MySQL Workbench

MySQL Workbench

Abstract

This manual documents the MySQL Workbench SE version 5.0 beta and the MySQL Workbench OSS version 5.0 beta.

Document generated on: 2009-04-03 (revision: 14506)

Table of Contents

List of Figures

Chapter 1. MySQL Enterprise

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:

1. The MySQL Enterprise Server – the most reliable, secure, and up-to-date version of the world's most popular open source database
2. The MySQL Enterprise Monitor – An automated virtual DBA assistant that monitors all your MySQL Servers around-the-clock, identifies exceptions to MySQL best practices, and provides expert advice on fixing any problems discovered
3. MySQL Production Support – Technical and consultative support when you need it, along with regularly scheduled service packs, hot-fixes, and more

For more information, visit http://www.mysql.com/enterprise.

Chapter 2. MySQL Workbench Editions

The Community Edition (OSS)

The Community Edition is the foundation of all MySQL Workbench editions—versions that are currently available or those that will become available in the future. All editions of MySQL Workbench are based on the Community Edition and all future improvements to the base framework and feature set will be included in this version. The Community Edition is a full feature product that puts a powerful database management tool into the hands of the MySQL community.

The Standard Edition

The Standard Edition is a commercial extension that builds on top of the OSS Edition and adds modules and plugins, allowing for an optimized work flow. The highlights of this edition are the added schema object privilege system, schema validation plugins, model reporting, and online printing, as well as reverse engineering and synchronization against live database connections. If you use the MySQL Workbench in a professional environment upgrading to the commercial edition can greatly improve your work flow.

Chapter 3. Installing MySQL Workbench

You can install MySQL Workbench using with an installer or a Zip file. The requirements for installing MySQL Workbench are:

1. Currently, MySQL Workbench is available for Windows only. MySQL Workbench runs on recent 32-bit Windows NT based operating systems, including Windows XP, Vista, and Windows 2003. It does not run on Windows 2000 and below.

2. The `.Net 2.0` Framework is a requirement. If you do not already have this framework installed, download .NET 2.0 Framework. You may also need the `VS2005.net Redistributable` but first try MySQL Workbench. If the application does not start download VS2005.net Redistributable.

3. On start up, the application checks the OpenGL version and selects between software and hardware rendering. To determine the rendering method that is being used, open the HELP menu and choose the SYSTEM INFO submenu.

3.1. Installing MySQL Workbench on Windows

MySQL Workbench may be installed using the Windows installer file or it may be installed manually from a ZIP file.

Installing MySQL Workbench Using the Installer

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.

Important

Installing MySQL Workbench using the installer 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.

Improving the MySQL Installation Wizard depends on the support and feedback of users. If you find that the MySQL Installation Wizard is lacking some feature important to you, or if you discover a bug, please report it in our bugs database. To do this use the REPORT A BUG option under the HELP menu.

1. To install MySQL Workbench, right click on the MSI file and select the INSTALL option from the pop-up menu.

2. At 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 5.0 `*`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 are having problems running the installer, as an alternative, you can download a ZIP file without an installer. That file is called `mysql-workbench-`*`version`*`-win32.zip`. Using a ZIP program, unpack it to the directory of your choice. You may also want to create a shortcut on your desktop or the quick launch bar.

To install using the ZIP file, download the ZIP file to a convenient location and decompress the file. You can place the resulting directory anywhere on you system. You do not need to install or configure the application before using it.

3.2. Starting MySQL Workbench

Installing MySQL Workbench using the Windows installer automatically creates two entries in the START menu under MYSQL. If your graphic card does not support OpenGL 1.5 select the `Software Rendering` menu entry.

You may also start MySQL Workbench from the command line. To view the available command-line options, issue the command `MySQLWorkbench -help | more` from the MySQL Workbench installation directory. You should see the following output:

```
MySQL Workbench 5.0.14 SE Beta. (C) 2006-2008 by MySQL AB. All rights reserved.

Usage: MySQLWorkbench [options] [model file]

Options
-help (-h) ...... Print this output
-open filename .. Open the given filename at startup
-nologo ......... Do not display the splash screen
-verbose (-v) ... Print verbose output in the GRT Shell
```

```
  -version [grt] .. Print the version information
  -swrendering .... Force the canvas to use software rendering instead of OpenGL
  -log ............ Instruction to save messages (other debug info) to file
```

The MySQL Workbench version number is displayed followed by a usage message and then the options. 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 or, optionally, the GRT shell version number. The other options are self-explanatory.

