTE: Log file 'log.txt' does not exist. Will be created.
# Checking privileges.

MySQL Replication Failover Utility
Failover Mode = auto      Next Interval = Fri Jul 26 10:17:52 2013

Master Information
-----
Binary Log File      Position  Binlog_Do_DB  Binlog_Ignore_DB
master-bin.000001    151

GTID Executed Set
None

Replication Health Status
+-----+-----+-----+-----+-----+
| host   | port  | role   | state  | gtid_mode | health |
+-----+-----+-----+-----+-----+
| server1 | 3311  | MASTER | UP     | ON       | OK      |
| server2 | 3312  | SLAVE  | UP     | ON       | OK      |
| server3 | 3313  | SLAVE  | UP     | ON       | OK      |
| server4 | 3314  | SLAVE  | UP     | ON       | OK      |
| server5 | 3315  | SLAVE  | UP     | ON       | OK      |
+-----+-----+-----+-----+-----+

Q-quit R-refresh H-health G-GTID Lists U-UUIDs L-log entries
```

Now imagine that the master crashed or is no longer reachable, then after a predefined time interval (by default 15 seconds) we can observe that the failover process will start automatically:

```
Failover starting in 'auto' mode...
# Candidate slave server2:3312 will become the new master.
# Checking slaves status (before failover).
# Preparing candidate for failover.
# Creating replication user if it does not exist.
# Stopping slaves.
# Performing STOP on all slaves.
# Switching slaves to new master.
# Disconnecting new master as slave.
# Starting slaves.
# Performing START on all slaves.
# Checking slaves for errors.
# Failover complete.

Failover console will restart in 5 seconds.

[...]

MySQL Replication Failover Utility
Failover Mode = auto      Next Interval = Fri Jul 26 10:25:17 2013

Master Information
-----
Binary Log File      Position  Binlog_Do_DB  Binlog_Ignore_DB
master-bin.000001    151

GTID Executed Set
None

Replication Health Status
+-----+-----+-----+-----+-----+
| host   | port  | role   | state  | gtid_mode | health |
+-----+-----+-----+-----+-----+
| server2 | 3312  | MASTER | UP     | ON       | OK      |
| server3 | 3313  | SLAVE  | UP     | ON       | OK      |
```

server4	3314	SLAVE	UP	ON	OK	
server5	3315	SLAVE	UP	ON	OK	

Q-quit R-refresh H-health G-GTID Lists U-UUIDs L-log entries

15.4.3.3.3. Discussion

The above example illustrates how to start the `mysqlfailover` utility to check the health of the replication topology and shows the displayed output when failover occurs.

Basically, we simply need to specify the master's connection with the `--master` [359] option, the list of slaves with the `--slaves` [360] option and the replication user (login and password) using the `--rpl-user` options. In alternative to the `--slaves` [360] options, one can use the `--discover-slaves-login` [358] specifying a user and password (or login-path) to connect to the slaves and the utility will attempt to discover all the slaves connected with the master using the specified login. For the above example, '`--discover-slaves-login=root`' could be used.

The `--discover-slaves-login` [358] can be very handy especially if there is a huge number of slaves in the topology, but bear in mind that the explicit specification of slaves is safer and that discovery can fail to find some servers. In particular, it is important to note that in order for slaves to be discovered, they must be started with the '`--report-host`' and '`--report-port`' options with appropriate values and they must be correctly connected to the master (IO thread running) otherwise discovery will fail.

It is also recommended to use the `--log` [358] options to specify a file to register all events, warning and errors. This is useful to keep a record of what happened, for example to later determine when failover occurred and if the process occurred without any errors.

An important matter to discuss is the order in which the servers are selected as candidates for failover. No distinction is made in terms of the number of transactions to select the most up-to-date slave to become the new master. The reason is very simple, this criteria is non-deterministic as many circumstances (i.e., network load, server maintenance operations) can temporarily influence the performance of a slave and could lead to an incorrect selection of the most appropriate candidate. For example, the slave with the best hardware should be in the long run the most appropriate candidate to become the new master, but for some unanticipated reason it might actually have had less transactions than other servers when the master crashed. Therefore, a more deterministic criteria based on the order in which the servers are specified is used, allowing the user to control the order in which the candidates are selected. The first server that will meet the required election criteria, consisting of simple sanity checks (server reachable and running with the required options: GTID ON and binary logging enabled), will be chosen. More specifically, the section of the new master will follow this order: first, sequentially check the list of servers specified by the `--candidates` [358] option, then the servers listed in the `--slaves` [360] option, finally check any discovered slave in an unordered way.

15.4.3.3.4. Permissions Required

You must have permissions to configure replication.

15.4.3.3.5. Tips and Tricks

In the above example the `mysqlfailover` utility was started in the default console mode, but it can also be executed as a daemon. For that purpose, the `--daemon` [358] option needs to be used, more specifically simply add '`--daemon=start`' to the command line. When `mysqlfailover` is executed as a daemon, no output is displayed and all the information is logged to file specified for the `--log` [358] option which is mandatory in this case. To stop the execution of the `mysqlfailover` daemon simply invoke the utility using the option '`--daemon=stop`'. No other options are required to stop the daemon, unless

a specific pidfile (to store the process PID) was specified with the [--pidfile](#) [359] option to start the daemon and in this case the same option value is also required to stop it.

Another useful feature is the possibility to run external script along the execution of the utility to perform customized actions. The following options can be used to execute different scripts at distinct moments of the `mysqlfailover` execution: [--exec-fail-check](#) [358] to specify a script to run periodically at each predefined interval instead of the default check (i.e., master is reachable and alive) to detect the need to failover, [--exec-before](#) [358] to specify a script to execute before starting failover, [--exec-after](#) [358] to execute a script at the end of failover process, [--exec-post-failover](#) [358] to run a script after completing the failover process (before displaying the health report).

15.4.3.4. Restore the Previous Master After Failover

After a successful failover it is sometime required to restore the initial topology and promote the crashed server to master again (or even a new server with distinctive hardware characteristics). Sometimes failover can be triggered by simple network issue (not affecting the health of the initial master server) and after being resolved it might be desirable to put the old master back in the replication topology.

15.4.3.4.1. Objectives

The goal of this task is simply to replace the new master of a replication topology with the previous one that might have been demoted as result of successful automatic failover execution due to some failure. It is assumed that the server to be restored as master is healthy and any previous issue (that triggered failover) have been resolved.

Let's consider the previous topology after failover, now with a new master (server2:3312) and three slaves (server3:3313, server4:3314, server5:3315) and that we want to promote the initial server (server1:3311) to master again.

Perform this task manually can be delicate as one wrong or missing step can lead to errors and incorrect replication topology or even to the lost of some transaction. Once more MySQL Utilities can provide a precious assistance to perform this task, in this case requiring the user to follow three simple steps to restore the initial topology as showed bellow.

15.4.3.4.2. Example Execution

There are several steps involved in solving this problem. We walk through each in turn.

You must first stop running the `mysqlfailover` utility instance and start the (old) master to be restored, i.e. server1:3311.

Next, set the old master (server1:3311) as a slave of the current new master (server2:3312):

```
$ mysqlreplicate --master=root@server2:3312 --slave=root@server1:3311 \
--rpl-user=rpl:rpl
# master on localhost: ... connected.
# slave on localhost: ... connected.
# Checking for binary logging on master...
# Setting up replication...
# ...done.
```

Next, switchover to the previous master to restore the initial replication topology:

```
$ mysqlrpladmin --master=root@server2:3312 \
--slaves=root@server2:3313,root@server4:3314,root@server5:3315 \
--rpl-user=rpl:rpl --new-master=root@server1:3311 --demote-master switchover
```

```

# Checking privileges.
# Performing switchover from master at server2:3312 to slave at server1:3311.
# Checking candidate slave prerequisites.
# Checking slaves configuration to master.
# Waiting for slaves to catch up to old master.
# Stopping slaves.
# Performing STOP on all slaves.
# Demoting old master to be a slave to the new master.
# Switching slaves to new master.
# Starting all slaves.
# Performing START on all slaves.
# Checking slaves for errors.
# Switchover complete.
#
# Replication Topology Health:
+-----+-----+-----+-----+-----+
| host | port | role | state | gtid_mode | health |
+-----+-----+-----+-----+-----+
| server1 | 3311 | MASTER | UP | ON | OK |
| server2 | 3312 | SLAVE | UP | ON | OK |
| server3 | 3313 | SLAVE | UP | ON | OK |
| server4 | 3314 | SLAVE | UP | ON | OK |
| server5 | 3315 | SLAVE | UP | ON | OK |
+-----+-----+-----+-----+-----+
# ...done.

```

The initial replication topology is now restored and `mysqlfailover` can be restarted (but using --force) as initially:

```

$ mysqlfailover --master=root@server1:3311 \
--slaves=root@server2:3312,root@server3:3313,root@server4:3314,server5:3315 \
--log=log.txt --rpl-user=rpl:rpl --force
# Checking privileges.

MySQL Replication Failover Utility
Failover Mode = auto      Next Interval = Sat Jul 27 02:17:12 2013

Master Information
-----
Binary Log File      Position  Binlog_Do_DB  Binlog_Ignore_DB
master-bin.000002    151

GTID Executed Set
None

Replication Health Status
+-----+-----+-----+-----+-----+
| host | port | role | state | gtid_mode | health |
+-----+-----+-----+-----+-----+
| server1 | 3311 | MASTER | UP | ON | OK |
| server2 | 3312 | SLAVE | UP | ON | OK |
| server3 | 3313 | SLAVE | UP | ON | OK |
| server4 | 3314 | SLAVE | UP | ON | OK |
| server5 | 3315 | SLAVE | UP | ON | OK |
+-----+-----+-----+-----+-----+
Q-quit R-refresh H-health G-GTID Lists U-UUIDs L-log entries

```

15.4.3.4.3. Discussion

In reality, the main and most important step is the execution of the `switchover` command with the `mysqlrpladmin` utility. The previous steps can be seen as a preparation for switchover. The first step simply makes sure that the server is running and that there is no `mysqlfailover` instance still running that could affect the correct execution of switchover. The second step sets the old master as a slave of the new master, because the `switchover` command can only be performed with slaves. This step will

also allows the old master to catch up with the new master. If many transaction have been performed on the new master it is recommended to wait a while here to let the old master catch up before switchover, otherwise the switchover command might take too much time.

As expected, the execution of the switchover command requires the specification of the current and new master with the `--master` [384] and `--new-master` [384] options, as well as the list of slaves in the topology using the `--slaves` [384] option (without need to list the new master). The replication user is specified with the `--rpl-user` [384] option. In this specific example, the use of the option `--demote-master` [383] is important, because without it the current master (server2:3312) will not be demoted and set as a slave of the new master (server1:3311).

The `mysqlrpladmin` utility will execute and display information about all required action to perform switchover. After completing the switchover process, an health report is displayed where it is possible to confirm the successful execution of the command and verify that the topology has changed as expected.

After completing these simple steps, the replication topology is back to its initial structure (before failover) with its old master. Therefore, `mysqlfailover` is ready to be executed again to monitor the topology and enable automatic failover. Note that in this particular situation, the use of the `--force` [358] option might be required because it is likely that some registration data from the previous failover instance execution might have been left on the recovered master (due to its previous crash).

The `mysqlfailover` utility registers its execution on the servers in order to avoid concurrent executions of the utility which will likely lead to errors and inconsistent state during failover. If the utility detects that another instance might be running, it will be started in "fail" mode (not taking any action when it detects that the master failed). The `mysqlfailover` instance registration is cleared when the utility exits, and it is expected that unregistration fails on crashed or not accessible servers. The `--force` [358] option overwrites the instance execution check allowing to surpass unregistration failure on (crashed) old masters, allowing the `mysqlfailover` utility to start in 'auto' mode.

15.4.3.4.4. Permissions Required

You must have permissions to configure replication.

15.4.3.4.5. Tips and Tricks

It is important to wait for the old master to catch up with the new master in order to guarantee that no transactions are lost. Depending on the time the old master was down or not accessible it might take a considerable time for the old master to execute all missing transactions. MySQL utilities provide tools that allow the visualizations of the slaves status, namely the 'health' command of the `mysqlrpladmin` utility.

An alternative set of steps could have been followed to perform the desired task, using the failover command from `mysqlrpladmin` instead of switchover. In this case, the old master should be specified in the candidates list using the option `--candidates` [358] to be chosen as the preferred slave to become the new master (no need for the `--master`, `--new-master` and `--demote-master` options). However, an additional step will be required to set the previous master (server2:3312) as a slave of the old master (server1:3311) using the `mysqlreplicate` utility, because failover will not demote the previous master as it assumes that it is not available. Notice that unlike switchover that will fail if the server specified by the `--new-master` [384] option does not meet the requirements to become master, failover will choose another server from the slaves list to become the new master if the one specified in by the `--candidates` [358] option is not suitable. It is important to keep this behavior differences in mind when deciding which command to apply.

15.4.3.5. How Can I Find All of the Slaves Attached to My Master Server?

When you have a topology that has grown over time - slaves have been added from time-to-time, it may not be so easy to remember which servers are connected as slaves.

Most often you want to know the state of those slaves at-a-glance. Rather than connect to each slave individually, it would be nice to know what the state of each slaves threads using a single command.

15.4.3.5.1. Objectives

Show a map of the slaves connected to a master including the state of each slaves threads (IO and SQL). We can do this with a single command using the `mysqlrplshow` utility.

15.4.3.5.2. Example Execution

```
$ mysqlrplshow --master=root:root@localhost:13001 \
                --disco=root:root --verbosity
# master on localhost: ... connected.
# Finding slaves for master: localhost:13001

# Replication Topology Graph
localhost:13001 (MASTER)
|
+--- localhost:13002 [IO: Yes, SQL: Yes] - (SLAVE)
|
+--- localhost:13003 [IO: Yes, SQL: Yes] - (SLAVE)
|
+--- localhost:13004 [IO: Yes, SQL: Yes] - (SLAVE)
|
+--- localhost:13005 [IO: Yes, SQL: Yes] - (SLAVE)
```

15.4.3.5.3. Discussion

Notice the use of the `mysqlrplshow` utility. Not only did it show us the slaves attached to the master, it also displayed the state of the IO and SQL thread for each slave.

We used the master server for the `--master` [394] option but for the slaves, we provided the option `--discover-slaves-login` [394] which provides the user name and password for the account used to connect to each slave. Without this, we would not be able to determine if the slave is attached (currently) or the state of its threads.

The `--discover-slaves-login` [394] option applies to all slaves. If you do not have the same user defined on all of your slaves, you can use the `--prompt` [394] option to prompt for the user and password for each slave.

To get the state of the slave threads, use the `--verbose` [394] option.

15.4.3.5.4. Permissions Required

The user connected to the master must have the REPLICATION SLAVE privilege.

The user specified with the `--discover-slaves-login` [394] option that logs into each slave must have the REPLICATION CLIENT privilege.

15.4.3.5.5. Tips and Tricks

You can also display multiple tiered topologies by providing the `--recurse` [394] option.

Notice in the example we used the option `--discover-slaves-login` [394]. This is a shortcut feature built into every utility. If you type the first N letters of a utility that uniquely identifies it among the options for said utility, the utility accepts it as if you typed the entire string. For example, the full name of the option we used is `--discover-slaves-login` [394].

15.4.4. Server Operations

The tasks described in this section are those used to perform server-wide operations such as cloning a server instance.

15.4.4.1. How Do I Make A Temporary Copy Of My Server For Testing?

When diagnosing a problem or needing to experiment with a server for developing new features or test modifications, you often need a duplicate of your running server so that you can ensure your solution works for the actual server. It would be really convenient if we had a process to make a copy of a running server for such processes.

While it is possible and indeed popular to use replication to replicate all of your data to multiple slaves and use one of the slaves for these purposes, cases where you are working with a particular server or if replication is not in use you will need some way to duplicate not only the data but also the server and its startup parameters.

15.4.4.1.1. Objectives

Create a new instance of a running server complete with the same options and the same data.

15.4.4.1.2. Example Execution

To meet this objective, we need to use several utilities. But before we get started, we need to know what specific options the host server is using. To do this, we use the `mysqlserverinfo` utility to discover the configuration file and the `my_print_defaults` tool to print the defaults. We can also show the process id to see what command-line options are being used. We get this from using the `--show-servers` [400] option with `mysqlserverinfo`. On POSIX systems, we can use the `ps` command to find the command line options.

```
$ mysqlserverinfo --server=root:root@localhost \
--format=vertical --show-servers
#
# The following MySQL servers are active on this host:
# Process id: 2377, Data path: /usr/local/mysql/data
# Process id: 2478, Data path: /Volumes/Source/source/temp_13001
# Process id: 2487, Data path: /Volumes/Source/source/temp_13002
#
# Source on localhost: ... connected.
***** 1. row *****
    server: localhost:3306
    version: 5.1.50-log
    datadir: /usr/local/mysql/data/
    basedir: /usr/local/mysql-5.1.50-osx10.6-x86_64/
    plugin_dir: /usr/local/mysql-5.1.50-osx10.6-x86_64/lib/plugin
    config_file: /etc/my.cnf
    binary_log: my_log.000287
    binary_log_pos: 106
    relay_log: None
    relay_log_pos: None
1 row.
#...done.
$ my_print_defaults mysqld /etc/my.cnf
--port=3306
--basedir=/usr/local/mysql
--datadir=/usr/local/mysql/data
--server_id=5
--log-bin=my_log
--general_log
--slow_query_log
--innodb_data_file_path=ibdata1:778M;ibdata2:50M:autoextend
$ ps -f 2377
```

```
UID PID PPID C STIME TTY TIME CMD
74 2377 2300 0 10:56AM ?? 0:02.04 /usr/local/mysql/bin/mysqld --basedir=/usr/local/mysql --data
```

Notice we now have all of the options from the configuration file as well as the startup options. We can now construct the proper options for creating a clone of this server using the `mysqlserverclone` utility. Specifically, we can set the following options using the `--mysqld` [397] option:

- `--log-bin=my_log`
- `--general_log`
- `--slow_query_log`
- `--user=mysql`
- `--log-error=<path>`

Using these options and choosing a new data directory, we can create a new instance of the host server using the following command.

```
$ mysqlserverclone --server=root:root@localhost \
--new-data=/source/temp_clone --new-port=3307 --root=root --delete \
--new-id=123 --mysqld="--log-bin=my_log --general-log --slow-query-log \
--user=mysql --log-error=/source/temp_clone"
# Cloning the MySQL server running on localhost.
# Creating new data directory...
# Configuring new instance...
# Locating mysql tools...
# Setting up empty database and mysql tables...
# Starting new instance of the server...
# Testing connection to new instance...
# Success!
# Setting the root password...
# Connection Information:
# -uroot -proot --socket=/source/temp_clone/mysql.sock
#...done.
```

Now that we have a running instance, we can export all of the data from the host to the clone.

```
$ mysqldbexport --server=root:root@localhost:3306 --export=both --all > data.sql
$ mysqldbimport --server=root:root@localhost:3307 --import=both data.sql
# Source on localhost: ... connected.
# Importing definitions and data from data.sql.
#...done.
```

15.4.4.1.3. Discussion

As you can see, this is a multiple step process. We saw examples of using the `mysqlserverinfo`, `mysqlserverclone`, `mysqldbexport`, and `mysqldbimport` utilities.

Notice in the example we used port 3307 for the clone which is reflected in the `mysqldbimport` utility `--server` [345] option.

15.4.4.1.4. Permissions Required

You must have permission to read all databases. Since we are using the root account for these examples (and you typically would), permissions are not generally a problem.

You also need permissions to create the new data directory and write data to it.

15.4.4.1.5. Tips and Tricks

If you want to copy all of the users and their permissions, check out the `mysqluserclone` utility.

15.4.4.2. How Can I Find What MySQL Servers Are Running?

One of the challenges for a database administrator or database developer when working with a development server that has multiple instances of MySQL running is knowing exactly how many are running.

In some cases, this may have come about by accident but mostly having multiple instances of MySQL running is intentional. Whichever the case, it would be nice to be able to use a single command to find all of the MySQL processes.

15.4.4.2.1. Objectives

Use the `mysqlserverinfo` utility to locate all of the MySQL processes running on a host.

15.4.4.2.2. Example Execution

```
$mysqlserverinfo --show-servers --server=root:root@localhost \
--format=vertical
#
# The following MySQL servers are active on this host:
# Process id: 3007, Data path: /usr/local/mysql/data
# Process id: 8191, Data path: /Volumes/Source/source/temp_13001
# Process id: 8196, Data path: /Volumes/Source/source/temp_13002
# Process id: 8201, Data path: /Volumes/Source/source/temp_13003
# Process id: 8207, Data path: /Volumes/Source/source/temp_13004
# Process id: 8212, Data path: /Volumes/Source/source/temp_13005
#
# Source on localhost: ... connected.
***** 1. row *****
    server: localhost:3306
    version: 5.1.50-log
    datadir: /usr/local/mysql/data/
    basedir: /usr/local/mysql-5.1.50-osx10.6-x86_64/
    plugin_dir: /usr/local/mysql-5.1.50-osx10.6-x86_64/lib/plugin
    config_file: /etc/my.cnf
    binary_log: my_log.000286
    binary_log_pos: 237
    relay_log: None
    relay_log_pos: None
1 row.
#...done.
```

15.4.4.2.3. Discussion

The `mysqlserverinfo` utility is normally used to find information about a particular server. We can see such results in the example above.

However, the utility also has an option, `--show-servers` [400] that displays a list of all of the MySQL server process ids that are executing on the host. This quick glance can help diagnose problems with multiple instances on the same machine.

15.4.4.2.4. Permissions Required

The permissions required include the ability to read the mysql database and to have read access to the data directory.

15.4.4.2.5. Tips and Tricks

Notice the output shows the data directory for each server. You can use this information to examine the files in that folder to discern more information such as what databases exist and find and examine the binary log, etc.

On POSIX systems, you can discover the command-line arguments such as the port number the server is using with the ps -f PID command. For example, to discover the complete information for PID 2487, you can do the following.

```
$ ps -f 2487
UID  PID  PPID  C STIME   TTY          TIME CMD
501  2487     1  0 10:58AM ttys001      0:00.41 /source/mysql-5.6/sql/mysqld --no-defaults --datadir=/source/
```

15.4.5. Specialized Operations

The tasks described in this section relate to specific situations or configurations and may not apply in the general case. For example, some tasks require a specific commercial plugin. More specifically, the following tasks are those for use with the Audit Log Plugin.

15.4.5.1. How Do I Record Only Login Events?

The audit log plugin records MySQL servers activity. By default, it is set to write all auditable events to the log file which can represent a considerable amount of information. Fortunately, it is possible to control the type of information that is written to the audit log file, changing the audit log plugin's policy. The policy should be set to log only the events of interest, avoiding wasting resources to log unnecessary events.

In particular, if the audit log plugin is only used to monitor access to the database server (for security purposes) then only the login events need to be recorded. The `mysqlauditadmin` utility allows us to perform such change in a simple way (as well as changes other settings).

15.4.5.1.1. Objectives

The goal is to set the audit log plugin to only write the login events to the log file. It is assumed that the audit log plugin is enabled and running with the default settings (logging all auditable events) on the localhost and default port (3306).

15.4.5.1.2. Example Execution

```
$ mysqlauditadmin --server=root@localhost:3306 policy --value=LOGINS \
--show-options
#
# Showing options before command.
#
# Audit Log Variables and Options
#
+-----+-----+
| Variable_name | Value |
+-----+-----+
| audit_log_buffer_size | 1048576 |
| audit_log_file | audit.log |
| audit_log_flush | OFF |
| audit_log_policy | ALL |
| audit_log_rotate_on_size | 0 |
| audit_log_strategy | ASYNCHRONOUS |
+-----+-----+

#
# Executing POLICY command.
#
#
# Showing options after command.
#
# Audit Log Variables and Options
#
+-----+-----+
| Variable_name | Value |
+-----+-----+
```

audit_log_buffer_size	1048576
audit_log_file	audit.log
audit_log_flush	OFF
audit_log_policy	LOGINS
audit_log_rotate_on_size	0
audit_log_strategy	ASYNCHRONOUS

15.4.5.1.3. Discussion

In order to change the type of events recorded to the audit log file, the policy settings must be changed. This is done with the `mysqlauditadmin` utility using the command 'policy' and specifying the desired policy value with the `--value [313]` option. As expected the specification of the target server is also required using the `--server [313]` option.

In the above example, the policy value was set to LOGINS to write only login events to the log file. Nevertheless, other values are also permitted to control the information written to the log file: ALL (write all events), QUERIES (write only query event), NONE (disable logging), DEFAULT (use the default policy).

15.4.5.1.4. Permissions Required

User must have the SELECT privilege for the mysql database. To view the log file, the user must have read access to the audit log file on the server.

15.4.5.1.5. Tips and Tricks

The policy value was specified in uppcases in this example, however upper and lower cases can be mixed to specify the policy value (e.g. LoGiNs). The values for this command will still be read correctly independently of the used cases (case insensitive), but if an unsupported value is specified an error will be issued.

In the above example the `--show-options [313]` option was additionally used, but it is not required. This option simply displays the audit log settings (variables). However, when this option is combined with a command that changes one of the audit log variables it display the audit log settings before and after the execution of the command which can be very handy to confirm that the desired change was performed as expected.

15.4.5.2. How Do I Copy/Move The Audit Log?

The audit log information can grow quickly and considerably depending on the type of information written to the audit log files and the activity of the MySQL server. Therefore, it might be a good idea to copy the audit log files to a different location and free some storage on the server.

The `mysqlauditadmin` utility also provides this useful functionality.

15.4.5.2.1. Objectives

The goal of this task is to copy an existing audit log file to a different location using the `mysqlauditadmin` utility.

It is assumed that the utility is executed on the destination host which must be a non-Windows system with the scp (Secure CoPy) command line program, and that must have access to the MySQL remote server and its data directory with the provided credentials (user and password). It is also assumed that the specified audit log file exists and user has write privileges on the target directory.

15.4.5.2.2. Example Execution

```
$ mysqlauditadmin --audit-log-name=/MySQL/SERVER/data/audit.log.13753706179878237 \
```

```
copy --copy-to=/ARCHIVE/Audit_Logs --remote-login=user1:server1
# Copying file from server1:/MySQL/SERVER/data/audit.log.13753706179878237 to /ARCHIVE/Audit_Logs:
user1@server1's password:
audit.log.13753706179878237                                         100% 4716      4.6KB/s   00:01
```

15.4.5.2.3. Discussion

The copy operation can be performed with the `mysqlauditadmin` utility using the 'copy' command and requiring the following options: the `--audit-log-name [313]` option to specify the path and filename of the audit log file to copy, the `--copy-to [313]` option to indicate the destination folder, and in this example the `--remote-login [313]` option to specify the user and remote host where the file is located (the user password will be prompted).

The `--remote-login [313]` option is not required if the source and destination location are on the same server where the utility is executed. Moreover, this option is not supported in Windows system where UNC paths should be used instead.

15.4.5.2.4. Permissions Required

The user must have permissions to read the audit log on disk and write the file to the remove location.

15.4.5.2.5. Tips and Tricks

The name of the audit log file (by default 'audit.log') is defined by the 'audit_log_file' variable displayed by `mysqlauditadmin` when using the `--show-options [313]` option. Existing audit log files have a timestamp extension except the one that is currently in use. That being said, it might be useful to know that it is possible to get information about the existing audit log files using `mysqlrpladmin`, for instance to determine which files need to be copied. To get this information use the `--file-stats [313]` option and the `--audit-log-name [313]` option specifying the full path of the current audit log file (i.e., without the timestamp extension). For example:

```
$ mysqlauditadmin --file-stats --audit-log-name=/MySQL/SERVER/data/audit.log
+-----+-----+-----+-----+
| File           | Size    | Created          | Last Modified |
+-----+-----+-----+-----+
| audit.log.13753706179878237 | 4716    | Thu Aug 1 16:23:37 2013 | Thu Aug 1 16:23:37 2013 |
| audit.log       | 6062    | Thu Aug 1 16:24:26 2013 | Thu Aug 1 16:24:26 2013 |
| audit.log.13753705495049727 | 335142503 | Thu Aug 1 16:22:29 2013 | Thu Aug 1 16:22:29 2013 |
+-----+-----+-----+-----+
```

Note that if the a audit log file with the timestamp extension is specified instead in this example for the `--audit-log-name [313]` option, only the information of the specified file will be displayed and not the file statistics of all existing ones.

15.4.5.3. How Do I Show All INSERT and UPDATE Queries That Failed?

Many useful information can be recored in the audit log files and also a considerable amount of it. However, how can someone easily filter this information and search for specific events, for instance in order to determine the possible cause of a problem.

For example, suppose that someone reported that some data changes are missing (INSERT or UPDATE queries failed) and you want to determine what might be the cause of those transaction failures. All queries are recorded to the audit log file, so you just need to get retrieve all queries of a given type that failed (with a MySQL Error) and analyze them.

This can be achieved using common 'grep' command line tools, but likely involves the use of very complex regular expression to filter the desired data. Fortunately, the `mysqlauditgrep` utility allows to perform

this kind of task in a much easier and simple way, taking advantage of the knowledge of the structure and semantics of the audit log files.

15.4.5.3.1. Objectives

The goal is display all INSERT and UPDATE queries that failed (independently of error) from the current audit log file.

It is assumed that the 'audit.log' file exists and is located in the directory '/MySQL/SERVER/data/'. The below example show how easy it is to perform the desired search with the `mysqlauditgrep` utility.

15.4.5.3.2. Example Execution

```
$ mysqlauditgrep --query-type=INSERT,UPDATE --status=1-9999 /MySQL/SERVER/data/audit.log
+-----+-----+-----+
| STATUS | TIMESTAMP          | NAME   | SQLTEXT
+-----+-----+-----+
| 1046   | 2013-08-01T18:20:46 | Query   | INSERT INTO tbl_not_exist (a,b,c) VALUES(1,2,3)
| 1146   | 2013-08-01T18:21:03 | Query   | INSERT INTO mysql.tbl_not_exist (a,b,c) VALUES(1,2,3)
| 1054   | 2013-08-01T18:23:10 | Query   | INSERT INTO test.t1 (a,b,not_col) VALUES(1,2,3)
| 1146   | 2013-08-01T18:26:14 | Query   | UPDATE tbl_not_exist SET a = 1
| 1054   | 2013-08-01T18:26:53 | Query   | UPDATE test.t1 SET not_col = 1
+-----+-----+-----+
| CONNE
| 37
| 37
| 37
| 37
| 37
+-----+-----+-----+
```

15.4.5.3.3. Discussion

As expected, the use of the `mysqlauditgrep` utility requires the specification of the target audit log file to search and a few options corresponding to the needed search criteria. In this case, the `--query-type` [320] option was used to restrict the displayed results to specific types of queries (i.e., only INSERT and UPDATE), and the `--status` [320] option was used to specify the considered MySQL error codes (i.e., all ranging from 1 to 9999).

The `--query-type` [320] option allows the specification of a comma separated list of different SQL commands. Apart from INSERT and UPDATE the list of supported values for this option also includes: CREATE, ALTER, DROP, TRUNCATE, RENAME, GRANT, REVOKE, SELECT, DELETE, COMMIT, SHOW, SET, CALL, PREPARE, EXECUTE, DEALLOCATE

The `--status` [320] option accepts a comma-separated list of non-negative integers (corresponding to MySQL error codes) or intervals marked with a dash. For example: 1051,1100-1199,1146. In this particular case, the range value 1-9999 was used to include all MySQL error codes and display all unsuccessful commands. To retrieve only successful command (no errors) simply use the value 0 for the `--status` [320] option.

15.4.5.3.4. Permissions Required

The user must have permissions to read the audit log on disk.

15.4.5.3.5. Tips and Tricks

The value specified for the `--query-type` [320] option are case insensitive, therefore you can mix lower and upper case to specify the list of query types. For example, 'insert,Update' will produce the same result as using 'INSERT,UPDATE'. Of course the use of non-supported values will raise an appropriate error.

Many other options and search criteria are provided by the `mysqlauditgrep` utility, check them in order to use the more appropriate one to meet your needs. Note that the utility provides the `--pattern` [320] option to search entries in the audit log file using regular expressions, like common grep tools. By default, this option will use standard SQL pattern matching (used by 'LIKE' comparison operator), unless the `--regexp` [320] option is used to allow more powerful standard regular expressions (POSIX extended).

15.4.5.4. How Do I Display Connections by the User 'root' and Show the Result in CSV Format?

The audit log plugin can be used to record information about different type of events which one might need to monitor or keep a record in a different format. For example, a security record with the list of all logins performed to the database serve might need to be kept to later track the responsible for some change. Moreover, the retrieved information might need to be converted to a specific format (e.g., CSV) to feed another application.

15.4.5.4.1. Objectives

The goal of this task is to retrieve from the audit log the information of all the connections established by the root user to the MySQL Server, and display the resulting information in the comma-separated-value (CSV) format.

Besides the search/filter functionalities using different criteria, the `mysqlauditgrep` utility also provides a feature to display the resulting information in different formats (including CSV). This allows this task to be performed easily with in a single step.

It is assumed that the 'audit.log' file exists and is located in the directory '/MySQL/SERVER/data/'.

15.4.5.4.2. Example Execution

```
$ mysqlauditgrep --user=root --event-type=Connect --format=CSV /MySQL/SERVER/data/audit.log
STATUS,NAME,TIMESTAMP,CONNECTION_ID,HOST,USER,PRIV_USER,IP
0,Connect,2013-08-01T15:24:26,33,localhost,root,root,127.0.0.1
0,Connect,2013-08-01T15:24:26,34,localhost,root,root,127.0.0.1
0,Connect,2013-08-01T15:24:26,35,localhost,root,root,127.0.0.1
0,Connect,2013-08-01T15:24:26,36,localhost,root,root,127.0.0.1
0,Connect,2013-08-01T18:18:43,37,localhost,root,root,127.0.0.1
0,Connect,2013-08-01T18:49:46,38,,root,root,192.168.1.104
1045,Connect,2013-08-01T19:18:08,39,localhost,root,,127.0.0.1
```

15.4.5.4.3. Discussion

To perform this operation the `mysqlauditgrep` utility requires the indication of the target audit log file as expected, two criteria search options, and one formating option to convert the output to the desired format. In this case, the `--users` [320] option was applied to search the records for the specified user (i.e., "root") and the `--event-type` [319] option to retrieve only event of a specific type (i.e., "connect"). The `--format` [320] option is the one used to define the output format of the obtained search results.

In this example, only the "Connect" value was used for the `--event-type` [319] option which correspond to the logging in event (when a client connects). Nevertheless, this option accepts a comma separated list of event types with the following supported values (beside "Connect"): Audit, Binlog Dump, Change user, Close stmt, Out, Connect, Create DB, Daemon, Debug, Delayed, insert, Drop DB, Execute, Fetch, Field List, Init DB, Kill, Long Data, NoAudit, Ping, Prepare, Processlist, Query, Quit, Refresh, Register Slave, Reset stmt, Set option, Shutdown, Sleep, Statistics, Table Dump, Time.

In terms of output formats the following are supported beside CSV: GRID (used by default), TAB, VERTICAL and RAW (corresponding to the original XML format of the audit log file).

15.4.5.4.4. Permissions Required

The user must have permissions to read the audit log on disk.

15.4.5.4.5. Tips and Tricks

The values for the `--event-type` [319] and `--format` [320] options are case insensitive, therefore lower and upper cases can be mixed to specify these values as long as a supported event type name or

format is used. Unlike them, the value specified for the `--users` [320] option is case sensitive, so be careful not to mix upper and lower cases here.

It is possible to find some event type values with a space in the middle, for example like "Binlog Dump" or "Init DB". If one of such values needs to be specified for the `--event-type` [319] option then it must be surrounded by double ("") or single ('') quotes depending on the operating system.

15.5. Overview of MySQL Utilities

This chapter presents an brief overview of each of the available utilities. The utilities are grouped into sections based on the type of administrative function that they perform.

15.5.1. Database Operations

These utilities are those designed to work at the database-level. They include utilities that can be used to administer databases on one or more servers.

- `mysqldbcompare`
 - Compare databases on two servers or the same server
 - Compare definitions and data
 - Generate a difference report
 - Generate SQL transformation statements
- `mysqldbcopy`
 - Copy databases between servers
 - Clone databases on the same server
 - Supports rename
- `mysqldbexport`
 - Export metadata and/or data from one or more databases
 - Formats: SQL, CSV, TAB, Grid, Vertical
- `mysqldbimport`
 - Import metadata and data from one or more files
 - Reads all formats from mysqldbexport
- `mysqldiff`
 - Compare object definitions
 - Generate a difference report

15.5.2. General Operations

These utilities are those designed to perform general operations such as reporting and searching.

- `mysqldiskusage`

- Show disk usage for databases
- Generate reports in SQL, CSV, TAB, Grid, Vertical
- `mysqlfrm`
 - Reads `.frm` files, optionally in byte-by-byte diagnostic mode
 - Generates `CREATE` statements from table definition data
- `mysqlindexcheck`
 - Read indexes for one or more tables
 - Check for redundant and duplicate indexes
 - Generate reports in SQL, CSV, TAB, Grid, Vertical
- `mysqlmetagrep`
 - Search metadata
 - Regexp, database search
 - Generate SQL statement for search query
- `mysqlprocgrep`
 - Search process information
 - Generate SQL statement for search
 - Kill processes that match query
- `mysqluserclone`
 - Clone a user account, to the same or different server
 - Show user grants
- `mysqluc`
 - Command line client for running MySQL Utilities
 - Allows a persistent connection to a MySQL Server
 - Tab completion for utility names and options
 - Allows calling the commands with shorter names, such as using "serverinfo" instead of `mysqlserverinfo`

15.5.3. High Availability Operations

These utilities are those designed to support replication and high availability operations for MySQL servers.

- `mysqlfailover`
 - Provides automatic failover on a replication topology

- Uses Global Transaction Identifiers (GTID, MySQL Server 5.6.5+)
- `mysqlreplicate`
 - Setup replication
 - Start from beginning, current, specific binlog, pos
- `mysqlrpladmin`
 - Administers the replication topology
 - Allows recovery of the master
 - Commands include elect, failover, gtid, health, start, stop, and switchover
- `mysqlrplcheck`
 - Check replication configuration
 - Tests binary logging on master
- `mysqlrplshow`
 - Show slaves attached to master
 - Can search recursively
 - Show the replication topology as a graph or list

15.5.4. Server Operations

These utilities are used to perform server-wide operations.

- `mysqlserverclone`
 - Start a new instance of a running server
- `mysqlserverinfo`
 - Show server information
 - Can search for running servers on a host
 - Access online or offline servers

15.5.5. Specialized Operations

These utilities are designed to be used with a specific commercial extension. In this case, these utilities require the Audit Log Plugin.

- `mysqlauditadmin`
 - Monitor the audit log
 - Copy, rotate, and configure the audit log
- `mysqlauditgrep`

- Search the audit log
- Output results to different formats

15.6. Manual Pages

This chapter includes the manual pages for each of the utilities. Each manual page is formatted similar to a typical Unix man page.

15.6.1. mysqlauditadmin

15.6.1.1. **mysqlauditadmin** — Allows users to perform maintenance action on the audit log

This utility allows you to maintain the [audit log](#), allowing you to monitor the audit log file growth and control its rotation. Rotation refers to the action of replacing the current audit log file by a new one for continuous use, renaming (with a timestamp extension) and copying the previously used audit log file to a defined location.

This utility allows you to view and modify a subset of audit log control variables, display the audit log file status, perform on-demand rotation of the log file, and copy files to other locations. These features enable you to easily monitor the audit log file growth and control its rotation (automatically based on the defined file size threshold, or manually by a on-demand command).

The available actions include the following:

1. **copy**

This command copies the audit log specified by [--audit-log-name \[313\]](#) to the destination path specified by [--copy-to \[313\]](#). The [--remote-login \[313\]](#) option can be used to copy log files from a remote location. Note: the destination path must be locally accessible by the current user.

2. **policy**

The policy command is used to change the audit logging policy. The accepted values are the following, which are set using the [--value \[313\]](#) option.



Note

The [--server \[313\]](#) option is also required to execute this command.

- [ALL](#): log all events
- [NONE](#): log nothing
- [LOGINS](#): only log login events
- [QUERIES](#): only log query events
- [DEFAULT](#): sets the default log policy

3. **rotate_on_size**

This command sets the file size threshold for automatic rotation of the audit log (the [audit_log_rotate_on_size](#) variable). The value is set using the [--value \[313\]](#) option, and

must be in the range (0, 4294967295). This command also requires the [--server \[313\]](#) option to be specified. Note: if the variable is set with a value that is not a multiple of 4096, then it is truncated to the nearest multiple.

4. **rotate**

This command is used to perform an on-demand audit log rotation, and only requires the [--server \[313\]](#) option to be passed. Note: this command has no effect if the audit log file size is smaller than 4096, which is the minimum value allowed that is greater than 0 for the [audit_log_rotate_on_size variable](#).

OPTIONS

`mysqlauditadmin` accepts the following command-line options:

- **--audit-log-name=<AUDIT_LOG_FILE>**

Full path and file name for the audit log file. Used by the [--file-stats \[313\]](#) option, and the `copy` command.

- **--copy-to=<COPY_DESTINATION>**

The location to copy the specified audit log file. The path must be locally accessible for the current user.

- **--file-stats**

Display the audit log file statistics.

- **--help**

Display a help message and exit.

- **--remote-login=<REMOTE_LOGIN>**

User name and host to be used for the remote login, for copying log files. It is defined using the following format: `<user>:<host or IP>`. Usage will prompt for the password.

- **--server=<SERVER>**

Connection information for the server in the format: `<user>[:<passwd>]@<host>[:<port>][:<socket>]` or `<login-path>[:<port>][:<socket>]`.

- **--show-options**

Display the audit log system variables.

- **--value=<VALUE>**

Value used to set variables based on the specified commands, such as `policy` and `rotate_on_size`.

- **--server1=<source>**

Connection information for the first server in the format: `<user>[:<passwd>]@<host>[:<port>][:<socket>]` or `<login-path>[:<port>][:<socket>]`.

- **--verbose, -v**

Specify how much information to display. Use this option multiple times to increase the amount of information. For example, `-v` = verbose, `-vv` = more verbose, `-vvv` = debug.

- **--version**

Display version information and exit.

NOTES

This utility is available as of μ 1.2.0.

This utility can only be applied to servers with the [audit log plugin enabled](#). And the audit log plugin is available as of MySQL Server versions 5.5.28 and 5.6.10.

This utility requires Python version 2.6 or higher, but does not support Python 3.

The path to the MySQL client tools should be included in the [PATH](#) environment variable in order to use the authentication mechanism with login-paths. This will allow the utility to use the [my_print_defaults](#) tools, which is required to read the login-path values from the login configuration file ([.mylogin.cnf](#)). This feature exists as of MySQL Server 5.6.6, see [mysql_config_editor — MySQL Configuration Utility](#).

LIMITATIONS

The [--remote-login \[313\]](#) option is not supported on Microsoft Windows platforms. For Microsoft Windows, use [UNC](#) paths and perform a local copy operation, omitting the [--remote-login \[313\]](#) option.

EXAMPLES

To display the audit log system variables, run the following command:

```
$ myslauditadmin --show-options --server=root@localhost:3310

#
# Audit Log Variables and Options
#
+-----+-----+
| Variable_name      | Value   |
+-----+-----+
| audit_log_buffer_size | 1048576
| audit_log_file     | audit.log
| audit_log_flush    | OFF
| audit_log_policy   | ALL
| audit_log_rotate_on_size | 0
| audit_log_strategy | ASYNCHRONOUS
+-----+-----+
```

To perform a (manual) rotation of the audit log file, use the following command:

```
shell> myslauditadmin --server=root@localhost:3310 rotate

#
# Executing ROTATE command.
#
```

To display the audit log file statistics, run the following command:

```
shell> mysqlauditadmin --file-stats --audit-log-name=../SERVER/data/audit.log

+-----+-----+-----+
| File | Size | Created | Last Modified |
+-----+-----+-----+
| audit.log | 3258 | Wed Sep 26 11:07:43 2012 | Wed Sep 26 11:07:43 2012 |
| audit.log.13486539046497235 | 47317 | Wed Sep 26 11:05:04 2012 | Wed Sep 26 11:05:04 2012 |
+-----+-----+-----+
```

To change the audit log policy to log only query events, and show the system variables before and after the execution of the *policy* command, use the following command:

```
shell> mysqlauditadmin --show-options --server=root@localhost:3310 policy \
--value=QUERIES

#
# Showing options before command.
#
# Audit Log Variables and Options
#
+-----+-----+
| Variable_name | Value |
+-----+-----+
| audit_log_buffer_size | 1048576
| audit_log_file | audit.log
| audit_log_flush | OFF
| audit_log_policy | ALL
| audit_log_rotate_on_size | 0
| audit_log_strategy | ASYNCHRONOUS
+-----+-----+

#
# Executing POLICY command.
#

#
# Showing options after command.
#
# Audit Log Variables and Options
#
+-----+-----+
| Variable_name | Value |
+-----+-----+
| audit_log_buffer_size | 1048576
| audit_log_file | audit.log
| audit_log_flush | OFF
| audit_log_policy | QUERIES
| audit_log_rotate_on_size | 0
| audit_log_strategy | ASYNCHRONOUS
+-----+-----+
```

To change the audit log automatic file rotation size to 32535, and show the system variables before and after the execution of the *rotate_on_size* command, use the following command. (Notice that the value set is actually 28672 because the specified *rotate_on_size* value is truncated to a multiple of 4096):

```
shell> mysqlauditadmin --show-options --server=root@localhost:3310 rotate_on_size \
--value=32535

#
```

```
# Showing options before command.  
#  
# Audit Log Variables and Options  
#  
+-----+-----+  
| Variable_name | Value |  
+-----+-----+  
| audit_log_buffer_size | 1048576  
| audit_log_file | audit.log  
| audit_log_flush | OFF  
| audit_log_policy | ALL  
| audit_log_rotate_on_size | 0  
| audit_log_strategy | ASYNCHRONOUS  
+-----+-----+  
  
#  
# Executing ROTATE_ON_SIZE command.  
#  
  
#  
# Showing options after command.  
#  
# Audit Log Variables and Options  
#  
+-----+-----+  
| Variable_name | Value |  
+-----+-----+  
| audit_log_buffer_size | 1048576  
| audit_log_file | audit.log  
| audit_log_flush | OFF  
| audit_log_policy | ALL  
| audit_log_rotate_on_size | 28672  
| audit_log_strategy | ASYNCHRONOUS  
+-----+-----+
```

To perform a copy of a audit log file to another location, use the following command:

```
shell> mysqlauditadmin --audit-log-name=../SERVER/data/audit.log.13486539046497235 \  
copy --copy-to=/BACKUP/Audit_Logs
```

To copy a audit log file from a remote server/location to the current location (user password will be prompted), use the following command:

```
shell> mysqlauditadmin --audit-log-name=audit.log.13486539046497235 \  
copy --remote-login=user:host --copy-to=.
```

15.6.2. mysqlauditgrep

15.6.2.1. **mysqlauditgrep** — Allows users to search the current or an archived audit log

This utility allows you to search the current or archived audit logs, allowing you to display data from the audit log file according to the defined search criterion. It also allows you to output the results in different formats, namely GRID (default), TAB, CSV, VERTICAL, and RAW (the original XML format).

This utility allows you to search and filter the returned audit log records by: users ([--users \[320\]](#)), date and time ranges ([--start-date \[320\]](#) and [--end-date \[319\]](#)), SQL query types ([--query-](#)

`type` [320]), logged event and record types (`--event-type` [319]), status (`--status` [320]), and matching patterns (`--pattern` [320]). Any of these search options can be combined and used together, with the retrieved records resulting from all passed in options being true.

The `--pattern` [320] supports two types of pattern matching: standard SQL, used with the SQL `LIKE` operator (SQL patterns), and standard `REGEXP` (POSIX regular expression patterns).

This utility always requires an audit log file to be passed in, so the `AUDIT_LOG_FILE` argument is searched as a full path and file name for the audit log file. If not specified, a notification concerning this requirement will be printed. And if `--format` [320] is passed in without search parameters, then all the records of the audit log are displayed in the specified format.

The `--file-stats` [319] option is not considered a search criteria, and is used to display the file statistics of a specified audit log. Other search options will be ignored when the `--file-stats` [319] option is used, except the `--format` [320] option will continue to format the results accordingly.

To specify the format of the generated results, use one of the following values with the `--format` [320] option:

1. *GRID (default)*

Display output in grid or table format like that of the `mysql` monitor.

2. *CSV*

Display output in comma-separated values format.

3. *VERTICAL*

Display output in single-column format like that of the `\G` command for the `mysql` monitor.

4. *RAW*

Display output results in the original raw format of the audit log records, which is written in XML.

Standard SQL Pattern Matching

The simple patterns defined by the SQL standard enables users to use two characters with special meanings: “`%`” (percent) matches zero or more characters, and “`_`” (underscore) matches exactly one arbitrary character. In standard SQL, these types of patterns are used with the `LIKE` comparison operator, and they are case-insensitive by default. This utility assumes that they are case-insensitive.

For example:

- `"audit%"`

Match any string that starts with "audit".

- `"%log%"`

Match any string containing the word "log".

- `"%_%"`

Match any string consisting of one or more characters.

For documentation about the standard SQL pattern matching syntax, see [Pattern Matching](#).

REGEXP Pattern Matching (POSIX)

Standard *REGEXP* patterns are more powerful than the simple patterns defined in the SQL standard. A regular expression is a string of ordinary and special characters specified to match other strings. Unlike SQL Patterns, *REGEXP* patterns are case-sensitive. The *REGEXP* syntax defines the following characters with special meaning:

- .
Match any character.
- ^
Match the beginning of a string.
- \$
Match the end of a string.
- \
Match zero or more repetitions of the preceding regular expression.
- +
Match one or more repetitions of the preceding regular expression.
- ?
Match zero or one repetition of the preceding regular expression.
- |
Match either the regular expressions from the left or right of |.
- []
Indicates a set of characters to match. Note that, special characters lose their special meaning inside sets. In particular, ^ acquires a different meaning if it is the first character of the set, matching the complementary set (i.e., all the characters that are not in the set will be matched).
- {m}
Match *m* repetitions of the preceding regular expression.
- {m,n}
Match from *m* to *n* repetitions of the preceding regular expression.
- ()
Define a matching group, and matches the regular expression inside the parentheses.

For example:

- "*a**"
Match a sequence of zero or more *a*.
- "*a*+"
Match a sequence of one or more *a*.

Match a sequence of one or more `a`.

- `"a?"`

Match zero or one `a`.

- `"ab/cd"`

Match `ab` or `cd`.

- `"[axy]"`

Match `a`, `x` or `y`.

- `"[a-f]"`

Match any character in the range `a` to `f` (that is, `a`, `b`, `c`, `d`, `e`, or `f`).

- `"[^axy]"`

Match any character *except* `a`, `x` or `y`.

- `"a{5}"`

Match exactly five copies of `a`.

- `"a{2,5}"`

Match from two to five copies of `a`.

- `"(abc)+"`

Match one or more repetitions of `abc`.

This is a brief overview of regular expressions that can be used to define this type of patterns. The full syntax is described in the [Python "re" module docs](#), supporting the definition of much more complex pattern matching expression.

OPTIONS

`mysqlauditgrep` accepts the following command-line options:

- `--end-date=<END_DATE>`

End date/time to retrieve log entries until the specified date/time range. If not specified or the value is 0, all entries to the end of the log are displayed. Accepted formats: "yyyy-mm-ddThh:mm:ss" or "yyyy-mm-dd".

- `--event-type=<EVENT_TYPE>`

Comma-separated list of event types to search in all audit log records matching the specified types. Supported values are: Audit, Binlog Dump, Change user, Close stmt, Connect Out, Connect, Create DB, Daemon, Debug, Delayed insert, Drop DB, Execute, Fetch, Field List, Init DB, Kill, Long Data, NoAudit, Ping, Prepare, Processlist, Query, Quit, Refresh, Register Slave, Reset stmt, Set option, Shutdown, Sleep, Statistics, Table Dump, Time.

- `--file-stats`

Display the audit log file statistics.

- **--format=FORMAT**, **-f FORMAT**

Output format to display the resulting data. Supported format values: GRID (default), TAB, CSV, VERTICAL and RAW.

- **--help**

Display a help message and exit.

- **--pattern=<PATTERN>**, **-e <PATTERN>**

Search pattern to retrieve all entries with at least one attribute value matching the specified pattern. By default the standard SQL *LIKE* patterns are used for matching. If the **--regexp** option is set, then *REGEXP* patterns must be specified for matching.

- **--query-type=<QUERY_TYPE>**

Comma-separated list of SQL statements/commands to search for and match. Supported values: CREATE, ALTER, DROP, TRUNCATE, RENAME, GRANT, REVOKE, SELECT, INSERT, UPDATE, DELETE, COMMIT, SHOW, SET, CALL, PREPARE, EXECUTE, DEALLOCATE.

- **--regexp**, **--basic-regexp**, **-G**

Indicates that pattern matching will be performed using a regular expression *REGEXP* (from the Python `re` module). By default, the simple standard SQL *LIKE* patterns are used for matching. This affects how the value specified by the **--pattern** option is interpreted.

- **--start-date=<START_DATE>**

Starting date/time to retrieve log entries from the specified date/time range. If not specified or the value is 0, all entries from the start of the log are displayed. Accepted formats: yyyy-mm-ddThh:mm:ss or yyyy-mm-dd.

- **--status=<STATUS>**

Comma-separated list of status values or intervals to search for all audit log records with a matching status. Status values are non-negative integers (corresponding to MySQL error codes). Status intervals are closed (i.e., include both endpoints) and defined simply using a dash between its endpoints. For Example: 1051,1068-1075,1109,1146.

The **--status** option is available as of MySQL Utilities 1.2.4 / 1.3.3.

- **--users=<USERS>**, **-u <USERS>**

Comma-separated list of user names, to search for their associated log entries. For example: "dan,jon,john,paul,philip,stefan".

- **--verbose**, **-v**

Specify how much information to display. Use this option multiple times to increase the amount of information. For example, **-v** = verbose, **-vv** = more verbose, **-vvv** = debug.

- **--version**

Display version information and exit.

NOTES

This utility is available as of μ 1.2.0.

This utility can only be applied to servers with the [audit log plugin enabled](#). And the audit log plugin is available as of MySQL Server versions 5.5.28 and 5.6.10.

This utility requires the use of Python version 2.6 or higher, but does not support Python 3.

Single or double quote characters (respectively, ' or ") can be used around option values. In fact, quotes are required to set some options values correctly, such as values with whitespace. For example, to specify the event types `Create DB` and `Drop DB` for the `--event-type` option, the following syntax must be used: `--event-type='Create DB,Drop DB'` or `--event-type="Create DB,Drop DB"`.

EXAMPLES

To display the audit log file statistics and output the results in CSV format, run the following command:

```
shell> mysqlauditgrep --file-stats --format=CSV /SERVER/data/audit.log

#
# Audit Log File Statistics:
#
File,Size,Created,Last Modified
audit.log,9101,Thu Sep 27 13:33:11 2012,Thu Oct 11 17:40:35 2012

#
# Audit Log Startup Entries:
#

SERVER_ID,STARTUP_OPTIONS,NAME,TIMESTAMP,MYSQL_VERSION,OS_VERSION,VERSION
1,/SERVER/sql/mysqld --defaults-file=/SERVER/my.cnf,Audit,2012-09-27T13:33:11,5.5.29-log,x86_64-Linux,1
```

To display the audit log entries of specific users, use the following command:

```
shell> mysqlauditgrep --users=tester1,tester2 /SERVER/data/audit.log
```

To display the audit log file statistics, run the following command:

```
shell> mysqlauditgrep --users=tester1,tester2 /SERVER/data/audit.log

+-----+-----+-----+-----+-----+-----+-----+-----+
| STATUS | SERVER_ID | NAME    | TIMESTAMP          | CONNECTION_ID | HOST    | USER    | PRIV   |
+-----+-----+-----+-----+-----+-----+-----+-----+
| 0      | 1         | Connect  | 2012-09-28T11:26:50 | 9            | localhost | root    | test   |
| 0      | 1         | Query    | 2012-09-28T11:26:50 | 9            | None    | root    | test   |
| 0      | 1         | Ping     | 2012-09-28T11:26:50 | 9            | None    | root    | test   |
| 0      | 1         | Query    | 2012-09-28T11:26:50 | 9            | None    | root    | test   |
| 0      | 1         | Query    | 2012-09-28T11:26:50 | 9            | None    | root    | test   |
| 0      | 1         | Ping     | 2012-09-28T11:26:50 | 9            | None    | root    | test   |
| 0      | 1         | Query    | 2012-09-28T11:26:50 | 9            | None    | root    | test   |
| 0      | 1         | Quit     | 2012-09-28T11:26:50 | 9            | None    | root    | test   |
| 0      | 1         | Connect  | 2012-10-10T15:55:55 | 11           | localhost | tester2 | root   |
| 0      | 1         | Query    | 2012-10-10T15:55:55 | 11           | None    | tester2 | root   |
| 0      | 1         | Query    | 2012-10-10T15:56:10 | 11           | None    | tester2 | root   |
| 1046   | 1         | Query    | 2012-10-10T15:57:26 | 11           | None    | tester2 | root   |
| 1046   | 1         | Query    | 2012-10-10T15:57:36 | 11           | None    | tester2 | root   |
```

mysqlauditgrep

0	1	Query	2012-10-10T15:57:51	11	None	tester2	root
0	1	Quit	2012-10-10T15:57:59	11	None	tester2	root
0	1	Connect	2012-10-10T17:35:42	12	localhost	tester2	root
0	1	Query	2012-10-10T17:35:42	12	None	tester2	root
0	1	Quit	2012-10-10T17:47:22	12	None	tester2	root

To display the audit log entries for a specific date/time range, use the following command:

```
shell> mysqlauditgrep --start-date=2012-09-27T13:33:47 --end-date=2012-09-28 /SERVER/data/audit.log
```

STATUS	TIMESTAMP	NAME	CONNECTION_ID	SQLTEXT
0	2012-09-27T13:33:47	Ping	7	None
0	2012-09-27T13:33:47	Query	7	SELECT * FROM INFORMATION_SCHEMA.PLUGINS WHERE
0	2012-09-27T13:33:47	Query	7	COMMIT
0	2012-09-27T13:34:48	Quit	7	None
0	2012-09-27T13:34:48	Quit	8	None

To display the audit log entries matching a specific SQL *LIKE* pattern, use the following command:

```
shell> mysqlauditgrep --pattern="% = ____"; /SERVER/data/audit.log
```

STATUS	TIMESTAMP	NAME	SQLTEXT	CONNECTION_ID
0	2012-09-27T13:33:39	Query	SET @@session.autocommit = OFF	7
0	2012-09-27T13:33:39	Query	SET @@session.autocommit = OFF	8
0	2012-09-28T11:26:50	Query	SET @@session.autocommit = OFF	9
0	2012-09-28T11:26:50	Query	SET @@session.autocommit = OFF	10

To display the audit log entries matching a specific *REGEXP* pattern, use the following command:

```
shell> mysqlauditgrep --pattern=".* = ..." --regexp /SERVER/data/audit.log
```

STATUS	TIMESTAMP	NAME	SQLTEXT	CONNECTION_ID
0	2012-09-27T13:33:39	Query	SET @@session.autocommit = OFF	7
0	2012-09-27T13:33:39	Query	SET @@session.autocommit = OFF	8
0	2012-09-28T11:26:50	Query	SET @@session.autocommit = OFF	9
0	2012-09-28T11:26:50	Query	SET @@session.autocommit = OFF	10

To display the audit log entries of specific query types, use the following command:

```
shell> mysqlauditgrep --query-type=show,SET /SERVER/data/audit.log
```

STATUS	TIMESTAMP	NAME	SQLTEXT	CONNECTION_ID
0	2012-09-27T13:33:39	Query	SET NAMES 'latin1' COLLATE 'latin1_swedish_ci'	7

mysqlauditgrep

0	2012-09-27T13:33:39	Query	SET @@session.autocommit = OFF	7
0	2012-09-27T13:33:39	Query	SHOW VARIABLES LIKE 'READ_ONLY'	7
0	2012-09-27T13:33:39	Query	SHOW VARIABLES LIKE 'datadir'	7
0	2012-09-27T13:33:39	Query	SHOW VARIABLES LIKE 'basedir'	7
0	2012-09-27T13:33:39	Query	SET NAMES 'latin1' COLLATE 'latin1_swedish_ci'	8
0	2012-09-27T13:33:39	Query	SET @@session.autocommit = OFF	8
0	2012-09-27T13:33:39	Query	SHOW VARIABLES LIKE 'READ_ONLY'	8
0	2012-09-27T13:33:39	Query	SHOW VARIABLES LIKE 'basedir'	8
0	2012-09-28T11:26:50	Query	SET NAMES 'latin1' COLLATE 'latin1_swedish_ci'	9
0	2012-09-28T11:26:50	Query	SET @@session.autocommit = OFF	9
0	2012-09-28T11:26:50	Query	SHOW VARIABLES LIKE 'READ_ONLY'	9
0	2012-09-28T11:26:50	Query	SET NAMES 'latin1' COLLATE 'latin1_swedish_ci'	10
0	2012-09-28T11:26:50	Query	SET @@session.autocommit = OFF	10
0	2012-09-28T11:26:50	Query	SHOW VARIABLES LIKE 'READ_ONLY'	10
0	2012-09-28T11:26:50	Query	SET @@GLOBAL.audit_log_flush = ON	10
0	2012-09-28T11:26:50	Query	SHOW VARIABLES LIKE 'audit_log_policy'	10
0	2012-09-28T11:26:50	Query	SHOW VARIABLES LIKE 'audit_log_rotate_on_size'	10
0	2012-10-10T15:56:10	Query	show databases	11
1046	2012-10-10T15:57:26	Query	show tables test	11
1046	2012-10-10T15:57:36	Query	show tables test	11
0	2012-10-10T15:57:51	Query	show tables in test	11

To display the audit log entries of specific event types, use the following command:

```
shell> mysqlauditgrep --event-type="Ping,Connect" /SERVER/data/audit.log
```

STATUS	NAME	TIMESTAMP	CONNECTION_ID	HOST	USER	PRIV_USER	IP
0	Connect	2012-09-27T13:33:39	7	localhost	root	root	127.0
0	Ping	2012-09-27T13:33:39	7	None	None	None	None
0	Ping	2012-09-27T13:33:39	7	None	None	None	None
0	Ping	2012-09-27T13:33:39	7	None	None	None	None
0	Ping	2012-09-27T13:33:39	7	None	None	None	None
0	Connect	2012-09-27T13:33:39	8	localhost	root	root	127.0
0	Ping	2012-09-27T13:33:39	8	None	None	None	None
0	Ping	2012-09-27T13:33:39	8	None	None	None	None
0	Ping	2012-09-27T13:33:47	7	None	None	None	None
0	Connect	2012-09-28T11:26:50	9	localhost	root	tester	127.0
0	Ping	2012-09-28T11:26:50	9	None	None	None	None
0	Ping	2012-09-28T11:26:50	9	None	None	None	None
0	Connect	2012-09-28T11:26:50	10	localhost	root	root	127.0
0	Ping	2012-09-28T11:26:50	10	None	None	None	None
0	Ping	2012-09-28T11:26:50	10	None	None	None	None
0	Ping	2012-09-28T11:26:50	10	None	None	None	None
0	Ping	2012-09-28T11:26:50	10	None	None	None	None
0	Connect	2012-10-10T15:55:55	11	localhost	tester	root	127.0
0	Connect	2012-10-10T17:35:42	12	localhost	tester	root	127.0

To display the audit log entries with a specific status, use the following command:

```
shell> mysqlauditgrep --status=1100-1199,1046 /SERVER/data/audit.log
```

STATUS	TIMESTAMP	NAME	SQLTEXT
1046	2012-10-10T15:57:26	Query	show tables test
1046	2012-10-10T15:57:36	Query	show tables test

```
| 1146 | 2012-10-10T17:44:55 | Query | select * from teste.employees where salary > 500 and salary < 1000
| 1046 | 2012-10-10T17:47:17 | Query | select * from test_encoding where value = '<>"&'
```

Note: You can view all successful commands with `--status=0` and all unsuccessful ones with `--status=1-9999`.

To display the audit log entries matching several search criteria, use the following command:

```
shell> mysqlauditgrep --users=root --start-date=0 --end-date=2012-10-10 --event-type=Query \
--query-type=SET --status=0 --pattern="%audit_log%" /SERVER/data/audit.log

+-----+-----+-----+-----+-----+-----+-----+
| STATUS | SERVER_ID | NAME   | TIMESTAMP           | CONNECTION_ID | USER    | PRIV_USER | SQLTEXT
+-----+-----+-----+-----+-----+-----+-----+
| 0      | 1          | Query  | 2012-09-28T11:26:50 | 10            | root    | root     | SET @@GLOBAL.
```

15.6.3. mysqldbcompare

15.6.3.1. mysqldbcompare — Compare Two Databases and Identify Differences

This utility compares the objects and data from two databases to find differences. It identifies objects having different definitions in the two databases and presents them in a diff-style format of choice. Differences in the data are shown using a similar diff-style format. Changed or missing rows are shown in a standard format of GRID, CSV, TAB, or VERTICAL.

Use the notation db1:db2 to name two databases to compare, or, alternatively just db1 to compare two databases with the same name. The latter case is a convenience notation for comparing same-named databases on different servers.

The comparison may be run against two databases of different names on a single server by specifying only the `--server1` [327] option. The user can also connect to another server by specifying the `--server2` [327] option. In this case, db1 is taken from server1 and db2 from server2.

Those objects considered in the database include tables, views, triggers, procedures, functions, and events. A count for each object type can be shown with the `-vv` option.

The check is performed using a series of steps called tests. By default, the utility stops on the first failed test, but you can specify the `--run-all-tests` [327] option to cause the utility to run all tests regardless of their end state.

Note: Using `--run-all-tests` [327] may produce expected cascade failures. For example, if the row counts differ among two tables being compared, the data consistency will also fail.

The tests include the following:

1. Check database definitions

A database existence precondition check ensures that both databases exist. If they do not, no further processing is possible and the `--run-all-tests` [327] option is ignored.

2. Check existence of objects in both databases

The test for objects in both databases identifies those objects missing from one or another database. The remaining tests apply only to those objects that appear in both databases. To skip this test, use the

--skip-object-compare [327] option. That can be useful when there are known missing objects among the databases.

3. Compare object definitions

The definitions (the **CREATE** statements) are compared and differences are presented. To skip this test, use the --skip-diff [327] option. That can be useful when there are object name differences only that you want to ignore.

4. Check table row counts

This check ensures that both tables have the same number of rows. This does not ensure that the table data is consistent. It is merely a cursory check to indicate possible missing rows in one table or the other. The data consistency check identifies the missing rows. To skip this test, use the --skip-row-count [327] option.

5. Check table data consistency

This check identifies both changed rows as well as missing rows from one or another of the tables in the databases. Changed rows are displayed as a diff-style report with the format chosen (**GRID** by default) and missing rows are also displayed using the format chosen. To skip this test, use the --skip-data-check [327] option.

You may want to use the --skip-xxx options to run only one of the tests. This might be helpful when working to bring two databases into synchronization, to avoid running all of the tests repeatedly during the process.

Each test completes with one of the following states:

- **pass**

The test succeeded.

- **FAIL**

The test failed. Errors are displayed following the test state line.

- **SKIP**

The test was skipped due to a missing prerequisite or a skip option.

- **WARN**

The test encountered an unusual but not fatal error.

- -

The test is not applicable to this object.

To specify how to display diff-style output, use one of the following values with the --diff-type [327] option:

- **unified** (default)

Display unified format output.

- **context**

Display context format output.

- **differ**

Display differ-style format output.

- **sql**

Display SQL transformation statement output.

To specify how to display output for changed or missing rows, use one of the following values with the [--format \[327\]](#) option:

- **grid** (default)

Display output in grid or table format like that of the [mysql](#) monitor.

- **csv**

Display output in comma-separated values format.

- **tab**

Display output in tab-separated format.

- **vertical**

Display output in single-column format like that of the `\G` command for the [mysql](#) monitor.

The [--changes-for \[327\]](#) option controls the direction of the difference (by specifying the object to be transformed) in either the difference report (default) or the transformation report (designated with the [--difftype=sql \[327\]](#) option). Consider the following command:

```
mysqldbcompare --server1=root@host1 --server2=root@host2 --difftype=sql \
    db1:dbx
```

The leftmost database (`db1`) exists on the server designated by the [--server1 \[327\]](#) option (`host1`). The rightmost database (`dbx`) exists on the server designated by the [--server2 \[327\]](#) option (`host2`).

- [--changes-for=server1 \[327\]](#): Produce output that shows how to make the definitions of objects on `server1` like the definitions of the corresponding objects on `server2`.
- [--changes-for=server2 \[327\]](#): Produce output that shows how to make the definitions of objects on `server2` like the definitions of the corresponding objects on `server1`.

The default direction is `server1`.

You must provide connection parameters (user, host, password, and so forth) for an account that has the appropriate privileges to access all objects in the operation.

If the utility is to be run on a server that has binary logging enabled, and you do not want the comparison steps logged, use the [--disable-binary-logging \[327\]](#) option.

OPTIONS

`mysqldbcompare` accepts the following command-line options:

- [--help](#)

Display a help message and exit.

- `--changes-for=<direction>`

Specify the server to show transformations to match the other server. For example, to see the transformation for transforming object definitions on server1 to match the corresponding definitions on server2, use `--changes-for=server1` [327]. Permitted values are **server1** and **server2**. The default is **server1**.

- `--difftype=<difftype>, -d<difftype>`

Specify the difference display format. Permitted format values are **unified**, **context**, **differ**, and **sql**. The default is **unified**.

- `--disable-binary-logging`

If binary logging is enabled, disable it during the operation to prevent comparison operations from being written to the binary log. Note: Disabling binary logging requires the **SUPER** privilege.

- `--format=<format>, -f<format>`

Specify the display format for changed or missing rows. Permitted format values are **grid**, **csv**, **tab**, and **vertical**. The default is **grid**.

- `--quiet, -q`

Do not print anything. Return only an exit code of success or failure.

- `--run-all-tests, -a`

Do not halt at the first difference found. Process all objects.

- `--server1=<source>`

Connection information for the first server in the format: `<user>[:<passwd>]@<host>[:<port>][:<socket>]` or `<login-path>[:<port>][:<socket>]`.

- `--server2=<source>`

Connection information for the second server in the format: `<user>[:<passwd>]@<host>[:<port>][:<socket>]` or `<login-path>[:<port>][:<socket>]`.

- `--show-reverse`

Produce a transformation report containing the SQL statements to conform the object definitions specified in reverse. For example, if --changes-for is set to server1, also generate the transformation for server2. Note: The reverse changes are annotated and marked as comments.

- `--skip-data-check`

Skip the data consistency check.

- `--skip-diff`

Skip the object definition difference check.

- `--skip-object-compare`

Skip the object comparison check.

- `--skip-row-count`

Skip the row count check.

- `--span-key-size=<number of bytes to use for key>`

Change the size of the key used for compare table contents. A higher value can help to get more accurate results comparing large databases, but may slow the algorithm.

Default value is 8.

- `--verbose, -v`

Specify how much information to display. Use this option multiple times to increase the amount of information. For example, `-v` = verbose, `-vv` = more verbose, `-vvv` = debug.

- `--version`

Display version information and exit.

- `--width=<number>`

Change the display width of the test report. The default is 75 characters.

NOTES

The login user must have the appropriate permissions to read all databases and tables listed.

For the `--difftype` [327] option, the permitted values are not case sensitive. In addition, values may be specified as any unambiguous prefix of a valid value. For example, `--difftype=d` [327] specifies the differ type. An error occurs if a prefix matches more than one valid value.

The path to the MySQL client tools should be included in the PATH environment variable in order to use the authentication mechanism with login-paths. This will allow the utility to use the my_print_defaults tools which is required to read the login-path values from the login configuration file (.mylogin.cnf).

If any database identifier specified as an argument contains special characters or is a reserved word, then it must be appropriately quoted with backticks (`). In turn, names quoted with backticks must also be quoted with single or double quotes depending on the operating system, i.e. ("") in Windows or (') in non-Windows systems, in order for the utilities to read backtick quoted identifiers as a single argument. For example, to compare a database with the name **weird`db.name** with **other:weird`db.name**, the database pair must be specified using the following syntax (in non-Windows): `'weird`db.name : other:weird`db.name'`.

EXAMPLES

Use the following command to compare the `emp1` and `emp2` databases on the local server, and run all tests even if earlier tests fail:

```
$ mysqldbcompare --server1=root@localhost emp1:emp2 --run-all-tests
# server1 on localhost: ... connected.
# Checking databases emp1 on server1 and emp2 on server2

WARNING: Objects in server2:emp2 but not in server1:emp1:
  TRIGGER: trg
  PROCEDURE: p1
  TABLE: t1
  VIEW: v1

          Defn      Row      Data
Type     Object Name    Diff    Count   Check
```

```

FUNCTION  f1
TABLE      departments          pass   -   -
                        pass   pass   FAIL

Data differences found among rows:
--- empl.departments
+++ emp2.departments
@@ -1,4 +1,4 @@
***** 1. row *****
dept_no: d002
-dept_name: dunno
+dept_name: Finance
1 rows.

Rows in empl.departments not in emp2.departments
***** 1. row *****
dept_no: d008
dept_name: Research
1 rows.

Rows in emp2.departments not in empl.departments
***** 1. row *****
dept_no: d100
dept_name: stupid
1 rows.

TABLE      dept_manager        pass   pass   pass

Database consistency check failed.

# ...done

```

Given: two databases with the same table layout. Data for each table contains:

```

mysql> select * from db1.t1;
+---+-----+
| a | b      |
+---+-----+
| 1 | Test 789 |
| 2 | Test 456 |
| 3 | Test 123 |
| 4 | New row - db1 |
+---+-----+
4 rows in set (0.00 sec)

mysql> select * from db2.t1;
+---+-----+
| a | b      |
+---+-----+
| 1 | Test 123 |
| 2 | Test 456 |
| 3 | Test 789 |
| 5 | New row - db2 |
+---+-----+
4 rows in set (0.00 sec)

```

To generate the SQL statements for data transformations to make `db1.t1` the same as `db2.t1`, use the `--changes-for=server1` [327] option. We must also include the `-a` option to ensure that the data consistency test is run. The following command illustrates the options used and an excerpt from the results generated:

```

$ mysqldbcompare --server1=root:root@localhost \
    --server2=root:root@localhost db1:db2 --changes-for=server1 -a \
    --difftype=mysql

[...]

```

mysqldbcompare

```
#                                         Defn      Row      Data
# Type      Object Name                Diff      Count
Check #  
-----  
# TABLE      t1                  pass      pass      FAIL  
# # Data transformations for direction = server1:  
  
# Data differences found among rows: UPDATE db1.t1 SET b = 'Test 123'  
WHERE a = '1'; UPDATE db1.t1 SET b = 'Test 789' WHERE a = '3'; DELETE  
FROM db1.t1 WHERE a = '4'; INSERT INTO db1.t1 (a, b) VALUES('5', 'New  
row - db2');  
  
# Database consistency check failed.  # # ...done
```

Similarly, when the same command is run with [--changes-for=server2 \[327\]](#) and [--difftype=mysql \[327\]](#), the following report is generated:

```
$ mysqldbcompare --server1=root:root@localhost \
    --server2=root:root@localhost db1:db2 --changes-for=server2 -a \
    --difftype=mysql  
  
[...]  
  
#                                         Defn      Row      Data
# Type      Object Name                Diff      Count
Check #  
-----  
# TABLE      t1                  pass      pass      FAIL  
# # Data transformations for direction = server2:  
  
# Data differences found among rows: UPDATE db2.t1 SET b = 'Test 789'  
WHERE a = '1'; UPDATE db2.t1 SET b = 'Test 123' WHERE a = '3'; DELETE  
FROM db2.t1 WHERE a = '5'; INSERT INTO db2.t1 (a, b) VALUES('4', 'New  
row - db1');
```

With the [--difftype=mysql \[327\]](#) SQL generation option set, [--show-reverse \[327\]](#) shows the object transformations in both directions. Here is an excerpt of the results:

```
$ mysqldbcompare --server1=root:root@localhost \
--server2=root:root@localhost db1:db2 --changes-for=server1 \
--show-reverse -a --difftype=mysql  
  
[...]  
  
#                                         Defn      Row      Data
# Type      Object Name                Diff      Count
Check #  
-----  
# TABLE      t1                  pass      pass      FAIL  
# # Data transformations for direction = server1:  
  
# Data differences found among rows: UPDATE db1.t1 SET b = 'Test 123'  
WHERE a = '1'; UPDATE db1.t1 SET b = 'Test 789' WHERE a = '3'; DELETE  
FROM db1.t1 WHERE a = '4'; INSERT INTO db1.t1 (a, b) VALUES('5', 'New  
row - db2');  
  
# Data transformations for direction = server2:  
  
# Data differences found among rows: UPDATE db2.t1 SET b = 'Test 789'  
WHERE a = '1'; UPDATE db2.t1 SET b = 'Test 123' WHERE a = '3'; DELETE  
FROM db2.t1 WHERE a = '5'; INSERT INTO db2.t1 (a, b) VALUES('4', 'New  
row - db1');
```

```
# Database consistency check failed. # # ...done
```

15.6.4. mysqldbcopy

15.6.4.1. mysqldbcopy — Copy Database Objects Between Servers

This utility copies a database on a source server to a database on a destination server. If the source and destination servers are different, the database names can be the same or different. If the source and destination servers are the same, the database names must be different.

The utility accepts one or more database pairs on the command line. To name a database pair, use *db_name:new_db_name* syntax to specify the source and destination names explicitly. If the source and destination database names are the same, *db_name* can be used as shorthand for *db_name:db_name*.

By default, the operation copies all objects (tables, views, triggers, events, procedures, functions, and database-level grants) and data to the destination server. There are options to turn off copying any or all of the objects as well as not copying the data.

To exclude specific objects by name, use the [--exclude \[332\]](#) option with a name in *db.*obj** format, or you can supply a search pattern. For example, [--exclude=db1.trig1 \[332\]](#) excludes the single trigger and [--exclude=trig_ \[332\]](#) excludes all objects from all databases having a name that begins with *trig* and has a following character.

By default, the utility creates each table on the destination server using the same storage engine as the original table. To override this and specify the storage engine to use for all tables created on the destination server, use the [--new-storage-engine \[332\]](#) option. If the destination server supports the new engine, all tables use that engine.

To specify the storage engine to use for tables for which the destination server does not support the original storage engine on the source server, use the [--default-storage-engine \[332\]](#) option.

The [--new-storage-engine \[332\]](#) option takes precedence over [--default-storage-engine \[332\]](#) if both are given.

If the [--new-storage-engine \[332\]](#) or [--default-storage-engine \[332\]](#) option is given and the destination server does not support the specified storage engine, a warning is issued and the server's default storage engine setting is used instead.

By default, the operation uses a consistent snapshot to read the source databases. To change the locking mode, use the [--locking \[332\]](#) option with a locking type value. Use a value of **no-locks** to turn off locking altogether or **lock-all** to use only table locks. The default value is **snapshot**. Additionally, the utility uses WRITE locks to lock the destination tables during the copy.

You can include replication statements for copying data among a master and slave or between slaves. The [--rpl \[333\]](#) option permits you to select from the following replication statements to include in the export.

- **master**

Include the **CHANGE MASTER** statement to start a new slave with the current server acting as the master. This executes the appropriate STOP and START slave statements. The **STOP SLAVE** statement is executed at the start of the copy and the **CHANGE MASTER** followed by the **START SLAVE** statements are executed after the copy.

- **slave**

Include the **CHANGE MASTER** statement to start a new slave using the current server's master information. This executes the appropriate STOP and START slave statements. The **STOP SLAVE**

statement is executed at the start of the copy and the **CHANGE MASTER** followed by the **START SLAVE** statements follow the copy.

To include the replication user in the **CHANGE MASTER** statement, use the [--rpl-user \[333\]](#) option to specify the user and password. If this option is omitted, the utility attempts to identify the replication user. In the event that there are multiple candidates or the user requires a password, the utility aborts with an error.

If you attempt to copy databases on a server with GTIDs enabled (GTID_MODE = ON), a warning will be generated if the copy does not include all databases. This is because the GTID statements generated include the GTIDs for all databases and not only those databases in the export.

The utility will also generate a warning if you copy databases on a GTID enabled server but use the [--skip-gtid \[333\]](#) option.

To make the most use of GTIDs, you should copy all of the databases on the server with the [--all \[333\]](#) option.

OPTIONS

`mysqldbcopy` accepts the following command-line options:

- `--help`

Display a help message and exit.

- `--default-storage-engine=<def_engine>`

The engine to use for tables if the destination server does not support the original storage engine on the source server.

- `--destination=<destination>`

Connection information for the destination server in the format: `<user>[:<passwd>]@<host>[:<port>][:<socket>]` or `<login-path>[:<port>][:<socket>]` (where `<passwd>` is optional and either `<port>` or `<socket>` must be provided).

- `--exclude=<exclude>, -x<exclude>`

Exclude one or more objects from the operation using either a specific name such as db1.t1 or a search pattern. Use this option multiple times to specify multiple exclusions. By default, patterns use **LIKE** matching. With the [--regexp \[333\]](#) option, patterns use **REGEXP** matching.

This option does not apply to grants.

- `--force`

Drop each database to be copied if exists before copying anything into it. Without this option, an error occurs if you attempt to copy objects into an existing database.

- `--locking=<locking>`

Choose the lock type for the operation. Permitted lock values are **no-locks** (do not use any table locks), **lock-all** (use table locks but no transaction and no consistent read), and **snapshot** (consistent read using a single transaction). The default is **snapshot**.

- `--new-storage-engine=<new_engine>`

The engine to use for all tables created on the destination server.

- `--quiet, -q`

Turn off all messages for quiet execution.

- `--regexp, --basic-regexp, -G`

Perform pattern matches using the **REGEXP** operator. The default is to use **LIKE** for matching.

- `--rpl=<dump_option>, --replication=<dump_option>`

Include replication information. Permitted values are **master** (include the **CHANGE MASTER** statement using the source server as the master), **slave** (include the **CHANGE MASTER** statement using the destination server's master information), and **both** (include the **master** and **slave** options where applicable).

- `--rpl-user=<replication_user>`

The user and password for the replication user requirement in the form: `<user>[:<password>]` or `<login-path>`. E.g. `rpl:passwd` Default = None.

- `I --skip-gtid`

Skip creation and execution of GTID statements during the copy operation.

- `--all`

Copy all of the databases on the server.

- `--skip=<objects>`

Specify objects to skip in the operation as a comma-separated list (no spaces). Permitted values are **CREATE_DB**, **DATA**, **EVENTS**, **FUNCTIONS**, **GRANTS**, **PROCEDURES**, **TABLES**, **TRIGGERS**, and **VIEWS**.

- `--source=<source>`

Connection information for the destination server in the format: `<user>[:<passwd>]@<host>[:<port>]` `[:<socket>]` or `<login-path>[:<port>][::<socket>]` (where `<passwd>` is optional and either `<port>` or `<socket>` must be provided).

- `--threads`

Use multiple threads for cross-server copy. The default is 1.

- `--verbose, -v`

Specify how much information to display. Use this option multiple times to increase the amount of information. For example, `-v` = verbose, `-vv` = more verbose, `-vvv` = debug.

- `--version`

Display version information and exit.

NOTES

You must provide connection parameters (user, host, password, and so forth) for an account that has the appropriate privileges to access all objects in the operation.

To copy all objects from a source, the user must have these privileges: **SELECT** and **SHOW VIEW** for the database, and **SELECT** for the `mysql` database.

To copy all objects to a destination, the user must have these privileges: **CREATE** for the database, **SUPER** (when binary logging is enabled) for procedures and functions, and **GRANT OPTION** to copy grants.

Actual privileges required may differ from installation to installation depending on the security privileges present and whether the database contains certain objects such as views or events and whether binary logging is enabled.

The `--new-storage-engine` [332] and `--default-storage-engine` [332] options apply to all destination tables in the operation.

Some option combinations may result in errors during the operation. For example, eliminating tables but not views may result in an error a the view is copied.

The `--rpl` [333] option is not valid for copying databases on the same server. An error will be generated.

When copying data and including the GTID commands, you may encounter an error similar to "GTID_PURGED can only be set when GTID_EXECUTED is empty". This occurs because the destination server is not in a clean replication state. To alleviate this problem, you can issue a "RESET MASTER" command on the destination prior to executing the copy.

Cloning databases that contain foreign key constraints does not change the constraint in the cloned table. For example, if table db1.t1 has a foreign key constraint on table db1.t2, when db1 is cloned to db2, table db2.t1 will have a foreign key constraint on db1.t2.

The path to the MySQL client tools should be included in the PATH environment variable in order to use the authentication mechanism with login-paths. This will allow the utility to use the `my_print_defaults` tools which is required to read the login-path values from the login configuration file (`.mylogin.cnf`).

If any database identifier specified as an argument contains special characters or is a reserved word, then it must be appropriately quoted with backticks (`). In turn, names quoted with backticks must also be quoted with single or double quotes depending on the operating system, i.e. ("") in Windows or ('') in non-Windows systems, in order for the utilities to read backtick quoted identifiers as a single argument. For example, to copy a database with the name **weird`db.name** with **other:weird`db.name**, the database pair must be specified using the following syntax (in non-Windows): `"weird``db.name":`other:weird``db.name`"`.

EXAMPLES

The following example demonstrates how to use the utility to copy a database named `util_test` to a new database named `util_test_copy` on the same server:

```
$ mysqldbcopy \
  --source=root:pass@localhost:3310:/test123/mysql.sock \
  --destination=root:pass@localhost:3310:/test123/mysql.sock \
  util_test:util_test_copy
# Source on localhost: ... connected.
# Destination on localhost: ... connected.
# Copying database util_test renamed as util_test_copy
# Copying TABLE util_test.t1
# Copying table data.
# Copying TABLE util_test.t2
# Copying table data.
# Copying TABLE util_test.t3
# Copying table data.
# Copying TABLE util_test.t4
```

```
# Copying table data.
# Copying VIEW util_test.v1
# Copying TRIGGER util_test.trg
# Copying PROCEDURE util_test.pl
# Copying FUNCTION util_test.fl
# Copying EVENT util_test.el
# Copying GRANTS from util_test
#...done.
```

If the database to be copied does not contain only InnoDB tables and you want to ensure data integrity of the copied data by locking the tables during the read step, add a `--locking=lock-all` [332] option to the command:

```
$ mysqldbcopy \
--source=root:pass@localhost:3310:/test123/mysql.sock \
--destination=root:pass@localhost:3310:/test123/mysql.sock \
util_test:util_test_copy --locking=lock-all
# Source on localhost: ... connected.
# Destination on localhost: ... connected.
# Copying database util_test renamed as util_test_copy
# Copying TABLE util_test.t1
# Copying table data.
# Copying TABLE util_test.t2
# Copying table data.
# Copying TABLE util_test.t3
# Copying table data.
# Copying TABLE util_test.t4
# Copying table data.
# Copying VIEW util_test.v1
# Copying TRIGGER util_test.trg
# Copying PROCEDURE util_test.pl
# Copying FUNCTION util_test.fl
# Copying EVENT util_test.el
# Copying GRANTS from util_test
#...done.
```

To copy one or more databases from a master to a slave, you can use the following command to copy the databases. Use the master as the source and the slave as the destination:

```
$ mysqldbcopy --source=root@localhost:3310 \
--destination=root@localhost:3311 test123 --rpl=master \
--rpl-user=rpl
# Source on localhost: ... connected.
# Destination on localhost: ... connected.
# Source on localhost: ... connected.
# Stopping slave
# Copying database test123
# Copying TABLE test123.t1
# Copying data for TABLE test123.t1
# Connecting to the current server as master
# Starting slave
#...done.
```

To copy a database from one slave to another attached to the same master, you can use the following command using the slave with the database to be copied as the source and the slave where the database needs to be copied as the destination:

```
$ mysqldbcopy --source=root@localhost:3311 \
--destination=root@localhost:3312 test123 --rpl=slave \
--rpl-user=rpl
# Source on localhost: ... connected.
# Destination on localhost: ... connected.
# Source on localhost: ... connected.
# Stopping slave
# Copying database test123
```

```
# Copying TABLE test123.t1
# Copying data for TABLE test123.t1
# Connecting to the current server's master
# Starting slave
#...done.
```

15.6.5. mysqldbexport

15.6.5.1. mysqldbexport — Export Object Definitions or Data from a Database

This utility exports metadata (object definitions) or data or both from one or more databases. By default, the export includes only definitions.

`mysqldbexport` differs from `mysqldump` in that it can produce output in a variety of formats to make your data extraction/transport much easier. It permits you to export your data in the format most suitable to an external tool, another MySQL server, or other use without the need to reformat the data.

To exclude specific objects by name, use the `--exclude` [338] option with a name in `db.*obj*` format, or you can supply a search pattern. For example, `--exclude=db1.trig1` [338] excludes the single trigger and `--exclude=trig_` [338] excludes all objects from all databases having a name that begins with `trig` and has a following character.

To skip objects by type, use the `--skip` [339] option with a list of the objects to skip. This enables you to extract a particular set of objects, say, for exporting only events (by excluding all other types). Similarly, to skip creation of **UPDATE** statements for **BLOB** data, specify the `--skip-blobs` [339] option.

To specify how to display output, use one of the following values with the `--format` [338] option:

- **sql** (default)

Display output using SQL statements. For definitions, this consists of the appropriate **CREATE** and **GRANT** statements. For data, this is an **INSERT** statement (or bulk insert if the `--bulk-insert` [338] option is specified).

- **grid**

Display output in grid or table format like that of the `mysql` monitor.

- **csv**

Display output in comma-separated values format.

- **tab**

Display output in tab-separated format.

- **vertical**

Display output in single-column format like that of the `\G` command for the `mysql` monitor.

To specify how much data to display, use one of the following values with the `--display` [338] option:

- **brief**

Display only the minimal columns for recreating the objects.

- **full**

Display the complete column list for recreating the objects.

- **names**

Display only the object names.

Note: For SQL-format output, the [--display \[338\]](#) option is ignored.

To turn off the headers for **csv** or **tab** display format, specify the [--no-headers \[339\]](#) option.

To turn off all feedback information, specify the [--quiet \[339\]](#) option.

To write the data for individual tables to separate files, use the [--file-per-table \[338\]](#) option. The name of each file is composed of the database and table names followed by the file format. For example, the following command produces files named db1.*table_name*.csv:

```
mysqldbexport --server=root@server1:3306 --format=csv db1 --export=data
```

By default, the operation uses a consistent snapshot to read the source databases. To change the locking mode, use the [--locking \[339\]](#) option with a locking type value. Use a value of **no-locks** to turn off locking altogether or **lock-all** to use only table locks. The default value is **snapshot**. Additionally, the utility uses WRITE locks to lock the destination tables during the copy.

You can include replication statements for exporting data among a master and slave or between slaves. The [--rpl \[339\]](#) option permits you to select from the following replication statements to include in the export.

- **master**

Include the **CHANGE MASTER** statement to start a new slave with the current server acting as the master. This places the appropriate STOP and START slave statements in the export whereby the **STOP SLAVE** statement is placed at the start of the export and the **CHANGE MASTER** followed by the **START SLAVE** statements are placed after the export stream.

- **slave**

Include the **CHANGE MASTER** statement to start a new slave using the current server's master information. This places the appropriate STOP and START slave statements in the export whereby the **STOP SLAVE** statement is placed at the start of the export and the **CHANGE MASTER** followed by the **START SLAVE** statements are placed after the export stream.

- **both**

Include both the 'master' and 'slave' information for **CHANGE MASTER** statements for either spawning a new slave with the current server's master or using the current server as the master. All statements generated are labeled and commented to enable the user to choose which to include when imported.

To include the replication user in the **CHANGE MASTER** statement, use the [--rpl-user \[339\]](#) option to specify the user and password. If this option is omitted, the utility attempts to identify the replication user. In the event that there are multiple candidates or the user requires a password, these statements are placed inside comments for the **CHANGE MASTER** statement.

You can also use the [--comment-rpl \[338\]](#) option to place the replication statements inside comments for later examination.

If you specify the [--rpl-file \[339\]](#) option, the utility writes the replication statements to the file specified instead of including them in the export stream.

If you attempt to export databases on a server with GTIDs enabled (GTID_MODE = ON), a warning will be generated if the export does not include all databases. This is because the GTID statements generated include the GTIDs for all databases and not only those databases in the export.

The utility will also generate a warning if you export databases on a GTID enabled server but use the [--skip-gtid \[339\]](#) option.

To make the most use of GTIDs and export/import, you should export all of the databases on the server with the [--all \[339\]](#) option. This will generate an export file with all of the databases and the GTIDs executed to that point.

Importing this file on another server will ensure that server has all of the data as well as all of the GTIDs recorded correctly in its logs.

OPTIONS

`mysqldbexport` accepts the following command-line options:

- `--help`

Display a help message and exit.

- `--bulk-insert, -b`

Use bulk insert statements for data.

- `--comment-rpl`

Place the replication statements in comment statements. Valid only with the [--rpl \[339\]](#) option.

- `--display=<display>, -d<display>`

Control the number of columns shown. Permitted display values are **brief** (minimal columns for object creation), **full* (all columns)**, and ****names** (only object names; not valid for [--format=sql \[338\]](#)). The default is **brief**.

- `--exclude=<exclude>, -x<exclude>`

Exclude one or more objects from the operation using either a specific name such as `db1.t1` or a search pattern. Use this option multiple times to specify multiple exclusions. By default, patterns use **LIKE** matching. With the [--regexp \[339\]](#) option, patterns use **REGEXP** matching.

This option does not apply to grants.

- `--export=<export>, -e<export>`

Specify the export format. Permitted format values are **definitions** = export only the definitions (metadata) for the objects in the database list, **data** = export only the table data for the tables in the database list, and **both** = export the definitions and the data. The default is **definitions**.

- `--file-per-table`

Write table data to separate files. This is Valid only if the export output includes data (that is, if [--export=data \[338\]](#) or [--export=both \[338\]](#) are given). This option produces files named `db_name.*tbl_name*.*format*`. For example, a **csv** export of two tables named `t1` and `t2` in database `d1`, results in files named `db1.t1.csv` and `db1.t2.csv`. If table definitions are included in the export, they are written to stdout as usual.

- `--format=<format>, -f<format>`

Specify the output display format. Permitted format values are **sql**, **grid**, **tab**, **csv**, and **vertical**. The default is **sql**.

- `--locking=<locking>`

Choose the lock type for the operation. Permitted lock values are **no-locks** (do not use any table locks), **lock-all** (use table locks but no transaction and no consistent read), and **snapshot** (consistent read using a single transaction). The default is **snapshot**.

- `--no-headers, -h`

Do not display column headers. This option applies only for **csv** and **tab** output.

- `--quiet, -q`

Turn off all messages for quiet execution.

- `--regexp, --basic-regexp, -G`

Perform pattern matches using the **REGEXP** operator. The default is to use **LIKE** for matching.

- `--rpl=<dump_option>, --replication=<dump_option>`

Include replication information. Permitted values are **master** (include the **CHANGE MASTER** statement using the source server as the master), **slave** (include the **CHANGE MASTER** statement using the destination server's master information), and **both** (include the **master** and **slave** options where applicable).

- `--rpl-file=RPL_FILE, --replication-file=RPL_FILE`

The path and file name where the generated replication information should be written. Valid only with the [--rpl](#) [339] option.

- `--rpl-user=<replication_user>`

The user and password for the replication user requirement, in the format: `<user>[:<password>]` or `<login-path>`. For example, `rpl:passwd`. The default is None.

- `--server=<server>`

Connection information for the server in `<user>[:<passwd>]@<host>[:<port>][:<socket>]` or `<login-path>[:<port>][:<socket>]`.

- `--skip=<skip-objects>`

Specify objects to skip in the operation as a comma-separated list (no spaces). Permitted values are **CREATE_DB**, **DATA**, **EVENTS**, **FUNCTIONS**, **GRANTS**, **PROCEDURES**, **TABLES**, **TRIGGERS**, and **VIEWS**.

- `--skip-blobs`

Do not export **BLOB** data.

- `--skip-gtid`

Skip creation of GTID_PURGED statements.

- `--all`

Generate an export file with all of the databases and the GTIDs executed to that point.

- `--verbose, -v`

Specify how much information to display. Use this option multiple times to increase the amount of information. For example, `-v` = verbose, `-vv` = more verbose, `-vvv` = debug.

- `--version`

Display version information and exit.

NOTES

You must provide connection parameters (user, host, password, and so forth) for an account that has the appropriate privileges to access all objects in the operation.

To export all objects from a source database, the user must have these privileges: **SELECT** and **SHOW VIEW** on the database as well as **SELECT** on the `mysql` database.

Actual privileges needed may differ from installation to installation depending on the security privileges present and whether the database contains certain objects such as views or events.

Some combinations of the options may result in errors when the export is imported later. For example, eliminating tables but not views may result in an error when a view is imported on another server.

For the `--format` [338], `--export` [338], and `--display` [338] options, the permitted values are not case sensitive. In addition, values may be specified as any unambiguous prefix of a valid value. For example, `--format=g` [338] specifies the grid format. An error occurs if a prefix matches more than one valid value.

The path to the MySQL client tools should be included in the PATH environment variable in order to use the authentication mechanism with login-paths. This will allow the utility to use the `my_print_defaults` tools which is required to read the login-path values from the login configuration file (`.mylogin.cnf`).

If any database identifier specified as an argument contains special characters or is a reserved word, then it must be appropriately quoted with backticks (`). In turn, names quoted with backticks must also be quoted with single or double quotes depending on the operating system, i.e. ("") in Windows or ('') in non-Windows systems, in order for the utilities to read backtick quoted identifiers as a single argument. For example, to export a database with the name `weird`db.name`, it must be specified as argument using the following syntax (in non-Windows): `"weird``db.name"`.

EXAMPLES

To export the definitions of the database `dev` from a MySQL server on the local host via port 3306, producing output consisting of **CREATE** statements, use this command:

```
$ mysqldbexport --server=root:pass@localhost \
  --skip=GRANTS --export=DEFINITIONS util_test
# Source on localhost: ... connected.
# Exporting metadata from util_test
DROP DATABASE IF EXISTS util_test;
CREATE DATABASE util_test;
USE util_test;
# TABLE: util_test.t1
CREATE TABLE `t1` (
  `a` char(30) DEFAULT NULL
) ENGINE=MEMORY DEFAULT CHARSET=latin1;
# TABLE: util_test.t2
CREATE TABLE `t2` (
  `a` char(30) DEFAULT NULL
) ENGINE=MyISAM DEFAULT CHARSET=latin1;
# TABLE: util_test.t3
CREATE TABLE `t3` (
```

```

`a` int(11) NOT NULL AUTO_INCREMENT,
`b` char(30) DEFAULT NULL,
PRIMARY KEY (`a`)
) ENGINE=InnoDB AUTO_INCREMENT=4 DEFAULT CHARSET=latin1;
# TABLE: util_test.t4
CREATE TABLE `t4` (
`c` int(11) NOT NULL,
`d` int(11) NOT NULL,
KEY `ref_t3` (`c`),
CONSTRAINT `ref_t3` FOREIGN KEY (`c`) REFERENCES `t3` (`a`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
# VIEW: util_test.v1
[...]
#...done.

```

Similarly, to export the data of the database `util_test`, producing bulk insert statements, use this command:

```

$ mysqldbexport --server=root:pass@localhost \
  --export=DATA --bulk-insert util_test
# Source on localhost: ... connected.
USE util_test;
# Exporting data from util_test
# Data for table util_test.t1:
INSERT INTO util_test.t1 VALUES  ('01 Test Basic database example'),
('02 Test Basic database example'),
('03 Test Basic database example'),
('04 Test Basic database example'),
('05 Test Basic database example'),
('06 Test Basic database example'),
('07 Test Basic database example');
# Data for table util_test.t2:
INSERT INTO util_test.t2 VALUES  ('11 Test Basic database example'),
('12 Test Basic database example'),
('13 Test Basic database example');
# Data for table util_test.t3:
INSERT INTO util_test.t3 VALUES  (1, '14 test fkeys'),
(2, '15 test fkeys'),
(3, '16 test fkeys');
# Data for table util_test.t4:
INSERT INTO util_test.t4 VALUES  (3, 2);
#...done.

```

If the database to be exported does not contain only InnoDB tables and you want to ensure data integrity of the exported data by locking the tables during the read step, add a `--locking=lock-all` [339] option to the command:

```

$ mysqldbexport --server=root:pass@localhost \
  --export=DATA --bulk-insert util_test --locking=lock-all
# Source on localhost: ... connected.
USE util_test;
# Exporting data from util_test
# Data for table util_test.t1:
INSERT INTO util_test.t1 VALUES  ('01 Test Basic database example'),
('02 Test Basic database example'),
('03 Test Basic database example'),
('04 Test Basic database example'),
('05 Test Basic database example'),
('06 Test Basic database example'),
('07 Test Basic database example');
# Data for table util_test.t2:
INSERT INTO util_test.t2 VALUES  ('11 Test Basic database example'),
('12 Test Basic database example'),
('13 Test Basic database example');
# Data for table util_test.t3:
INSERT INTO util_test.t3 VALUES  (1, '14 test fkeys'),

```

```
(2, '15 test fkeys'),
(3, '16 test fkeys');
# Data for table util_test.t4:
INSERT INTO util_test.t4 VALUES  (3, 2);
#...done.
```

To export a database and include the replication commands to use the current server as the master (for example, to start a new slave using the current server as the master), use the following command:

```
$ mysqldbexport --server=root@localhost:3311 util_test \
--export=both --rpl-user=rpl:rpl --rpl=master -v
# Source on localhost: ... connected.
#
# Stopping slave
STOP SLAVE;
#
# Source on localhost: ... connected.
# Exporting metadata from util_test
DROP DATABASE IF EXISTS util_test;
CREATE DATABASE util_test;
USE util_test;
# TABLE: util_test.t1
CREATE TABLE `t1` (
  `a` char(30) DEFAULT NULL
) ENGINE=MEMORY DEFAULT CHARSET=latin1;
#...done.
# Source on localhost: ... connected.
USE util_test;
# Exporting data from util_test
# Data for table util_test.t1:
INSERT INTO util_test.t1 VALUES ('01 Test Basic database example');
INSERT INTO util_test.t1 VALUES ('02 Test Basic database example');
INSERT INTO util_test.t1 VALUES ('03 Test Basic database example');
INSERT INTO util_test.t1 VALUES ('04 Test Basic database example');
INSERT INTO util_test.t1 VALUES ('05 Test Basic database example');
INSERT INTO util_test.t1 VALUES ('06 Test Basic database example');
INSERT INTO util_test.t1 VALUES ('07 Test Basic database example');
#...done.
#
# Connecting to the current server as master
CHANGE MASTER TO MASTER_HOST = 'localhost',
MASTER_USER = 'rpl',
MASTER_PASSWORD = 'rpl',
MASTER_PORT = 3311,
MASTER_LOG_FILE = 'clone-bin.000001' ,
MASTER_LOG_POS = 106;
#
# Starting slave
START SLAVE;
#
```

Similarly, to export a database and include the replication commands to use the current server's master (for example, to start a new slave using the same the master), use the following command:

```
$ mysqldbexport --server=root@localhost:3311 util_test \
--export=both --rpl-user=rpl:rpl --rpl=slave -v
# Source on localhost: ... connected.
#
# Stopping slave
STOP SLAVE;
#
# Source on localhost: ... connected.
# Exporting metadata from util_test
DROP DATABASE IF EXISTS util_test;
CREATE DATABASE util_test;
USE util_test;
```

```

# TABLE: util_test.t1
CREATE TABLE `t1` (
  `a` char(30) DEFAULT NULL
) ENGINE=MEMORY DEFAULT CHARSET=latin1;
#...done.
# Source on localhost: ... connected.
USE util_test;
# Exporting data from util_test
# Data for table util_test.t1:
INSERT INTO util_test.t1 VALUES ('01 Test Basic database example');
INSERT INTO util_test.t1 VALUES ('02 Test Basic database example');
INSERT INTO util_test.t1 VALUES ('03 Test Basic database example');
INSERT INTO util_test.t1 VALUES ('04 Test Basic database example');
INSERT INTO util_test.t1 VALUES ('05 Test Basic database example');
INSERT INTO util_test.t1 VALUES ('06 Test Basic database example');
INSERT INTO util_test.t1 VALUES ('07 Test Basic database example');
#...done.
#
# Connecting to the current server's master
CHANGE MASTER TO MASTER_HOST = 'localhost',
  MASTER_USER = 'rpl',
  MASTER_PASSWORD = 'rpl',
  MASTER_PORT = 3310,
  MASTER_LOG_FILE = 'clone-bin.000001' ,
  MASTER_LOG_POS = 1739;
#
# Starting slave
START SLAVE;
#

```

15.6.6. mysqldbimport

15.6.6.1. `mysqldbimport` — Import Object Definitions or Data into a Database

This utility imports metadata (object definitions) or data or both for one or more databases from one or more files.

If an object exists on the destination server with the same name as an imported object, it is dropped first before importing the new object.

To skip objects by type, use the `--skip` [345] option with a list of the objects to skip. This enables you to extract a particular set of objects, say, for importing only events (by excluding all other types). Similarly, to skip creation of `UPDATE` statements for `BLOB` data, specify the `--skip-blobs` [345] option.

To specify the input format, use one of the following values with the `--format` [345] option. These correspond to the output formats of the `mysqldbexport` utility:

- **sql** (default)

Input consists of SQL statements. For definitions, this consists of the appropriate `CREATE` and `GRANT` statements. For data, this is an `INSERT` statement (or bulk insert if the `--bulk-insert` [344] option is specified).

- **grid**

Display output in grid or table format like that of the `mysql` monitor.

- **csv**

Input is formatted in comma-separated values format.

- **raw_csv**

Input is a simple CSV file containing uniform rows with values separated with commas. The file can contain a header (the first row) that lists the table columns. The option [--table](#) [346] is required to use this format.

- **tab**

Input is formatted in tab-separated format.

- **vertical**

Display output in single-column format like that of the `\G` command for the `mysql` monitor.

To indicate that input in **csv** or **tab** format does not contain column headers, specify the [--no-headers](#) [345] option.

To turn off all feedback information, specify the [--quiet](#) [345] option.

By default, the utility creates each table on the destination server using the same storage engine as the original table. To override this and specify the storage engine to use for all tables created on the destination server, use the [--new-storage-engine](#) [345] option. If the destination server supports the new engine, all tables use that engine.

To specify the storage engine to use for tables for which the destination server does not support the original storage engine on the source server, use the [--default-storage-engine](#) [345] option.

The [--new-storage-engine](#) [345] option takes precedence over [--default-storage-engine](#) [345] if both are given.

If the [--new-storage-engine](#) [345] or [--default-storage-engine](#) [345] option is given and the destination server does not support the specified storage engine, a warning is issued and the server's default storage engine setting is used instead.

You must provide connection parameters (user, host, password, and so forth) for an account that has the appropriate privileges to access all objects in the operation. For details, see NOTES.

If you attempt to import databases on a server with GTIDs enabled (GTID_MODE = ON), a warning will be generated if the import file did not include the GTID statements generated by mysqldbexport.

The utility will also generate a warning if you import databases on a server without GTIDs enabled and there are GTID statements present in the file. Use the [--skip-gtid](#) [345] option to ignore the GTID statements.

To make the most use of GTIDs and export/import, you should export all of the databases on the server with the [--all](#) [339] option. This will generate an export file with all of the databases and the GTIDs executed to that point. Importing this file on another server will ensure that server has all of the data as well as all of the GTIDs recorded correctly in its logs.

OPTIONS

`mysqldbimport` accepts the following command-line options:

- `--help`

Display a help message and exit.

- `--bulk-insert, -b`

Use bulk insert statements for data.

- `--default-storage-engine=<def_engine>`

The engine to use for tables if the destination server does not support the original storage engine on the source server.

- `--drop-first, -d`

Drop each database to be imported if exists before importing anything into it.

- `--dryrun`

Import the files and generate the statements but do not execute them. This is useful for testing input file validity.

- `--format=<format>, -f<format>`

Specify the input format. Permitted format values are **sql**, **grid**, **tab**, **csv**, **raw_csv**, and **vertical**. The default is **sql**.

- `--import=<import_type>, -i<import_type>`

Specify the import format. Permitted format values are **definitions** = import only the definitions (metadata) for the objects in the database list, **data** = import only the table data for the tables in the database list, and **both** = import the definitions and the data. The default is **definitions**.

If you attempt to import objects into an existing database, the result depends on the import format. If the format is **definitions** or **both**, an error occurs unless [--drop-first \[345\]](#) is given. If the format is **data**, imported table data is added to existing table data.

- `--new-storage-engine=<new_engine>`

The engine to use for all tables created on the destination server.

- `--no-headers, -h`

Input does not contain column headers. This option applies only for **csv** and **tab** output.

- `--quiet, -q`

Turn off all messages for quiet execution.

- `--server=<server>`

Connection information for the server in the format: `<user>[:<passwd>]@<host>[:<port>][:<socket>]` or `<login-path>[:<port>][:<socket>]`.

- `--skip=<skip_objects>`

Specify objects to skip in the operation as a comma-separated list (no spaces). Permitted values are **CREATE_DB**, **DATA**, **EVENTS**, **FUNCTIONS**, **GRANTS**, **PROCEDURES**, **TABLES**, **TRIGGERS**, and **VIEWS**.

- `--skip-blobs`

Do not import **BLOB** data.

- `--skip-gtid`

Skip execution of GTID_PURGED statements.

- **--skip-rpl**

Do not execute replication commands.

- **--table=<db>,<table>**

Specify the table for importing. This option is required while using [--format=raw_csv](#).

- **--verbose, -v**

Specify how much information to display. Use this option multiple times to increase the amount of information. For example, [-v](#) = verbose, [-vv](#) = more verbose, [-vvv](#) = debug.

- **--version**

Display version information and exit.

NOTES

The login user must have the appropriate permissions to create new objects, access (read) the [mysql](#) database, and grant privileges. If a database to be imported already exists, the user must have read permission for it, which is needed to check the existence of objects in the database.

Actual privileges needed may differ from installation to installation depending on the security privileges present and whether the database contains certain objects such as views or events and whether binary logging is enabled.

Some combinations of the options may result in errors during the operation. For example, excluding tables but not views may result in an error when a view is imported.

The [--new-storage-engine](#) [345] and [--default-storage-engine](#) [345] options apply to all destination tables in the operation.

For the [--format](#) [345] and [--import](#) [345] options, the permitted values are not case sensitive. In addition, values may be specified as any unambiguous prefix of a valid value. For example, [--format=g](#) [345] specifies the grid format. An error occurs if a prefix matches more than one valid value.

When importing data and including the GTID commands, you may encounter an error similar to "GTID_PURGED can only be set when GTID_EXECUTED is empty". This occurs because the destination server is not in a clean replication state. To alleviate this problem, you can issue a "RESET MASTER" command on the destination prior to executing the import.

The path to the MySQL client tools should be included in the PATH environment variable in order to use the authentication mechanism with login-paths. This will allow the utility to use the my_print_defaults tools which is required to read the login-path values from the login configuration file (.mylogin.cnf).

EXAMPLES

To import the metadata from the [util_test](#) database to the server on the local host using a file in CSV format, use this command:

```
$ mysqldbimport --server=root@localhost --import=definitions \
  --format=csv data.csv
# Source on localhost: ... connected.
# Importing definitions from data.csv.
#...done.
```

Similarly, to import the data from the [util_test](#) database to the server on the local host, importing the data using bulk insert statements, use this command:

```
$ mysqlimport --server=root@localhost --import=data \
  --bulk-insert --format=csv data.csv
# Source on localhost: ... connected.
# Importing data from data.csv.
#...done.
```

To import both data and definitions from the `util_test` database, importing the data using bulk insert statements from a file that contains SQL statements, use this command:

```
$ mysqlimport --server=root@localhost --import=both --bulk-insert --format=sql data.sql
# Source on localhost: ... connected.
# Importing definitions and data from data.sql.
#...done.
```

15.6.7. mysqldiff

15.6.7.1. `mysqldiff` — Identify Differences Among Database Objects

This utility reads the definitions of objects and compares them using a diff-like method to determine whether they are the same. The utility displays the differences for objects that are not the same.

Use the notation `db1:db2` to name two databases to compare, or, alternatively just `db1` to compare two databases with the same name. The latter case is a convenience notation for comparing same-named databases on different servers.

The comparison may be run against two databases of different names on a single server by specifying only the `--server1` [349] option. The user can also connect to another server by specifying the `--server2` [349] option. In this case, `db1` is taken from `server1` and `db2` from `server2`.

When a database pair is specified, all objects in one database are compared to the corresponding objects in the other. Any objects not appearing in either database produce an error.

To compare a specific pair of objects, add an object name to each database name in `db.obj` format. For example, use `db1.obj1:db2.obj2` to compare two named objects, or `db1.obj1` to compare an object with the same name in databases with the same name. It is not legal to mix a database name with an object name. For example, `db1.obj1:db2` and `db1:db2.obj2` are illegal.

The comparison may be run against a single server for comparing two databases of different names on the same server by specifying only the `--server1` [349] option. Alternatively, you can also connect to another server by specifying the `--server2` [349] option. In this case, the first object to compare is taken from `server1` and the second from `server2`.

By default, the utility generates object differences as a difference report. However, you can generate a transformation report containing SQL statements for transforming the objects for conformity instead. Use the `'sql'` value for the `--difftype` [348] option to produce a listing that contains the appropriate `ALTER` commands to conform the object definitions for the object pairs specified. If a transformation cannot be formed, the utility reports the diff of the object along with a warning statement. See important limitations in the NOTES section.

To specify how to display diff-style output, use one of the following values with the `--difftype` [348] option:

- **unified** (default)

Display unified format output.

- **context**

Display context format output.

- **differ**

Display differ-style format output.

- **sql**

Display SQL transformation statement output.

The [--changes-for \[348\]](#) option controls the direction of the difference (by specifying the object to be transformed) in either the difference report (default) or the transformation report (designated with the [--difftype=sq \[348\]](#) option). Consider the following command:

```
mysqldiff --server1=root@host1 --server2=root@host2 --difftype=sql \
  db1.table1:dbx.table3
```

The leftmost database (`db1`) exists on the server designated by the [--server1 \[349\]](#) option (`host1`). The rightmost database (`dbx`) exists on the server designated by the [--server2 \[349\]](#) option (`host2`).

- [--changes-for=server1 \[348\]](#): Produce output that shows how to make the definitions of objects on `server1` like the definitions of the corresponding objects on `server2`.
- [--changes-for=server2 \[348\]](#): Produce output that shows how to make the definitions of objects on `server2` like the definitions of the corresponding objects on `server1`.

The default direction is `server1`.

For **sql** difference format, you can also see the reverse transformation by specifying the [--show-reverse \[349\]](#) option.

The utility stops on the first occurrence of missing objects or when an object does not match. To override this behavior, specify the [--force \[348\]](#) option to cause the utility to attempt to compare all objects listed as arguments.

OPTIONS

`mysqldiff` accepts the following command-line options:

- **--help**

Display a help message and exit.

- **--changes-for=<direction>**

Specify the server to show transformations to match the other server. For example, to see the transformation for transforming object definitions on `server1` to match the corresponding definitions on `server2`, use [--changes-for=server1 \[348\]](#). Permitted values are `server1` and `server2`. The default is `server1`.

- **--difftype=<difftype>, -d<difftype>**

Specify the difference display format. Permitted format values are **unified**, **context**, **differ**, and **sql**. The default is **unified**.

- **--force**

Do not halt at the first difference found. Process all objects to find all differences.

- `--quiet, -q`

Do not print anything. Return only an exit code of success or failure.

- `--server1=<source>`

Connection information for the first server in the format: `<user>[:<passwd>]@<host>[:<port>][:<socket>]` or `<login-path>[:<port>][:<socket>]`.

- `--server2=<source>`

Connection information for the second server in the format: `<user>[:<passwd>]@<host>[:<port>][:<socket>]` or `<login-path>[:<port>][:<socket>]`.

- `--show-reverse`

Produce a transformation report containing the SQL statements to conform the object definitions specified in reverse. For example, if `--changes-for [348]` is set to server1, also generate the transformation for server2. Note: The reverse changes are annotated and marked as comments.

- `--verbose, -v`

Specify how much information to display. Use this option multiple times to increase the amount of information. For example, `-v` = verbose, `-vv` = more verbose, `-vvv` = debug.

- `--version`

Display version information and exit.

- `--width=<number>`

Change the display width of the test report. The default is 75 characters.

NOTES

You must provide connection parameters (user, host, password, and so forth) for an account that has the appropriate privileges to access all objects to be compared.

The SQL transformation feature has these known limitations:

- When tables with partition differences are encountered, the utility generates the **ALTER TABLE** statement for all other changes but prints a warning and omits the partition differences.
- If the transformation detects table options in the source table (specified with the `--changes-for [348]` option) that are not changed or do not exist in the target table, the utility generates the **ALTER TABLE** statement for all other changes but prints a warning and omits the table option differences.
- Rename for events is not supported. This is because `mysqldiff` compares objects by name. In this case, depending on the direction of the diff, the event is identified as needing to be added or a **DROP EVENT** statement is generated.
- Changes in the definer clause for events are not supported.
- SQL extensions specific to MySQL Cluster are not supported.

For the `--difftype [348]` option, the permitted values are not case sensitive. In addition, values may be specified as any unambiguous prefix of a valid value. For example, `--difftype=d [348]` specifies the differ type. An error occurs if a prefix matches more than one valid value.

The path to the MySQL client tools should be included in the PATH environment variable in order to use the authentication mechanism with login-paths. This will allow the utility to use the my_print_defaults tools which is required to read the login-path values from the login configuration file (.mylogin.cnf).

If any database object identifier specified as an argument contains special characters or is a reserved word, then it must be appropriately quoted with backticks (`). In turn, names quoted with backticks must also be quoted with single or double quotes depending on the operating system, i.e. ("") in Windows or ('') in non-Windows systems, in order for the utilities to read backtick quoted identifiers as a single argument. For example, to show the difference between table **weird`table1** from database **weird`db.name** and table **weird`table2** from database **other:weird`db.name**, the objects pair must be specified using the following syntax (in non-Windows): "**weird``db.name`.**weird`table1`:**other:weird`db.name`.**weird`table2`******".**

EXAMPLES

To compare the `employees` and `emp` databases on the local server, use this command:

```
$ mysqldiff --server1=root@localhost employees:emp1
# server1 on localhost: ... connected.
WARNING: Objects in server1:employees but not in server2:emp1:
  EVENT: e1
Compare failed. One or more differences found.

$ mysqldiff --server1=root@localhost \
    employees.t1:emp1.t1 employees.t3:emp1.t3
# server1 on localhost: ... connected.                                [PASS]
# Comparing employees.t1 to emp1.t1
# server1 on localhost: ... connected.                                [PASS]
# Comparing employees.t3 to emp1.t3
Success. All objects are the same.

$ mysqldiff --server1=root@localhost \
    employees.salaries:emp1.salaries --differ
# server1 on localhost: ... connected.                                [FAIL]
# Comparing employees.salaries to emp1.salaries
# Object definitions are not the same:
CREATE TABLE `salaries` (
  `emp_no` int(11) NOT NULL,
  `salary` int(11) NOT NULL,
  `from_date` date NOT NULL,
  `to_date` date NOT NULL,
  PRIMARY KEY (`emp_no`, `from_date`),
  KEY `emp_no` (`emp_no`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1
?
^~~~~~+
+ ) ENGINE=MyISAM DEFAULT CHARSET=latin1
?
++ ^^^
Compare failed. One or more differences found.
```

The following examples show how to generate a transformation report. Assume the following object definitions:

Host1:

```
CREATE TABLE db1.table1 (num int, misc char(30));
```

Host2:

```
CREATE TABLE dbx.table3 (num int, notes char(30), misc char(55));
```

To generate a set of SQL statements that transform the definition of `db1.table1` to `dbx.table3`, use this command:

```
$ mysqldiff --server1=root@host1 --server2=root@host2 \
```

```
--changes-for=server1 --difftype=mysql \
  db1.table1:dbx.table3
# server1 on host1: ... connected.
# server2 on host2: ... connected.
# Comparing db1.table1 to dbx.table3 [FAIL]
# Transformation statements:

ALTER TABLE db1.table1
  ADD COLUMN notes char(30) AFTER a,
  CHANGE COLUMN misc misc char(55);

Compare failed. One or more differences found.
```

To generate a set of SQL statements that transform the definition of `dbx.table3` to `db1.table1`, use this command:

```
$ mysqldiff --server1=root@host1 --server2=root@host2 \
  --changes-for=server2 --difftype=mysql \
  db1.table1:dbx.table3
# server1 on host1: ... connected.
# server2 on host2: ... connected.
# Comparing db1.table1 to dbx.table3 [FAIL]
# Transformation statements:

ALTER TABLE dbx.table3
  DROP COLUMN notes,
  CHANGE COLUMN misc misc char(30);

Compare failed. One or more differences found.
```

To generate a set of SQL statements that transform the definitions of `dbx.table3` and `db1.table1` in both directions, use this command:

```
$ mysqldiff --server1=root@host1 --server2=root@host2 \
  --show-reverse --difftype=mysql \
  db1.table1:dbx.table3
# server1 on host1: ... connected.
# server2 on host2: ... connected.
# Comparing db1.table1 to dbx.table3 [FAIL]
# Transformation statements:

# --destination=server1:
ALTER TABLE db1.table1
  ADD COLUMN notes char(30) AFTER a,
  CHANGE COLUMN misc misc char(55);

# --destination=server2:
# ALTER TABLE dbx.table3
#   DROP COLUMN notes,
#   CHANGE COLUMN misc misc char(30);

Compare failed. One or more differences found.
```

15.6.8. mysqldiskusage

15.6.8.1. `mysqldiskusage` — Show Database Disk Usage

This utility displays disk space usage for one or more databases. The utility optionally displays disk usage for the binary log, slow query log, error log, general query log, relay log, and InnoDB tablespaces. The default is to show only database disk usage.

If the command line lists no databases, the utility shows the disk space usage for all databases.

Sizes displayed without a unit indicator such as MB are in bytes.

The utility determines the the location of the data directory by requesting it from the server. For a local server, the utility obtains size information directly from files in the data directory and InnoDB home directory. In this case, you must have file system access to read those directories. Disk space usage shown includes the sum of all storage engine- specific files such as the .MYI and .MYD files for MyISAM and the tablespace files for InnoDB.

If the file system read fails, or if the server is not local, the utility cannot determine exact file sizes. It is limited to information that can be obtained from the system tables, which therefore should be considered an estimate. For information read from the server, the account used to connect to the server must have the appropriate permissions to read any objects accessed during the operation.

If information requested requires file system access but is not available that way, the utility prints a message that the information is not accessible. This occurs, for example, if you request log usage but the server is not local and the log files cannot be examined directly.

To specify how to display output, use one of the following values with the [--format \[352\]](#) option:

- **grid** (default)

Display output in grid or table format like that of the [mysql](#) monitor.

- **csv**

Display output in comma-separated values format.

- **tab**

Display output in tab-separated format.

- **vertical**

Display output in single-column format like that of the [\G](#) command for the [mysql](#) monitor.

To turn off the headers for **csv** or **tab** display format, specify the [--no-headers \[353\]](#) option.

OPTIONS

[mysqldiskusage](#) accepts the following command-line options:

- **--help**

Display a help message and exit.

- **--all, -a**

Display all disk usage. This includes usage for databases, logs, and InnoDB tablespaces.

- **--binlog, -b**

Display binary log usage.

- **--empty, -m**

Include empty databases.

- **--format=<format>, -f<format>**

Specify the output display format. Permitted format values are **grid**, **csv**, **tab**, and **vertical**. The default is **grid**.

- **--innodb, -i**

Display InnoDB tablespace usage. This includes information about the shared InnoDB tablespace as well as .idb files for InnoDB tables with their own tablespace.

- **--logs, -l**

Display general query log, error log, and slow query log usage.

- **--no-headers, -h**

Do not display column headers. This option applies only for **csv** and **tab** output.

- **--quiet, -q**

Suppress informational messages.

- **--relaylog, -r**

Display relay log usage.

- **--server=<server>**

Connection information for the server in the format: *<user>[:<passwd>]@<host>[:<port>][:<socket>]* or *<login-path>[:<port>][:<socket>]*.

- **--verbose, -v**

Specify how much information to display. Use this option multiple times to increase the amount of information. For example, **-v** = verbose, **-vv** = more verbose, **-vvv** = debug.

- **--version**

Display version information and exit.

For the **--format** [352] option, the permitted values are not case sensitive. In addition, values may be specified as any unambiguous prefix of a valid value. For example, **--format=g** [352] specifies the grid format. An error occurs if a prefix matches more than one valid value.

NOTES

You must provide connection parameters (user, host, password, and so forth) for an account that has the appropriate privileges for all objects accessed during the operation.

The path to the MySQL client tools should be included in the PATH environment variable in order to use the authentication mechanism with login-paths. This will allow the utility to use the my_print_defaults tools which is required to read the login-path values from the login configuration file (.mylogin.cnf).

EXAMPLES

To show only the disk space usage for the `employees` and `test` databases in grid format (the default), use this command:

```
$ mysqldiskusage --server=root@localhost employees test
# Source on localhost: ... connected.
# Database totals:
+-----+-----+
| db_name |      total   |
+-----+-----+
```

```
+-----+-----+
| employees | 205,979,648 |
| test      |      4,096 |
+-----+-----+
Total database disk usage = 205,983,744 bytes or 196.00 MB
#...done.
```

To see all disk usage for the server in CSV format, use this command:

```
$ mysqldiskusage --server=root@localhost --format=csv -a -vv
# Source on localhost: ... connected.
# Database totals:
db_name,db_dir_size,data_size,misc_files,total
test1,0,0,0,0
db3,0,0,0,0
db2,0,0,0,0
db1,0,0,0,0
backup_test,19410,1117,18293,19410
employees,242519463,205979648,242519463,448499111
mysql,867211,657669,191720,849389
t1,9849,1024,8825,9849
test,56162,4096,52066,56162
util_test_a,19625,2048,17577,19625
util_test_b,17347,0,17347,17347
util_test_c,19623,2048,17575,19623

Total database disk usage = 449,490,516 bytes or 428.00 MB

# Log information.
# The general_log is turned off on the server.
# The slow_query_log is turned off on the server.

# binary log information:
Current binary log file = ./mysql-bin.000076
log_file,size
./data/mysql-bin.000076,125
./data/mysql-bin.000077,125
./data/mysql-bin.000078,556
./data/mysql-bin.000079,168398223
./data/mysql-bin.index,76

Total size of binary logs = 168,399,105 bytes or 160.00 MB

# Server is not an active slave - no relay log information.
# InnoDB tablespace information:
InnoDB_file,size,type,specification
./data/ib_logfile0,5242880,log file,
./data/ib_logfile1,5242880,log file,
./data/ibdata1,220200960,shared tablespace,ibdata1:210M
./data/ibdata2,10485760,shared tablespace,ibdata2:10M:autoextend
./data/employees/departments.ibd,114688,file tablespace,
./data/employees/dept_emp.ibd,30408704,file tablespace,
./data/employees/dept_manager.ibd,131072,file tablespace,
./data/employees/employees.ibd,23068672,file tablespace,
./data/employees/salaries.ibd,146800640,file tablespace,
./data/employees/titles.ibd,41943040,file tablespace,

Total size of InnoDB files = 494,125,056 bytes or 471.00 MB
#...done.
```

15.6.9. mysqlfailover

15.6.9.1. mysqlfailover — Automatic replication health monitoring and failover

This utility permits users to perform replication health monitoring and automatic failover on a replication topology consisting of a master and its slaves. The utility is designed to run interactively or continuously refreshing the health information at periodic intervals. Its primary mission is to monitor the master for failure and when a failure occurs, execute failover to the best slave available. The utility accepts a list of slaves to be considered the candidate slave.

This utility is designed to work exclusively for servers that support global transaction identifiers (GTIDs) and have GTID_MODE=ON. MySQL server versions 5.6.9 and higher support GTIDs. See the MySQL server online reference manual for more information about setting up replication with GTIDs enabled.

The user can specify the interval in seconds to use for detecting the master status and generating the health report using the [--interval \[358\]](#) option. At each interval, the utility will check to see if the server is alive via a ping operation followed by a check of the connector to detect if the server is still reachable. The ping operation can be controlled with the [--ping \[359\]](#) option (see below).

If the master is found to be offline or unreachable, the utility will execute one of the following actions based on the value of the [--failover-mode \[358\]](#) option.

auto Execute automatic failover to the list of candidates first and if no slaves are viable, continue to locate a viable candidate from the list of slaves. If no slaves are found to be a viable candidate, the utility will generate and error and exit.

Once a candidate is found, the utility will conduct failover to the best slave. The command will test each candidate slave listed for the prerequisites. Once a candidate slave is elected, it is made a slave of each of the other slaves thereby collecting any transactions executed on other slaves but not the candidate. In this way, the candidate becomes the most up-to-date slave.

elect This mode is the same as auto except if no candidates specified in the list of candidate slaves are viable, it does not check the remaining slaves and generates and error and exits.

fail This mode produces an error and does not failover when the master is downed. This mode is used to provide periodic health monitoring without the failover action taken.

For all options that permit specifying multiple servers, the options require a comma-separated list of connection parameters in the following form (where the password, port, and socket are optional):