When using command-line options that display output to a console window, namely `-help` and `-version`, be sure that you pipe the output through the `more` command otherwise nothing will be displayed.

3.3. Uninstalling MySQL Workbench – Windows

The method for uninstalling MySQL Workbench will depend on how you install MySQL Workbench in the first place.

Rmoving MySQL Workbench when installed Using the Installer

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. Doing this will remove MySQL Workbench.

2. Any modules added to the `C:\Program Files\MySQL\MySQL Workbench *version*\modules` directory will **not** be deleted.

> **Note**
>
> It is not possible to remove MySQL Workbench from the command line if you have installed MySQL Workbench using the installer. Although you can manually remove some of the compoentns 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.

When installed from a ZIP file

If you installed MySQL Workbench using a ZIP file, to remove MySQL Workbench you can just delete the MySQL Workbench directory.

> **Note**
>
> If you installed any additional modules within the `modules` directory and you want to keep them, make sure you copy those modules to a different directory before deleting the MySQL Workbench directory.

Chapter 4. MySQL Workbench 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 4.2, “Using the Default Schema” and exploring the graphic design capabilities of MySQL Workbench is touched upon in Section 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.

4.1. Importing a Data Definition SQL Script

For this tutorial use the `sakila` database script found in the `Example Databases` section of the http://dev.mysql.com/doc/ page.

After downloading the file, extract it to a convenient location. Open MySQL Workbench and find the REVERSE ENGINEER MYSQL CREATE SCRIPT menu option by first selecting 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 only be able to view 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 labelled **SAKILA**.

Yo may also wish to remove the default schema tab, `mydb`. Do this by selecting this tab and then clicking 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. To do this 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.

Once you've 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 15.1, “Reverse Engineering Using a Create Script”.

4.1.1. Adding an EER Diagram

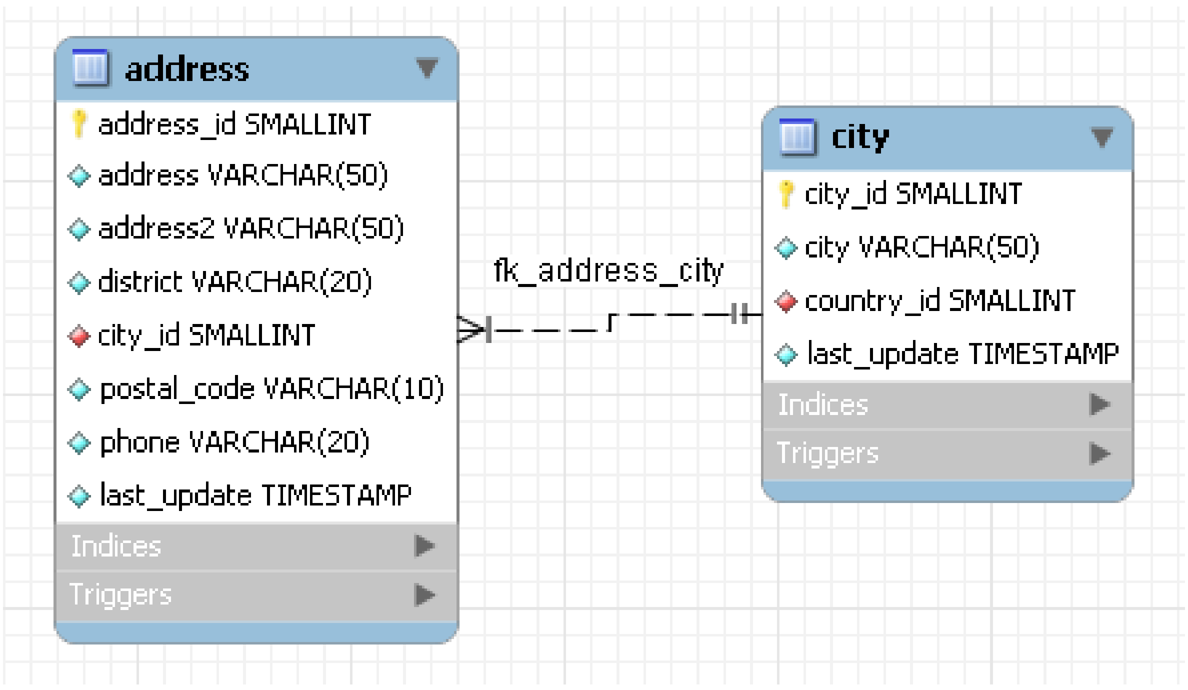
To create an EER diagram for the `sakila` database, first add an EER diagram by double clicking the `Add Diagram` icon in the **EER DIAGRAMS** panel. This should create and open a new `EER Diagram`.

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. This should 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 picking them up from the **CATALOG** panel and placing them on the canvas. Drop the `address` table and the `city` table onto the canvas.

Figure 4.1. 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. (See Figure 5.1, "The `sakila` EER diagram" to view a PNG file of all the relationships in the `sakila` database.)

Choose the `Properties` tab of the panel on the lower right and 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.

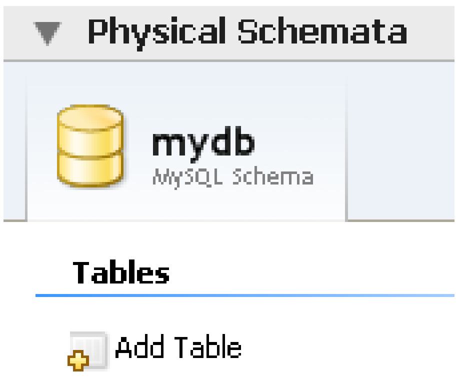
Changing the color of a table is a good way to identify a table quickly — something that becomes more important as the number of tables increases. Changing the color of a table is also an easy way to identify a table in the `Model Navigator` panel. This panel, the uppermost panel on the left side of the screen, gives a bird's eye view of the entire EER canvas.

Save your changes to a `MySQL Workbench Models` file (mwb) by choosing SAVE from the FILE menu or by using the keyboard command **Ctrl S**.

4.2. Using the Default Schema

When you first open MySQL Workbench a default schema, `mydb` appears as the left-most tab of the **PHYSICAL SCHEMATA** section of MySQL Workbench. You can begin designing a database by using this default schema.

Figure 4.2. 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, simply double click anywhere in the editor title bar.

To rename the schema, use the text box labeled **NAME**. Once you have renamed the schema a lightning bolt icon appears right aligned in the **NAME** text box, indicating that other changes are pending. Click in the **COMMENTS** text area 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.

4.2.1. Creating a New Table

Create a new table by double clicking the **ADD TABLE** icon in the `Physical Schemata` panel. Doing 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.

4.2.2. Creating Other Schema Objects

Additional objects such as views or routines can be added in the same way as tables.

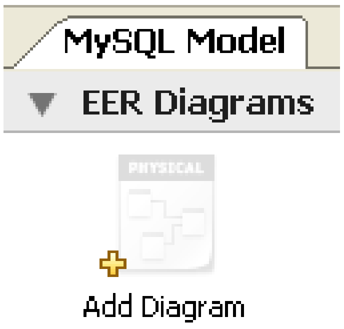
Any objects you have created can be found in the **CATALOG** palette on the right. To view these schema objects select the **CATALOG** tab in the middle palette on the right. View all the objects by clicking the + button to the left of the schema name.

Save your changes to a `MySQL Workbench Models` file (mwb) by choosing SAVE from the FILE menu or by using the keyboard command **Ctrl S**.

4.3. Basic Modeling

On the `MySQL Model` page double click the **ADD DIAGRAM** icon. This creates and opens a new `EER Diagram`.

Figure 4.3. Adding an EER Diagram



From an EER diagram page you can graphically design a database.

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 on 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:** text box. Notice that the name of the tab in the table editor and the name of the table on the canvas, both change to this new value.

Pressing **Tab** or **Enter** while the cursor is in the table name text box, 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` drop-down list box 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.

4.3.2. Create a Foreign Key

Select the table tool again and place another table on the canvas. Name this table `invoice_item`. Next click on the `1:n Non-Identifying Relationship` tool.

First click on the `invoice_item` table; notice that a red border indicates that this table is selected. Next click on the `invoice` table. Doing 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 in the **FOREIGN KEY NAME** text box. 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 on the line joining the tables and then press **Ctrl Delete**.