```
<*user*>[:<*passwd*>]@<*host*>[:<*port*>][:<*socket*>] or  
<*login-path*>[:<*port*>][:<*socket*>]
```

The utility permits users to discover slaves connected to the master. In order to use the discover slaves feature, all slaves must use the --report-host and --report-port startup variables to specify the correct hostname and ip port of the slave. If these are missing or report the incorrect information, the slaves health may not be reported correctly or the slave may not be listed at all. The discover slaves feature ignores any slaves it cannot connect to.

The discover slaves feature is run automatically on each interval.

The utility permits the user to specify an external script to execute before and after the switchover and failover commands. The user can specify these with the [--exec-before \[358\]](#) and [--exec-after \[358\]](#) options. The return code of the script is used to determine success thus each script must report 0 (success) to be considered successful. If a script returns a value other than 0, the result code is presented in an error message.

The utility also permits the user to specify a script to be used for detecting a downed master or an application-level event to trigger failover. This can be specified using the [--exec-fail-check \[358\]](#) option. The return code for the script is used to invoke failover. A return code of 0 indicates failover should not take place. A return code other than 0 indicates failover should take place. This is checked at the start

of each interval if a script is supplied. The timeout option is not used in this case and the script is run once at the start of each interval.

The utility permits the user to log all actions taken during the commands. The [--log \[358\]](#) option requires a valid path and file name of the file to use for logging operations. The log is active only when this option is specified. The option [--log-age \[358\]](#) specifies the age in days that log entries are kept. The default is seven (7) days. Older entries are automatically deleted from the log file (but only if the [--log \[358\]](#) option is specified).

The format of the log file includes the date and time of the event, the level of the event (informational - INFO, warning - WARN, error - ERROR, critical failure - CRITICAL), and the message reported by the utility.

The interface provides the user with a number of options for displaying additional information. The user can choose to see the replication health report (default), or choose to see the list of GTIDs in use, the UUIDs in use, and if logging is enabled the contents of the log file. Each of these reports is described below.

health Display the replication health of the topology. This report is the default view for the interface. By default, this includes the host name, port, role (MASTER or SLAVE) of the server, state of the server (UP = is connected, WARN = not connected but can ping, DOWN = not connected and cannot ping), the GTID_MODE, and health state.

The master health state is based on the following; if GTID_MODE=ON, the server must have binary log enabled, and there must exist a user with the REPLICATE SLAVE privilege.

The slave health state is based on the following; the IO_THREAD and SQL_THREADS must be running, it must be connected to the master, there are no errors, the slave delay for non-gtid enabled scenarios is not more than the threshold provided by the [--max-position \[359\]](#) and the slave is reading the correct master log file, and slave delay is not more than the [--seconds-behind \[360\]](#) threshold option.

At each interval, if the discover slaves option was specified at startup and new slaves are discovered, the health report is refreshed.

gtid: Display the master's list of executed GTIDs, contents of the GTID variables; [@GLOBAL.GTID_EXECUTED](#), [@GLOBAL.GTID_PURGED](#), and [@@GLOBAL.GTID_OWNED](#). Thus, the user can toggle through four screens by pressing the 'G' key repeatedly. The display will cycle through all four screens restarting after the fourth screen.

UUID: Display universally unique identifiers (UUIDs) for all servers.

Log: This option is visible only if the [--log \[358\]](#) option is specified. Show the contents of the log file. This can be helpful to see at a later time when failover occurred and the actions or messages recorded at the time.

The user interface is designed to match the size of the terminal window in which it is run. A refresh option is provided to permit users to resize their terminal windows or refresh the display at any time. However, the interface will automatically resize to the terminal window on each interval.

The interface will display the name of the utility, the master's status including binary log file, position, and filters as well as the date and time of the next interval event.

The interface will also permit the user to scroll up or down through a list longer than what the terminal window permits. When a long list is presented, the scroll options become enabled. The user can scroll the list up with the up arrow key and down with the down arrow key.

Use the [--verbose \[360\]](#) option to see additional information in the health report and additional messages during failover.

The utility supports two modes of operation. The default mode, running as a console, works as described above. An additional mode that permits you to run the utility as a daemon is provided for POSIX platforms.

When run as a daemon, the utility does not have interactivity. However, all events are written to the log file. You can control what is written to the log by using the [--report-values](#) [359] option.

To run the utility as a daemon, use the [--daemon](#) [358] option. There are three commands that can be used in [--daemon](#) [358] option. These include:

- start

Starts the daemon. The [--log](#) [358] option is required.

- stop

Stops the daemon. If you used the option [--pidfile](#) [359], the value must be the same when starting the daemon.

- restart

Restarts the daemon. If you used the option [--pidfile](#) [359], the value must be the same when starting the daemon.

- nodetach

Starts the daemon, but it will not detach the process from the console. The [--log](#) [358] option is required.

The utility supports two modes of operation. The default mode, running as a console, works as described above. An additional mode that permits you to run the utility as a daemon is provided for POSIX platforms.

When run as a daemon, the utility does not have interactivity. However, all events are written to the log file. You can control what is written to the log by using the [--report-values](#) [359] option.

To run the utility as a daemon, use the [--daemon](#) [358] option. There are three commands that can be used in [--daemon](#) [358] option. These include:

- start

Starts the daemon. The [--log](#) [358] option is required.

- stop

Stops the daemon. The option [--pidfile](#) [359] must be the same when starting the daemon.

- restart

Restarts the daemon. The option [--pidfile](#) [359] must be the same when starting the daemon.

- nodetach

Starts the daemon, but it will not detach the process from the console. The [--log](#) [358] option is required.

OPTIONS

`mysqlfailover` accepts the following command-line options:

- [--help](#)

Display a help message and exit.

- **--candidates=<candidate slave connections>**

Connection information for candidate slave servers for failover in the form:

`<user>[:<passwd>]@<host>[:<port>][:<socket>]` or `<login-path>[:<port>][:<socket>]`. Valid only with failover command. List multiple slaves in comma-separated list.

- **--daemon=<command>**

Run as a daemon. It can be [start \[357\]](#) (start daemon), [stop \[357\]](#) (stop daemon), [restart \[357\]](#) (stop then start the daemon) or [nodelatch \[357\]](#) (start but do not detach the process). This option is only available for POSIX systems.

- **--discover-slaves-login=<user:password>**

At startup, query master for all registered slaves and use the user name and password specified to connect. Supply the user and password in the form `<user>[:<passwd>]` or `<login-path>`. For example, `--discover=joe:secret` will use 'joe' as the user and 'secret' as the password for each discovered slave.

- **--exec-after=<script>**

Name of script to execute after failover or switchover. Script name may include the path.

- **--exec-before=<script>**

Name of script to execute before failover or switchover. Script name may include the path.

- **--exec-fail-check=<script>**

Name of script to execute on each interval to invoke failover.

- **--exec-post-failover=<script>**

Name of script to execute after failover is complete and the utility has refreshed the health report.

- **--failover-mode=<mode>, -f <mode>**

Action to take when the master fails. 'auto' = automatically fail to best slave, 'elect' = fail to candidate list or if no candidate meets criteria fail, 'fail' = take no action and stop when master fails. Default = 'auto'.

- **--force**

Override the registration check on master for multiple instances of the console monitoring the same master. See notes.

- **--interval=<seconds>, -i <seconds>**

Interval in seconds for polling the master for failure and reporting health. Default = 15 seconds. Minimum is 5 seconds.

- **--log=<log_file>**

Specify a log file to use for logging messages

- **--log-age=<days>**

Specify maximum age of log entries in days. Entries older than this will be purged on startup. Default = 7 days.

- **--master=<connection>**

Connection information for the master server in the format: <*user*>[:<*passwd*>]@<*host*>[:<*port*>]
[:<*socket*>] or <*login-path*>[:<*port*>][:<*socket*>].

- **--max-position=<position>**

Used to detect slave delay. The maximum difference between the master's log position and the slave's reported read position of the master. A value greater than this means the slave is too far behind the master. Default = 0.

- **--pedantic, -p**

Used to stop failover if some inconsistencies are found (e.g. errant transactions on slaves or SQL thread errors) during servers checks. By default, the utility will only issue warnings if issues are found when checking slaves status during failover and will continue its execution unless this option is specified.

- **--pidfile=<pidfile>**

Pidfile for running mysqlfailover as a daemon. This file contains the PID (process identifier), that uniquely identify a process. It is needed to identify and control the process forked by mysqlfailover.

- **--ping=<number>**

Number of ping attempts for detecting downed server. Note: on some platforms this is the same as number of seconds to wait for ping to return. Default is 3 seconds.

- **--report-values=<report_values>**

Report values used in mysqlfailover running as a daemon. It can be health, gtid or uuid. Multiple values can be used separated by commas.

- **health**

Display the replication health of the topology.

- **gtid**

Display the master's list of executed GTIDs, contents of the GTID variables;
`@@GLOBAL.GTID_EXECUTED`, `@@GLOBAL.GTID_PURGED` and `@@GLOBAL.GTID_OWNED`.

- **uuid**

Display universally unique identifiers (UUIDs) for all servers.

Default = health.

- **--rpl-user=:<replication_user>**

The user and password for the replication user requirement , in the form: <*user*>[:<*password*>] or <*login-path*>. E.g. rpl:passwd

Default = None.

- **--script-threshold=<return_code>**

Value for external scripts to trigger aborting the operation if result is greater than or equal to the threshold.

Default = None (no threshold checking).

- `--seconds-behind=<seconds>`

Used to detect slave delay. The maximum number of seconds behind the master permitted before slave is considered behind the master. Default = 0.

- `--slaves=<slave connections>`

Connection information for slave servers in the form: `<user>[:<passwd>]@<host>[:<port>][:<socket>]` or `<login-path>[:<port>][:<socket>]`. List multiple slaves in comma-separated list. The list will be evaluated literally whereby each server is considered a slave to the master listed regardless if they are a slave of the master.

- `--timeout=<seconds>`

Maximum timeout in seconds to wait for each replication command to complete. For example, timeout for slave waiting to catch up to master.

Default = 3.

- `--verbose, -v`

Specify how much information to display. Use this option multiple times to increase the amount of information. For example, `-v` = verbose, `-vv` = more verbose, `-vvv` = debug.

- `--version`

Display version information and exit.

NOTES

The login user must have the appropriate permissions to execute **SHOW SLAVE STATUS**, **SHOW MASTER STATUS**, and **SHOW VARIABLES** on the appropriate servers as well as grant the REPLICATE SLAVE privilege. Different permission are required by the failover utility to run successfully for master and slaves. In particular, users connectioned to slaves and candidates require **SUPER**, **GRANT OPTION**, **REPLICATION SLAVE**, and **RELOAD** privileges.

In addition, the user connected to the master requires **DROP**, **CREATE**, **INSERT** and **SELECT** privileges to register the failover console. The utility checks permissions for the master, slaves, and candidates at startup.

At startup, the console will attempt to register itself with the master. If another console is already registered, and the failover mode is auto or elect, the console will be blocked from running failover. When a console quits, it deregisters itself from the master. If this process is broken, the user may override the registration check by using the `--force` [358] option.

Mixing IP and hostnames is not recommended. The replication-specific utilities will attempt to compare hostnames and IP addresses as aliases for checking slave connectivity to the master. However, if your installation does not support reverse name lookup, the comparison could fail. Without the ability to do a reverse name lookup, the replication utilities could report a false negative that the slave is (not) connected to the master.

For example, if you setup replication using `MASTER_HOST=ubuntu.net` on the slave and later connect to the slave with `mysqlrplcheck` and have the master specified as `--master=192.168.0.6` using the valid IP address for `ubuntu.net`, you must have the ability to do a reverse name lookup to compare the IP (`192.168.0.6`) and the hostname (`ubuntu.net`) to determine if they are the same machine.

Similarly, in order to avoid issues mixing local IP '127.0.0.1' with 'localhost', all the addresses '127.0.0.1' will be internally converted to 'localhost' by the utility. Nevertheless, It is best to use the actual hostname of the master when connecting or setting up replication.

The utility will check to see if the slaves are using the option `--master-info-repository=TABLE`. If they are not, the utility will stop with an error.

The path to the MySQL client tools should be included in the PATH environment variable in order to use the authentication mechanism with login-paths. This will allow the utility to use the `my_print_defaults` tools which is required to read the login-path values from the login configuration file (`.mylogin.cnf`).

The console creates a special table in the `mysql` database that is used to keep track of which instance is communicating with the master. If you use the `--force` [358] option, the console will remove the rows in this table. The table is constructed with:

```
CREATE TABLE IF NOT EXISTS mysql.failover_console (host char(30), port char(10))
```

When the console starts, a row is inserted containing the hostname and port of the master. On startup, if a row matches these values, the console will not start. If you use the `--force` [358] option, the row is deleted.

When running the utility using the `--daemon` [358] option with `nodetach` [357] the `--pidfile` [359] option can be omitted, will be ignored if used.

When running the utility using the `--daemon` [358] option with `nodetach` [357] the `--pidfile` [359] option can be omitted, will be ignored if used.

EXAMPLES

To launch the utility, you must specify at a minimum the `--master` [359] option and either the `--discover-slaves-login` [358] option or the `--slaves` [360] option. The option: option can be used in conjunction with the `--slaves` [360] option to specify a list of known slaves (or slaves that do not report their host and ip) and to discover any other slaves connected to the master.

An example of the user interface and some of the report views are shown in the following examples.



Note

The "GTID Executed Set" will display the first GTID listed in the `SHOW MASTER STATUS` view. If there are multiple GTIDs listed, the utility shall display [...] to indicate there are additional GTIDs to view. You can view the complete list of GTIDs on the GTID display screens.

The default interface will display the replication health report like the following. In this example the log file is enabled. A sample startup command is shown below:

```
$ mysqlfailover --master=root@localhost:3331 --discover-slaves-login=root --log=log.txt

MySQL Replication Monitor and Failover Utility
Failover Mode = auto      Next Interval = Mon Mar 19 15:56:03 2012

Master Information
-----
Binary Log File  Position  Binlog_Do_DB  Binlog_Ignore_DB
mysql-bin.000001  571

GTID Executed Set
2A67DE00-2DA1-11E2-A711-00764F2BE90F:1-7 [...]

Replication Health Status
```

```
+-----+-----+-----+-----+-----+
| host | port | role | state | gtid_mode | health |
+-----+-----+-----+-----+-----+
| localhost | 3331 | MASTER | UP | ON | OK |
| localhost | 3332 | SLAVE | UP | ON | OK |
| localhost | 3333 | SLAVE | UP | ON | OK |
| localhost | 3334 | SLAVE | UP | ON | OK |
+-----+-----+-----+-----+-----+
Q-quit R-refresh H-health G-GTID Lists U-UUIDs L-log entries
```

Pressing the 'q' key will exit the utility. Pressing the 'r' key will refresh the current display. Pressing the 'h' key will return to the replication health report.

If the user presses the 'g' key, the gtid report is shown like the following. The first page shown is the master's executed GTID set:

```
MySQL Replication Monitor and Failover Utility
Failover Mode = auto      Next Interval = Mon Mar 19 15:59:33 2012

Master Information
-----
Binary Log File  Position  Binlog_Do_DB  Binlog_Ignore_DB
mysql-bin.000001  571

GTID Executed Set
2A67DE00-2DA1-11E2-A711-00764F2BE90F:1-7 [ ... ]

Master GTID Executed Set
+-----+
| gtid           |
+-----+
| 2A67DE00-2DA1-11E2-A711-00764F2BE90F:1-7 |
| 5503D37E-2DB2-11E2-A781-8077D4C14B33:1-3 |
+-----+
Q-quit R-refresh H-health G-GTID Lists U-UUIDs L-log entries Up|Down-scroll
```

If the user continues to press the 'g' key, the display will cycle through the three gtid lists.

If the list is longer than the screen permits as shown in the example above, the scroll up and down help is also shown. In this case, if the user presses the down arrow, the list will scroll down.

If the user presses the 'u' key, the list of UUIDs used in the topology are shown.:

```
MySQL Replication Monitor and Failover Utility
Failover Mode = auto      Next Interval = Mon Mar 19 16:02:34 2012

Master Information
-----
Binary Log File  Position  Binlog_Do_DB  Binlog_Ignore_DB
mysql-bin.000001  571

GTID Executed Set
2A67DE00-2DA1-11E2-A711-00764F2BE90F:1-7 [ ... ]

UUIDs
+-----+-----+-----+-----+
| host | port | role | uuid          |
+-----+-----+-----+-----+
| localhost | 3331 | MASTER | 55c65a00-71fd-11e1-9f80-ac64ef85c961 |
| localhost | 3332 | SLAVE | 5dd30888-71fd-11e1-9f80-dc242138b7ec |
| localhost | 3333 | SLAVE | 65ccb38-71fd-11e1-9f80-bda8146bdb0a |
| localhost | 3334 | SLAVE | 6dd6abf4-71fd-11e1-9f80-d406a0117519 |
+-----+-----+-----+-----+
Q-quit R-refresh H-health G-GTID Lists U-UUIDs L-log entries
```

If, once the master is detected as down and failover mode is auto or elect and there are viable candidate slaves, the failover feature will engage automatically and the user will see the failover messages appear. When failover is complete, the interface returns to monitoring replication health after 5 seconds. The following shows an example of failover occurring.:

```

Failover starting...
# Candidate slave localhost:3332 will become the new master.
# Preparing candidate for failover.
# Creating replication user if it does not exist.
# Stopping slaves.
# Performing STOP on all slaves.
# Switching slaves to new master.
# Starting slaves.
# Performing START on all slaves.
# Checking slaves for errors.
# Failover complete.
# Discovering slaves for master at localhost:3332

Failover console will restart in 5 seconds.

```

After the failover event, the new topology is shown in the replication health report.:

```

MySQL Replication Monitor and Failover Utility
Failover Mode = auto      Next Interval = Mon Mar 19 16:05:12 2012

Master Information
-----
Binary Log File  Position  Binlog_Do_DB  Binlog_Ignore_DB
mysql-bin.000001  1117

GTID Executed Set
2A67DE00-2DA1-11E2-A711-00764F2BE90F:1-7 [ ... ]

UUIDs
+-----+-----+-----+-----+-----+
| host    | port   | role    | state   | gtid_mode | health |
+-----+-----+-----+-----+-----+
| localhost | 3332   | MASTER  | UP      | ON        | OK      |
| localhost | 3333   | SLAVE   | UP      | ON        | OK      |
| localhost | 3334   | SLAVE   | UP      | ON        | OK      |
+-----+-----+-----+-----+-----+
Q-quit R-refresh H-health G-GTID Lists U-UUIDs L-log entries

```

If the user presses the 'l' key and the `--log` [358] option was specified, the interface will show the entries in the log file. Note: example truncated for space allowance.:

```

MySQL Replication Monitor and Failover Utility
Failover Mode = auto      Next Interval = Mon Mar 19 16:06:13 2012

Master Information
-----
Binary Log File  Position  Binlog_Do_DB  Binlog_Ignore_DB
mysql-bin.000001  1117

GTID Executed Set
2A67DE00-2DA1-11E2-A711-00764F2BE90F:1-7 [ ... ]

Log File
+-----+-----+-----+-----+
| Date          | Entry           | ... | ...
+-----+-----+-----+-----+
| 2012-03-19 15:55:33 PM | INFO Failover console started. | ... | ...
| 2012-03-19 15:55:33 PM | INFO Failover mode = auto. | ... | ...
| 2012-03-19 15:55:33 PM | INFO Getting health for master: localhos ... | ...
| 2012-03-19 15:55:33 PM | INFO Master status: binlog: mysql-bin.00 ... | ...

```

+-----+ ... --+
Q-quit R-refresh H-health G-GTID Lists U-UUIDs L-log entries Up|Down-scroll\

15.6.10. mysqlfrm

15.6.10.1. `mysqlfrm` — File reader for .frm files.

The mysqlfrm utility is designed as a recovery tool that reads .frm files and produces facsimile *CREATE* statements from the table definition data found in the file. In most cases, the *CREATE* statement produced will be usable for recreating the table on another server or for extended diagnostics. However, some features are not saved in the .frm files and therefore will be omitted. The exclusions include but are not limited to:

- foreign key constraints
 - auto increment number sequences

The mysqlfrm utility has two modes of operation. The default mode is designed to spawn an instance of an installed server by reference to the base directory using the `--basedir` [364] option or by connecting to the server with the `--server` [365] option. The process will not alter the original .frm file(s). This mode also requires the `--port` [365] option to specify a port to use for the spawned server. It must be different than the port for the installed server and no other server must be using the port. The spawned server will be shutdown and all temporary files removed after the .frm files are read.

A diagnostic mode is available by using the `--diagnostic` [364] option. This will switch the utility to reading the .frm files byte-by-byte to recover as much information as possible. The diagnostic mode has additional limitations in that it cannot decipher character set or collation values without using an existing server installation specified with either the `--server` [365] or `--basedir` [364] option. This can also affect the size of the columns if the table uses multi-byte characters. Use this mode when the default mode cannot read the file or if there is no server installed on the host.

To read .frm files, list each file as a separate argument for the utility as shown in the following examples. You will need to specify the path for each .frm file you want to read or supply a path to a directory and all of the .frm files in that directory will be read.

You can specify the database name to be used in the resulting *CREATE* statement by prepending the .frm file with the name of the database followed by a colon. For example, oltp:t1.frm will use 'oltp' for the database name in the *CREATE* statement. The optional database name can also be used with paths. For example, /home/me/oltp:t1.frm will use 'oltp' as the database name. If you leave off the optional database name and include a path, the last folder will be the database name. For example /home/me/data1/t1.frm will use 'data1' as the database name. If you do not want to use the last folder as the database name, simply specify the colon like this: /home/me/data1:t1.frm. In this case, the database will be omitted from the *CREATE* statement.

OPTIONS

- **--help**
show the program's help page
 - **--basedir=<basedir>**
The base directory for the server installed. Use this or [--server \[365\]](#) for the default mode.
 - **--diagnostic**
Turn on diagnostic mode to read .frm files byte-by-byte and generate best-effort *CREATE* statement.

- `--new-storage-engine=<engine>`

Set the ENGINE= option for all .frm files read.

- `--port=<port>`

The port to use for the spawned server in the default mode. Must be a free port. Required for default mode.

- `--server=<server>`

onnection information for a server in the format: `<user>[:<passwd>]@<host>[:<port>][:<socket>]` or `<login-path>[:<port>][:<socket>]`. Use this option or [--basedir \[364\]](#) for the default mode. If provided with the diagnostic mode, the storage engine and character set information will be validated against this server.

- `--show-stats, -s`

Show file statistics and general table information for each .frm file read.

- `--start-timeout=<timeout_in_seconds>`

Number of seconds to wait for spawned server to start.

Default = 10 seconds.

- `--user`

Execute the spawned server using this user account. Permits the execution of the utility as one user but the spawned server as another. Required if running the utility as the root user (e.g. su or sudo).

- `--quiet`

Turn off all messages for quiet execution except *CREATE* statements and errors.

- `--verbose, -v`

control how much information is displayed. For example, `-v` = verbose, `-vv` = more verbose, `-vvv` = debug

- `--version`

show program's version number and exit

NOTES

Tables with certain storage engines cannot be read in the default mode. These include *PARTITION*, *PERFORMANCE_SCHEMA*. You must read these with the [--diagnostic \[364\]](#) mode.

Use the [--diagnostic \[364\]](#) mode for tables that fail to open correctly in the default mode or if there is no server installed on the host.

To change the storage engine in the *CREATE* statement generated for all .frm files read, use the [--new-storage-engine \[365\]](#) option

To turn off all messages except the *CREATE* statement and warnings or errors, use the [--quiet \[365\]](#) option.

Use the [--show-stats \[365\]](#) option to see file statistics for each .frm file.

If you need to run the utility with elevated privileges, use the [--user \[365\]](#) option to execute the spawned server using a normal user account.

If you encounter connection or similar errors when running in default mode, re-run the command with the [--verbose \[365\]](#) option and view the output from the spawned server and repair any errors in launching the server. If mysqlfrm fails in the middle, you may need to manually shutdown the server on the port specified with [--port \[365\]](#).

EXAMPLES

The following example will read a single .frm file in the default mode from the current working directory using the server installed in `/usr/local/bin/mysql` and port 3333 for the spawned server. Notice the use of the `db:table.frm` format for specifying the database name for the table. The database name appears to the left of ':' and the .frm name to the right. So in this case, we have database = test1 and table = db1 so the *CREATE* statement will read *CREATE test1.db1*.

```
$ mysqlfrm --basedir=/usr/local/bin/mysql test1:city.frm --port=3333
# Starting the spawned server on port 3333 ... done.
# Reading .frm files
#
# Reading the city.frm file.
#
# CREATE statement for city.frm:
#
CREATE TABLE `test1`.`city` (
  `city_id` smallint(5) unsigned NOT NULL AUTO_INCREMENT,
  `city` varchar(50) NOT NULL,
  `country_id` smallint(5) unsigned NOT NULL,
  `last_update` timestamp NOT NULL DEFAULT CURRENT_TIMESTAMP ON UPDATE CURRENT_TIMESTAMP,
  PRIMARY KEY (`city_id`),
  KEY `idx_fk_country_id` (`country_id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8
#
#...done.
```

The following demonstrates reading multiple .frm files in the default mode using a running server. The .frm files are located in different folders. Notice the use of the database name option for each of the files. The t1 file was given the database name temp1 since that is the folder in which it resides, t2 was given db1 since that was specified in the path, and t3 was not given a database name since we used the ':' without providing a database name.

```
$ mysqlfrm --server=root:pass@localhost:3306 /mysql/data/temp1/t1.frm \
          /mysql/data/temp2/db1:t2.frm --port=3310
# Starting the spawned server on port 3333 ... done.
# Reading .frm files
#
#
# Reading the t1.frm file.
#
# CREATE statement for ./mysql-test/std_data/frm_files/t1.frm:
#
CREATE TABLE `temp1`.`t1` (
  `a` int(11) DEFAULT NULL
) ENGINE=MyISAM DEFAULT CHARSET=latin1
#
# Reading the t2.frm file.
#
# CREATE statement for ./mysql-test/std_data/frm_files/t2.frm:
#
CREATE TABLE `db1`.`t2` (
```

```
`a` int(11) DEFAULT NULL  
) ENGINE=MyISAM DEFAULT CHARSET=latin1  
  
##  
# Reading the t3.frm file.  
##  
# CREATE statement for ./mysql-test/std_data/frm_files/t3.frm:  
##  
  
CREATE TABLE `t3` (  
  `a` int(11) DEFAULT NULL  
) ENGINE=MyISAM DEFAULT CHARSET=latin1  
  
#...done.
```

The following demonstrates running the utility in diagnostic mode to read all of the .frm files in a directory.

```
$ mysqlfrm --diagnostic /mysql/data/sakila  
# WARNING: Cannot generate character set or collation names without the --server option.  
# CAUTION: The diagnostic mode is a best-effort parse of the .frm file. As such, it may not identify all o  
# Reading .frm file for /mysql/data/sakila/city.frm:  
# The .frm file is a TABLE.  
# CREATE TABLE Statement:  
  
CREATE TABLE `city` (  
  `city_id` smallint(5) unsigned NOT NULL AUTO_INCREMENT,  
  `city` varchar(150) NOT NULL,  
  `country_id` smallint(5) unsigned NOT NULL,  
  `last_update` timestamp NOT NULL DEFAULT CURRENT_TIMESTAMP ON UPDATE CURRENT_TIMESTAMP,  
PRIMARY KEY `PRIMARY` (`city_id`),  
KEY `idx_fk_country_id` (`country_id`)  
) ENGINE=InnoDB;  
  
#...done.
```

15.6.11. mysqlindexcheck

15.6.11.1. [mysqlindexcheck](#) — Identify Potentially Redundant Table Indexes

This utility reads the indexes for one or more tables and identifies duplicate and potentially redundant indexes.

To check all tables in a database, specify only the database name. To check a specific table, name the table in *db.table* format. It is possible to mix database and table names.

You can scan tables in any database except the internal databases **mysql**, **INFORMATION_SCHEMA**, and **performance_schema**.

Depending on the index type, the utility applies the following rules to compare indexes (designated as *idx_a* and *idx_b*):

- **BTREE**

idx_b is redundant to *idx_a* if and only if the first *n* columns in *idx_b* also appear in *idx_a*. Order and uniqueness count.

- **HASH**

idx_a and *idx_b* are duplicates if and only if they contain the same columns in the same order. Uniqueness counts.

- **SPATIAL**

`idx_a` and `idx_b` are duplicates if and only if they contain the same column (only one column is permitted).

- **FULLTEXT**

`idx_b` is redundant to `idx_a` if and only if all columns in `idx_b` are included in `idx_a`. Order counts.

To see **DROP** statements to drop redundant indexes, specify the `--show-drops` [369] option. To examine the existing indexes, use the `--verbose` [369] option, which prints the equivalent **CREATE INDEX** or **ALTER TABLE** for primary keys.

To display the best or worst nonprimary key indexes for each table, use the `--best` [368] or `--worst` [369] option. This causes the output to show the best or worst indexes from tables with 10 or more rows. By default, each option shows five indexes. To override that, provide an integer value for the option.

To change the format of the index lists displayed for the `--show-indexes` [369], `--best` [368], and `--worst` [369] options, use one of the following values with the `--format` [368] option:

- **grid** (default)

Display output in grid or table format like that of the `mysql` monitor.

- **csv**

Display output in comma-separated values format.

- **tab**

Display output in tab-separated format.

- **sql**

print SQL statements rather than a list.

- **vertical**

Display output in single-column format like that of the `\G` command for the `mysql` monitor.

Note: The `--best` [368] and `--worst` [369] lists cannot be printed as SQL statements.

OPTIONS

`mysqlindexcheck` accepts the following command-line options:

- **--help**

Display a help message and exit.

- **--best[=<N>]**

If `--stats` [369] is given, limit index statistics to the best *N* indexes. The default value of *N* is 5 if omitted.

- **--format=<index_format>, -f<index_format>**

Specify the index list display format for output produced by `--stats` [369]. Permitted format values are **grid**, **csv**, **tab**, **sql**, and **vertical**. The default is **grid**.

- `--server=<source>`

Connection information for the server in the format: `<user>[:<passwd>]@<host>[:<port>][:<socket>]` or `<login-path>[:<port>][:<socket>]`.

- `--show-drops, -d`

Display **DROP** statements for dropping indexes.

- `--show-indexes, -i`

Display indexes for each table.

- `--skip, -s`

Skip tables that do not exist.

- `--stats`

Show index performance statistics.

- `--verbose, -v`

Specify how much information to display. Use this option multiple times to increase the amount of information. For example, `-v` = verbose, `-vv` = more verbose, `-vvv` = debug.

- `--version`

Display version information and exit.

- `--worst[=<N>]`

If `--stats` [369] is given, limit index statistics to the worst *N* indexes. The default value of *N* is 5 if omitted.

NOTES

You must provide connection parameters (user, host, password, and so forth) for an account that has the appropriate privileges to read all objects accessed during the operation.

For the `--format` [368] option, the permitted values are not case sensitive. In addition, values may be specified as any unambiguous prefix of a valid value. For example, `--format=g` [368] specifies the grid format. An error occurs if a prefix matches more than one valid value.

The path to the MySQL client tools should be included in the PATH environment variable in order to use the authentication mechanism with login-paths. This will allow the utility to use the my_print_defaults tools which is required to read the login-path values from the login configuration file (.mylogin.cnf).

EXAMPLES

To check all tables in the `employees` database on the local server to see the possible redundant and duplicate indexes, use this command:

```
$ mysqlindexcheck --server=root@localhost employees
# Source on localhost: ... connected.
# The following indexes are duplicates or redundant \
  for table employees.dept_emp:
#
CREATE INDEX emp_no ON employees.dept_emp (emp_no) USING BTREE
#      may be redundant or duplicate of:
```

```
ALTER TABLE employees.dept_emp ADD PRIMARY KEY (emp_no, dept_no)
# The following indexes are duplicates or redundant \
  for table employees.dept_manager:
#
CREATE INDEX emp_no ON employees.dept_manager (emp_no) USING BTREE
#      may be redundant or duplicate of:
ALTER TABLE employees.dept_manager ADD PRIMARY KEY (emp_no, dept_no)
# The following indexes are duplicates or redundant \
  for table employees.salaries:
#
CREATE INDEX emp_no ON employees.salaries (emp_no) USING BTREE
#      may be redundant or duplicate of:
ALTER TABLE employees.salaries ADD PRIMARY KEY (emp_no, from_date)
# The following indexes are duplicates or redundant \
  for table employees.titles:
#
CREATE INDEX emp_no ON employees.titles (emp_no) USING BTREE
#      may be redundant or duplicate of:
ALTER TABLE employees.titles ADD PRIMARY KEY (emp_no, title, from_date)
```

15.6.12. mysqlmetagrep

15.6.12.1. mysqlmetagrep — Search Database Object Definitions

This utility searches for objects matching a given pattern on all the servers specified using instances of the [--server](#) [373] option. It produces output that displays the matching objects. By default, the first nonoption argument is taken to be the pattern unless the [--pattern](#) [372] option is given. If the [--pattern](#) [372] option is given, all nonoption arguments are treated as connection specifications.

Internally, the utility generates an SQL statement for searching the necessary tables in the **INFORMATION_SCHEMA** database on the designated servers and executes it in turn before collecting the result and printing it as a table. Use the [--sql](#) [373] option to have the utility display the statement rather than execute it. This can be useful if you want to feed the output of the statement to another application such as the [mysql](#) monitor.

The MySQL server supports two forms of patterns when matching strings: SQL Simple Patterns (used with the **LIKE** operator) and POSIX Regular Expressions (used with the **REGEXP** operator).

By default, the utility uses the **LIKE** operator to match the name (and optionally, the body) of objects. To use the **REGEXP** operator instead, use the [--regexp](#) [372] option.

Note that since the **REGEXP** operator does substring searching, it is necessary to anchor the expression to the beginning of the string if you want to match the beginning of the string.

To specify how to display output, use one of the following values with the [--format](#) [372] option:

- **grid** (default)

Display output in grid or table format like that of the [mysql](#) monitor.

- **csv**

Display output in comma-separated values format.

- **tab**

Display output in tab-separated format.

- **vertical**

Display output in single-column format like that of the [\G](#) command for the [mysql](#) monitor.

SQL Simple Patterns

The simple patterns defined by the SQL standard consist of a string of characters with two characters that have special meaning: `%` (percent) matches zero or more characters and `_` (underscore) matches exactly one character.

For example:

- `'mats%`

Match any string that starts with 'mats'.

- `'%kindahl%`

Match any string containing the word 'kindahl'.

- `'%_'`

Match any string consisting of one or more characters.

POSIX Regular Expressions

POSIX regular expressions are more powerful than the simple patterns defined in the SQL standard. A regular expression is a string of characters, optionally containing characters with special meaning:

- `.`

Match any character.

- `^`

Match the beginning of a string.

- `$`

Match the end of a string.

- `[axy]`

Match `a`, `x`, or `y`.

- `[a-f]`

Match any character in the range `a` to `f` (that is, `a`, `b`, `c`, `d`, `e`, or `f`).

- `[^axy]`

Match any character *except* `a`, `x`, or `y`.

- `a*`

Match a sequence of zero or more `a`.

- `a+`

Match a sequence of one or more `a`.

- `a?`

Match zero or one `a`.

- **ab|cd**

Match **ab** or **cd**.

- **a{5}**

Match five instances of **a**.

- **a{2,5}**

Match from two to five instances of **a**.

- **(abc)+**

Match one or more repetitions of **abc**.

This is but a brief set of examples of regular expressions. The full syntax is described in the [MySQL manual](#), but can often be found in *regex(7)*.

OPTIONS

`mysqlmetagrep` accepts the following command-line options:

- **--help**

Display a help message and exit.

- **--body, -b**

Search the body of stored programs (procedures, functions, triggers, and events). The default is to match only the name.

- **--database=<pattern>**

Look only in databases matching this pattern.

- **--format=<format>, -f<format>**

Specify the output display format. Permitted format values are **grid**, **csv**, **tab**, and **vertical**. The default is **grid**.

- **--object-types=<types>, --search-objects=<types>**

Search only the object types named in *types*, which is a comma-separated list of one or more of the values **procedure**, **function**, **event**, **trigger**, **table**, and **database**.

The default is to search in objects of all types.

- **--pattern=<pattern>, -e=<pattern>**

The pattern to use when matching. This is required when the first nonoption argument looks like a connection specification rather than a pattern.

If the **--pattern** [372] option is given, the first nonoption argument is treated as a connection specifier, not as a pattern.

- **--regexp, --basic-regexp, -G**

Perform pattern matches using the **REGEXP** operator. The default is to use **LIKE** for matching. This affects the **--database** [372] and **--pattern** [372] options.

- **--server=<source>**

Connection information for a server to search in the format: `<user>[:<passwd>]@<host>[:<port>][:<socket>]` or `<login-path>[:<port>][:<socket>]`. Use this option multiple times to search multiple servers.

- **--sql, --print-sql, -p**

Print rather than executing the SQL code that would be executed to find all matching objects. This can be useful to save the statement for later execution or to use it as input for other programs.

- **--version**

Display version information and exit.

NOTES

For the [--format](#) [372] option, the permitted values are not case sensitive. In addition, values may be specified as any unambiguous prefix of a valid value. For example, [--format=g](#) [372] specifies the grid format. An error occurs if a prefix matches more than one valid value.

The path to the MySQL client tools should be included in the PATH environment variable in order to use the authentication mechanism with login-paths. This will allow the utility to use the my_print_defaults tools which is required to read the login-path values from the login configuration file (.mylogin.cnf).

EXAMPLES

Find all objects with a name that matches the pattern '`t_`' (the letter t followed by any single character):

```
$ mysqlmetagrep --pattern="t_ " --server=mats@localhost
+-----+-----+-----+-----+
| Connection | Object Type | Object Name | Database |
+-----+-----+-----+-----+
| mats:@localhost:3306 | TABLE | t1 | test |
| mats:@localhost:3306 | TABLE | t2 | test |
| mats:@localhost:3306 | TABLE | t3 | test |
+-----+-----+-----+-----+
```

To find all object that contain '`t2`' in the name or the body (for routines, triggers, and events):

```
$ mysqlmetagrep -b --pattern="%t2%" --server=mats@localhost:3306
+-----+-----+-----+-----+
| Connection | Object Type | Object Name | Database |
+-----+-----+-----+-----+
| root:@localhost:3306 | TRIGGER | tr_foo | test |
| root:@localhost:3306 | TABLE | t2 | test |
+-----+-----+-----+-----+
```

In the preceding output, the trigger name does not match the pattern, but is displayed because its body does.

This is the same as the previous example, but using the **REGEXP** operator. Note that in the pattern it is not necessary to add wildcards before or after t2:

```
$ mysqlmetagrep -Gb --pattern="t2" --server=mats@localhost
+-----+-----+-----+-----+
| Connection | Object Type | Object Name | Database |
+-----+-----+-----+-----+
| root:@localhost:3306 | TRIGGER | tr_foo | test |
| root:@localhost:3306 | TABLE | t2 | test |
+-----+-----+-----+-----+
```

15.6.13. mysqlprocgrep

15.6.13.1. mysqlprocgrep — Search Server Process Lists

This utility scans the process lists for the servers specified using instances of the `--server` [375] option and selects those that match the conditions specified using the `--age` [374] and `--match-xxx` options. For a process to match, all conditions given must match. The utility then either prints the selected processes (the default) or executes certain actions on them.

If no `--age` [374] or `--match-xxx` options are given, the utility selects all processes.

The `--match-xxx` options correspond to the columns in the **INFORMATION_SCHEMA.PROCESSLIST** table. For example, `--match-command` [375] specifies a matching condition for **PROCESSLIST.COMMAND** column values. There is no `--match-time` option. To specify a condition based on process time, use `--age` [374].

Processes that can be seen and killed are subject to whether the account used to connect to the server has the **PROCESS** and **SUPER** privileges. Without **PROCESS**, the account cannot see processes belonging to other accounts. Without **SUPER**, the account cannot kill processes belonging to other accounts.

To specify how to display output, use one of the following values with the `--format` [375] option:

- **grid** (default)

Display output in grid or table format like that of the `mysql` monitor.

- **csv**

Display output in comma-separated values format.

- **tab**

Display output in tab-separated format.

- **vertical**

Display output in single-column format like that of the `\G` command for the `mysql` monitor.

Options

`mysqlprocgrep` accepts the following command-line options:

- `--help`

Display a help message and exit.

- `--age=<time>`

Select only processes that have been in the current state more than a given time. The time value can be specified in two formats: either using the `hh:mm:ss` format, with hours and minutes optional, or as a sequence of numbers with a suffix giving the period size.

The permitted suffixes are **s** (second), **m** (minute), **h** (hour), **d** (day), and **w** (week). For example, **4h15m** mean 4 hours and 15 minutes.

For both formats, the specification can optionally be preceded by `+` or `-`, where `+` means older than the given time, and `-` means younger than the given time.

- **--format=<format>, -f<format>**
Specify the output display format. Permitted format values are **grid**, **csv**, **tab**, and **vertical**. The default is **grid**.
- **--kill-connection**
Kill the connection for all matching processes (like the **KILL CONNECTION** statement).
- **--kill-query**
Kill the query for all matching processes (like the **KILL QUERY** statement).
- **--match-command=<pattern>**
Match all processes where the **Command** field matches the pattern.
- **--match-db=<pattern>**
Match all processes where the **Db** field matches the pattern.
- **--match-host=<pattern>**
Match all processes where the **Host** field matches the pattern.
- **--match-info=<pattern>**
Match all processes where the **Info** field matches the pattern.
- **--match-state=<pattern>**
Match all processes where the **State** field matches the pattern.
- **--match-user=<pattern>**
Match all processes where the **User** field matches the pattern.
- **--print**
Print information about the matching processes. This is the default if no [--kill-connection \[375\]](#) or [--kill-query \[375\]](#) option is given. If a kill option is given, [--print \[375\]](#) prints information about the processes before killing them.
- **--regexp, --basic-regexp, -G**
Perform pattern matches using the **REGEXP** operator. The default is to use **LIKE** for matching. This affects the [--match-xxx](#) options.
- **--server=<source>**
Connection information for a server to search in the format: **<user>[:<passwd>]@<host>[:<port>]** **[:<socket>]** or **<login-path>[:<port>][:<socket>]**. Use this option multiple times to search multiple servers.
- **--sql, --print-sql, -Q**
Instead of displaying the selected processes, emit the **SELECT** statement that retrieves information about them. If the [--kill-connection \[375\]](#) or [--kill-query \[375\]](#) option is given, the utility generates a stored procedure named **kill_processes()** for killing the queries rather than a **SELECT** statement.

- **--sql-body**

Like [--sql](#) [375], but produces the output as the body of a stored procedure without the **CREATE PROCEDURE** part of the definition. This could be used, for example, to generate an event for the server Event Manager.

When used with a kill option, code for killing the matching queries is generated. Note that it is not possible to execute the emitted code unless it is put in a stored routine, event, or trigger. For example, the following code could be generated to kill all idle connections for user [www-data](#):

```
$ mysqlprocgrep --kill-connection --sql-body \
>   --match-user=www-data --match-state=sleep
DECLARE kill_done INT;
DECLARE kill_cursor CURSOR FOR
    SELECT
        Id, User, Host, Db, Command, Time, State, Info
    FROM
        INFORMATION_SCHEMA.PROCESSLIST
    WHERE
        user LIKE 'www-data'
        AND
        State LIKE 'sleep'
OPEN kill_cursor;
BEGIN
    DECLARE id BIGINT;
    DECLARE EXIT HANDLER FOR NOT FOUND SET kill_done = 1;
    kill_loop: LOOP
        FETCH kill_cursor INTO id;
        KILL CONNECTION id;
    END LOOP kill_loop;
END;
CLOSE kill_cursor;
```

- **--verbose, -v**

Specify how much information to display. Use this option multiple times to increase the amount of information. For example, [-v](#) = verbose, [-vv](#) = more verbose, [-vvv](#) = debug.

- **--version**

Display version information and exit.

NOTES

For the [--format](#) [375] option, the permitted values are not case sensitive. In addition, values may be specified as any unambiguous prefix of a valid value. For example, [--format=g](#) [375] specifies the grid format. An error occurs if a prefix matches more than one valid value.

The path to the MySQL client tools should be included in the PATH environment variable in order to use the authentication mechanism with login-paths. This will allow the utility to use the my_print_defaults tools which is required to read the login-path values from the login configuration file (.mylogin.cnf).

EXAMPLES

For each example, assume that the [root](#) user on [localhost](#) has sufficient privileges to kill queries and connections.

Kill all queries created by user [mats](#) that are younger than 1 minute:

```
mysqlprocgrep --server=root@localhost \
--match-user=mats --age=-1m --kill-query
```

Kill all connections that have been idle for more than 1 hour:

```
mysqlprocgrep --server=root@localhost \
--match-command=sleep --age=1h --kill-connection
```

15.6.14. mysqlreplicate

15.6.14.1. [mysqlreplicate](#) — Set Up and Start Replication Between Two Servers

This utility permits an administrator to start replication from one server (the master) to another (the slave). The user provides login information for the slave and connection information for connecting to the master. It is also possible to specify a database to be used to test replication.

The utility reports conditions where the storage engines on the master and the slave differ. It also reports a warning if the InnoDB storage engine differs on the master and slave. For InnoDB to be the same, both servers must be running the same “type” of InnoDB (built-in or the InnoDB Plugin), and InnoDB on both servers must have the same major and minor version numbers and enabled state.

By default, the utility issues warnings for mismatches between the sets of storage engines, the default storage engine, and the InnoDB storage engine. To produce errors instead, use the [--pedantic](#) [378] option, which requires storage engines to be the same on the master and slave.

The [-vv](#) option displays any discrepancies between the storage engines and InnoDB values, with or without the [--pedantic](#) [378] option.

Replication can be started using one of the following strategies.

- Start from the current position (default)

Start replication from the current master binary log file and position. The utility uses the **SHOW MASTER STATUS** statement to retrieve this information.

- Start from the beginning

Start replication from the first event recorded in the master binary log. To do this, use the [--start-from-beginning](#) [378] option.

- Start from a binary log file

Start replication from the first event in a specific master binary log file. To do this, use the [--master-log-file](#) [378] option.

- Start from a specific event

Start replication from specific event coordinates (specific binary log file and position). To do this, use the [--master-log-file](#) [378] and [--master-log-pos](#) [378] options.

OPTIONS

`mysqlreplicate` accepts the following command-line options:

- `--help`

Display a help message and exit.

- `--master=<master>`

Connection information for the master server in the format: `<user>[:<passwd>]@<host>[:<port>][:<socket>]` or `<login-path>[:<port>][:<socket>]`.

- `--master-log-file=<master_log_file>`

Begin replication from the beginning of this master log file.

- `--master-log-pos=<master_log_pos>`

Begin replication from this position in the master log file. This option is not valid unless `--master-log-file` [378] is given.

- `--pedantic, -p`

Fail if both servers do not have the same set of storage engines, the same default storage engine, and the same InnoDB storage engine.

- `--rpl-user=<replication_user>`

The user and password for the replication user, in the format: `<user>[:<password>]` or `<login-path>`. The default is `rpl:rpl`.

- `--slave=<slave>`

Connection information for the slave server in the format: `<user>[:<passwd>]@<host>[:<port>]` `[:<socket>]` or `<login-path>[:<port>][:<socket>]`.

- `--start-from-beginning, -b`

Start replication at the beginning of events logged in the master binary log. This option is not valid unless both `--master-log-file` [378] and `--master-log-pos` [378] are given.

- `--test-db=<test_database>`

The database name to use for testing the replication setup. If this option is not given, no testing is done, only error checking.

- `--verbose, -v`

Specify how much information to display. Use this option multiple times to increase the amount of information. For example, `-v` = verbose, `-vv` = more verbose, `-vvv` = debug.

- `--version`

Display version information and exit.

NOTES

The login user for the master server must have the appropriate permissions to grant access to all databases and the ability to create a user account. For example, the user account used to connect to the master must have the **WITH GRANT OPTION** privilege.

The server IDs on the master and slave must be nonzero and unique. The utility reports an error if the server ID is 0 on either server or the same on the master and slave. Set these values before starting this utility.

Mixing IP and hostnames is not recommended. The replication-specific utilities will attempt to compare hostnames and IP addresses as aliases for checking slave connectivity to the master. However, if your installation does not support reverse name lookup, the comparison could fail. Without the ability to do a reverse name lookup, the replication utilities could report a false negative that the slave is (not) connected to the master.

For example, if you setup replication using `MASTER_HOST=ubuntu.net` on the slave and later connect to the slave with `mysqlrlpcheck` and have the master specified as `--master=192.168.0.6` using the valid IP address for `ubuntu.net`, you must have the ability to do a reverse name lookup to compare the IP (`192.168.0.6`) and the hostname (`ubuntu.net`) to determine if they are the same machine.

The path to the MySQL client tools should be included in the `PATH` environment variable in order to use the authentication mechanism with `login-paths`. This will allow the utility to use the `my_print_defaults` tools which is required to read the `login-path` values from the `login configuration file (.mylogin.cnf)`.

EXAMPLES

To set up replication between two MySQL instances running on different ports of the same host using the default settings, use this command:

```
$ mysqlreplicate --master=root@localhost:3306 \
  --slave=root@localhost:3307 --rpl-user=rpl:rpl
# master on localhost: ... connected.
# slave on localhost: ... connected.
# Checking for binary logging on master...
# Setting up replication...
# ...done.
```

The following command uses [--pedantic \[378\]](#) to ensure that replication between the master and slave is successful if and only if both servers have the same storage engines available, the same default storage engine, and the same InnoDB storage engine:

```
$ mysqlreplicate --master=root@localhost:3306 \
  --slave=root@localhost:3307 --rpl-user=rpl:rpl -vv --pedantic
# master on localhost: ... connected.
# slave on localhost: ... connected.
# master id = 2
# slave id = 99
# Checking InnoDB statistics for type and version conflicts.
# Checking storage engines...
# Checking for binary logging on master...
# Setting up replication...
# Flushing tables on master with read lock...
# Connecting slave to master...
# CHANGE MASTER TO MASTER_HOST = [...omitted...]
# Starting slave...
# status: Waiting for master to send event
# error: 0:
# Unlocking tables on master...
# ...done.
```

The following command starts replication from the current position of the master (which is the default):

```
$ mysqlreplicate --master=root@localhost:3306 \
  --slave=root@localhost:3307 --rpl-user=rpl:rpl
# master on localhost: ... connected.
# slave on localhost: ... connected.
# Checking for binary logging on master...
# Setting up replication...
# ...done.
```

The following command starts replication from the beginning of recorded events on the master:

```
$ mysqlreplicate --master=root@localhost:3306 \
  --slave=root@localhost:3307 --rpl-user=rpl:rpl \
  --start-from-beginning
# master on localhost: ... connected.
# slave on localhost: ... connected.
# Checking for binary logging on master...
# Setting up replication...
```

```
# ...done.
```

The following command starts replication from the beginning of a specific master binary log file:

```
$ mysqlreplicate --master=root@localhost:3306 \
    --slave=root@localhost:3307 --rpl-user=rpl:rpl \
    --master-log-file=my_log.000003
# master on localhost: ... connected.
# slave on localhost: ... connected.
# Checking for binary logging on master...
# Setting up replication...
# ...done.
```

The following command starts replication from specific master binary log coordinates (specific log file and position):

```
$ mysqlreplicate --master=root@localhost:3306 \
    --slave=root@localhost:3307 --rpl-user=rpl:rpl \
    --master-log-file=my_log.000001 --master-log-pos=96
# master on localhost: ... connected.
# slave on localhost: ... connected.
# Checking for binary logging on master...
# Setting up replication...
# ...done.
```

RECOMMENDATIONS

You should set `read_only=1` in the `my.cnf` file for the slave to ensure that no accidental data changes, such as **INSERT**, **DELETE**, **UPDATE**, and so forth, are permitted on the slave other than those produced by events read from the master.

Use the `--pedantic` [378] and `-vv` options for setting up replication on production servers to avoid possible problems with differing storage engines.

15.6.15. mysqlrpladmin

15.6.15.1. `mysqlrpladmin` — Administration utility for MySQL replication

This utility permits users to perform administrative actions on a replication topology consisting of a master and its slaves. The utility is designed to make it easy to recover from planned maintenance of the master or from an event that takes the master offline unexpectedly.

The act of taking the master offline intentionally and switching control to another slave is called switchover. In this case, there is no loss of transactions as the master is locked and all slaves are allowed to catch up to the master. Once the slaves have read all events from the master, the master is shutdown and control switched to a slave (in this case called a candidate slave).

Recovering from the loss of a downed master is more traumatic and since there is no way to know what transactions the master may have failed to send, the new master (called a candidate slave) must be the slave that is most up-to-date. How this is determined depends on the version of the server (see below). However, it can result in the loss of some transactions that were executed on the downed master but not sent. The utility accepts a list of slaves to be considered the candidate slave. If no slave is found to meet the requirements, the operation will search the list of known slaves.

The utility also provides a number of useful commands for managing a replication topology including the following.

elect This command is available to only those servers supporting global transaction identifiers (GTIDs), perform best slave election and report best slave to use in the event a switchover or failover is required. Best slave election is simply the first slave to meet the prerequisites. GTIDs are supported in version 5.6.5

and higher. This command requires the options [--master \[384\]](#) and either [--slaves \[384\]](#) or [--discover-slaves-login \[383\]](#).

failover This command is available to only those servers supporting GTIDs. Conduct failover to the best slave. The command will test each candidate slave listed for the prerequisites. Once a candidate slave is elected, it is made a slave of each of the other slaves thereby collecting any transactions executed on other slaves but not the candidate. In this way, the candidate becomes the most up-to-date slave. This command requires the [--slaves \[384\]](#) option. The [--discover-slaves-login \[383\]](#) option is not allowed because, for failover, the master is presumed to be offline or otherwise unreachable (so there is no way to discover the slaves). The [--master \[384\]](#) option is ignored for this command.

gtid This command is available to only those servers supporting GTIDs. It displays the contents of the GTID variables, @@GLOBAL.GTID_EXECUTED, @@GLOBAL.GTID_PURGED, and @@GLOBAL.GTID_OWNED. The command also displays universally unique identifiers (UUIDs) for all servers. This command requires one of the following combinations: [--master \[384\]](#) and [--slaves \[384\]](#), or [--master \[384\]](#) and [--discover-slaves-login \[383\]](#).

health Display the replication health of the topology. By default, this includes the host name, port, role (MASTER or SLAVE) of the server, state of the server (UP = is connected, WARN = not connected but can ping, DOWN = not connected and cannot ping), the GTID_MODE, and health state. This command requires one of the following combinations:

- [--master \[384\]](#) and [--slaves \[384\]](#);
- [--master \[384\]](#) and [--discover-slaves-login \[383\]](#);

The master health state is based on the following; if GTID_MODE=ON, the server must have binary log enabled, and there must exist a user with the REPLICATE SLAVE privilege.

The slave health state is based on the following; the IO_THREAD and SQL_THREADS must be running, it must be connected to the master, there are no errors, the slave delay for non-gtid enabled scenarios is not more than the threshold provided by the [--max-position \[384\]](#) and the slave is reading the correct master log file, and slave delay is not more than the [--seconds-behind \[384\]](#) threshold option.

reset Execute the STOP SLAVE and RESET SLAVE commands on all slaves. This command requires the [--slaves \[384\]](#) option. The [--discover-slaves-login \[383\]](#) option is not allowed because it might not provide the expected result, excluding slaves with the IO thread stopped. Optionally, the [--master \[384\]](#) option can also be used and in this case the utility will perform an additional check to verify if the specified slaves are associated (i.e. replication configured) to the given master.

start Execute the START SLAVE command on all slaves. This command requires the [--slaves \[384\]](#) option. The [--discover-slaves-login \[383\]](#) option is not allowed because it might not provide the expected result, excluding slaves with the IO thread stopped. Optionally, the [--master \[384\]](#) option can also be used and in this case the utility will perform an additional check to verify if the specified slaves are associated (i.e. replication configured) to the given master.

stop Execute the STOP SLAVE command on all slaves. This command requires the [--slaves \[384\]](#) option. The [--discover-slaves-login \[383\]](#) option is not allowed because it might not provide the expected result, excluding slaves with the IO thread stopped. Optionally, the [--master \[384\]](#) option can also be used and in this case the utility will perform an additional check to verify if the specified slaves are associated (i.e. replication configured) to the given master.

switchover Perform slave promotion to a specified candidate slave as designated by the [--new-master \[384\]](#) option. This command is available for both gtid-enabled servers and non-gtid-enabled scenarios. This command requires one of the following combinations:

- [--master \[384\]](#), [--new-master \[384\]](#) and [--slaves \[384\]](#);

- [--master](#) [384], [--new-master](#) [384] and [--discover-slaves-login](#) [383];

Detection of a downed master is performed as follows. If the connection to the master is lost, wait [--ping](#) [384] seconds and check again. If the master connection is lost and the master cannot be pinged or reconnected, the failover event occurs.

For all commands that require specifying multiple servers, the options require a comma-separated list of connection parameters in the following form (where the password, port, and socket are optional).:

```
<*user*>[:<*passwd*>]@<*host*>[:<*port*>][:<*socket*>] or  
<*login-path*>[:<*port*>][:<*socket*>]
```

The utility permits users to discover slaves connected to the master. In order to use the discover slaves feature, all slaves must use the [--report-host](#) and [--report-port](#) startup variables to specify the correct hostname and ip port of the slave. If these are missing or report the incorrect information, the slaves health may not be reported correctly or the slave may not be listed at all. The discover slaves feature ignores any slaves it cannot connect to or with the IO thread stopped (i.e. not connected to the master).

The utility permits the user to demote a master to a slave during the switchover operation. The [--demote-master](#) [383] option tells the utility to, once the new master is established, make the old master a slave of the new master. This permits rotation of the master role among a set of servers.

The utility permits the user to specify an external script to execute before and after the switchover and failover commands. The user can specify these with the [--exec-before](#) [383] and [--exec-after](#) [383] options. The return code of the script is used to determine success thus each script must report 0 (success) to be considered successful. If a script returns a value other than 0, the result code is presented in an error message.

The utility permits the user to log all actions taken during the commands. The [--log](#) [383] option requires a valid path and file name of the file to use for logging operations. The log is active only when this option is specified. The option [--log-age](#) [383] specifies the age in days that log entries are kept. The default is seven (7) days. Older entries are automatically deleted from the log file (but only if the [--log](#) [383] option is specified).

The format of the log file includes the date and time of the event, the level of the event (informational - INFO, warning - WARN, error - ERROR, critical failure - CRITICAL), and the message reported by the utility.

The utility has a number of options each explained in more detail below. Some of the options are specific to certain commands. Warning messages are issued whenever an option is used that does not apply to the command requested. A brief overview of each command and its options is presented in the following paragraphs.

The start, stop, and reset commands require the [--slaves](#) [384] option to list all of the slaves in the topology. Optionally, the [--master](#) [384] option can be specified for the utility to check if the specified slaves are associated to the given master before executing the command, making sure that the command is only applied to slaves connected to the right replication master.

The options required for the elect, health and gtid commands include the [--master](#) [384] option to specify the existing master, and either the [--slaves](#) [384] option to list all of the slaves in the topology or the [--discover-slaves-login](#) [383] option to provide the user name and password to discover any slaves in the topology that are registered and connected to the master.

The options required for switchover include the [--master](#) [384] option to specify the existing master, the [--new-master](#) [384] option to specify the candidate slave (the slave to become the new master), and either the [--slaves](#) [384] option to list the considered slaves in the topology or the [--discover-](#)

[slaves-login](#) [383] option to provide the user name and password to discover any slaves in the topology that are registered and connected to the master.

The failover command requires only the [--slaves](#) [384] option to explicitly list all of the slaves in the topology because it is expected that the master is down when this command is used.

Use the [--verbose](#) [385] option to see additional information in the health report and additional messages during switchover or failover.

OPTIONS

[mysqlrpladmin](#) accepts the following command-line options:

- [--help](#)

Display a help message and exit.

- [--candidates=<candidate slave connections>](#)

Connection information for candidate slave servers for failover in the form:

<user>[:<passwd>]@<host>[:<port>][:<socket>] or <login-path>[:<port>][:<socket>]. Valid only with failover command. List multiple slaves in comma-separated list.

- [--demote-master](#)

Make master a slave after switchover.

- [--discover-slaves-login=<slave_login>](#)

At startup, query master for all registered slaves and use the user name and password specified to connect. Supply the user and password in the form <user>[:<passwd>] or <login-path>. For example, --discover=joe:secret will use 'joe' as the user and 'secret' as the password for each discovered slave.

- [--exec-after=<script>](#)

Name of script to execute after failover or switchover. Script name may include the path.

- [--exec-before=<script>](#)

Name of script to execute before failover or switchover. Script name may include the path.

- [--force](#)

Ignore prerequisite checks or any inconsistencies found (e.g. errant transactions on the slaves or SQL thread errors) forcing the execution of the specified command. This option need to be used carefully as it will not solve any detected issue, but will only ignore them displaying a warning message.

- [--format=<format>, -f <format>](#)

Display the replication health output in either grid (default), tab, csv, or vertical format.

- [--log=<log_file>](#)

Specify a log file to use for logging messages

- [--log-age=<days>](#)

Specify maximum age of log entries in days. Entries older than this will be purged on startup. Default = 7 days.

- **--master=<connection>**

Connection information for the master server in the format: `<user>[:<passwd>]@<host>[:<port>][:<socket>]` or `<login-path>[:<port>][:<socket>]`.

- **--max-position=<position>**

Used to detect slave delay. The maximum difference between the master's log position and the slave's reported read position of the master. A value greater than this means the slave is too far behind the master. Default = 0.

- **--new-master=<connection>**

Connection information for the slave to be used to replace the master for switchover in the form: `<user>[:<passwd>]@<host>[:<port>][:<socket>]` or `<login-path>[:<port>][:<socket>]`. Valid only with switchover command.

- **--no-health**

Turn off health report after switchover or failover.

- **--ping=<number>**

Number of ping attempts for detecting downed server. Note: on some platforms this is the same as number of seconds to wait for `ping` to return. This value is also used to check down status of master. Failover will wait `ping` seconds to check master response. If no response, failover event occurs.

- **--quiet, -q**

Turn off all messages for quiet execution.

- **--rpl-user=<replication_user>**

The user and password for the replication user requirement, in the format: `<user>[:<password>]` or `<login-path>`. E.g. `rpl:passwd` Default = None.

- **--script-threshold=<return_code>**

Value for external scripts to trigger aborting the operation if result is greater than or equal to the threshold.

Default = None (no threshold checking).

- **--seconds-behind=<seconds>**

Used to detect slave delay. The maximum number of seconds behind the master permitted before slave is considered behind the master. Default = 0.

- **--slaves=<slave connections>**

Connection information for slave servers in the form: `<user>[:<passwd>]@<host>[:<port>][:<socket>]` or `<login-path>[:<port>][:<socket>]`. List multiple slaves in comma-separated list. The list will be evaluated literally whereby each server is considered a slave to the master listed regardless if they are a slave of the master.

- **--timeout=<seconds>**

Maximum timeout in seconds to wait for each replication command to complete. For example, timeout for slave waiting to catch up to master. Default = 300 seconds.

- **--verbose**, **-v**

Specify how much information to display. Use this option multiple times to increase the amount of information. For example, **-v** = verbose, **-vv** = more verbose, **-vvv** = debug.

- **--version**

Display version information and exit.

NOTES

The login user must have the appropriate permissions to execute **SHOW SLAVE STATUS**, **SHOW MASTER STATUS**, and **SHOW VARIABLES** on the appropriate servers as well as grant the REPLICATE SLAVE privilege. The utility checks permissions for the master, slaves, and candidates at startup.

Mixing IP and hostnames is not recommended. The replication-specific utilities will attempt to compare hostnames and IP addresses as aliases for checking slave connectivity to the master. However, if your installation does not support reverse name lookup, the comparison could fail. Without the ability to do a reverse name lookup, the replication utilities could report a false negative that the slave is (not) connected to the master.

For example, if you setup replication using **MASTER_HOST=ubuntu.net** on the slave and later connect to the slave with **mysqlrlcheck** and have the master specified as **--master=192.168.0.6** using the valid IP address for **ubuntu.net**, you must have the ability to do a reverse name lookup to compare the IP (192.168.0.6) and the hostname (**ubuntu.net**) to determine if they are the same machine.

Similarly, if you use **localhost** to connect to the master, the health report may not show all of the slaves. It is best to use the actual hostname of the master when connecting or setting up replication.

If the user does not specify the **--rpl-user** [384] and the user has specified the switchover or failover command, the utility will check to see if the slaves are using **--master-info-repository=TABLE**. If they are not, the utility will stop with an error.

All the commands require either the **--slaves** [384] or **--discover-slaves-login** [383] option but both cannot be used at the same time. In fact, some commands only allow the use of the **--slaves** [384] option which is safer to specify the list slaves, because **--discover-slaves-login** [383] might not provide an up to date list of available slaves.

The path to the MySQL client tools should be included in the PATH environment variable in order to use the authentication mechanism with login-paths. This will allow the utility to use the **my_print_defaults** tools which is required to read the login-path values from the login configuration file (**.mylogin.cnf**).

EXAMPLES

To perform best slave election for a topology with **GTID_MODE=ON** (server version 5.6.5 or higher) where all slaves are specified with the **--slaves** [384] option, run the following command.:

```
$ mysqlrpladmin --master=root@localhost:3331 \
  --slaves=root@localhost:3332,root@localhost:3333,root@localhost:3334 elect
# Electing candidate slave from known slaves.
# Best slave found is located on localhost:3332.
# ...done.
```

To perform best slave election supplying a candidate list, use the following command.:

```
$ mysqlrpladmin --master=root@localhost:3331 \
  --slaves=root@localhost:3332,root@localhost:3333,root@localhost:3334 \
  --candidates=root@localhost:3333,root@localhost:3334 elect
```

```
# Electing candidate slave from candidate list then slaves list.
# Best slave found is located on localhost:3332.
# ...done.
```

To perform failover after a master has failed, use the following command.:

```
$ mysqlrpladmin \
--slaves=root@localhost:3332,root@localhost:3333,root@localhost:3334 \
--candidates=root@localhost:3333,root@localhost:3334 failover
# Performing failover.
# Candidate slave localhost:3333 will become the new master.
# Preparing candidate for failover.
# Creating replication user if it does not exist.
# Stopping slaves.
# Performing STOP on all slaves.
# Switching slaves to new master.
# Starting slaves.
# Performing START on all slaves.
# Checking slaves for errors.
# Failover complete.
# ...done.
```

To see the replication health of a topology with GTID_MODE=ON (server version 5.6.5 or higher) and discover all slaves attached to the master, run the following command. We use the result of the failover command above.:

```
$ mysqlrpladmin --master=root@localhost:3333 \
--slaves=root@localhost:3332,root@localhost:3334 health
# Getting health for master: localhost:3333.
#
# Replication Topology Health:
+-----+-----+-----+-----+-----+
| host | port | role | state | gtid_mode | health |
+-----+-----+-----+-----+-----+
| localhost | 3333 | MASTER | UP | ON | OK |
| localhost | 3332 | SLAVE | UP | ON | OK |
| localhost | 3334 | SLAVE | UP | ON | OK |
+-----+-----+-----+-----+-----+
# ...done.
```

To view a detailed replication health report but with all of the replication health checks revealed, use the [--verbose \[385\]](#) option as shown below. In this example, we use vertical format to make viewing easier.:

```
$ mysqlrpladmin --master=root@localhost:3331 \
--slaves=root@localhost:3332,root@localhost:3333,root@localhost:3334 \
--verbose health
# Getting health for master: localhost:3331.
# Attempting to contact localhost ... Success
#
# Replication Topology Health:
***** 1. row *****
host: localhost
port: 3331
role: MASTER
state: UP
gtid_mode: ON
health: OK
version: 5.6.5-m8-debug-log
master_log_file: mysql-bin.000001
master_log_pos: 571
IO_Thread:
SQL_Thread:
Secs_Behind:
```

```

Remaining_Delay:
  IO_Error_Num:
    IO_Error:
*****
          host: localhost
          port: 3332
          role: SLAVE
          state: UP
        gtid_mode: ON
          health: OK
          version: 5.6.5-m8-debug-log
master_log_file: mysql-bin.000001
master_log_pos: 571
  IO_Thread: Yes
  SQL_Thread: Yes
  Secs_Behind: 0
Remaining_Delay: No
  IO_Error_Num: 0
    IO_Error:
*****
          host: localhost
          port: 3333
          role: SLAVE
          state: UP
        gtid_mode: ON
          health: OK
          version: 5.6.5-m8-debug-log
master_log_file: mysql-bin.000001
master_log_pos: 571
  IO_Thread: Yes
  SQL_Thread: Yes
  Secs_Behind: 0
Remaining_Delay: No
  IO_Error_Num: 0
    IO_Error:
*****
          host: localhost
          port: 3334
          role: SLAVE
          state: UP
        gtid_mode: ON
          health: OK
          version: 5.6.5-m8-debug-log
master_log_file: mysql-bin.000001
master_log_pos: 571
  IO_Thread: Yes
  SQL_Thread: Yes
  Secs_Behind: 0
Remaining_Delay: No
  IO_Error_Num: 0
    IO_Error:
4 rows.
# ...done.

```

To run the same failover command above, but specify a log file, use the following command.:

```

$ mysqlrpladmin \
--slaves=root@localhost:3332,root@localhost:3333,root@localhost:3334 \
--candidates=root@localhost:3333,root@localhost:3334 \
--log=test_log.txt failover
# Performing failover.
# Candidate slave localhost:3333 will become the new master.
# Preparing candidate for failover.
# Creating replication user if it does not exist.
# Stopping slaves.
# Performing STOP on all slaves.
# Switching slaves to new master.

```

```
# Starting slaves.
# Performing START on all slaves.
# Checking slaves for errors.
# Failover complete.
# ...done.
```

After this command, the log file will contain entries like the following:

```
2012-03-19 14:44:17 PM INFO Executing failover command...
2012-03-19 14:44:17 PM INFO Performing failover.
2012-03-19 14:44:17 PM INFO Candidate slave localhost:3333 will become the new master.
2012-03-19 14:44:17 PM INFO Preparing candidate for failover.
2012-03-19 14:44:19 PM INFO Creating replication user if it does not exist.
2012-03-19 14:44:19 PM INFO Stopping slaves.
2012-03-19 14:44:19 PM INFO Performing STOP on all slaves.
2012-03-19 14:44:19 PM INFO Switching slaves to new master.
2012-03-19 14:44:20 PM INFO Starting slaves.
2012-03-19 14:44:20 PM INFO Performing START on all slaves.
2012-03-19 14:44:20 PM INFO Checking slaves for errors.
2012-03-19 14:44:21 PM INFO Failover complete.
2012-03-19 14:44:21 PM INFO ...done.
```

To perform switchover and demote the current master to a slave, use the following command.:

```
$ mysqlrpladmin --master=root@localhost:3331 \
--slaves=root@localhost:3332,root@localhost:3333,root@localhost:3334 \
--new-master=root@localhost:3332 --demote-master switchover
# Performing switchover from master at localhost:3331 to slave at localhost:3332.
# Checking candidate slave prerequisites.
# Waiting for slaves to catch up to old master.
# Stopping slaves.
# Performing STOP on all slaves.
# Demoting old master to be a slave to the new master.
# Switching slaves to new master.
# Starting all slaves.
# Performing START on all slaves.
# Checking slaves for errors.
# Switchover complete.
# ...done.
```

If the replication health report is generated on the topology following the above command, it will display the old master as a slave as shown below.:

# Replication Topology Health:					
host	port	role	state	gtid_mode	health
localhost	3332	MASTER	UP	ON	OK
localhost	3331	SLAVE	UP	ON	OK
localhost	3333	SLAVE	UP	ON	OK
localhost	3334	SLAVE	UP	ON	OK

You can use the discover slaves feature, if and only if all slaves report their host and port to the master. A sample command to generate a replication health report with discovery is shown below. Note that the option [--discover-slaves-login \[383\]](#) cannot be used in conjunction with the [--slaves \[384\]](#) option.:

```
$ mysqlrpladmin --master=root@localhost:3332 --discover-slaves-login=root  health
# Discovering slaves for master at localhost:3332
# Discovering slave at localhost:3331
# Found slave: localhost:3331
# Discovering slave at localhost:3333
# Found slave: localhost:3333
```

```
# Discovering slave at localhost:3334
# Found slave: localhost:3334
# Checking privileges.
#
# Replication Topology Health:
+-----+-----+-----+-----+-----+-----+
| host | port | role | state | gtid_mode | health |
+-----+-----+-----+-----+-----+-----+
| localhost | 3332 | MASTER | UP | ON | OK |
| localhost | 3331 | SLAVE | UP | ON | OK |
| localhost | 3333 | SLAVE | UP | ON | OK |
| localhost | 3334 | SLAVE | UP | ON | OK |
+-----+-----+-----+-----+-----+-----+
# ...done.
```

15.6.16. mysqlrplcheck

15.6.16.1. `mysqlrplcheck` — Check Replication Prerequisites

This utility checks the prerequisites for replication between a master and a slave. These checks (called tests) are designed to ensure a healthy replication setup. The utility performs the following tests:

1. Is the binary log enabled on the master?
2. Are there binary logging exceptions (such as `*_do_db` or `*_ignore_db` settings)? If so, display them.
3. Does the replication user exist on the master with the correct privileges?
4. Are there `server_id` conflicts?
5. Is the slave connected to this master? If not, display the master host and port.
6. Are there conflicts between the `master.info` file on the slave and the values shown in **SHOW SLAVE STATUS** on the master?
7. Are the InnoDB configurations compatible (plugin vs. native)?
8. Are the storage engines compatible (have same on slave as master)?
9. Are the `lower_case_table_names` settings compatible? Warn if there are settings for lowercase/uppercase table names that can cause problems. See Bug #59240.
10. Is the slave behind the master?

The utility runs each test in turn unless there is a fatal error preventing further testing, such as a loss of connection to the servers.

Each test can complete with one of the following states: pass (the prerequisites are met), fail (the prerequisites were met but one or more errors occurred or there are exceptions to consider), or warn (the test found some unusual settings that should be examined further but may not be in error).

Use the `--verbose` [390] option to see additional information such as server IDs, `lower_case_table_name` settings, and the contents of the master information file on the slave.

To see the values from the **SHOW SLAVE STATUS** statement, use the `--show-slave-status` [390] option.

OPTIONS

`mysqlrplcheck` accepts the following command-line options:

- **--help**

Display a help message and exit.

- **--master=<source>**

Connection information for the master server in the format: `<user>[:<passwd>]@<host>[:<port>][:<socket>]` or `<login-path>[:<port>][:<socket>]`.

- **--master-info-file=<file>**

The name of the master information file on the slave. The default is `master.info` read from the data directory. Note: This option requires that you run the utility on the slave and that you have appropriate read access for the file.

- **--quiet, -q**

Turn off all messages for quiet execution. Note: Errors and warnings are not suppressed.

- **--show-slave-status, -s**

Display the values from **SHOW SLAVE STATUS** on the master.

- **--slave=<source>**

Connection information for the slave server in the format: `<user>[:<passwd>]@<host>[:<port>][:<socket>]` or `<login-path>[:<port>][:<socket>]`.

- **--suppress**

Suppress warning messages.

- **--verbose, -v**

Specify how much information to display. Use this option multiple times to increase the amount of information. For example, `-v` = verbose, `-vv` = more verbose, `-vvv` = debug.

- **--version**

Display version information and exit.

- **--width=<number>**

Change the display width of the test report. The default is 75 characters.

NOTES

The login user must have the appropriate permissions to execute **SHOW SLAVE STATUS**, **SHOW MASTER STATUS**, and **SHOW VARIABLES** on the appropriate servers.

Mixing IP and hostnames is not recommended. The replication-specific utilities will attempt to compare hostnames and IP addresses as aliases for checking slave connectivity to the master. However, if your installation does not support reverse name lookup, the comparison could fail. Without the ability to do a reverse name lookup, the replication utilities could report a false negative that the slave is (not) connected to the master.

For example, if you setup replication using `MASTER_HOST=ubuntu.net` on the slave and later connect to the slave with `mysqlrplcheck` and have the master specified as `--master=192.168.0.6` using the valid

IP address for ubuntu.net, you must have the ability to do a reverse name lookup to compare the IP (192.168.0.6) and the hostname (ubuntu.net) to determine if they are the same machine.

The path to the MySQL client tools should be included in the PATH environment variable in order to use the authentication mechanism with login-paths. This will allow the utility to use the my_print_defaults tools which is required to read the login-path values from the login configuration file (.mylogin.cnf).

EXAMPLES

To check the prerequisites of a master and slave that currently are actively performing replication, use the following command:

```
$ mysqlrplcheck --master=root@host1:3310 --slave=root@host2:3311
# master on host1: ... connected.
# slave on host2: ... connected.

Test Description          Status
-----
Checking for binary logging on master      [pass]
Are there binlog exceptions?            [pass]
Replication user exists?                [pass]
Checking server_id values              [pass]
Is slave connected to master?           [pass]
Check master information file          [pass]
Checking InnoDB compatibility          [pass]
Checking storage engines compatibility  [pass]
Checking lower_case_table_names settings [pass]
Checking slave delay (seconds behind master) [pass]
# ...done.
```

As shown in the example, you must provide valid login information for both the master and the slave.

To perform the same command but also display the contents of the master information file on the slave and the values of **SHOW SLAVE STATUS** as well as additional details, use this command:

```
$ mysqlrplcheck --master=root@host1:3310 --slave=root@host2:3311 \
--show-slave-status -vv
# master on host1: ... connected.
# slave on host2: ... connected.

Test Description          Status
-----
Checking for binary logging on master      [pass]
Are there binlog exceptions?            [pass]
Replication user exists?                [pass]
Checking server_id values              [pass]

master id = 10
slave id = 11

Is slave connected to master?           [pass]
Check master information file          [pass]

#
# Master information file:
#
      Master_Log_File : clone-bin.000001
      Read_Master_Log_Pos : 482
      Master_Host : host1
      Master_User : rpl
      Master_Password : XXXX
      Master_Port : 3310
      Connect_Retry : 60
      Master_SSL_Allowed : 0
      Master_SSL_CA_File :
      Master_SSL_CA_Path :
      Master_SSL_Cert :
```

```

        Master_SSL_Cipher :
        Master_SSL_Key :
Master_SSL_Verify_Server_Cert : 0

Checking InnoDB compatibility                                [pass]
Checking storage engines compatibility                      [pass]
Checking lower_case_table_names settings                   [pass]

    Master lower_case_table_names: 2
    Slave lower_case_table_names: 2

Checking slave delay (seconds behind master)                [pass]

#
# Slave status:
#
        Slave_IO_State : Waiting for master to send event
        Master_Host : host1
        Master_User : rpl
        Master_Port : 3310
        Connect_Retry : 60
        Master_Log_File : clone-bin.000001
        Read_Master_Log_Pos : 482
        Relay_Log_File : clone-relay-bin.000006
        Relay_Log_Pos : 251
        Relay_Master_Log_File : clone-bin.000001
        Slave_IO_Running : Yes
        Slave_SQL_Running : Yes
        Replicate_Do_DB :
        Replicate_Ignore_DB :
        Replicate_Do_Table :
        Replicate_Ignore_Table :
        Replicate_Wild_Do_Table :
        Replicate_Wild_Ignore_Table :
            Last_Error :
            Skip_Counter : 0
        Exec_Master_Log_Pos : 482
        Relay_Log_Space : 551
        Until_Condition : None
        Until_Log_File :
        Until_Log_Pos : 0
        Master_SSL_Allowed : No
        Master_SSL_CA_File :
        Master_SSL_CA_Path :
            Master_SSL_Cert :
            Master_SSL_Cipher :
            Master_SSL_Key :
        Seconds_Behind_Master : 0
Master_SSL_Verify_Server_Cert : No
            Last_IO_Error :
            Last_SQL_Error :
Last_SQL_Error : 0
Last_SQL_Error :

# ...done.

```

15.6.17. mysqlrplshow

15.6.17.1. mysqlrplshow — Show Slaves for Master Server

This utility shows the replication slaves for a master. It prints a graph of the master and its slaves labeling each with the host name and port number.

You must specify the `--discover-slaves-login` [394] option to provide the user name and password to discover any slaves in the topology.

To explore the slaves for each client, use the `--recurse` [394] option. This causes the utility to connect to each slave found and attempt to determine whether it has any slaves. If slaves are found, the process continues until the slave is found in the list of servers serving as masters (a circular topology). The graph displays the topology with successive indents. A notation is made for circular topologies.

If you use the `--recurse` [394] option, the utility attempts to connect to the slaves using the user name and password provided for the master. By default, if the connection attempt fails, the utility throws an error and stops. To change this behavior, use the `--prompt` [394] option, which permits the utility to prompt for the user name and password for each slave that fails to connect. You can also use the `--num-retries=n` [394] option to reattempt a failed connection 'n' times before the utility fails.

An example graph for a typical topology with relay slaves is shown here:

```
# Replication Topology Graph::

localhost:3311 (MASTER)
 |
 +--- localhost:3310 - (SLAVE)
 |
 +--- localhost:3312 - (SLAVE + MASTER)
 |
 +--- localhost:3313 - (SLAVE)
```

`MASTER`, `SLAVE`, and `SLAVE+MASTER` indicate that a server is a master only, slave only, and both slave and master, respectively.

A circular replication topology is shown like this, where `<-->` indicates circularity:

```
# Replication Topology Graph

localhost:3311 (MASTER)
 |
 +--- localhost:3312 - (SLAVE + MASTER)
 |
 +--- localhost:3313 - (SLAVE + MASTER)
 |
 +--- localhost:3311 <--> (SLAVE)
```

To produce a column list in addition to the graph, specify the `--show-list` [394] option. In this case, to specify how to display the list, use one of the following values with the `--format` [394] option:

- **grid** (default)

Display output in grid or table format like that of the `mysql` monitor.

- **csv**

Display output in comma-separated values format.

- **tab**

Display output in tab-separated format.

- **vertical**

Display output in single-column format like that of the `\G` command for the `mysql` monitor.

The utility uses of the **SHOW SLAVE HOSTS** statement to determine which slaves the master has. If you want to use the `--recurse` [394] option, slaves should have been started with the `--report-host` and `--report-port` options set to their actual host name and port number or the utility may not be able to connect to the slaves to determine their own slaves.

OPTIONS

`mysqlrplshow` accepts the following command-line options:

- `--help`

Display a help message and exit.
- `--discover-slaves-login=<slave-login>`

Supply the user and password in the form `<user>[:<passwd>]` or `<login-path>` for discovering slaves and relay slaves in the topology. For example, `--discover=joe:secret` will use 'joe' as the user and 'secret' as the password for each discovered slave.
- `--format=<format>, -f<format>`

Specify the display format for column list output. Permitted format values are **grid**, **csv**, **tab**, and **vertical**. The default is **grid**. This option applies only if [--show-list \[394\]](#) is given.
- `--master=<source>`

Connection information for the master server in the format: `<user>[:<passwd>]@<host>[:<port>][:<socket>]` or `<login-path>[:<port>][:<socket>]`.
- `--max-depth=<N>`

The maximum recursion depth. This option is valid only if [--recurse \[394\]](#) is given.
- `--num-retries=<num_retries>, -n<num_retries>`

The number of retries permitted for failed slave login attempts. This option is valid only if [--prompt \[394\]](#) is given.
- `--prompt, -p`

Prompt for the slave user and password if different from the master user and password.

If you give this option, the utility sets [--num-retries \[394\]](#) to 1 if that option is not set explicitly. This ensures at least one attempt to retry and prompt for the user name and password should a connection fail.
- `--quiet, -q`

Turn off all messages for quiet execution. This option does not suppress errors or warnings.
- `--recurse, -r`

Traverse the list of slaves to find additional master/slave connections. User this option to map a replication topology.
- `--show-list, -l`

Display a column list of the topology.
- `--verbose, -v`

Specify how much information to display. If this option is used, the IO thread status of each slave is also displayed. Use this option multiple times to increase the amount of information. For example, `-v =`

verbose, `--vv` = more verbose, `--vvv` = debug. If you use `-vvv`, the output will contain the state of the IO and SQL threads for each slave.

- `--version`

Display version information and exit.

NOTES

The login user must have the **REPLICATE SLAVE** and **REPLICATE CLIENT** privileges to successfully execute this utility. Specifically, the login user must have appropriate permissions to execute **SHOW SLAVE STATUS**, **SHOW MASTER STATUS**, and **SHOW SLAVE HOSTS**.

For the `--format` [394] option, the permitted values are not case sensitive. In addition, values may be specified as any unambiguous prefix of a valid value. For example, `--format=g` [394] specifies the grid format. An error occurs if a prefix matches more than one valid value.

Mixing IP and hostnames is not recommended. The replication-specific utilities will attempt to compare hostnames and IP addresses as aliases for checking slave connectivity to the master. However, if your installation does not support reverse name lookup, the comparison could fail. Without the ability to do a reverse name lookup, the replication utilities could report a false negative that the slave is (not) connected to the master.

For example, if you setup replication using `MASTER_HOST=ubuntu.net` on the slave and later connect to the slave with `mysqlrplcheck` and have the master specified as `--master=192.168.0.6` using the valid IP address for `ubuntu.net`, you must have the ability to do a reverse name lookup to compare the IP (`192.168.0.6`) and the hostname (`ubuntu.net`) to determine if they are the same machine.

The path to the MySQL client tools should be included in the PATH environment variable in order to use the authentication mechanism with login-paths. This will allow the utility to use the `my_print_defaults` tools which is required to read the login-path values from the login configuration file (`.mylogin.cnf`).

EXAMPLES

To show the slaves for a master running on port 3311 on the local host, use the following command:

```
$ mysqlrplshow --master=root@localhost:3311 --discover-slaves-login=root
# master on localhost: ... connected.
# Finding slaves for master: localhost:3311

# Replication Topology Graph
localhost:3311 (MASTER)
|
+--- localhost:3310 - (SLAVE)
|
+--- localhost:3312 - (SLAVE)
```

As shown in the example, you must provide valid login information for the master.

To show additional information about the IO thread status (to confirm if the slaves are really connected to the master) use the option `--verbose` [394]:

```
$ mysqlrplshow --master=root@localhost:3311 --discover-slaves-login=root --verbose
# master on localhost: ... connected.
# Finding slaves for master: localhost:3311

# Replication Topology Graph
localhost:3311 (MASTER)
|
+--- localhost:3310 [IO: Yes, SQL: Yes] - (SLAVE)
```

```
|  
+--- localhost:3312 [IO: Yes, SQL: Yes] - (SLAVE)
```

To show the full replication topology of a master running on the local host, use the following command:

```
$ mysqlrlplshow --master=root@localhost:3311 --recurse --discover-slaves-login=root  
# master on localhost: ... connected.  
# Finding slaves for master: localhost:3311  
  
# Replication Topology Graph  
localhost:3311 (MASTER)  
|  
+--- localhost:3310 - (SLAVE)  
|  
+--- localhost:3312 - (SLAVE + MASTER)  
|  
+--- localhost:3313 - (SLAVE)
```

To show the full replication topology of a master running on the local host, prompting for the user name and password for slaves that do not have the same user name and password credentials as the master, use the following command:

```
$ mysqlrlplshow --recurse --prompt --num-retries=1 \  
--master=root@localhost:3331 --discover-slaves-login=root  
  