Experiment with the other tools on the vertical toolbar. Delete a relationship by selecting the eraser tool and clicking on 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) by choosing SAVE from the FILE menu or by using the keyboard command **Ctrl S**.

Chapter 5. Documenting the `sakila` Database

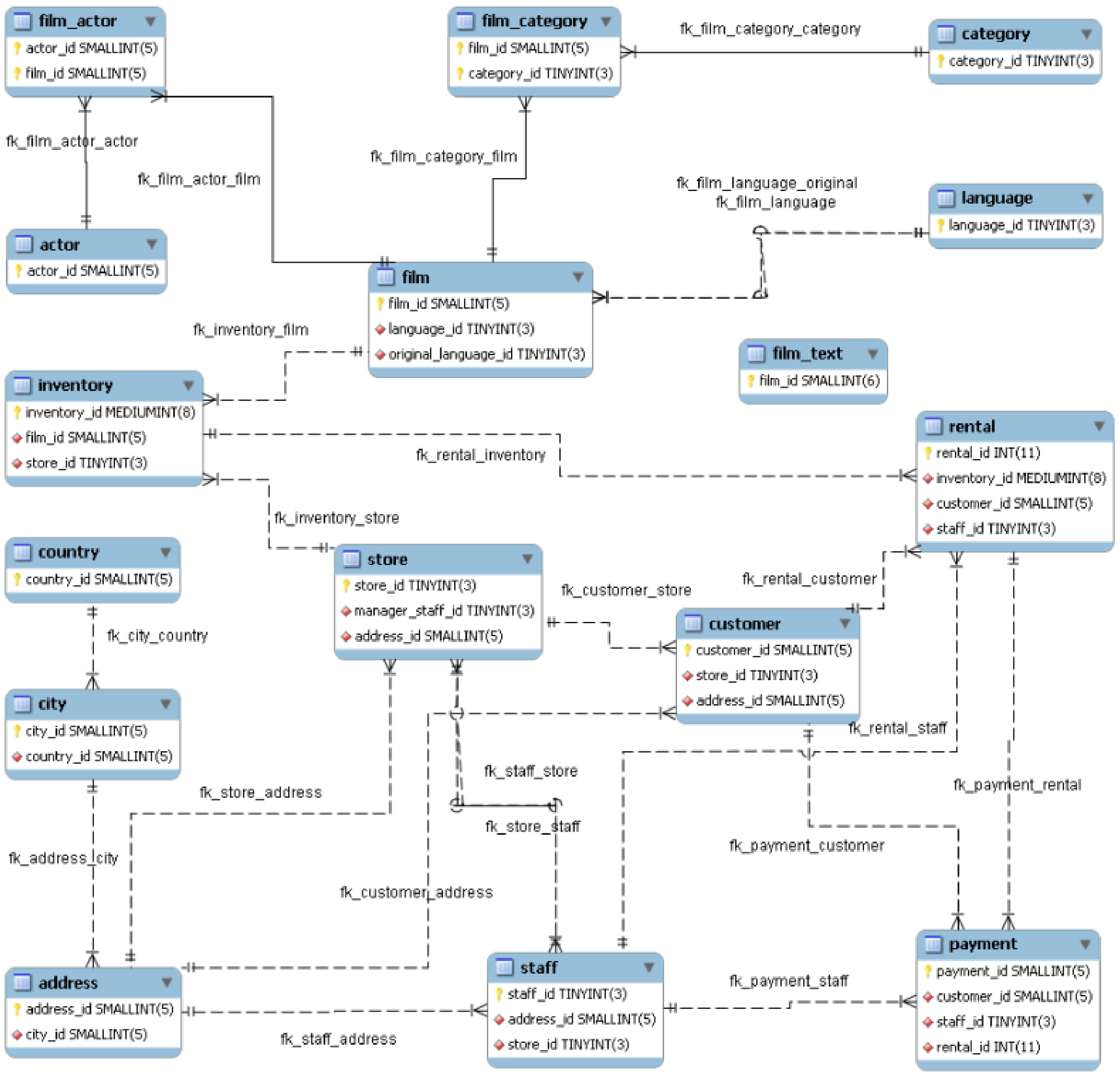
This chapter highlights the capabilities of MySQL Workbench as a documentation tool using the `sakila` database as an example. This is a sample database provided by MySQL and found in the `Example Databases` section of the http://dev.mysql.com/doc/ page. An EER diagram is an invaluable aid to a quick 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.

5.1. A PNG File of the `sakila` Database

Find below an EER digram showing all the tables in the `sakila` database. This image was created using the menu options FILE, EXPORT, EXPORT AS PNG ....

Figure 5.1. The `sakila` EER diagram



The object notation style used in Figure 5.1, "The `sakila` EER diagram" is `Workbench (PKs only)`. This notation only shows primary keys and no other columns so it 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 there is more than one relationship between two tables, the connection lines run concurrently.

Identifying and non-identifying 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.

Chapter 6. MySQL Workbench Reference

This chapter gives an overview of the MySQL Workbench application including a review of the menu items, the toolbar, the MySQL Model page, an EER diagram page, and the various windows that make up the application.

6.1. The Application Windows

The MySQL Workbench application is composed of a number of different windows some of which can be detached from the main window. Others cannot be detached but can be docked a various locations.

6.1.1. The Application Layout

When MySQL Workbench is first opened and you have added an EER Diagram the default layout of the application is as pictured in the following:

Figure 6.1. The default application layout

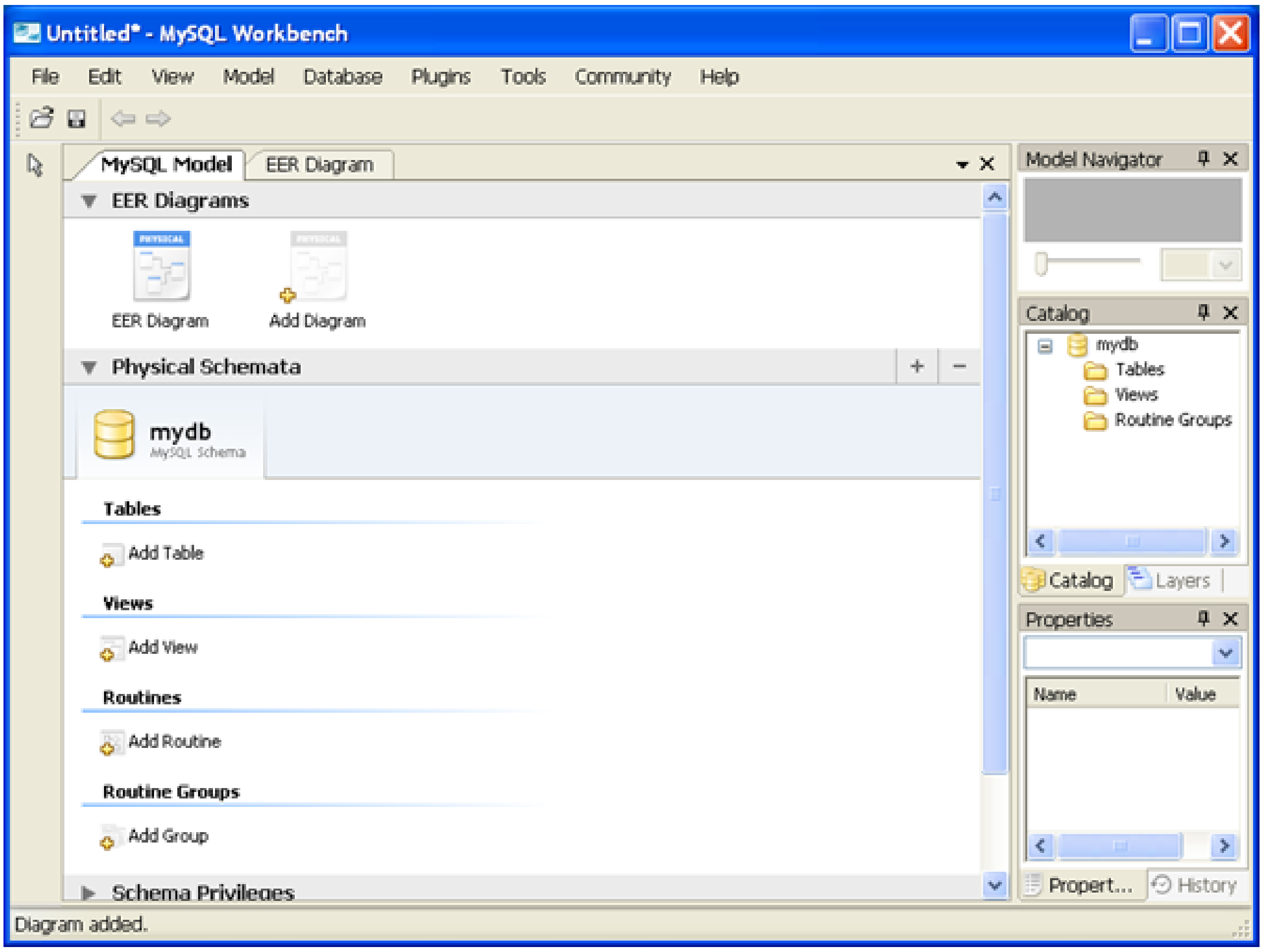


Diagram added.

The `MySQL Model` and the `EER Diagram` windows appear as tabbed windows in the larger window on the left. On the right, the `Model Navigator` is docked at the top, the `Catalog` and `Layers` are tabbed and docked in the middle and the `Properties` and `History` windows are tabbed and docked on the lower right hand side.

The windows on the right cannot be detached from the main window but they may be docked at any position. You also have the option of autohiding these windows by clicking the push pin icon on the window title bar. Clicking this push pin creates a vertical tab on the right side of the main window. Mouse over this tab to display the window.

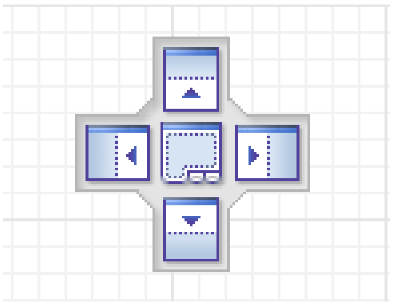
There are five different docking locations within the application; at the top and the bottom of the larger window on the left and one in each of the three areas on the right. When more than one window is placed within any one of these areas you may dock a window on any one of the four sides or you may create tabbed windows.

6.1.2. Docking and Undocking Windows