Server localhost:3331 is running on localhost.  
# master on localhost: ... connected.  
# Finding slaves for master: localhost:3331  
Server localhost:3332 is running on localhost.  
# master on localhost: ... FAILED.  
Connection to localhost:3332 has failed.  
Please enter the following information to connect to this server.  
User name: root  
Password:  
# master on localhost: ... connected.  
# Finding slaves for master: localhost:3332  
Server localhost:3333 is running on localhost.  
# master on localhost: ... FAILED.  
Connection to localhost:3333 has failed.  
Please enter the following information to connect to this server.  
User name: root  
Password:  
# master on localhost: ... connected.  
# Finding slaves for master: localhost:3333  
Server localhost:3334 is running on localhost.  
# master on localhost: ... FAILED.  
Connection to localhost:3334 has failed.  
Please enter the following information to connect to this server.  
User name: root  
Password:  
# master on localhost: ... connected.  
# Finding slaves for master: localhost:3334  
  
# Replication Topology Graph  
localhost:3331 (MASTER)  
|  
+--- localhost:3332 - (SLAVE)  
|  
+--- localhost:3333 - (SLAVE + MASTER)  
|  
+--- localhost:3334 - (SLAVE)
```

15.6.18. mysqlserverclone

15.6.18.1. [mysqlserverclone](#) — Clone Existing Server to Create New Server

This utility permits an administrator to clone an existing MySQL server instance to start a new server instance on the same host. The utility creates a new datadir ([--new-data \[397\]](#)), and starts the server with a socket file. You can optionally add a password for the login user account on the new instance.

If the user does not have read and write access to the folder specified by the [--new-data \[397\]](#) option, the utility shall issue an error.

Similarly, if the folder specified by [--new-data \[397\]](#) exists and is not empty, the utility will not delete the folder and will issue an error message. Users must specify the [--delete-data \[397\]](#) option to permit the utility to remove the folder prior to starting the cloned server.

OPTIONS

`mysqlserverclone` accepts the following command-line options:

- `--help`

Display a help message and exit.

- `--delete-data`

Delete the folder specified by [--new-data](#) if it exists and is not empty.

- `--mysqld=<options>`

Additional options for `mysqld`. To specify multiple options, separate them by spaces. Use appropriate quoting as necessary. For example, to specify `--log-bin=binlog` and `--general-log-file="mylogfile"`, use:

```
--mysqld="--log-bin=binlog --general-log-file='my log file'"
```

- `--new-data=<path_to_new_datadir>`

The full path name of the location of the data directory for the new server instance. If the directory does not exist, the utility will create it.

- `--new-id=<server_id>`

The `server_id` value for the new server instance. The default is 2.

- `--new-port=<port>`

The port number for the new server instance. The default is 3307.

- `--quiet, -q`

Turn off all messages for quiet execution.

- `--root-password=<password>`

The password for the `root` user of the new server instance.

- `--server=<source>`

Connection information for the server to be cloned in `<user>[:<passwd>]@<host>[:<port>][:<socket>]` format.

- `--start-timeout=<timeout_in_seconds>`

Number of seconds to wait for server to start. Default = 10 seconds.

- **--verbose, -v**

Specify how much information to display. Use this option multiple times to increase the amount of information. For example, **-v** = verbose, **-vv** = more verbose, **-vvv** = debug.

- **--version**

Display version information and exit.

- **--write-command=<file_name>, -w<file_name>**

Path name of file in which to write the command used to launch the new server instance.

EXAMPLES

The following command demonstrates how to create a new instance of a running server, set the `root` user password and enable binary logging:

```
$ mkdir /source/test123
$ mysqlserverclone --server=root:pass@localhost \
    --new-data=/Users/cbell/source/test123 --new-port=3310 \
    --root-password=pass --mysqld=--log-bin=mysql-bin
# Cloning the MySQL server running on localhost.
# Creating new data directory...
# Configuring new instance...
# Locating mysql tools...
# Setting up empty database and mysql tables...
# Starting new instance of the server...
# Testing connection to new instance...
# Success!
# Setting the root password...
# ...done.
```

15.6.19. mysqlserverinfo

15.6.19.1. [mysqlserverinfo](#) — Display Common Diagnostic Information from a Server

This utility displays critical information about a server for use in diagnosing problems. The information displayed includes the following:

- Server connection information
- Server version number
- Data directory path name
- Base directory path name
- Plugin directory path name
- Configuration file location and name
- Current binary log coordinates (file name and position)
- Current relay log coordinates (file name and position)

This utility can be used to see the diagnostic information for servers that are running or offline. If you want to see information about an offline server, the utility starts the server in read-only mode. In this case, you must specify the [--basedir \[399\]](#), [--datadir \[399\]](#), and [--start \[400\]](#) options to prevent the utility from starting an offline server accidentally. Note: Be sure to consider the ramifications of starting an offline server on the error and similar logs. It is best to save this information prior to running this utility.

To specify how to display output, use one of the following values with the [--format](#) [399] option:

- **grid** (default)

Display output in grid or table format like that of the [mysql](#) monitor.

- **csv**

Display output in comma-separated values format.

- **tab**

Display output in tab-separated format.

- **vertical**

Display output in single-column format like that of the [\G](#) command for the [mysql](#) monitor.

To turn off the headers for **csv** or **tab** display format, specify the [--no-headers](#) [399] option.

To see the common default settings for the local server's configuration file, use the [--show-defaults](#) [400] option. This option reads the configuration file on the machine where the utility is run, not the machine for the host that the [--server](#) [400] option specifies.

To run the utility against several servers, specify the [--server](#) [400] option multiple times. In this case, the utility attempts to connect to each server and read the information.

To see the MySQL servers running on the local machine, use the [--show-servers](#) [400] option. This shows all the servers with their process ID and data directory. On Windows, the utility shows only the process ID and port.

OPTIONS

[mysqlserverinfo](#) accepts the following command-line options:

- **--help**

Display a help message and exit.

- **--basedir=<basedir>**

The base directory for the server. This option is required for starting an offline server.

Is also used to access server tools, such as `my_print_defaults` that is required to read the login-path values from the login configuration file (`.mylogin.cnf`).

- **--datadir=<datadir>**

The data directory for the server. This option is required for starting an offline server.

- **--format=<format>, -f<format>**

Specify the output display format. Permitted format values are **grid**, **csv**, **tab**, and **vertical**. The default is **grid**.

- **--no-headers, -h**

Do not display column headers. This option applies only for **csv** and **tab** output.

- **--port-range=<start:end>**

The port range to check for finding running servers. This option applies only to Windows and is ignored unless [--show-servers \[400\]](#) is given. The default range is 3306:3333.

- **--server=<server>**

Connection information for a server in the format: *<user>[:<passwd>]@<host>[:<port>][:<socket>]* or *<login-path>[:<port>][:<socket>]*. Use this option multiple times to see information for multiple servers.

- **--show-defaults, -d**

Display default settings for `mysqld` from the local configuration file. It uses [my_print_defaults](#) to obtain the options.

- **--show-servers**

Display information about servers running on the local host. The utility examines the host process list to determine which servers are running.

- **--start, -s**

Start the server in read-only mode if it is offline. With this option, you must also give the [--basedir \[399\]](#) and [--datadir \[399\]](#) options.

- **--start-timeout**

Number of seconds to wait for the server to be online when started in read-only mode using the [--start \[400\]](#) option. The default value is 10 seconds.

The [--start-timeout](#) option is available as of MySQL Utilities 1.2.4 / 1.3.3.

- **--verbose, -v**

Specify how much information to display. Use this option multiple times to increase the amount of information. For example, `-v` = verbose, `-vv` = more verbose, `-vvv` = debug.

- **--version**

Display version information and exit.

For the [--format \[399\]](#) option, the permitted values are not case sensitive. In addition, values may be specified as any unambiguous prefix of a valid value. For example, [--format=g \[399\]](#) specifies the grid format. An error occurs if a prefix matches more than one valid value.

The path to the MySQL client tools should be included in the PATH environment variable in order to use the authentication mechanism with login-paths. This will allow the utility to use the `my_print_defaults` tools which is required to read the login-path values from the login configuration file (.mylogin.cnf).

EXAMPLES

To display the server information for the local server and the settings for `mysqld` in the configuration file with the output in a vertical list, use this command:

```
$ mysqlserverinfo --server=root:pass@localhost -d --format=vertical
# Source on localhost: ... connected.
*****          1. row *****
server: localhost:3306
version: 5.1.50-log
```

```

datadir: /usr/local/mysql/data/
basedir: /usr/local/mysql-5.1.50-osx10.6-x86_64/
plugin_dir: /usr/local/mysql-5.1.50-osx10.6-x86_64/lib/plugin
config_file: /etc/my.cnf
binary_log: my_log.000068
binary_log_pos: 212383
relay_log: None
relay_log_pos: None
1 rows.

Defaults for server localhost:3306
--port=3306
--basedir=/usr/local/mysql
--datadir=/usr/local/mysql/data
--server_id=5
--log-bin=my_log
--general_log
--slow_query_log
--innodb_data_file_path=ibdata1:778M;ibdata2:50M:autoextend
#...done.

```

15.6.20. mysqluc

15.6.20.1. [mysqluc](#) — Command line client for running MySQL Utilities

This utility provides a command line environment for running MySQL Utilities.

The mysqluc utility, hence console, allows users to execute any of the currently installed MySQL Utilities command. The option [--utildir \[403\]](#) is used to provide a path to the MySQL Utilities if the location is different from when the utility is executed.

The console has a list of console or base commands. These allow the user to interact with the features of the console itself. The list of base commands is shown below along with a brief description.:

Command	Description
help utilities	Display list of all utilities supported.
help <utility>	Display help for a specific utility.
help help commands	Show this list.
exit quit	Exit the console.
set <variable>=<value>	Store a variable for recall in commands.
show options	Display list of options specified by the user on launch.
show variables	Display list of variables.
<ENTER>	Press ENTER to execute command.
<ESCAPE>	Press ESCAPE to clear the command entry.
<DOWN>	Press DOWN to retrieve the previous command.
<UP>	Press UP to retrieve the next command in history.
<TAB>	Press TAB for type completion of utility, option, or variable names.
<TAB><TAB>	Press TAB twice for list of matching type completion (context sensitive).

One of the most helpful base commands is the ability to see the options for a given utility by typing 'help <utility>'. When the user types this command and presses ENTER, the console will display a list of all of the options for the utility.

The console provides tab completion for all commands, options for utilities, and user-defined variables. Tab completion for commands allows users to specify the starting N characters of a command and press TAB to complete the command. If there are more than one command that matches the prefix, and the user presses TAB twice, a list of all possible matches is displayed.

Tab completion for options is similar. The user must first type a valid MySQL Utility command then types the first N characters of a command and presses TAB, for example --verb<TAB>. In this case, the console will complete the option. For the cases where an option requires a value, the console will complete the option name and append the '=' character. Tab completion for options works for both the full name and the alias (if available). If the user presses TAB twice, the console will display a list of matching options. Pressing TAB twice immediately after typing the name of a MySQL Utility will display a list of all options for that utility.

Tab completion for variables works the same as that for options. In this case, the user must first type the '\$' character then press TAB. For example, if a variable \$SERVER1 exists, when the user types --server=\$SER<TAB>, the console will complete the \$SERVER variable name. For cases where there are multiple variables, pressing TAB twice will display a list of all matches to the first \$+N characters. Pressing TAB twice after typing only the \$ character will display a list of all variables.

Note: the console does not require typing the 'mysql' prefix for the utility. For example, if the user types 'disku<TAB>' the console will complete the command with 'diskusage'.

Executing utilities is accomplished by typing the complete command and pressing ENTER. The user does not have to type 'python' or provide the '.py' file extension. The console will add these if needed.

The user can also run commands using the option [--execute \[403\]](#). The value for this option is a semi-colon separated list of commands to execute. These can be base commands or MySQL Utility commands. The console will execute each command and display the output. All commands to be run by the console must appear inside a quoted string and separated by semi-colons. Commands outside of the quoted string will be treated as arguments for the mysqluc utility itself and thus ignored for execution.

Note: if there is an error in the console or related code, the console will stop executing commands at the point of failure. Commands may also be piped into the console using a mechanism like 'echo "<commands>" | mysqluc".

The console also allows users to set user-defined variables for commonly used values in options. The syntax is simply 'set VARNAME=VALUE'. The user can see a list of all variables by entering the 'show variables' command. To use the values of these variables in utility commands, the user must prefix the value with a '\$'. For example, --server=\$SERVER1 will substitute the value of the SERVER1 user-defined variable when the utility is executed.

Note: user-defined variables have a session lifetime. They are not saved from one execution to another of the users console.

User-defined variables may also be set by passing them as arguments to the mysqluc command. For example, to set the SERVER1 variable and launch the console, the user can launch the console using this command.:

```
$ mysqluc SERVER1=root@localhost
```

The user can provide any number of user-defined variables but they must contain a value and no spaces around the '=' character. Once the console is launched, the user can see all variables using the 'show variables' command.

OPTIONS

- --version
 - show program's version number and exit
- --help
 - show the program's help page

- **--verbose, -v**

control how much information is displayed. For example, **-v** = verbose, **-vv** = more verbose, **-vvv** = debug

- **--quiet**

suppress all informational messages

- **--execute <commands>, -e <commands>**

Execute commands and exit. Multiple commands are separated with semi-colons. Note: some platforms may require double quotes around command list.

- **--utildir <path>**

location of utilities

- **--width <number>**

Display width

NOTES

Using the **--execute** option or piping commands to the console may require quotes or double quotes (for example, on Windows).

EXAMPLES

To launch the console, use this command:

```
$ mysqluc
```

The following demonstrates launching the console and running the console command 'help utilities' to see a list of all utilities supported. The console will execute the command then exit.:

```
$ mysqluc -e "help utilities"

Utility          Description
-----
mysqlindexcheck  check for duplicate or redundant indexes
mysqrlpcheck     check replication
mysqluserclone   clone a MySQL user account to one or more new users
mysqldbcompare   compare databases for consistency
mysqldiff        compare object definitions among objects where the
                  difference is how db1.obj1 differs from db2.obj2
mysqldbcopy      copy databases from one server to another
mysqlreplicate   establish replication with a master
mysqldbexport    export metadata and data from databases
mysqldbimport    import metadata and data from files
mysqlmetagrep    search metadata
mysqlprocgrep    search process information
mysqldiskusage   show disk usage for databases
mysqlserverinfo  show server information
mysqlserverclone start another instance of a running server
```

The following demonstrates launching the console to run several commands using the **--execute** option to including setting a variable for a server connection and executing a utility using variable substitution. Note: it may be necessary to escape the '\$' on some platforms (for example, Linux). Output below is an excerpt and is representational only.:

```
$ mysqluc -e "set SERVER=root@host123; mysqldiskusage --server=\$SERVER"
```

```
# Source on host123: ... connected.

NOTICE: Your user account does not have read access to the datadir. Data
sizes will be calculated and actual file sizes may be omitted. Some features
may be unavailable.

# Database totals:
+-----+-----+
| db_name |      total |
+-----+-----+
...
| world   |          0 |
...
+-----+-----+

Total database disk usage = 1,072,359,052 bytes or 1022.00 MB

#...done.
```

The following demonstrates launching the console using the commands shown above but piped into the console on the command line. The results are the same as above.:

```
$ echo "set SERVER=root@host123; mysqldiskusage --server=\$SERVER" | mysqluc
```

The following demonstrates launching the console and setting variables via the command line.:

```
$ mysqluc SERVER=root@host123 VAR_A=57 -e "show variables"

Variable  Value
-----
SERVER    root@host123
VAR_A     57
```

15.6.21. mysqluserclone

15.6.21.1. mysqluserclone — Clone Existing User to Create New User

This utility uses an existing MySQL user account on one server as a template, and clones it to create one or more new user accounts with the same privileges as the original user. The new users can be created on the original server or a different server.

To list users for a server, specify the [--list \[405\]](#) option. This prints a list of the users on the source (no destination is needed). To control how to display list output, use one of the following values with the [--format \[405\]](#) option:

- **grid** (default)

Display output in grid or table format like that of the `mysql` monitor.

- **csv**

Display output in comma-separated values format.

- **tab**

Display output in tab-separated format.

- **vertical**

Display output in single-column format like that of the `\G` command for the `mysql` monitor.

OPTIONS

`mysqluserclone` accepts the following command-line options:

- `--help`

Display a help message and exit.

- `--destination=<destination>`

Connection information for the destination server in the format: `<user>[:<passwd>]@<host>[:<port>][:<socket>]` or `<login-path>[:<port>][:<socket>]`.

- `--dump, -d`

Display the **GRANT** statements to create the account rather than executing them. In this case, the utility does not connect to the destination server and no `--destination` [405] option is needed.

- `--format=<list_format>, -f<list_format>`

Specify the user display format. Permitted format values are **grid**, **csv**, **tab**, and **vertical**. The default is **grid**. This option is valid only if `--list` [405] is given.

- `--force`

Drop the new user account if it exists before creating the new account. Without this option, it is an error to try to create an account that already exists.

- `--include-global-privileges`

Include privileges that match `base_user@%` as well as `base_user@host`.

- `--list`

List all users on the source server. With this option, a destination server need not be specified.

- `--quiet, -q`

Turn off all messages for quiet execution.

- `--source=<source>`

Connection information for the source server in the format: `<user>[:<passwd>]@<host>[:<port>][:<socket>]` or `<login-path>[:<port>][:<socket>]`.

- `--verbose, -v`

Specify how much information to display. Use this option multiple times to increase the amount of information. For example, `-v` = verbose, `-vv` = more verbose, `-vvv` = debug.

- `--version`

Display version information and exit.

NOTES

You must provide connection parameters (user, host, password, and so forth) for an account that has the appropriate privileges to access all objects in the operation.

The account used to connect to the source server must have privileges to read the **mysql** database.

The account used to connect to the destination server must have privileges to execute **CREATE USER** (and **DROP USER** if the `--force` [405] option is given), and privileges to execute **GRANT** for all privileges to be granted to the new accounts.

For the `--format` [405] option, the permitted values are not case sensitive. In addition, values may be specified as any unambiguous prefix of a valid value. For example, `--format=g` [405] specifies the grid format. An error occurs if a prefix matches more than one valid value.

The path to the MySQL client tools should be included in the PATH environment variable in order to use the authentication mechanism with login-paths. This will allow the utility to use the `my_print_defaults` tools which is required to read the login-path values from the login configuration file (`.mylogin.cnf`).

EXAMPLES

To clone `joe` as `sam` and `sally` with passwords and logging in as `root` on the local machine, use this command:

```
$ mysqluserclone --source=root@localhost \
  --destination=root@localhost \
  joe@localhost sam:secret1@localhost sally:secret2@localhost
# Source on localhost: ... connected.
# Destination on localhost: ... connected.
# Cloning 2 users...
# Cloning joe@localhost to user sam:secret1@localhost
# Cloning joe@localhost to user sally:secret2@localhost
# ...done.
```

The following command shows all users on the local server in the most verbose output in CSV format:

```
$ mysqluserclone --source=root@localhost --list --format=csv -vvv
# Source on localhost: ... connected.
user,host,database
joe,localhost,util_test
rpl,localhost,
sally,localhost,util_test
sam,localhost,util_test
joe,user,util_test
```

15.7. Extending MySQL Utilities

This chapter introduces the architecture for the MySQL Utilities library and demonstrates how to get started building your own utilities.

15.7.1. Introduction to extending the MySQL Utilities

Administration and maintenance on the MySQL server can at times be complicated. Sometimes tasks require tedious or even repetitive operations that can be time consuming to type and re-type. For these reasons and more, the MySQL Utilities were created to help both beginners and experienced database administrators perform common tasks.

What are the internals of the MySQL Utilities?

MySQL Utilities are designed as a collection of easy to use Python scripts that can be combined to provide more powerful features. Internally, the scripts use the `mysql.utilities` module library to perform its various tasks. Since a library of common functions is available, it is easy for a database administrator to create

scripts for common tasks. These utilities are located in the `/scripts` folder of the installation or source tree.

If you have a task that is not met by these utilities or one that can be met by combining one or more of the utilities or even parts of the utilities, you can easily form your own custom solution. The following sections present an example of a custom utility, discussing first the anatomy of a utility and then what the `mysql.utilities` module library has available.

Anatomy of a MySQL Utility

MySQL Utilities use a three-tier module organization. At the top is the command script, which resides in the `/scripts` folder of the installation or source tree. Included in the script is a command module designed to encapsulate and isolate the bulk of the work performed by the utility. The command module resides in the `/mysql/utilities/command` folder of the source tree. Command modules have names similar to the script. A command module includes classes and methods from one or more common modules where the abstract objects and method groups are kept. The common modules reside in the `/mysql/utilities/common` folder of the source tree. The following illustrates this arrangement using the `mysqlserverinfo` utility:

```
/scripts/mysqlserverinfo.py
  |
  +--- /mysql/utilities/command/serverinfo.py
    |
    +--- /mysql/utilities/common/options.py
    |
    +--- /mysql/utilities/common/server.py
    |
    +--- /mysql/utilities/common/tools.py
    |
    +--- /mysql/utilities/common/format.py
```

Each utility script is designed to process the user input and option settings and pass them on to the command module. Thus, the script contains only such logic for managing and validating options. The work of the operation resides in the command module.

Command modules are designed to be used from other Python applications. For example, one could call the methods in the `serverinfo.py` module from another Python script. This enables developers to create their own interfaces to the utilities. It also permits developers to combine several utilities to form a macro-level utility tailored to a specified need. For example, if there is a need to gather server information as well as disk usage, it is possible to import the `serverinfo.py` and `diskusage.py` modules and create a new utility that performs both operations.

Common modules are the heart of the MySQL Utilities library. These modules contain classes that abstract MySQL objects, devices, and mechanisms. For example, there is a server class that contains operations to be performed on servers, such as connecting (logging in) and running queries.

The MySQL Utilities Library

While the library is growing, the following lists the current common modules and the major classes and methods as of the 1.0.1 release:

Module	Class/Method	Description
database	Database	Perform database-level operations
dbcompare	get_create_object	Retrieve object create statement
	diff_objects	Diff definitions of two objects
	check_consistency	Check data consistency of two tables
format	format_tabular_list	Format list in either GRID or delimited format to a file

	format_vertical_list	Format list in a vertical format to a file
	print_list	Print list based on format (CSV, GRID, TAB, or VERTICAL)
options	setup_common_options	Set up option parser and options common to all MySQL Utilities
	add_skip_options	Add common --skip options
	check_skip_options	Check skip options for validity
	check_format_option	Check format option for validity
	add_verbosity	Add verbosity and quiet options
	check_verbosity	Check whether both verbosity and quiet options are being used
	add_difftype	Add difftype option
	add_engines	Add engine, default-storage-engine options
	check_engine_options	Check whether storage engines listed in options exist
rpl	parse_connection	Parse connection values
	Replication	Establish replication connection between a master and a slave
	get_replication_tests	Return list of replication test function pointers
server	get_connection_dictionary	Get connection dictionary
	find_running_servers	Check whether any servers are running on the local host
	connect_servers	Connect to source and destination server
	Server	Connect to running MySQL server and perform server-level operations
table	Index	Encapsulate index for a given table as defined by SHOW INDEXES
	Table	Encapsulate table for given database to perform table-level operations
tools	get_tool_path	Search for MySQL tool and return its full path
	delete_directory	Remove directory (folder) and contents
user	parse_user_host	Parse user, passwd, host, port from user:passwd@host
	User	Clone user and its grants to another user and perform user-level operations

General Interface Specifications and Code Practices

The MySQL Utilities are designed and coded using mainstream coding practices and techniques common to the Python community. Effort has been made to adhere to the most widely accepted specifications and techniques. This includes limiting the choice of libraries used to the default libraries found in the Python distributions. This ensures easier installation, enhanced portability, and fewer problems with missing libraries. Similarly, external libraries that resort to platform-specific native code are also not used.

The class method and function signatures are designed to make use of a small number of required parameters and all optional parameters as a single dictionary. Consider the following method:

```
def do_something_wonderful(position, obj1, obj2, options={}):
    """Does something wonderful

    A fictional method that does something to object 2 based on the
    location of something in object 1.

    position[in]      Position in obj1
    obj1[in]         First object to manipulate
    obj2[in]         Second object to manipulate
    options[in]       Option dictionary
        width        width of printout (default 75)
        iter          max iterations (default 2)
        ok_to_fail   if True, do not throw exception
                      (default True)
```

```
Returns bool - True = success, Fail = failed
"""

```

This example is typical of the methods and classes in the library. Notice that this method has three required parameters and a dictionary of options that may exist.

Each method and function that uses this mechanism defines its own default values for the items in the dictionary. A quick look at the method documentation shows the key names for the dictionary. This can be seen in the preceding example where the dictionary contains three keys and the documentation lists their defaults.

To call this method and pass different values for one or more of the options, the code may look like this:

```
opt_dictionary = {
    'width'      : 100,
    'iter'       : 10,
    'ok_to_fail' : False,
}
result = do_something_wonderful(1, obj_1, obj_2, opt_dictionary)
```

The documentation block for the preceding method is the style used throughout the library.

Example

Now that you are familiar with the MySQL utilities and the supporting library modules, let us take a look at an example that combines some of these modules to solve a problem.

Suppose that you want to develop a new database solution and need to use real world data and user accounts for testing. The `mysqlserverclone` MySQL utility looks like a possibility but it makes only an instance of a running server. It does not copy data. However, `mysqldbcopy` makes a copy of the data and `mysqluserclone` clones the users. You could run each of these utilities in sequence, and that would work, but we are lazy at heart and want something that not only copies everything but also finds it for us. That is, we want a one-command solution.

The good news is that this is indeed possible and very easy to do. Let us start by breaking the problem down into its smaller components. In a nutshell, we must perform these tasks:

- Connect to the original server
- Find all of the databases
- Find all of the users
- Make a clone of the original server
- Copy all of the databases
- Copy all of the users

If you look at the utilities and the modules just listed, you see that we have solutions and primitives for each of these operations. So you need not even call the MySQL utilities directly (although you could). Now let us dive into the code for this example.

The first task is to connect to the original server. We use the same connection mechanism as the other MySQL utilities by specifying a `--server` option like this:

```
parser.add_option("--server", action="store", dest="server",
                  type="string", default="root@localhost:3306",
```

```
help="connection information for original server in " + \
"the form: <user>:<password>@<host>:<port>:<socket>")
```

Once we process the options and arguments, connecting to the server is easy: Use the `parse_connection` method to take the server option values and get a dictionary with the connection values. All of the heavy diagnosis and error handling is done for us, so we just need to check for exceptions:

```
from mysql.utilities.common.options import parse_connection

try:
    conn = parse_connection(opt.server)
except:
    parser.error("Server connection values invalid or cannot be parsed.")
```

Now that we have the connection parameters, we create a class instance of the server using the `Server` class from the `server` module and then connect. Once again, we check for exceptions:

```
from mysql.utilities.common.server import Server

server_options = {
    'conn_info' : conn,
    'role'      : "source",
}
server1 = Server(server_options)
try:
    server1.connect()
except UtilError, e:
    print "ERROR:", e errmsg
```

The next item is to get a list of all of the databases on the server. We use the new server class instance to retrieve all of the databases on the server:

```
db_list = []
for db in server1.get_all_databases():
    db_list.append((db[0], None))
```

If you wanted to supply your own list of databases, you could use an option like the following. You could also add an `else` clause which would enable you to either get all of the databases by omitting the `--databases` option or supply your own list of databases (for example, `--databases=db1,db2,db3`):

```
parser.add_option("-d", "--databases", action="store", dest="dbs_to_copy",
                  type="string", help="comma-separated list of databases "
                  "to include in the copy (omit for all databases)",
                  default=None)

if opt.dbs_to_copy is None:
    for db in server1.get_all_databases():
        db_list.append((db[0], None))
else:
    for db in opt.dbs_to_copy.split(","):
        db_list.append((db, None))
```

Notice we are creating a list of tuples. This is because the `dbcopy` module uses a list of tuples in the form (`old_db, new_db`) to enable you to copy a database to a new name. For our purposes, we do not want a rename so we leave the new name value set to `None`.

Next, we want a list of all of the users. Once again, you could construct the new solution to be flexible by permitting the user to specify the users to copy. We leave this as an exercise.

In this case, we do not have a primitive for getting all users created on a server. But we do have the ability to run a query and process the results. Fortunately, there is a simple SQL statement that can retrieve all of

the users on a server. For our purposes, we get all of the users except the root and anonymous users, then add each to a list for processing later:

```
users = server1.exec_query("SELECT user, host "
                           "FROM mysql.user "
                           "WHERE user != 'root' and user != ''")
for user in users:
    user_list.append(user[0] +'@'+user[1])
```

Now we must clone the original server and create a viable running instance. When you examine the `mysqlserverclone` utility code, you see that it calls another module located in the `/mysql/utilities/command` sub folder. These modules are where all of the work done by the utilities take place. This enables you to create new combinations of the utilities by calling the actual operations directly. Let's do that now to clone the server.

The first thing you notice in examining the `serverclone` module is that it takes a number of parameters for the new server instance. We supply those in a similar way as options:

```
parser.add_option("--new-data", action="store", dest="new_data",
                  type="string", help="the full path to the location "
                  "of the data directory for the new instance")
parser.add_option("--new-port", action="store", dest="new_port",
                  type="string", default="3307", help="the new port "
                  "for the new instance - default=%default")
parser.add_option("--new-id", action="store", dest="new_id",
                  type="string", default="2", help="the server_id for "
                  "the new instance - default=%default")

from mysql.utilities.command import serverclone

try:
    res = serverclone.clone_server(conn, opt.new_data, opt.new_port,
                                    opt.new_id, "root", None, False, True)
except exception.UtilError, e:
    print "ERROR:", e.errmsg
    sys.exit()
```

As you can see, the operation is very simple. We just added a few options we needed like `--new-data`, `--new-port`, and `--new-id` (much like `mysqlserverclone`) and supplied some default values for the other parameters.

Next, we need to copy the databases. Once again, we use the command module for `mysqldbcopy` to do all of the work for us. First, we need the connection parameters for the new instance. This is provided in the form of a dictionary. We know the instance is a clone, so some of the values are going to be the same and we use a default root password, so that is also known. Likewise, we specified the data directory and, since we are running on a Linux machine, we know what the socket path is. (For Windows machines, you can leave the socket value `None`.) We pass this dictionary to the copy method:

```
dest_values = {
    "user"      : conn.get("user"),
    "passwd"    : "root",
    "host"      : conn.get("host"),
    "port"      : opt.new_port,
    "unix_socket" : os.path.join(opt.new_data, "mysql.sock")
}
```

In this case, a number of options are needed to control how the copy works (for example, if any objects are skipped). For our purposes, we want all objects to be copied so we supply only the minimal settings and let the library use the defaults. This example shows how you can 'fine tune' the scripts to meet your specific needs without having to specify a lot of additional options in your script. We enable the quiet option on so as not to clutter the screen with messages, and tell the copy to skip databases that do not exist (in case we supply the `--databases` option and provide a database that does not exist):

```
options = {
    "quiet" : True,
    "force" : True
}
```

The actual copy of the databases is easy. Just call the method and supply the list of databases:

```
from mysql.utilities.command import dbcopy

try:
    dbcopy.copy_db(conn, dest_values, db_list, options)
except exception.UtilError, e:
    print "ERROR:", e errmsg
    sys.exit()
```

Lastly, we copy the user accounts. Once again, we must provide a dictionary of options and call the command module directly. In this case, the `userclone` module provides a method that clones one user to one or more users so we must loop through the users and clone them one at a time:

```
from mysql.utilities.command import userclone

options = {
    "overwrite" : True,
    "quiet"     : True,
    "globals"   : True
}

for user in user_list:
    try:
        res = userclone.clone_user(conn, dest_values, user,
                                    (user,), options)
    except exception.UtilError, e:
        print "ERROR:", e errmsg
        sys.exit()
```

We are done. As you can see, constructing new solutions from the MySQL utility command and common modules is easy and is limited only by your imagination.

Enhancing the Example

A complete solution for the example named `copy_server.py` is located in the appendix. It is complete in so far as this document explains, but it can be enhanced in a number of ways. The following briefly lists some of the things to consider adding to make this example utility more robust.

- Table locking: Currently, databases are not locked when copied. To achieve a consistent copy of the data on an active server, you may want to add table locking or use transactions (for example, if you are using InnoDB) for a more consistent copy.
- Skip users not associated with the databases being copied.
- Do not copy users with only global privileges.
- Start replication after all of the users are copied (makes this example a clone and replicate scale out solution).
- Stop new client connections to the server during the copy.

Conclusion

If you find some primitives missing or would like to see more specific functionality in the library or scripts, please contact us with your ideas or better still, write them yourselves! We welcome all suggestions in

code or text. To file a feature request or bug report, visit <http://bugs.mysql.com>. For discussions, visit <http://forums.mysql.com/list.php?155>.

15.7.2. MySQL Utilities copy_server.py sample

```
#  
# Copyright (c) 2010, 2013, Oracle and/or its affiliates. All rights reserved.  
#  
# This program is free software; you can redistribute it and/or modify  
# it under the terms of the GNU General Public License as published by  
# the Free Software Foundation; version 2 of the License.  
#  
# This program is distributed in the hope that it will be useful,  
# but WITHOUT ANY WARRANTY; without even the implied warranty of  
# MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the  
# GNU General Public License for more details.  
#  
# You should have received a copy of the GNU General Public License  
# along with this program; if not, write to the Free Software  
# Foundation, Inc., 51 Franklin St, Fifth Floor, Boston, MA 02110-1301 USA  
#  
#  
"""  
This file contains an example of how to build a customized utility using  
the MySQL Utilities scripts and libraries.  
"""  
  
import optparse  
import os  
import sys  
  
from mysql.utilities import VERSION_FRM  
from mysql.utilities.command import dbcopy  
from mysql.utilities.command import serverclone  
from mysql.utilities.command import userclone  
from mysql.utilities.common.server import Server  
from mysql.utilities.common.options import parse_connection  
from mysql.utilities.exception import UtilError  
  
# Constants  
NAME = "example - copy_server"  
DESCRIPTION = "copy_server - copy an existing server"  
USAGE = "%prog --server=user:pass@host:port:socket " \  
       "--new-dir=<path> --new-id=<server_id> " \  
       "--new-port=<port> --databases=<db list> " \  
       "--users=<user list>"  
  
# Setup the command parser  
parser = optparse.OptionParser(  
    version=VERSION_FRM.format(program=os.path.basename(sys.argv[0])),  
    description=DESCRIPTION,  
    usage=USAGE,  
    add_help_option=False)  
parser.add_option("--help", action="help")  
  
# Setup utility-specific options:  
  
# Connection information for the source server  
parser.add_option("--server", action="store", dest="server",  
                 type="string", default="root@localhost:3306",  
                 help="connection information for original server in " + \  
                      "the form: <user>:<password>@<host>:<port>:<socket>")  
  
# Data directory for new instance
```

```

parser.add_option("--new-data", action="store", dest="new_data",
                  type="string", help="the full path to the location "
                  "of the data directory for the new instance")

# Port for the new instance
parser.add_option("--new-port", action="store", dest="new_port",
                  type="string", default="3307", help="the new port "
                  "for the new instance - default=%default")

# Server id for the new instance
parser.add_option("--new-id", action="store", dest="new_id",
                  type="string", default="2", help="the server_id for "
                  "the new instance - default=%default")

# List of databases
parser.add_option("-d", "--databases", action="store", dest="dbs_to_copy",
                  type="string", help="comma-separated list of databases "
                  "to include in the copy (omit for all databases)",
                  default=None)

# List of users
parser.add_option("-u", "--users", action="store", dest="users_to_copy",
                  type="string", help="comma-separated list of users "
                  "to include in the copy (omit for all users)",
                  default=None)

# Now we process the rest of the arguments.
opt, args = parser.parse_args()

# Parse source connection values
try:
    conn = parse_connection(opt.server)
except:
    parser.error("Server connection values invalid or cannot be parsed.")

# Get a server class instance
print "# Connecting to server..."
server_options = {
    'conn_info' : conn,
    'role'      : "source",
}
server1 = Server(server_options)
try:
    server1.connect()
except UtilError, e:
    print "ERROR:", e.errmsg

# Get list of databases from the server if not specified in options
print "# Getting databases..."
db_list = []
if opt.dbs_to_copy is None:
    for db in server1.get_all_databases():
        db_list.append((db[0], None))
else:
    for db in opt.dbs_to_copy.split(","):
        db_list.append((db, None))

# Get list of all users from the server
print "# Getting users..."
user_list=[]
if opt.users_to_copy is None:
    users = server1.exec_query("SELECT user, host "
                               "FROM mysql.user "
                               "WHERE user != 'root' and user != ''")
    for user in users:
        user_list.append(user[0] + '@' + user[1])
else:

```

```
for user in opt.users_to_copy.split(", "):
    user_list.append(user)

# Build options
options = {
    'new_data'        : opt.new_data,
    'new_port'        : opt.new_port,
    'new_id'          : opt.new_id,
    'root_pass'       : 'root',
    'mysqld_options' : '--report-host=localhost --report-port=%s' % opt.new_port,
}

# Clone the server
print "# Cloning server instance..."
try:
    res = serverclone.clone_server(conn, options)
except UtilError, e:
    print "ERROR:", eerrmsg
    sys.exit()

# Set connection values
dest_values = {
    "user"      : conn.get("user"),
    "passwd"    : "root",
    "host"      : conn.get("host"),
    "port"      : opt.new_port,
    "unix_socket": os.path.join(opt.new_data, "mysql.sock")
}

# Build dictionary of options
options = {
    "quiet" : True,
    "force" : True
}

print "# Copying databases..."
try:
    dbcopy.copy_db(conn, dest_values, db_list, options)
except UtilError, e:
    print "ERROR:", eerrmsg
    sys.exit()

# Build dictionary of options
options = {
    "overwrite" : True,
    "quiet"     : True,
    "globals"   : True
}

print "# Cloning the users..."
for user in user_list:
    try:
        res = userclone.clone_user(conn, dest_values, user,
                                    (user,), options)
    except UtilError, e:
        print "ERROR:", eerrmsg
        sys.exit()

print "# ...done."
```

15.7.3. Specialized Operations

15.7.3.1. `mysql.utilities.command.grep` — Search Databases for Objects

This module provides utilities to search for objects on a server. The module defines a set of *object types* that can be searched by searching the *fields* of each object. The notion of an object field is very loosely defined and means any names occurring as part of the object definition. For example, the fields of a table include the table name, the column names, and the partition names (if it is a partitioned table).

Constants

The following constants denote the object types that can be searched.

- `mysql.utilities.command.grep.ROUTINE`
- `mysql.utilities.command.grep.EVENT`
- `mysql.utilities.command.grep.TRIGGER`
- `mysql.utilities.command.grep.TABLE`
- `mysql.utilities.command.grep.DATABASE`
- `mysql.utilities.command.grep.VIEW`
- `mysql.utilities.command.grep.USER`

The following constant is a sequence of all the object types that are available. It can be used to generate a version-independent list of object types that can be searched; for example, options and help texts.

- `mysql.utilities.command.grep.OBJECT_TYPES`

Classes

```
class mysql.utilities.command.grep.ObjectGrep(pattern[, database_pattern=None,  
types=OBJECT_TYPES, check_body=False, use_regex=False])
```

Search MySQL server instances for objects where the name (or content, for routines, triggers, or events) matches a given pattern.

sql() - string

Return the SQL code for executing the search in the form of a `SELECT` statement.

Returns:	SQL code for executing the operation specified by the options.
Return type:	string

execute(connections[, output=sys.output, connector=mysql.connector])

Execute the search on each of the connections in turn and print an aggregate of the result as a grid table.

Parameters:	<ul style="list-style-type: none">• connections Sequence of connection specifiers to send the query to• output File object to use for writing the result• connector Connector to use for connecting to the servers
-------------	---

15.7.3.2. `mysql.utilities.command.proc` — Search Processes on Servers

This module searches processes on a server and optionally kills either the query or the connection for all matching processes.

Processes are matched by searching the fields of the [INFORMATION_SCHEMA.PROCESSLIST](#) table (which is available only for servers from MySQL 5.1.7 and later). Internally, the module operates by constructing a [SELECT](#) statement for finding matching processes, and then sending it to the server. Instead of performing the search, the module can return the SQL code that performs the query. This can be useful if you want to execute the query later or feed it to some other program that processes SQL queries further.

Constants

The following constants correspond to columns in the [INFORMATION_SCHEMA.PROCESSLIST](#) table. They indicate which columns to examine when searching for processes matching the search conditions.

- `mysql.utilities.command.proc.ID`
- `mysql.utilities.command.proc.USER`
- `mysql.utilities.command.proc.HOST`
- `mysql.utilities.command.proc.DB`
- `mysql.utilities.command.proc.COMMAND`
- `mysql.utilities.command.proc.TIME`
- `mysql.utilities.command.proc.STATE`
- `mysql.utilities.command.proc.INFO`

The following constants indicate actions to perform on processes that match the search conditions.

- `mysql.utilities.command.proc.KILL_QUERY`
Kill the process query
- `mysql.utilities.command.proc.KILL_CONNECTION`
Kill the process connection
- `mysql.utilities.command.proc.PRINT_PROCESS`
Print the processes

Classes

`class mysql.utilities.command.proc.ProcessGrep(matches, actions=[], use_regex=False)`

This class searches the [INFORMATION_SCHEMA.PROCESSLIST](#) table for processes on MySQL servers and optionally kills them. It can both be used to actually perform the search or kill operation, or to generate the SQL statement for doing the job.

To kill all queries with user 'mats', the following code can be used:

```
>>> from mysql.utilities.command.proc import *
>>> grep = ProcessGrep(matches=[(USER, "mats")], actions=[KILL_QUERY])
>>> grep.execute("root@server-1.example.com", "root@server-2.example.com")
```

Parameters:

- **matches** (List of *(var, pat)* pairs) Sequence of field comparison conditions. In each condition, *var* is one of the constants listed earlier that specify [PROCESSLIST](#)

	table fields and <i>pat</i> is a pattern. For a process to match, all field conditions must match.
--	--

sql([only_body=False])

Return the SQL code for executing the search (and optionally, the kill).

If *only_body* is `True`, only the body of the function is shown. This is useful if the SQL code is to be used with other utilities that generate the routine declaration. If *only_body* is `False`, a complete procedure will be generated if there is any kill action supplied, and just a select statement if it is a plain search.

Parameters:	<ul style="list-style-type: none"> • only_body (<i>boolean</i>) Show only the body of the procedure. If this is <code>False</code>, a complete procedure is returned.
Returns:	SQL code for executing the operation specified by the options.
Return type:	string

execute(connections, ...[, output=sys.stdout, connector=mysql.connector])

Execute the search on each of the connections supplied. If *output* is not `None`, the value is treated as a file object and the result of the execution is printed on that stream. Note that the output and connector arguments *must* be supplied as keyword arguments. All other arguments are treated as connection specifiers.

Parameters:	<ul style="list-style-type: none"> • connections Sequence of connection specifiers to send the search to • output File object to use for writing the result • connector Connector to use for connecting to the servers
-------------	--

15.7.4. Parsers

15.7.4.1. mysql.utilities.parser — Parse MySQL Log Files

This module provides classes for parsing MySQL log files. Currently, *Slow Query Log* and *General Query Log* are supported.

Classes

class mysql.utilities.parser.GeneralQueryLog(stream)

This class parses the MySQL General Query Log. Instances are iterable, but the class does not provide multiple independent iterators.

For example, to read the log and print the entries:

```
>>> general_log = open( "/var/lib/mysql/mysql.log" )
>>> log = GeneralQueryLog(general_log)
>>> for entry in log:
...     print entry
```

Parameters:	<ul style="list-style-type: none"> • stream (<i>file type</i>) – a valid file type; for example, the result of the built-in Python function <code>open()</code>
-------------	---

version

Returns:	Version of the MySQL server that produced the log
Return type:	tuple

program

Returns:	Full path of the MySQL server executable
Return type:	str

port

Returns:	TCP/IP port on which the MySQL server was listening
Return type:	int

socket

Returns:	Full path of the MySQL server Unix socket
Return type:	str

start_datetime

Returns:	Date and time of the first read log entry
Return type:	datetime.datetime

lastseen_datetime

Returns:	Date and time of the last read log entry
Return type:	datetime.datetime

class mysql.utilities.parser.SlowQueryLog(stream)

This class parses the MySQL Slow Query Log. Instances are iterable, but the class does not provide multiple independent iterators.

For example, to read the log and print the entries:

```
>>> slow_log = open("/var/lib/mysql/mysql-slow.log")
>>> log = SlowQueryLog(slow_log)
>>> for entry in log:
...     print entry
```

Parameters:	<ul style="list-style-type: none"> • stream (<i>file type</i>) – a valid file type; for example, the result of the built-in Python function open()
-------------	--

version

Returns:	Version of the MySQL server that produced the log
Return type:	tuple

program

Returns:	Full path of the MySQL server executable
----------	--

Return type:	str
--------------	-----

port

Returns:	TCP/IP port on which the MySQL server was listening
Return type:	int

socket

Returns:	Full path of the MySQL server Unix socket
Return type:	str

start_datetime

Returns:	Date and time of the first read log entry
Return type:	datetime.datetime

lastseen_datetime

Returns:	Date and time of the last read log entry
Return type:	datetime.datetime

15.8. MySQL Utilities Testing (MUT)

15.8.1. `mut` — MySQL Utilities Testing

This utility executes predefined tests to test the MySQL Utilities. The tests are located under the `/mysql-test` directory and divided into suites (stored as folders). By default, all tests located in the `/t` folder are considered the 'main' suite.

You can select any number of tests to run, select one or more suites to restrict the tests, exclude suites and tests, and specify the location of the utilities and tests.

The utility requires the existence of at least one server to clone for testing purposes. You must specify at least one server, but you may specify multiple servers for tests designed to use additional servers.

The utility has a special test suite named 'performance' where performance-related tests are placed. This suite is not included by default and must be specified with the `--suite [421]` option to execute the performance tests.

OPTIONS

`mut` accepts the following command-line options:

- `--help`

Display a help message and exit.

- `--do-tests=<prefix>`

Execute all tests that begin with *prefix*.

- `--force`

Do not abort when a test fails.

- `--record`

Record the output of the specified test if successful. With this option, you must specify exactly one test to run.

- `--server=<server>`

Connection information for the server to use in the tests, in `<user>[:<passwd>]@<host>[:<port>]` [`:<socket>`] format. Use this option multiple times to specify multiple servers.

- `--skip-long`

Exclude tests that require greater resources or take a long time to run.

- `--skip-suite=<name>`

Exclude the named test suite. Use this option multiple times to specify multiple suites.

- `--skip-test=<name>`

Exclude the named test. Use this option multiple times to specify multiple tests.

- `--skip-tests=<prefix>`

Exclude all tests that begin with *prefix*.

- `--sort`

Execute tests sorted by suite.name either ascending (asc) or descending (desc). Default is ascending (asc).

- `--start-port=<port>`

The first port to use for spawned servers. If you run the entire test suite, you may see up to 12 new instances created. The default is to use ports 3310 to 3321.

- `--start-test=<prefix>`

Start executing tests that begin with *prefix*.

- `--stop-test=<prefix>`

Stop executing tests at the first test that begins with *prefix*.

- `--suite=<name>`

Execute the named test suite. Use this option multiple times to specify multiple suites.

- `--testdir=<path>`

The path to the test directory.

- `--utildir=<path>`

The location of the utilities.

- `--verbose, -v`

Specify how much information to display. Use this option multiple times to increase the amount of information. For example, `-v` = verbose, `-vv` = more verbose, `-vvv` = debug. To diagnose test execution problems, use `-vvvv` to display the actual results of test cases and ignore result processing.

- `--version`
Display version information and exit.
- `--width=<number>`
Specify the display width. The default is 75 characters.

NOTES

The connection specifier must name a valid account for the server.

Any test named `???.template.py` is skipped. This enables the developer to create a base class to import for a collection of tests based on a common code base.

EXAMPLES

The following example demonstrates how to invoke `mut` to execute a subset of the tests using an existing server which is cloned. The example displays the test name, status, and relative time:

```
$ mut --server=root@localhost --do-tests=clone_user --width=70

MySQL Utilities Testing - MUT

Parameters used:
  Display Width      = 70
  Sorted             = True
  Force              = False
  Test directory     = './t'
  Utilities directory = '../scripts'
  Starting port      = 3310
  Test wildcard      = 'clone_user%'

Servers:
  Connecting to localhost as user root on port 3306: CONNECTED

-----
TEST NAME                      STATUS    TIME
=====
main.clone_user                  [pass]    54
main.clone_user_errors           [pass]    27
main.clone_user_parameters       [pass]    17
-----

Testing completed: Friday 03 December 2010 09:50:06

All 3 tests passed.
```

15.9. Appendix

This chapter includes additional information about MySQL Utilities including a list of frequently asked questions and the change history.

15.9.1. MySQL Utilities FAQ

Frequently Asked Questions with answers.

Questions

- [15.9.1.1.1: \[423\] Are these utilities present in the community version of MySQL?](#)
- [15.9.1.2.1: \[423\] Can the utilities be used with MyISAM or CSV?](#)
- [15.9.1.3.1: \[423\] Can the .frm reader read a .frm file without the associated data files?](#)
- [15.9.1.3.2: \[423\] Will the .frm reader modify my original .frm file?](#)
- [15.9.1.3.3: \[423\] What is diagnostic mode and why doesn't it produce the same output as the default mode?](#)
- [15.9.1.3.4: \[423\] If the diagnostic mode is only a best-effort compilation, why use it?](#)
- [15.9.1.3.5: \[423\] Why does the default mode require a server?](#)
- [15.9.1.3.6: \[424\] Can the .frm reader read any .frm file?](#)
- [15.9.1.3.7: \[424\] My .frm files are tucked away in a restricted folder. How do I get access to them to run the .frm reader without copying or modifying file privileges?](#)
- [15.9.1.3.8: \[424\] Will the default mode display a 100% accurate CREATE statement?](#)

Questions and Answers

15.9.1.1.1: Are these utilities present in the community version of MySQL?

They are included in the community version of MySQL Workbench, and available from Launchpad.

15.9.1.2.1: Can the utilities be used with MyISAM or CSV?

Yes. There are no storage engine specific limitations in using the utilities. There are some features written specifically for InnoDB so those may not apply but in general no utility is storage engine specific. For example, the `mysqldiskusage` utility shows exact sizes for MyISAM and InnoDB files but uses estimated sizes for any other storage engine based on number of rows and row size.

15.9.1.3.1: Can the .frm reader read a .frm file without the associated data files?

Yes! The .frm reader was designed to read the contents of an .frm file without requiring the data files.

15.9.1.3.2: Will the .frm reader modify my original .frm file?

No, it does not modify the original .frm file in either default or diagnostic mode.

15.9.1.3.3: What is diagnostic mode and why doesn't it produce the same output as the default mode?

The diagnostic mode does not use a spawned server to read the .frm file. Instead, it attempts to read the contents of the file byte-by-byte and forms a best-effort approximation of the CREATE statement. Due to the many complexities of the server code, the diagnostic mode does not currently process all features of a table. Future revisions will improve the accuracy of the diagnostic mode.

15.9.1.3.4: If the diagnostic mode is only a best-effort compilation, why use it?

The diagnostic mode is used to attempt to read corrupt or otherwise damaged .frm files. You would also use it if you had no access to a server installation on the local machine.

15.9.1.3.5: Why does the default mode require a server?

The default mode uses a server to create a temporary working copy of the server instance. It does *not* access the donor server in any way other than to execute the mysqld[.exe] process.

15.9.1.3.6: Can the .frm reader read any .frm file?

While it can read most .frm files, there are known limits to which storage engines it can process correctly. Currently, tables with storage engines partition and performance_schema cannot be read. However, these .frm files can be read by the diagnostic mode,

15.9.1.3.7: My .frm files are tucked away in a restricted folder. How do I get access to them to run the .frm reader without copying or modifying file privileges?

You can use elevated privileges such as su or sudo to execute the .frm reader. You must use the --user option to specify a user to launch the spawned server, however. This will permit the .frm reader to read the original .frm file and copy it to the spawned server and access the copy without requiring additional privileges.

15.9.1.3.8: Will the default mode display a 100% accurate CREATE statement?

For most tables and all views, yes. However, there are at least two features that are not stored in the .frm file and therefore will not be included. These are autoincrement values and foreign keys. That being said, the CREATE statement produced will be syntactically correct.

15.9.2. MySQL Utilities Change History

This appendix lists the changes from version to version in the MySQL Utilities source code.

Note that we tend to update the manual at the same time we make changes to MySQL Utilities. If you find a recent version of the MySQL Utilities listed here that you can't find on our download page (<http://dev.mysql.com/downloads/>), it means that the version has not yet been released.

The date mentioned with a release version is the date of the last Bazaar ChangeSet on which the release was based, not the date when the packages were made available. The binaries are usually made available a few days after the date of the tagged ChangeSet, because building and testing all packages takes some time.

The manual included in the source and binary distributions may not be fully accurate when it comes to the release changelog entries, because the integration of the manual happens at build time. For the most up-to-date release changelog, please refer to the online version instead.

Release notes for the changes in each release of MySQL Utilities are located at [MySQL Utilities Release Notes](#).

Appendix A. Third Party Licenses

Table of Contents

A.1. .NET Flat TabControl License	426
A.2. ANTLR License	427
A.3. Bitstream Vera License	427
A.4. Boost Library License	428
A.5. Cairo License	429
A.6. CTemplate (Google Template System) License	429
A.7. cURL (libcurl) License	430
A.8. DockPanel Suite License	430
A.9. Dojo Toolkit v1.7.0b1 License	431
A.10. GLib License (for MySQL Workbench)	431
A.11. Glitz License	432
A.12. GNU Lesser General Public License Version 2.1, February 1999	432
A.13. HtmlRenderer (System.Drawing.Html)	440
A.14. iODBC License	441
A.15. Libiconv License	442
A.16. Libintl License	442
A.17. Libxml2 License	443
A.18. Libzip License	443
A.19. Lua (liblua) License	444
A.20. Paramiko License	444
A.21. PCRE License	444
A.22. Pixman License	446
A.23. PyCrypto 2.6 License	447
A.24. PyODBC License	449
A.25. PySQLite License	450
A.26. Python License	450
A.27. Scintilla License	460
A.28. ScintillaNET License	462
A.29. SQLCipher License	462
A.30. TinyXML License	463
A.31. TreeViewAdv for .NET License	463
A.32. VSSQLite++ License	464
A.33. zlib License	464

Use of any of this software is governed by the terms of the licenses that follow.

MySQL Workbench 6.0

- [Section A.1, “.NET Flat TabControl License”](#)
- [Section A.2, “ANTLR License”](#)
- [Section A.3, “Bitstream Vera License”](#)
- [Section A.4, “Boost Library License”](#)
- [Section A.5, “Cairo License”](#)
- [Section A.6, “CTemplate \(Google Template System\) License”](#)

- [Section A.7, “cURL \(`libcurl`\) License”](#)
- [Section A.8, “DockPanel Suite License”](#)
- [Section A.9, “Dojo Toolkit v1.7.0b1 License”](#)
- [Section A.10, “GLib License \(for MySQL Workbench\)”](#)
- [Section A.11, “Glitz License”](#)
- [Section A.12, “GNU Lesser General Public License Version 2.1, February 1999”](#)
- [Section A.13, “HtmlRenderer \(System.Drawing.Html\)”](#)
- [Section A.14, “iODBC License”](#)
- [Section A.15, “Libiconv License”](#)
- [Section A.16, “LIBINTL License”](#)
- [Section A.17, “Libxml2 License”](#)
- [Section A.18, “Libzip License”](#)
- [Section A.19, “Lua \(`libleu`\) License”](#)
- [Section A.20, “Paramiko License”](#)
- [Section A.21, “PCRE License”](#)
- [Section A.22, “Pixman License”](#)
- [Section A.23, “PyCrypto 2.6 License”](#)
- [Section A.24, “PyODBC License”](#)
- [Section A.25, “PySQLite License”](#)
- [Section A.26, “Python License”](#)
- [Section A.27, “Scintilla License”](#)
- [Section A.28, “ScintillaNET License”](#)
- [Section A.29, “SQLCipher License”](#)
- [Section A.30, “TinyXML License”](#)
- [Section A.31, “TreeViewAdv for .NET License”](#)
- [Section A.32, “VSQLite++ License”](#)
- [Section A.33, “`zlib` License”](#)

A.1. .NET Flat TabControl License

The following software may be included in this product:

- **.NET Flat TabControl**

Use of any of this software is governed by the terms of the license below:

It is free. Public domain!

Oscar Londono

A.2. ANTLR License

The following software may be included in this product:

ANTLR

This product was build using ANTLR, which was provided to Oracle under the following terms:

Copyright (c) 2010 Terence Parr
All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

Neither the name of the author nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

A.3. Bitstream Vera License

The following software may be included in this product:

Bitstream Vera

Copyright (c) 2003 by Bitstream, Inc. All Rights Reserved. Bitstream Vera is a trademark of Bitstream, Inc.

Permission is hereby granted, free of charge, to any person obtaining a copy of the fonts accompanying this license ("Fonts") and associated documentation files (the "Font Software"), to reproduce and distribute the Font Software, including without limitation the rights to use, copy, merge, publish, distribute, and/or sell copies of the Font Software, and to permit persons to whom the Font Software is furnished to do so, subject to the following conditions:

The above copyright and trademark notices and this permission notice shall be included in all copies of one or more of the Font Software typefaces.

The Font Software may be modified, altered, or added to, and in particular the designs of glyphs or characters

in the Fonts may be modified and additional glyphs or characters may be added to the Fonts, only if the fonts are renamed to names not containing either the words "Bitstream" or the word "Vera".

This License becomes null and void to the extent applicable to Fonts or Font Software that has been modified and is distributed under the "Bitstream Vera" names.

The Font Software may be sold as part of a larger software package but no copy of one or more of the Font Software typefaces may be sold by itself.

THE FONT SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT OF COPYRIGHT, PATENT, TRADEMARK, OR OTHER RIGHT. IN NO EVENT SHALL BITSTREAM OR THE GNOME FOUNDATION BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, INCLUDING ANY GENERAL, SPECIAL, INDIRECT, INCIDENTAL, OR CONSEQUENTIAL DAMAGES, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF THE USE OR INABILITY TO USE THE FONT SOFTWARE OR FROM OTHER DEALINGS IN THE FONT SOFTWARE.

Except as contained in this notice, the names of Gnome, the Gnome Foundation, and Bitstream Inc., shall not be used in advertising or otherwise to promote the sale, use or other dealings in this Font Software without prior written authorization from the Gnome Foundation or Bitstream Inc., respectively. For further information, contact: fonts at gnome dot org.

A.4. Boost Library License

The following software may be included in this product:

Boost C++ Libraries

Use of any of this software is governed by the terms of the license below:

Boost Software License - Version 1.0 - August 17th, 2003

Permission is hereby granted, free of charge, to any person or organization obtaining a copy of the software and accompanying documentation covered by this license (the "Software") to use, reproduce, display, distribute, execute, and transmit the Software, and to prepare derivative works of the Software, and to permit third-parties to whom the Software is furnished to do so, all subject to the following:

The copyright notices in the Software and this entire statement, including the above license grant, this restriction and the following disclaimer, must be included in all copies of the Software, in whole or in part, and all derivative works of the Software, unless such copies or derivative works are solely in the form of machine-executable object code generated by a source language processor.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, TITLE AND NON-INFRINGEMENT. IN NO EVENT SHALL THE COPYRIGHT HOLDERS OR ANYONE

DISTRIBUTING THE SOFTWARE BE LIABLE FOR ANY DAMAGES OR OTHER LIABILITY, WHETHER IN CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

A.5. Cairo License

The following software may be included in this product:

Cairo

You are receiving a copy of the Cairo in both source and object code in the following DLL (libcairo.dll) or dynamic libraries (MySQLWorkbench.app/Contents/Frameworks/libcairo.2.dylib and MySQLWorkbench.app/Contents/Frameworks/libpixman-1.0.dylib). The terms of the Oracle license do NOT apply to Cairo; Oracle distributes it under the GNU Lesser General Public License Version 2.1 separately from the Oracle programs you receive. You can also separately obtain and use Cairo independent of the Oracle programs under a dual license subject to the terms of the LGPL or the Mozilla Public License Version 1.1. If you do not wish to install this program, you may delete libcairo.dll or libcairo.2.dylib and libpixman-1.0.dylib from the installation directory or uninstall MySQL Workbench completely.

This component is licensed under [Section A.12, “GNU Lesser General Public License Version 2.1, February 1999”](#).

A.6. CTemplate (Google Template System) License

The following software may be included in this product:

CTemplate (Google Template System)

Copyright (c) 2005, Google Inc.
All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

- * Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
- * Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
- * Neither the name of Google Inc. nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,

SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

A.7. cURL ([libcurl](#)) License

The following software may be included in this product:

cURL (libcurl)

Use of any of this software is governed by the terms of the license below:

COPYRIGHT AND PERMISSION NOTICE

Copyright (c) 1996 - 2009, Daniel Stenberg, <daniel@haxx.se>. All rights reserved.

Permission to use, copy, modify, and distribute this software for any purpose with or without fee is hereby granted, provided that the above copyright notice and this permission notice appear in all copies.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT OF THIRD PARTY RIGHTS. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

Except as contained in this notice, the name of a copyright holder shall not be used in advertising or otherwise to promote the sale, use or other dealings in this Software without prior written authorization of the copyright holder.

A.8. DockPanel Suite License

The following software may be included in this product:

DockPanel Suite

The MIT License

Copyright (c) 2007 Weifen Luo (email: weifenluo@yahoo.com)

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY

OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

A.9. Dojo Toolkit v1.7.0b1 License

The following software may be included in this product:

Dojo Toolkit v1.7.0b1
Copyright (c) 2005-2006, The Dojo Foundation
All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

- * Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
- * Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
- * Neither the name of the Dojo Foundation nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

A.10. GLib License (for MySQL Workbench)

The following software may be included in this product:

GLib

You are receiving a copy of the GLib library in both source and object code in the following folder: C:\Program Files (x86)\MySQL\MySQLWorkbench 5.2\ on Windows and MySQLWorkbench.app/Contents/Frameworks on Mac OS X. The terms of the Oracle license do NOT apply to the GLib library; it is licensed under the following license, separately from the Oracle programs you receive. If you do not wish to install this library, you may go to the folder C:\Program Files (x86)\MySQL\MySQL Workbench 5.2\ and remove or replace the libglib-2.0-0.dll, libgmodule-2.0-0.dll, libgobject-2.0-0.dll and libgthread-2.0-0.dll files if present on Windows or go to the folder MySQLWorkbench.app/Contents/Frameworks and remove or replace the files libglib-2.*.dylib, libgmodule-2.*.dylib

and libgthread-2.*.dylib on Mac OS X, but the Oracle program might not operate properly or at all without the library.

This component is licensed under [Section A.12, "GNU Lesser General Public License Version 2.1, February 1999".](#)

A.11. Glitz License

The following software may be included in this product:

Glitz

Copyright © 2004 David Reveman, Peter Nilsson

Permission to use, copy, modify, distribute, and sell this software and its documentation for any purpose is hereby granted without fee, provided that the above copyright notice appear in all copies and that both that copyright notice and this permission notice appear in supporting documentation, and that the names of David Reveman and Peter Nilsson not be used in advertising or publicity pertaining to distribution of the software without specific, written prior permission. David Reveman and Peter Nilsson makes no representations about the suitability of this software for any purpose. It is provided "as is" without express or implied warranty.

DAVID REVEMAN AND PETER NILSSON DISCLAIMS ALL WARRANTIES WITH REGARD TO THIS SOFTWARE, INCLUDING ALL IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS, IN NO EVENT SHALL DAVID REVEMAN AND PETER NILSSON BE LIABLE FOR ANY SPECIAL, INDIRECT OR CONSEQUENTIAL DAMAGES OR ANY DAMAGES WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER IN AN ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION, ARISING OUT OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE.

A.12. GNU Lesser General Public License Version 2.1, February 1999

The following applies to all products licensed under the GNU Lesser General Public License, Version 2.1: You may not use the identified files except in compliance with the GNU Lesser General Public License, Version 2.1 (the "License"). You may obtain a copy of the License at <http://www.gnu.org/licenses/lgpl-2.1.html>. A copy of the license is also reproduced below. Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

GNU LESSER GENERAL PUBLIC LICENSE
Version 2.1, February 1999

Copyright (C) 1991, 1999 Free Software Foundation, Inc.
51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA
Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.

[This is the first released version of the Lesser GPL. It also counts as the successor of the GNU Library Public License, version 2, hence the version number 2.1.]

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public Licenses are intended to guarantee your freedom to share and change free software--to make sure the software is free for all its users.

This license, the Lesser General Public License, applies to some specially designated software packages--typically libraries--of the Free Software Foundation and other authors who decide to use it. You can use it too, but we suggest you first think carefully about whether this license or the ordinary General Public License is the better strategy to use in any particular case, based on the explanations below.

When we speak of free software, we are referring to freedom of use, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish); that you receive source code or can get it if you want it; that you can change the software and use pieces of it in new free programs; and that you are informed that you can do these things.

To protect your rights, we need to make restrictions that forbid distributors to deny you these rights or to ask you to surrender these rights. These restrictions translate to certain responsibilities for you if you distribute copies of the library or if you modify it.

For example, if you distribute copies of the library, whether gratis or for a fee, you must give the recipients all the rights that we gave you. You must make sure that they, too, receive or can get the source code. If you link other code with the library, you must provide complete object files to the recipients, so that they can relink them with the library after making changes to the library and recompiling it. And you must show them these terms so they know their rights.

We protect your rights with a two-step method: (1) we copyright the library, and (2) we offer you this license, which gives you legal permission to copy, distribute and/or modify the library.

To protect each distributor, we want to make it very clear that there is no warranty for the free library. Also, if the library is modified by someone else and passed on, the recipients should know that what they have is not the original version, so that the original author's reputation will not be affected by problems that might be introduced by others.

Finally, software patents pose a constant threat to the existence of any free program. We wish to make sure that a company cannot effectively restrict the users of a free program by obtaining a restrictive license from a patent holder. Therefore, we insist that any patent license obtained for a version of the library must be consistent with the full freedom of use specified in this license.

Most GNU software, including some libraries, is covered by the ordinary GNU General Public License. This license, the GNU Lesser General Public License, applies to certain designated libraries, and is quite different from the ordinary General Public License. We use this license for certain libraries in order to permit linking those libraries into non-free programs.

When a program is linked with a library, whether statically or using a shared library, the combination of the two is legally speaking a combined work, a derivative of the original library. The ordinary

General Public License therefore permits such linking only if the entire combination fits its criteria of freedom. The Lesser General Public License permits more lax criteria for linking other code with the library.

We call this license the "Lesser" General Public License because it does Less to protect the user's freedom than the ordinary General Public License. It also provides other free software developers Less of an advantage over competing non-free programs. These disadvantages are the reason we use the ordinary General Public License for many libraries. However, the Lesser license provides advantages in certain special circumstances.

For example, on rare occasions, there may be a special need to encourage the widest possible use of a certain library, so that it becomes a de-facto standard. To achieve this, non-free programs must be allowed to use the library. A more frequent case is that a free library does the same job as widely used non-free libraries. In this case, there is little to gain by limiting the free library to free software only, so we use the Lesser General Public License.

In other cases, permission to use a particular library in non-free programs enables a greater number of people to use a large body of free software. For example, permission to use the GNU C Library in non-free programs enables many more people to use the whole GNU operating system, as well as its variant, the GNU/Linux operating system.

Although the Lesser General Public License is Less protective of the users' freedom, it does ensure that the user of a program that is linked with the Library has the freedom and the wherewithal to run that program using a modified version of the Library.

The precise terms and conditions for copying, distribution and modification follow. Pay close attention to the difference between a "work based on the library" and a "work that uses the library". The former contains code derived from the library, whereas the latter must be combined with the library in order to run.

GNU LESSER GENERAL PUBLIC LICENSE
TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License Agreement applies to any software library or other program which contains a notice placed by the copyright holder or other authorized party saying it may be distributed under the terms of this Lesser General Public License (also called "this License"). Each licensee is addressed as "you".

A "library" means a collection of software functions and/or data prepared so as to be conveniently linked with application programs (which use some of those functions and data) to form executables.

The "Library", below, refers to any such software library or work which has been distributed under these terms. A "work based on the Library" means either the Library or any derivative work under copyright law: that is to say, a work containing the Library or a portion of it, either verbatim or with modifications and/or translated straightforwardly into another language. (Hereinafter, translation is included without limitation in the term "modification".)

"Source code" for a work means the preferred form of the work for making modifications to it. For a library, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the library.

Activities other than copying, distribution and modification are not

covered by this License; they are outside its scope. The act of running a program using the Library is not restricted, and output from such a program is covered only if its contents constitute a work based on the Library (independent of the use of the Library in a tool for writing it). Whether that is true depends on what the Library does and what the program that uses the Library does.

1. You may copy and distribute verbatim copies of the Library's complete source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and distribute a copy of this License along with the Library.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Library or any portion of it, thus forming a work based on the Library, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

a) The modified work must itself be a software library.

b) You must cause the files modified to carry prominent notices stating that you changed the files and the date of any change.

c) You must cause the whole of the work to be licensed at no charge to all third parties under the terms of this License.

d) If a facility in the modified Library refers to a function or a table of data to be supplied by an application program that uses the facility, other than as an argument passed when the facility is invoked, then you must make a good faith effort to ensure that, in the event an application does not supply such function or table, the facility still operates, and performs whatever part of its purpose remains meaningful.

(For example, a function in a library to compute square roots has a purpose that is entirely well-defined independent of the application. Therefore, Subsection 2d requires that any application-supplied function or table used by this function must be optional: if the application does not supply it, the square root function must still compute square roots.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Library, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Library, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Library.

In addition, mere aggregation of another work not based on the Library with the Library (or with a work based on the Library) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may opt to apply the terms of the ordinary GNU General Public License instead of this License to a given copy of the Library. To do this, you must alter all the notices that refer to this License, so that they refer to the ordinary GNU General Public License, version 2, instead of to this License. (If a newer version than version 2 of the ordinary GNU General Public License has appeared, then you can specify that version instead if you wish.) Do not make any other change in these notices.

Once this change is made in a given copy, it is irreversible for that copy, so the ordinary GNU General Public License applies to all subsequent copies and derivative works made from that copy.

This option is useful when you wish to copy part of the code of the Library into a program that is not a library.

4. You may copy and distribute the Library (or a portion or derivative of it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange.

If distribution of object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place satisfies the requirement to distribute the source code, even though third parties are not compelled to copy the source along with the object code.

5. A program that contains no derivative of any portion of the Library, but is designed to work with the Library by being compiled or linked with it, is called a "work that uses the Library". Such a work, in isolation, is not a derivative work of the Library, and therefore falls outside the scope of this License.

However, linking a "work that uses the Library" with the Library creates an executable that is a derivative of the Library (because it contains portions of the Library), rather than a "work that uses the library". The executable is therefore covered by this License. Section 6 states terms for distribution of such executables.

When a "work that uses the Library" uses material from a header file that is part of the Library, the object code for the work may be a derivative work of the Library even though the source code is not. Whether this is true is especially significant if the work can be linked without the Library, or if the work is itself a library. The threshold for this to be true is not precisely defined by law.

If such an object file uses only numerical parameters, data structure layouts and accessors, and small macros and small inline functions (ten lines or less in length), then the use of the object file is unrestricted, regardless of whether it is legally a derivative work. (Executables containing this object code plus portions of the Library will still fall under Section 6.)

Otherwise, if the work is a derivative of the Library, you may distribute the object code for the work under the terms of Section 6. Any executables containing that work also fall under Section 6, whether or not they are linked directly with the Library itself.

6. As an exception to the Sections above, you may also combine or link a "work that uses the Library" with the Library to produce a work containing portions of the Library, and distribute that work under terms of your choice, provided that the terms permit modification of the work for the customer's own use and reverse engineering for debugging such modifications.

You must give prominent notice with each copy of the work that the Library is used in it and that the Library and its use are covered by this License. You must supply a copy of this License. If the work during execution displays copyright notices, you must include the copyright notice for the Library among them, as well as a reference directing the user to the copy of this License. Also, you must do one of these things:

- a) Accompany the work with the complete corresponding machine-readable source code for the Library including whatever changes were used in the work (which must be distributed under Sections 1 and 2 above); and, if the work is an executable linked with the Library, with the complete machine-readable "work that uses the Library", as object code and/or source code, so that the user can modify the Library and then relink to produce a modified executable containing the modified Library. (It is understood that the user who changes the contents of definitions files in the Library will not necessarily be able to recompile the application to use the modified definitions.)
- b) Use a suitable shared library mechanism for linking with the Library. A suitable mechanism is one that (1) uses at run time a copy of the library already present on the user's computer system, rather than copying library functions into the executable, and (2) will operate properly with a modified version of the library, if the user installs one, as long as the modified version is interface-compatible with the version that the work was made with.
- c) Accompany the work with a written offer, valid for at least three years, to give the same user the materials specified in Subsection 6a, above, for a charge no more than the cost of performing this distribution.
- d) If distribution of the work is made by offering access to copy from a designated place, offer equivalent access to copy the above specified materials from the same place.
- e) Verify that the user has already received a copy of these materials or that you have already sent this user a copy.

For an executable, the required form of the "work that uses the Library" must include any data and utility programs needed for reproducing the executable from it. However, as a special exception, the materials to be distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

It may happen that this requirement contradicts the license restrictions of other proprietary libraries that do not normally accompany the operating system. Such a contradiction means you cannot use both them and the Library together in an executable that you distribute.

7. You may place library facilities that are a work based on the Library side-by-side in a single library together with other library facilities not covered by this License, and distribute such a combined library, provided that the separate distribution of the work based on the Library and of the other library facilities is otherwise permitted, and provided that you do these two things:

- a) Accompany the combined library with a copy of the same work based on the Library, uncombined with any other library facilities. This must be distributed under the terms of the Sections above.

b) Give prominent notice with the combined library of the fact that part of it is a work based on the Library, and explaining where to find the accompanying uncombined form of the same work.

8. You may not copy, modify, sublicense, link with, or distribute the Library except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense, link with, or distribute the Library is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

9. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Library or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Library (or any work based on the Library), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Library or works based on it.

10. Each time you redistribute the Library (or any work based on the Library), the recipient automatically receives a license from the original licensor to copy, distribute, link with or modify the Library subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties with this License.

11. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Library at all. For example, if a patent license would not permit royalty-free redistribution of the Library by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Library.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply, and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

12. If the distribution and/or use of the Library is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Library under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among

countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

13. The Free Software Foundation may publish revised and/or new versions of the Lesser General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Library specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Library does not specify a license version number, you may choose any version ever published by the Free Software Foundation.

14. If you wish to incorporate parts of the Library into other free programs whose distribution conditions are incompatible with these, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

15. BECAUSE THE LIBRARY IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE LIBRARY, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE LIBRARY "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE LIBRARY IS WITH YOU. SHOULD THE LIBRARY PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

16. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE LIBRARY AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE LIBRARY (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE LIBRARY TO OPERATE WITH ANY OTHER SOFTWARE), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Libraries

If you develop a new library, and you want it to be of the greatest possible use to the public, we recommend making it free software that everyone can redistribute and change. You can do so by permitting redistribution under these terms (or, alternatively, under the terms of the ordinary General Public License).

To apply these terms, attach the following notices to the library. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

<one line to give the library's name and a brief idea of what it does.>
Copyright (C) <year> <name of author>

This library is free software; you can redistribute it and/or modify it under the terms of the GNU Lesser General Public License as published by the Free Software Foundation; either version 2.1 of the License, or (at your option) any later version.

This library is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU Lesser General Public License for more details.

You should have received a copy of the GNU Lesser General Public License along with this library; if not, write to the Free Software Foundation, Inc., 51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA

Also add information on how to contact you by electronic and paper mail.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a "copyright disclaimer" for the library, if necessary. Here is a sample; alter the names:

Yoyodyne, Inc., hereby disclaims all copyright interest in the library 'Frob' (a library for tweaking knobs) written by James Random Hacker.

<signature of Ty Coon>, 1 April 1990
Ty Coon, President of Vice

That's all there is to it!

A.13. HtmlRenderer (System.Drawing.Html)

The following software may be included in this product:

HtmlRenderer (System.Drawing.Html)

Copyright (c) 2009, José Manuel Menéndez Poo
All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

- * Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
- * Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
- * Neither the name of menendezpoo.com nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED

TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

A.14. iODBC License

The following software may be included in this product:

iODBC

In accordance with the terms of the iODBC licensing scheme set forth below, Oracle is hereby making the election to license iODBC under the BSD license.

iODBC Driver Manager

Copyright (C) 1995 by Ke Jin <kejin@empress.com>

Copyright (C) 1996-2009 by OpenLink Software <iodbc@openlinksw.com>

All Rights Reserved.

This software is released under either the GNU Library General Public License (see LICENSE.LGPL) or the BSD License (see LICENSE.BSD).

Note that the only valid version of the LGPL license as far as this project is concerned is the original GNU Library General Public License Version 2, dated June 1991.

While not mandated by the BSD license, any patches you make to the iODBC may be contributed back into the iODBC project at your discretion. Contributions will benefit the Open Source and Data Access community as a whole. Submissions may be made at <http://www.iodbc.org>.

LICENSE.BSD:

Copyright (C) 1995-2009, OpenLink Software Inc and Ke Jin.
All rights reserved.

.
Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
3. Neither the name of OpenLink Software Inc. nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL OPENLINK OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

A.15. Libiconv License

The following software may be included in this product:

libiconv

You are receiving a copy of the GNU LIBICONV Library. The terms of the Oracle license do NOT apply to the GNU LIBICONV Library; it is licensed under the following license, separately from the Oracle programs you receive. If you do not wish to install this program, you may delete iconv.dll or libiconv.* files.

This component is licensed under [Section A.12, “GNU Lesser General Public License Version 2.1, February 1999”](#).

A.16. Libintl License

The following software may be included in this product:

libintl

Copyright (C) 1994 X Consortium

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE X CONSORTIUM BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

Except as contained in this notice, the name of the X Consortium shall not be used in advertising or otherwise to promote the sale, use or other dealings in this Software without prior written authorization from the X Consortium.

FSF changes to this file are in the public domain.

Copyright 1996-2007 Free Software Foundation, Inc. Taken from GNU libtool, 2001

Originally by Gordon Matzigkeit <gord@gnu.ai.mit.edu>, 1996

This file is free software; the Free Software Foundation gives unlimited permission to copy and/or distribute it, with or without modifications, as long as this notice is preserved.

You are receiving a copy of the libintl library. The terms of the Oracle license do NOT apply to the libintl library; it is licensed under the following license, separately from the Oracle programs you receive. If you do not wish to install this program, you may delete the intl.dll or libintl.* files.

This component is licensed under [Section A.12, “GNU Lesser General Public License Version 2.1, February 1999”](#).

A.17. Libxml2 License

The following software may be included in this product:

Libxml2

Except where otherwise noted in the source code (e.g. the files hash.c, list.c and the trio files, which are covered by a similar licence but with different Copyright notices) all the files are:

Copyright (C) 1998-2003 Daniel Veillard. All Rights Reserved.

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE DANIEL VEILLARD BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

Except as contained in this notice, the name of Daniel Veillard shall not be used in advertising or otherwise to promote the sale, use or other dealings in this Software without prior written authorization from him.

A.18. Libzip License

The following software may be included in this product:

libzip

Copyright (C) 1999-2008 Dieter Baron and Thomas Klausner
The authors can be contacted at <libzip@nih.at>

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
3. The names of the authors may not be used to endorse or promote products derived from this software without specific prior

written permission.

THIS SOFTWARE IS PROVIDED BY THE AUTHORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

A.19. Lua (liblua) License

The following software may be included in this product:

Lua (liblua)

Copyright © 1994-2008 Lua.org, PUC-Rio.

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

A.20. Paramiko License

The following software may be included in this product:

Paramiko

You are receiving a copy of Paramiko in both source and object code. The terms of the Oracle license do NOT apply to the Paramiko program; it is licensed under the following license, separately from the Oracle programs you receive. If you do not wish to install this program, you may delete the Paramiko folder and all its contents.

This component is licensed under [Section A.12, "GNU Lesser General Public License Version 2.1, February 1999".](#)

A.21. PCRE License

The following software may be included in this product:

PCRE (Perl Compatible Regular Expressions) Library

PCRE LICENCE

PCRE is a library of functions to support regular expressions whose syntax and semantics are as close as possible to those of the Perl 5 language.

Release 7 of PCRE is distributed under the terms of the "BSD" licence, as specified below. The documentation for PCRE, supplied in the "doc" directory, is distributed under the same terms as the software itself.

The basic library functions are written in C and are freestanding. Also included in the distribution is a set of C++ wrapper functions.

THE BASIC LIBRARY FUNCTIONS

Written by: Philip Hazel
Email local part: ph10
Email domain: cam.ac.uk

University of Cambridge Computing Service,
Cambridge, England. Phone: +44 1223 334714.

Copyright (c) 1997-2006 University of Cambridge
All rights reserved.

THE C++ WRAPPER FUNCTIONS

Contributed by: Google Inc.

Copyright (c) 2006, Google Inc.
All rights reserved.

THE "BSD" LICENCE

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

- * Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
- * Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
- * Neither the name of the University of Cambridge nor the name of Google Inc. nor the names of their contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF

THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

End

A.22. Pixman License

The following software may be included in this product:

Pixman

Pixman v0.21.2

The following is the MIT license, agreed upon by most contributors. Copyright holders of new code should use this license statement where possible. They may also add themselves to the list below.

Copyright 1987, 1988, 1989, 1998 The Open Group
Copyright 1987, 1988, 1989 Digital Equipment Corporation
Copyright 1999, 2004, 2008 Keith Packard
Copyright 2000 SuSE, Inc.
Copyright 2000 Keith Packard, member of The XFree86 Project, Inc.
Copyright 2004, 2005, 2007, 2008, 2009, 2010 Red Hat, Inc.
Copyright 2004 Nicholas Miell
Copyright 2005 Lars Knoll & Zack Rusin, Trolltech
Copyright 2005 Trolltech AS
Copyright 2007 Luca Barbato
Copyright 2008 Aaron Plattner, NVIDIA Corporation
Copyright 2008 Rodrigo Kumpera
Copyright 2008 André Tupinambá
Copyright 2008 Mozilla Corporation
Copyright 2008 Frederic Plourde
Copyright 2009, Oracle and/or its affiliates. All rights reserved.

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice (including the next paragraph) shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

Pixman v0.17.4 and lower:

The following is the 'standard copyright' agreed upon by most contributors, and is currently the canonical license, though a modification is currently under discussion. Copyright holders of new code should use this license statement where possible, and append their name to this list.

Copyright 1987, 1988, 1989, 1998 The Open Group

```
Copyright 1987, 1988, 1989 Digital Equipment Corporation
Copyright 1999, 2004, 2008 Keith Packard
Copyright 2000 SuSE, Inc.
Copyright 2000 Keith Packard, member of The XFree86 Project, Inc.
Copyright 2004, 2005, 2007, 2008 Red Hat, Inc.
Copyright 2004 Nicholas Miell
Copyright 2005 Lars Knoll & Zack Rusin, Trolltech
Copyright 2005 Trolltech AS
Copyright 2007 Luca Barbato
Copyright 2008 Aaron Plattner, NVIDIA Corporation
Copyright 2008 Rodrigo Kumpera
Copyright 2008 AndrÃ© TupinambÃ¡;
Copyright 2008 Mozilla Corporation
Copyright 2008 Frederic Plourde
Copyright 2009 Sun Microsystems, Inc.
```

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice (including the next paragraph) shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

A.23. PyCrypto 2.6 License

The following software may be included in this product:

```
Copyright and licensing of the Python Cryptography Toolkit ("PyCrypto"):
```

Previously, the copyright and/or licensing status of the Python Cryptography Toolkit ("PyCrypto") had been somewhat ambiguous. The original intention of Andrew M. Kuchling and other contributors has been to dedicate PyCrypto to the public domain, but that intention was not necessarily made clear in the original disclaimer (see `LEGAL/copy/LICENSE.orig`).

Additionally, some files within PyCrypto had specified their own licenses that differed from the PyCrypto license itself. For example, the original RIPEMD.c module simply had a copyright statement and warranty disclaimer, without clearly specifying any license terms. (An updated version on the author's website came with a license that contained a GPL-incompatible advertising clause.)

To rectify this situation for PyCrypto 2.1, the following steps have been taken:

1. Obtaining explicit permission from the original contributors to dedicate their contributions to the public domain if they have not already done so.

(See the "LEGAL/copy/stmts" directory for contributors' statements.)

2. Replacing some modules with clearly-licensed code from other sources (e.g. the DES and DES3 modules were replaced with new ones based on Tom St. Denis's public-domain LibTomCrypt library.)

3. Replacing some modules with code written from scratch (e.g. the RIPEMD and Blowfish modules were re-implemented from their respective algorithm specifications without reference to the old implementations).

4. Removing some modules altogether without replacing them.

To the best of our knowledge, with the exceptions noted below or within the files themselves, the files that constitute PyCrypto are in the public domain. Most are distributed with the following notice:

The contents of this file are dedicated to the public domain. To the extent that dedication to the public domain is not available, everyone is granted a worldwide, perpetual, royalty-free, non-exclusive license to exercise all rights associated with the contents of this file for any purpose whatsoever. No rights are reserved.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

Exception:

- Portions of HMAC.py and setup.py are derived from Python 2.2, and are therefore Copyright (c) 2001, 2002, 2003 Python Software Foundation (All Rights Reserved). They are licensed by the PSF under the terms of the Python 2.2 license. (See the file `LEGAL/copy/LICENSE.python-2.2` for details.)

EXPORT RESTRICTIONS:

Note that the export or re-export of cryptographic software and/or source code may be subject to regulation in your jurisdiction.

```
=====
# HMAC.py - Implements the HMAC algorithm as described by RFC 2104.
#
# =====
# Portions Copyright (c) 2001, 2002, 2003 Python Software Foundation;
# All Rights Reserved
#
# This file contains code from the Python 2.2 hmac.py module (the
# "Original Code"), with modifications made after it was incorporated
# into PyCrypto (the "Modifications").
#
# To the best of our knowledge, the Python Software Foundation is the
# copyright holder of the Original Code, and has licensed it under the
# Python 2.2 license. See the file LEGAL/copy/LICENSE.python-2.2 for
# details.
#
# The Modifications to this file are dedicated to the public domain.
# To the extent that dedication to the public domain is not available,
# everyone is granted a worldwide, perpetual, royalty-free,
# non-exclusive license to exercise all rights associated with the
# contents of this file for any purpose whatsoever. No rights are
# reserved.
#
# THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND,
```

```
# EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF
# MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND
# NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS
# BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN
# ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN
# CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE
# SOFTWARE.
# =====

#!/usr/bin/env python
#
# setup.py : Distutils setup script
#
# Part of the Python Cryptography Toolkit
#
# =====
# Portions Copyright (c) 2001, 2002, 2003 Python Software Foundation;
# All Rights Reserved
#
# This file contains code from the Python 2.2 setup.py module (the
# "Original Code"), with modifications made after it was incorporated
# into PyCrypto (the "Modifications").
#
# To the best of our knowledge, the Python Software Foundation is the
# copyright holder of the Original Code, and has licensed it under the
# Python 2.2 license. See the file LEGAL/copy/LICENSE.python-2.2 for
# details.
#
# The Modifications to this file are dedicated to the public domain.
# To the extent that dedication to the public domain is not available,
# everyone is granted a worldwide, perpetual, royalty-free,
# non-exclusive license to exercise all rights associated with the
# contents of this file for any purpose whatsoever. No rights are
# reserved.
#
# THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND,
# EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF
# MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND
# NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS
# BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN
# ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN
# CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE
# SOFTWARE.
# =====
```

A.24. PyODBC License

The following software may be included in this product:

PyODBC

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

A.25. PySQLite License

This product uses `pysqlite 2.6.3` Copyright (c) 2004-2007 Gerhard Haering

A.26. Python License

The following software may be included in this product:

Python Programming Language

This is the official license for the Python 2.7 release:

A. HISTORY OF THE SOFTWARE

Python was created in the early 1990s by Guido van Rossum at Stichting Mathematisch Centrum (CWI, see <http://www.cwi.nl>) in the Netherlands as a successor of a language called ABC. Guido remains Python's principal author, although it includes many contributions from others.

In 1995, Guido continued his work on Python at the Corporation for National Research Initiatives (CNRI, see <http://www.cnri.reston.va.us>) in Reston, Virginia where he released several versions of the software.

In May 2000, Guido and the Python core development team moved to BeOpen.com to form the BeOpen PythonLabs team. In October of the same year, the PythonLabs team moved to Digital Creations (now Zope Corporation, see <http://www.zope.com>). In 2001, the Python Software Foundation (PSF, see <http://www.python.org/psf/>) was formed, a non-profit organization created specifically to own Python-related Intellectual Property. Zope Corporation is a sponsoring member of the PSF.

All Python releases are Open Source (see <http://www.opensource.org> for the Open Source Definition). Historically, most, but not all, Python releases have also been GPL-compatible; the table below summarizes the various releases.

Release	Derived from	Year	Owner	GPL- compatible? (1)
0.9.0 thru 1.2		1991-1995	CWI	yes
1.3 thru 1.5.2	1.2	1995-1999	CNRI	yes
1.6	1.5.2	2000	CNRI	no
2.0	1.6	2000	BeOpen.com	no
1.6.1	1.6	2001	CNRI	yes (2)
2.1	2.0+1.6.1	2001	PSF	no
2.0.1	2.0+1.6.1	2001	PSF	yes
2.1.1	2.1+2.0.1	2001	PSF	yes
2.2	2.1.1	2001	PSF	yes
2.1.2	2.1.1	2002	PSF	yes
2.1.3	2.1.2	2002	PSF	yes
2.2.1	2.2	2002	PSF	yes
2.2.2	2.2.1	2002	PSF	yes
2.2.3	2.2.2	2003	PSF	yes
2.3	2.2.2	2002-2003	PSF	yes
2.3.1	2.3	2002-2003	PSF	yes
2.3.2	2.3.1	2002-2003	PSF	yes
2.3.3	2.3.2	2002-2003	PSF	yes
2.3.4	2.3.3	2004	PSF	yes
2.3.5	2.3.4	2005	PSF	yes
2.4	2.3	2004	PSF	yes
2.4.1	2.4	2005	PSF	yes
2.4.2	2.4.1	2005	PSF	yes
2.4.3	2.4.2	2006	PSF	yes

2.5	2.4	2006	PSF	yes
2.5.1	2.5	2007	PSF	yes
2.5.2	2.5.1	2008	PSF	yes
2.5.3	2.5.2	2008	PSF	yes
2.6	2.5	2008	PSF	yes
2.6.1	2.6	2008	PSF	yes
2.6.2	2.6.1	2009	PSF	yes
2.6.3	2.6.2	2009	PSF	yes
2.6.4	2.6.3	2010	PSF	yes
2.7	2.6	2010	PSF	yes

Footnotes:

- (1) GPL-compatible doesn't mean that we're distributing Python under the GPL. All Python licenses, unlike the GPL, let you distribute a modified version without making your changes open source. The GPL-compatible licenses make it possible to combine Python with other software that is released under the GPL; the others don't.
- (2) According to Richard Stallman, 1.6.1 is not GPL-compatible, because its license has a choice of law clause. According to CNRI, however, Stallman's lawyer has told CNRI's lawyer that 1.6.1 is "not incompatible" with the GPL.

Thanks to the many outside volunteers who have worked under Guido's direction to make these releases possible.

B. TERMS AND CONDITIONS FOR ACCESSING OR OTHERWISE USING PYTHON

PYTHON SOFTWARE FOUNDATION LICENSE VERSION 2

1. This LICENSE AGREEMENT is between the Python Software Foundation ("PSF"), and the Individual or Organization ("Licensee") accessing and otherwise using this software ("Python") in source or binary form and its associated documentation.
2. Subject to the terms and conditions of this License Agreement, PSF hereby grants Licensee a nonexclusive, royalty-free, world-wide license to reproduce, analyze, test, perform and/or display publicly, prepare derivative works, distribute, and otherwise use Python alone or in any derivative version, provided, however, that PSF's License Agreement and PSF's notice of copyright, i.e., "Copyright (c) 2001, 2002, 2003, 2004, 2005, 2006 Python Software Foundation; All Rights Reserved" are retained in Python alone or in any derivative version prepared by Licensee.
3. In the event Licensee prepares a derivative work that is based on or incorporates Python or any part thereof, and wants to make the derivative work available to others as provided herein, then Licensee hereby agrees to include in any such work a brief summary of the changes made to Python.
4. PSF is making Python available to Licensee on an "AS IS" basis. PSF MAKES NO REPRESENTATIONS OR WARRANTIES, EXPRESS OR IMPLIED. BY WAY OF EXAMPLE, BUT NOT LIMITATION, PSF MAKES NO AND DISCLAIMS ANY REPRESENTATION OR WARRANTY OF MERCHANTABILITY OR FITNESS FOR ANY PARTICULAR PURPOSE OR THAT THE USE OF PYTHON WILL NOT INFRINGE ANY THIRD PARTY RIGHTS.
5. PSF SHALL NOT BE LIABLE TO LICENSEE OR ANY OTHER USERS OF PYTHON FOR ANY INCIDENTAL, SPECIAL, OR CONSEQUENTIAL DAMAGES OR LOSS AS A RESULT OF MODIFYING, DISTRIBUTING, OR OTHERWISE USING PYTHON, OR ANY DERIVATIVE THEREOF, EVEN IF ADVISED OF THE POSSIBILITY THEREOF.
6. This License Agreement will automatically terminate upon a material

breach of its terms and conditions.

7. Nothing in this License Agreement shall be deemed to create any relationship of agency, partnership, or joint venture between PSF and Licensee. This License Agreement does not grant permission to use PSF trademarks or trade name in a trademark sense to endorse or promote products or services of Licensee, or any third party.

8. By copying, installing or otherwise using Python, Licensee agrees to be bound by the terms and conditions of this License Agreement.

BOPEN.COM LICENSE AGREEMENT FOR PYTHON 2.0

BOPEN PYTHON OPEN SOURCE LICENSE AGREEMENT VERSION 1

1. This LICENSE AGREEMENT is between BeOpen.com ("BeOpen"), having an office at 160 Saratoga Avenue, Santa Clara, CA 95051, and the Individual or Organization ("Licensee") accessing and otherwise using this software in source or binary form and its associated documentation ("the Software").

2. Subject to the terms and conditions of this BeOpen Python License Agreement, BeOpen hereby grants Licensee a non-exclusive, royalty-free, world-wide license to reproduce, analyze, test, perform and/or display publicly, prepare derivative works, distribute, and otherwise use the Software alone or in any derivative version, provided, however, that the BeOpen Python License is retained in the Software, alone or in any derivative version prepared by Licensee.

3. BeOpen is making the Software available to Licensee on an "AS IS" basis. BEOPEN MAKES NO REPRESENTATIONS OR WARRANTIES, EXPRESS OR IMPLIED. BY WAY OF EXAMPLE, BUT NOT LIMITATION, BEOPEN MAKES NO AND DISCLAIMS ANY REPRESENTATION OR WARRANTY OF MERCHANTABILITY OR FITNESS FOR ANY PARTICULAR PURPOSE OR THAT THE USE OF THE SOFTWARE WILL NOT INFRINGE ANY THIRD PARTY RIGHTS.

4. BEOPEN SHALL NOT BE LIABLE TO LICENSEE OR ANY OTHER USERS OF THE SOFTWARE FOR ANY INCIDENTAL, SPECIAL, OR CONSEQUENTIAL DAMAGES OR LOSS AS A RESULT OF USING, MODIFYING OR DISTRIBUTING THE SOFTWARE, OR ANY DERIVATIVE THEREOF, EVEN IF ADVISED OF THE POSSIBILITY THEREOF.

5. This License Agreement will automatically terminate upon a material breach of its terms and conditions.

6. This License Agreement shall be governed by and interpreted in all respects by the law of the State of California, excluding conflict of law provisions. Nothing in this License Agreement shall be deemed to create any relationship of agency, partnership, or joint venture between BeOpen and Licensee. This License Agreement does not grant permission to use BeOpen trademarks or trade names in a trademark sense to endorse or promote products or services of Licensee, or any third party. As an exception, the "BeOpen Python" logos available at <http://www.pythonlabs.com/logos.html> may be used according to the permissions granted on that web page.

7. By copying, installing or otherwise using the software, Licensee agrees to be bound by the terms and conditions of this License Agreement.

CNRI LICENSE AGREEMENT FOR PYTHON 1.6.1

1. This LICENSE AGREEMENT is between the Corporation for National

Python License

Research Initiatives, having an office at 1895 Preston White Drive, Reston, VA 20191 ("CNRI"), and the Individual or Organization ("Licensee") accessing and otherwise using Python 1.6.1 software in source or binary form and its associated documentation.

2. Subject to the terms and conditions of this License Agreement, CNRI hereby grants Licensee a nonexclusive, royalty-free, world-wide license to reproduce, analyze, test, perform and/or display publicly, prepare derivative works, distribute, and otherwise use Python 1.6.1 alone or in any derivative version, provided, however, that CNRI's License Agreement and CNRI's notice of copyright, i.e., "Copyright (c) 1995-2001 Corporation for National Research Initiatives; All Rights Reserved" are retained in Python 1.6.1 alone or in any derivative version prepared by Licensee. Alternately, in lieu of CNRI's License Agreement, Licensee may substitute the following text (omitting the quotes): "Python 1.6.1 is made available subject to the terms and conditions in CNRI's License Agreement. This Agreement together with Python 1.6.1 may be located on the Internet using the following unique, persistent identifier (known as a handle): 1895.22/1013. This Agreement may also be obtained from a proxy server on the Internet using the following URL: <http://hdl.handle.net/1895.22/1013>".

3. In the event Licensee prepares a derivative work that is based on or incorporates Python 1.6.1 or any part thereof, and wants to make the derivative work available to others as provided herein, then Licensee hereby agrees to include in any such work a brief summary of the changes made to Python 1.6.1.

4. CNRI is making Python 1.6.1 available to Licensee on an "AS IS" basis. CNRI MAKES NO REPRESENTATIONS OR WARRANTIES, EXPRESS OR IMPLIED. BY WAY OF EXAMPLE, BUT NOT LIMITATION, CNRI MAKES NO AND DISCLAIMS ANY REPRESENTATION OR WARRANTY OF MERCHANTABILITY OR FITNESS FOR ANY PARTICULAR PURPOSE OR THAT THE USE OF PYTHON 1.6.1 WILL NOT INFRINGE ANY THIRD PARTY RIGHTS.

5. CNRI SHALL NOT BE LIABLE TO LICENSEE OR ANY OTHER USERS OF PYTHON 1.6.1 FOR ANY INCIDENTAL, SPECIAL, OR CONSEQUENTIAL DAMAGES OR LOSS AS A RESULT OF MODIFYING, DISTRIBUTING, OR OTHERWISE USING PYTHON 1.6.1, OR ANY DERIVATIVE THEREOF, EVEN IF ADVISED OF THE POSSIBILITY THEREOF.

6. This License Agreement will automatically terminate upon a material breach of its terms and conditions.

7. This License Agreement shall be governed by the federal intellectual property law of the United States, including without limitation the federal copyright law, and, to the extent such U.S. federal law does not apply, by the law of the Commonwealth of Virginia, excluding Virginia's conflict of law provisions. Notwithstanding the foregoing, with regard to derivative works based on Python 1.6.1 that incorporate non-separable material that was previously distributed under the GNU General Public License (GPL), the law of the Commonwealth of Virginia shall govern this License Agreement only as to issues arising under or with respect to Paragraphs 4, 5, and 7 of this License Agreement. Nothing in this License Agreement shall be deemed to create any relationship of agency, partnership, or joint venture between CNRI and Licensee. This License Agreement does not grant permission to use CNRI trademarks or trade name in a trademark sense to endorse or promote products or services of Licensee, or any third party.

8. By clicking on the "ACCEPT" button where indicated, or by copying, installing or otherwise using Python 1.6.1, Licensee agrees to be bound by the terms and conditions of this License Agreement.

ACCEPT

CWI LICENSE AGREEMENT FOR PYTHON 0.9.0 THROUGH 1.2

Copyright (c) 1991 - 1995, Stichting Mathematisch Centrum Amsterdam,
The Netherlands. All rights reserved.

Permission to use, copy, modify, and distribute this software and its
documentation for any purpose and without fee is hereby granted,
provided that the above copyright notice appear in all copies and that
both that copyright notice and this permission notice appear in
supporting documentation, and that the name of Stichting Mathematisch
Centrum or CWI not be used in advertising or publicity pertaining to
distribution of the software without specific, written prior
permission.

STICHTING MATHEMATISCH CENTRUM DISCLAIMS ALL WARRANTIES WITH REGARD TO
THIS SOFTWARE, INCLUDING ALL IMPLIED WARRANTIES OF MERCHANTABILITY AND
FITNESS, IN NO EVENT SHALL STICHTING MATHEMATISCH CENTRUM BE LIABLE
FOR ANY SPECIAL, INDIRECT OR CONSEQUENTIAL DAMAGES OR ANY DAMAGES
WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER IN AN
ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION, ARISING OUT
OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE.

Licenses and Acknowledgements for Incorporated Software

This section is an incomplete, but growing list of licenses and acknowledgements
for third-party software incorporated in the Python distribution.

Mersenne Twister

The `_random` module includes code based on a download from
<http://www.math.keio.ac.jp/matumoto/MT2002/emt19937ar.html>.
The following are the verbatim comments from the original code:

A C-program for MT19937, with initialization improved 2002/1/26.
Coded by Takuji Nishimura and Makoto Matsumoto.

Before using, initialize the state by using `init_genrand(seed)`
or `init_by_array(init_key, key_length)`.

Copyright (C) 1997 - 2002, Makoto Matsumoto and Takuji Nishimura,
All rights reserved.

Redistribution and use in source and binary forms, with or without
modification, are permitted provided that the following conditions
are met:

1. Redistributions of source code must retain the above copyright
notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright
notice, this list of conditions and the following disclaimer in the
documentation and/or other materials provided with the
distribution.
3. The names of its contributors may not be used to endorse or promote
products derived from this software without specific prior written
permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
"AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR
CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL,
EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO,
PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR

PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Any feedback is very welcome.
<http://www.math.keio.ac.jp/matumoto/emt.html>
email: matumoto@math.keio.ac.jp

Sockets

=====

The socket module uses the functions, getaddrinfo(), and getnameinfo(), which are coded in separate source files from the WIDE Project, <http://www.wide.ad.jp/>.

Copyright (C) 1995, 1996, 1997, and 1998 WIDE Project.
All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
3. Neither the name of the project nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Floating point exception control

=====

The source for the fpectl module includes the following notice:

/ Copyright (c) 1996.
The Regents of the University of California.
All rights reserved.
\\

Permission to use, copy, modify, and distribute this software for any purpose without fee is hereby granted, provided that this entire notice is included in all copies of any software which is or includes a copy or modification of this software and in all copies of the supporting documentation for such software.

This work was produced at the University of California, Lawrence Livermore National Laboratory under contract no. W-7405-ENG-48 between the U.S. Department of Energy and The Regents of the University of California for the operation of UC LLNL.

DISCLAIMER

This software was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor the University of California nor any of their employees, makes any warranty, express or implied, or assumes any

```
| liability or responsibility for the accuracy, completeness, or  
| usefulness of any information, apparatus, product, or process  
| disclosed, or represents that its use would not infringe  
| privately-owned rights. Reference herein to any specific commer-  
| cial products, process, or service by trade name, trademark,  
| manufacturer, or otherwise, does not necessarily constitute or  
| imply its endorsement, recommendation, or favoring by the United  
| States Government or the University of California. The views and  
| opinions of authors expressed herein do not necessarily state or  
| reflect those of the United States Government or the University  
| of California, and shall not be used for advertising or product  
\\ endorsement purposes.
```

MD5 message digest algorithm

=====

The source code for the md5 module contains the following notice:

Copyright (C) 1999, 2002 Aladdin Enterprises. All rights reserved.

This software is provided 'as-is', without any express or implied
warranty. In no event will the authors be held liable for any damages
arising from the use of this software.

Permission is granted to anyone to use this software for any purpose,
including commercial applications, and to alter it and redistribute it
freely, subject to the following restrictions:

1. The origin of this software must not be misrepresented; you must not
claim that you wrote the original software. If you use this software
in a product, an acknowledgment in the product documentation would
be appreciated but is not required.
2. Altered source versions must be plainly marked as such, and must not
be misrepresented as being the original software.
3. This notice may not be removed or altered from any source
distribution.

L. Peter Deutsch
ghost@aladdin.com

Independent implementation of MD5 (RFC 1321).

This code implements the MD5 Algorithm defined in RFC 1321, whose
text is available at

<http://www.ietf.org/rfc/rfc1321.txt>

The code is derived from the text of the RFC, including the test suite
(section A.5) but excluding the rest of Appendix A. It does not include
any code or documentation that is identified in the RFC as being
copyrighted.

The original and principal author of md5.h is L. Peter Deutsch
<ghost@aladdin.com>. Other authors are noted in the change history
that follows (in reverse chronological order):

2002-04-13 lpd Removed support for non-ANSI compilers; removed
references to Ghostscript; clarified derivation from RFC 1321;
now handles byte order either statically or dynamically.
1999-11-04 lpd Edited comments slightly for automatic TOC extraction.
1999-10-18 lpd Fixed typo in header comment (ansi2knr rather than md5);
added conditionalization for C++ compilation from Martin
Purschke <purschke@bnl.gov>.
1999-05-03 lpd Original version.

Asynchronous socket services

=====

The asynchat and asyncore modules contain the following notice:

Copyright 1996 by Sam Rushing

All Rights Reserved

Permission to use, copy, modify, and distribute this software and its documentation for any purpose and without fee is hereby granted, provided that the above copyright notice appear in all copies and that both that copyright notice and this permission notice appear in supporting documentation, and that the name of Sam Rushing not be used in advertising or publicity pertaining to distribution of the software without specific, written prior permission.

SAM RUSHING DISCLAIMS ALL WARRANTIES WITH REGARD TO THIS SOFTWARE, INCLUDING ALL IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS, IN NO EVENT SHALL SAM RUSHING BE LIABLE FOR ANY SPECIAL, INDIRECT OR CONSEQUENTIAL DAMAGES OR ANY DAMAGES WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER IN AN ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION, ARISING OUT OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE.

Cookie management

=====

The Cookie module contains the following notice:

Copyright 2000 by Timothy O'Malley <timo@alum.mit.edu>

All Rights Reserved

Permission to use, copy, modify, and distribute this software and its documentation for any purpose and without fee is hereby granted, provided that the above copyright notice appear in all copies and that both that copyright notice and this permission notice appear in supporting documentation, and that the name of Timothy O'Malley not be used in advertising or publicity pertaining to distribution of the software without specific, written prior permission.

Timothy O'Malley DISCLAIMS ALL WARRANTIES WITH REGARD TO THIS SOFTWARE, INCLUDING ALL IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS, IN NO EVENT SHALL Timothy O'Malley BE LIABLE FOR ANY SPECIAL, INDIRECT OR CONSEQUENTIAL DAMAGES OR ANY DAMAGES WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER IN AN ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION, ARISING OUT OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE.

Profiling

=====

The profile and pstats modules contain the following notice:

Copyright 1994, by InfoSeek Corporation, all rights reserved.
Written by James Roskind

Permission to use, copy, modify, and distribute this Python software and its associated documentation for any purpose (subject to the restriction in the following sentence) without fee is hereby granted, provided that the above copyright notice appears in all copies, and that both that copyright notice and this permission notice appear in supporting documentation, and that the name of InfoSeek not be used in advertising or publicity pertaining to distribution of the software without specific, written prior permission. This permission is explicitly restricted to the copying and modification of the software to remain in Python, compiled Python, or other languages (such as C) wherein the modified or derived code is exclusively imported into a Python module.

INFOSEEK CORPORATION DISCLAIMS ALL WARRANTIES WITH REGARD TO THIS SOFTWARE, INCLUDING ALL IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS. IN NO EVENT SHALL INFOSEEK CORPORATION BE LIABLE FOR ANY SPECIAL, INDIRECT OR CONSEQUENTIAL DAMAGES OR ANY DAMAGES WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER IN AN ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION, ARISING OUT OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE.

Execution tracing

The trace module contains the following notice:

portions copyright 2001, Autonomous Zones Industries, Inc., all rights...
err... reserved and offered to the public under the terms of the
Python 2.2 license.

Author: Zooko O'Whielacronx
<http://zooko.com/>
mailto:zooko@zooko.com

Copyright 2000, Mojam Media, Inc., all rights reserved.
Author: Skip Montanaro

Copyright 1999, Bioreason, Inc., all rights reserved.
Author: Andrew Dalke

Copyright 1995-1997, Automatrix, Inc., all rights reserved.
Author: Skip Montanaro

Copyright 1991-1995, Stichting Mathematisch Centrum, all rights reserved.

Permission to use, copy, modify, and distribute this Python software and its associated documentation for any purpose without fee is hereby granted, provided that the above copyright notice appears in all copies, and that both that copyright notice and this permission notice appear in supporting documentation, and that the name of neither Automatrix, Bioreason or Mojam Media be used in advertising or publicity pertaining to distribution of the software without specific, written prior permission.

UUencode and UUdecode functions

The uu module contains the following notice:

Copyright 1994 by Lance Ellinghouse
Cathedral City, California Republic, United States of America.

All Rights Reserved

Permission to use, copy, modify, and distribute this software and its documentation for any purpose and without fee is hereby granted, provided that the above copyright notice appear in all copies and that both that copyright notice and this permission notice appear in supporting documentation, and that the name of Lance Ellinghouse not be used in advertising or publicity pertaining to distribution of the software without specific, written prior permission.

LANCE ELLINGHOUSE DISCLAIMS ALL WARRANTIES WITH REGARD TO THIS SOFTWARE, INCLUDING ALL IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS, IN NO EVENT SHALL LANCE ELLINGHOUSE CENTRUM BE LIABLE FOR ANY SPECIAL, INDIRECT OR CONSEQUENTIAL DAMAGES OR ANY DAMAGES WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER IN AN ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION, ARISING OUT OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE.

Modified by Jack Jansen, CWI, July 1995:

- Use binascii module to do the actual line-by-line conversion between ascii and binary. This results in a 1000-fold speedup. The C version is still 5 times faster, though.
- Arguments more compliant with Python standard

XML Remote Procedure Calls¶

The `xmlrpclib` module contains the following notice:

The XML-RPC client interface is

Copyright (c) 1999-2002 by Secret Labs AB
Copyright (c) 1999-2002 by Fredrik Lundh

By obtaining, using, and/or copying this software and/or its associated documentation, you agree that you have read, understood, and will comply with the following terms and conditions:

Permission to use, copy, modify, and distribute this software and its associated documentation for any purpose and without fee is hereby granted, provided that the above copyright notice appears in all copies, and that both that copyright notice and this permission notice appear in supporting documentation, and that the name of Secret Labs AB or the author not be used in advertising or publicity pertaining to distribution of the software without specific, written prior permission.

SECRET LABS AB AND THE AUTHOR DISCLAIMS ALL WARRANTIES WITH REGARD TO THIS SOFTWARE, INCLUDING ALL IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS. IN NO EVENT SHALL SECRET LABS AB OR THE AUTHOR BE LIABLE FOR ANY SPECIAL, INDIRECT OR CONSEQUENTIAL DAMAGES OR ANY DAMAGES WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER IN AN ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION, ARISING OUT OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE.

test_epoll

=====

The `test_epoll` contains the following notice:

Copyright (c) 2001-2006 Twisted Matrix Laboratories.

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

Select kqueue

=====

The `select` and `kqueue` contains the following notice for the `kqueue` interface:

Copyright (c) 2000 Doug White, 2006 James Knight, 2007 Christian Heimes
All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright

```
    notice, this list of conditions and the following disclaimer.  
2. Redistributions in binary form must reproduce the above copyright  
    notice, this list of conditions and the following disclaimer in the  
    documentation and/or other materials provided with the distribution.
```

THIS SOFTWARE IS PROVIDED BY THE AUTHOR AND CONTRIBUTORS ``AS IS'' AND
ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE
ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHOR OR CONTRIBUTORS BE LIABLE
FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL
DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS
OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION)
HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT
LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY
OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF
SUCH DAMAGE.

strtod and dtoa

=====

The file Python/dtoa.c, which supplies C functions dtoa and strtod for conversion
of C doubles to and from strings, is derived from the file of the same name by
David M. Gay, currently available from <http://www.netlib.org/fp/>. The original
file, as retrieved on March 16, 2009, contains the following copyright and
licensing notice:

```
*****  
*  
* The author of this software is David M. Gay.  
*  
* Copyright (c) 1991, 2000, 2001 by Lucent Technologies.  
*  
* Permission to use, copy, modify, and distribute this software for  
* any purpose without fee is hereby granted, provided that this entire  
* notice is included in all copies of any software which is or  
* includes a copy or modification of this software and in all copies  
* of the supporting documentation for such software.  
*  
* THIS SOFTWARE IS BEING PROVIDED "AS IS", WITHOUT ANY EXPRESS OR  
* IMPLIED WARRANTY. IN PARTICULAR, NEITHER THE AUTHOR NOR LUCENT  
* MAKES ANY REPRESENTATION OR WARRANTY OF ANY KIND CONCERNING THE  
* MERCHANTABILITY OF THIS SOFTWARE OR ITS FITNESS FOR ANY PARTICULAR  
* PURPOSE.  
*  
***** /
```

A.27. Scintilla License

The following software may be included in this product:

```
Scintilla  
  
License for Scintilla and SciTE  
  
Copyright 1998-2003 by Neil Hodgson <neilh@scintilla.org>  
  
All Rights Reserved
```

Permission to use, copy, modify, and distribute this software and its
documentation for any purpose and without fee is hereby granted,
provided that the above copyright notice appear in all copies and that
both that copyright notice and this permission notice appear in
supporting documentation.

NEIL HODGSON DISCLAIMS ALL WARRANTIES WITH REGARD TO THIS
SOFTWARE, INCLUDING ALL IMPLIED WARRANTIES OF MERCHANTABILITY

AND FITNESS, IN NO EVENT SHALL NEIL HODGSON BE LIABLE FOR ANY SPECIAL, INDIRECT OR CONSEQUENTIAL DAMAGES OR ANY DAMAGES WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER IN AN ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION, ARISING OUT OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE.

Scintilla includes some files copyright Adobe Systems Incorporated:

Copyright (c) 2007 Adobe Systems Incorporated

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

Scintilla includes some files copyright Apple Computer, Inc.:

Disclaimer: IMPORTANT: This Apple software is supplied to you by Apple Computer, Inc. ("Apple") in consideration of your agreement to the following terms, and your use, installation, modification or redistribution of this Apple software constitutes acceptance of these terms. If you do not agree with these terms, please do not use, install, modify or redistribute this Apple software.

In consideration of your agreement to abide by the following terms, and subject to these terms, Apple grants you a personal, non-exclusive license, under Apple's copyrights in this original Apple software (the "Apple Software"), to use, reproduce, modify and redistribute the Apple Software, with or without modifications, in source and/or binary forms; provided that if you redistribute the Apple Software in its entirety and without modifications, you must retain this notice and the following text and disclaimers in all such redistributions of the Apple Software. Neither the name, trademarks, service marks or logos of Apple Computer, Inc. may be used to endorse or promote products derived from the Apple Software without specific prior written permission from Apple. Except as expressly stated in this notice, no other rights or licenses, express or implied, are granted by Apple herein, including but not limited to any patent rights that may be infringed by your derivative works or by other works in which the Apple Software may be incorporated.

The Apple Software is provided by Apple on an "AS IS" basis. APPLE MAKES NO WARRANTIES, EXPRESS OR IMPLIED, INCLUDING WITHOUT LIMITATION THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE, REGARDING THE APPLE SOFTWARE OR ITS USE AND OPERATION ALONE OR IN COMBINATION WITH YOUR PRODUCTS.

IN NO EVENT SHALL APPLE BE LIABLE FOR ANY SPECIAL, INDIRECT, INCIDENTAL OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) ARISING IN ANY WAY OUT OF THE USE, REPRODUCTION, MODIFICATION AND/OR DISTRIBUTION OF THE APPLE SOFTWARE, HOWEVER CAUSED AND WHETHER UNDER THEORY OF CONTRACT, TORT (INCLUDING NEGLIGENCE), STRICT LIABILITY OR

OTHERWISE, EVEN IF APPLE HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Copyright (c) 2002 Apple Computer, Inc., All Rights Reserved

A.28. ScintillaNET License

The following software may be included in this product:

ScintillaNET

ScintillaNET is based on the Scintilla component by Neil Hodgson.

ScintillaNET is released on this same license.

The ScintillaNET bindings are Copyright 2002-2006 by Garrett Serack
<gserack@gmail.com>

All Rights Reserved

Permission to use, copy, modify, and distribute this software and its documentation for any purpose and without fee is hereby granted, provided that the above copyright notice appear in all copies and that both that copyright notice and this permission notice appear in supporting documentation.

GARRETT SERACK AND ALL EMPLOYERS PAST AND PRESENT DISCLAIM ALL WARRANTIES WITH REGARD TO THIS SOFTWARE, INCLUDING ALL IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS, IN NO EVENT SHALL GARRETT SERACK AND ALL EMPLOYERS PAST AND PRESENT BE LIABLE FOR ANY SPECIAL, INDIRECT OR CONSEQUENTIAL DAMAGES OR ANY DAMAGES WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER IN AN ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION, ARISING OUT OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE.

The license for Scintilla is as follows:

Copyright 1998-2006 by Neil Hodgson <neilh@scintilla.org>

All Rights Reserved

Permission to use, copy, modify, and distribute this software and its documentation for any purpose and without fee is hereby granted, provided that the above copyright notice appear in all copies and that both that copyright notice and this permission notice appear in supporting documentation.

NEIL HODGSON DISCLAIMS ALL WARRANTIES WITH REGARD TO THIS SOFTWARE, INCLUDING ALL IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS, IN NO EVENT SHALL NEIL HODGSON BE LIABLE FOR ANY SPECIAL, INDIRECT OR CONSEQUENTIAL DAMAGES OR ANY DAMAGES WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER IN AN ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION, ARISING OUT OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE.

A.29. SQLCipher License

Copyright (c) 2008, ZETETIC LLC All rights reserved. Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met: * Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

* Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution. *

Neither the name of the ZETETIC LLC nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission. THIS SOFTWARE IS PROVIDED BY ZETETIC LLC "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL ZETETIC LLC BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

A.30. TinyXML License

The following software may be included in this product:

TinyXML

TinyXML is released under the zlib license:

This software is provided 'as-is', without any express or implied warranty. In no event will the authors be held liable for any damages arising from the use of this software.

Permission is granted to anyone to use this software for any purpose, including commercial applications, and to alter it and redistribute it freely, subject to the following restrictions:

1. The origin of this software must not be misrepresented; you must not claim that you wrote the original software. If you use this software in a product, an acknowledgment in the product documentation would be appreciated but is not required.
2. Altered source versions must be plainly marked as such, and must not be misrepresented as being the original software.
3. This notice may not be removed or altered from any source distribution.

A.31. TreeViewAdv for .NET License

The following software may be included in this product:

TreeViewAdv for .NET

The BSD License

Copyright (c) 2009, Andrey Gliznetsov (a.gliznetsov@gmail.com)

All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met

- Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
- Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation

and other materials provided with the distribution.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS AS IS AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

A.32. VSQLite++ License

The following software may be included in this product:

VSQLite++

VSQLite++ - virtuosic bytes SQLite3 C++ wrapper

Copyright (c) 2006 Vinzenz Feenstra vinzenz.deenstra@virtuosic-bytes.com
All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

- * Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
- * Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
- * Neither the name of virtuosic bytes nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

A.33. zlib License

The following software may be included in this product:

zlib

Oracle gratefully acknowledges the contributions of Jean-loup Gailly and Mark Adler in creating the zlib general purpose compression library which is used in this product.

```
zlib.h -- interface of the 'zlib' general purpose compression library  
Copyright (C) 1995-2004 Jean-loup Gailly and Mark Adler
```

```
zlib.h -- interface of the 'zlib' general purpose compression library  
version 1.2.3, July 18th, 2005  
Copyright (C) 1995-2005 Jean-loup Gailly and Mark Adler
```

```
zlib.h -- interface of the 'zlib' general purpose compression library  
version 1.2.5, April 19th, 2010  
Copyright (C) 1995-2010 Jean-loup Gailly and Mark Adler
```

This software is provided 'as-is', without any express or implied warranty.
In no event will the authors be held liable for any damages arising from the
use of this software. Permission is granted to anyone to use this software
for any purpose, including commercial applications, and to alter it and
redistribute it freely, subject to the following restrictions:

1. The origin of this software must not be misrepresented; you must not
claim that you wrote the original software. If you use this software
in a product, an acknowledgment in the product documentation would
be appreciated but is not required.
2. Altered source versions must be plainly marked as such, and must not
be misrepresented as being the original software.
3. This notice may not be removed or altered from any source distribution.

Jean-loup Gailly jloup@gzip.org
Mark Adler madler@alumni.caltech.edu

Appendix B. MySQL Workbench FAQ

Frequently Asked Questions with answers.

Questions

- [B.1: \[467\] How does MySQL Workbench increase import performance?](#)
- [B.2: \[467\] MySQL Workbench 5.0 appears to run slowly. How can I increase performance?](#)
- [B.3: \[468\] I get errors when creating or placing objects on an EER Diagram. I am using OpenGL rendering, AMD processor, and ATI graphics hardware.](#)
- [B.4: \[468\] What do the column flag acronyms \(PK, NN, UQ, BIN, UN, ZF, AI\) in the MySQL Workbench Table Editor mean?](#)

Questions and Answers

B.1: How does MySQL Workbench increase import performance?

When a model is exported using the main menu item `File`, Export, Forward Engineer SQL CREATE Script, some server variables are temporarily set to enable faster SQL import by the server. The statements added at the start of the code are:

```
SET @OLD_UNIQUE_CHECKS=@@UNIQUE_CHECKS, UNIQUE_CHECKS=0;
SET @OLD_FOREIGN_KEY_CHECKS=@@FOREIGN_KEY_CHECKS, FOREIGN_KEY_CHECKS=0;
SET @OLD_SQL_MODE=@@SQL_MODE, SQL_MODE='TRADITIONAL';
```

These statements function as follows:

- `SET @OLD_UNIQUE_CHECKS=@@UNIQUE_CHECKS, UNIQUE_CHECKS=0;`: Determines whether `InnoDB` performs duplicate key checks. Import is much faster for large data sets if this check is not performed.
- `SET @OLD_FOREIGN_KEY_CHECKS=@@FOREIGN_KEY_CHECKS, FOREIGN_KEY_CHECKS=0;`: Determines whether the server should check that a referenced table exists when defining a foreign key. Due to potential circular references, this check must be turned off for the duration of the import, to permit defining foreign keys.
- `SET @OLD_SQL_MODE=@@SQL_MODE, SQL_MODE='TRADITIONAL' ;`: Sets `SQL_MODE` to `TRADITIONAL`, causing the server to operate in a more restrictive mode.

These server variables are then reset at the end of the script using the following statements:

```
SET SQL_MODE=@OLD_SQL_MODE;
SET FOREIGN_KEY_CHECKS=@OLD_FOREIGN_KEY_CHECKS;
SET UNIQUE_CHECKS=@OLD_UNIQUE_CHECKS;
```

B.2: MySQL Workbench 5.0 appears to run slowly. How can I increase performance?

Although graphics rendering may appear slow, there are several other reasons why performance may be less than expected. The following tips may offer improved performance:

- Upgrade to the latest version. MySQL Workbench 5.0 is still being continually maintained and some performance-related issues may have been resolved.

-
- Limit the number of steps to save in the **Undo History** facility. Depending on the operations performed, having an infinite undo history can use a lot of memory after a few hours of work. In **Tools, Options, General**, enter a number in the range 10 to 20 into the **Undo History Size** spinbox.
 - Disable relationship line crossing rendering. In large diagrams, there may be a significant overhead when drawing these line crossings. In **Tools, Options, Diagram**, uncheck the option named **Draw Line Crossings**.
 - Check your graphics card driver. The GDI rendering used in MySQL Workbench 5.0 is not inherently slow, as most video drivers support hardware acceleration for GDI functions. It can help if you have the latest native video drivers for your graphics card.
 - Upgrade to MySQL Workbench 5.1. MySQL Workbench 5.1 has had many operations optimized. For example, opening an object editor, such as the table editor, is much faster, even with a large model loaded. However, these core optimizations will not be back-ported to 5.0.

B.3: I get errors when creating or placing objects on an EER Diagram. I am using OpenGL rendering, AMD processor, and ATI graphics hardware.

To solve this problem renew the ATI drivers pack, which can be downloaded from the [AMD Web site](#).

B.4: What do the column flag acronyms (PK, NN, UQ, BIN, UN, ZF, AI) in the MySQL Workbench Table Editor mean?

Checking these boxes will alter the table column by assigning the checked constraints to the designated columns.

Hover over an acronym to view a description, and see the [MySQL Workbench Table Editor](#) and [MySQL CREATE TABLE](#) documentation for further information.

Appendix C. Reporting A Useful MySQL Workbench Bug

The following is a list of tips and information that is helpful for reporting a MySQL Workbench bug.

A useful bug report includes:

- The exact steps taken to repeat the bug, ideally as a video if the bug is tricky to repeat
- A screenshot, if the bug is visual
- The error messages, which includes text sent to stdout and the GUI
- The MySQL Workbench Log file

The log file location can be found using [Help](#), Locate Log Files from within MySQL Workbench.

Bugs that cannot be reproduced are difficult and nearly impossible to fix, so it is important to provide the steps necessary to reproduce the bug.

Log Levels

There are six different log levels, with increasing levels of verbosity: `error`, `warning`, `info`, `debug1`, `debug2`, and `debug3`. By default, the `error`, `warning` and `info` levels are enabled. There is also a "none" level that completely switches off logging.



Important

Please enable the `debug3` level before generating a log for the report.

The enabled error log levels can be configured using an environment variable, or by using a command line parameter.

Both the environment variable and command line variants accept a single error level, but enabling a more verbose option will implicitly enable the levels below it. For example, passing in "info" will also enable the "error" and "warning" levels.

- Environment variable: `WB_LOG_LEVEL`

Command line option: `--log-level` on Mac OS X and Linux, and `-log-level` on Microsoft Windows



Note

If both the command line and environment variable are set, the command line takes precedence.

If the `info` level is enabled, the system information and all paths used in the application are also logged. On Microsoft Windows, this also means that the log file contains the full set of current environment variables that are active for the program.

Operating System Specific Notes

Microsoft Windows

- Log file location: Near the user's app data folder, such as `C:\Users\[user]\AppData\Roaming\MySQL\Workbench\` for Microsoft Windows 7.

- In case of errors (or exceptions), the log file contains the stack trace to the point MySQL Workbench can track it (usually only C# code, and not C++ code). Also, all warnings are added to the log if the warning (or greater) log level is enabled.
- If it is a crash and that cannot be replicated by the MySQL Workbench team, and the stack trace cannot be obtained, we will request a crashdump. Instructions for enabling a crashdump can be [found here](#), and please also read the MSDN details for this as we need a full dump, and not the mini dump.
- For crashes related to display issues, start MySQL Workbench with the `-swrrendering` parameter (and only then, as it switches off OpenGL rendering, which is of no use in WBA or WQE). This output will be added to the log file.
- If it is a crash when MySQL Workbench is started (especially if the error report includes something about `kernelbase.dll`), we will ask you to run `depends.exe` on the `MySQLWorkbench.exe` binary, and ask for the reported errors.
- If it is a crash when MySQL Workbench is started, and it is a 64-bit version of Microsoft Windows, check that the correct MSVC runtimes are installed. Often people install the 64-bit version of them, but only the 32-bit will function. More precisely: `MSVC 2010 runtime x86 (32 bit)`.

Mac OS X

- Log File Location: `~/Library/Application Support/MySQL/Workbench/logs`
- System crash logs generated for Workbench are in `~/Library/Logs/DiagnosticReports/MySQLWorkbench*`

Linux

- Log File Location: `~/.mysql/workbench/logs`
- For a crash, we might ask for a stack trace that can be generated by `gdb` by using the following steps:



Note

Because published MySQL Workbench builds lack debug symbols, this step is optional and will probably not be necessary.

- In shell, execute `source /usr/bin/mysql-workbench`
- Quit MySQL Workbench
- In shell, execute `gdb /usr/bin/mysql-workbench-bin`
- In the gdb interface, type `run`
- In MySQL Workbench, repeat the crash
- In the gdb interface, type `bt`
- If it is a crash, also run `glxinfo`. If that also crashes, then it is a driver/X server problem related to OpenGL that is not specific to MySQL Workbench.

Appendix D. MySQL Workbench and Utilities Change History

Table of Contents

D.1. MySQL Workbench Change History	471
D.2. MySQL Utilities Change History	471

This appendix lists the changes from version to version in the MySQL Workbench and MySQL Utilities source code.

Note that we tend to update the manual at the same time we make changes to MySQL. If you find a recent version of the MySQL Workbench or Utilities listed here that you can't find on our download page (<http://dev.mysql.com/downloads/>), it means that the version has not yet been released.

The date mentioned with a release version is the date of the last Bazaar ChangeSet on which the release was based, not the date when the packages were made available. The binaries are usually made available a few days after the date of the tagged ChangeSet, because building and testing all packages takes some time.

The manual included in the source and binary distributions may not be fully accurate when it comes to the release changelog entries, because the integration of the manual happens at build time. For the most up-to-date release changelog, please refer to the online version instead.

D.1. MySQL Workbench Change History

MySQL Workbench release notes are no longer published in the MySQL Workbench Manual.

Release notes for the changes in each release of MySQL Workbench are located at [MySQL Workbench Release Notes](#).

D.2. MySQL Utilities Change History

MySQL Utilities release notes are no longer published in the MySQL Workbench Manual.

Release notes for the changes in each release of MySQL Utilities are located at [MySQL Utilities Release Notes](#).