Of the windows shown in the default view of MySQL Workbench only EER diagrams and the `MySQL Model` page may be undocked. To detach one of these widows double click the desired tab. Doing this immediately undocks a window. Redock a window by double clicking the title bar. Doiong this docks the window at its previous location in the main application window .

To change the docking location of a window, left click and drag its title bar, or if the window is a tabbed window, you may also drag it by left clicking its tab. Doing this displays the docking tool shown in the following.

Figure 6.2. The docking tool



Drag a window over the main application window and a docking tool is displayed for each of the five docking areas identified in Section 6.1.1, "The Application Layout". Drag the window you wish to dock towards your preferred docking area. Notice that the shading changes indicating the area where the window will be docked. To create a tabbed window drop the window at the center of the docking tool.

If you close a window by clicking the x button you can reopen it by choosing the appropriate option under the VIEW, WINDOWS menu option; it will reappear at its last location. For more information about the **WINDOWS** menu options see Section 6.2.3, "The View Menu". To reopen the `MySQL Model` window use the **OVERVIEW** option found under the VIEW menu.

6.2. MySQL Workbench Menus

Some menu options are not available in the OSS version of this application. Menu items marked `SE` in MySQL Workbench OSS version, indicate menu options that are enabled in the standard edition only.

6.2.1. The File Menu

Use this menu item to open a project, begin a new project, or save a project. Choosing NEW opens the default schema, `mydb`. Choosing OPEN opens a file dialog box with the default file type set to MySQL Workbench Models (MWB). To display a list of recently opened MWB files, choose the OPEN RECENT menu option. The keyboard command to create a new project is **Ctrl N** and the command to open an existing project is **Ctrl O**.

To close the currently active `MySQL Model` or `EER Diagram` tab, use the CLOSE TAB option. You can also do this from the keyboard by pressing **Ctrl W**. To reopen the `MySQL Model` tab, see Section 6.2.3, "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 or SAVE AS menu options 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 haven't 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 option to import a MySQL data definition (DDL) script file, one created by issuing the command `mysqldump --no-data`, for example. If the script does not contain a `CREATE` *`db_name`*`;` statement, the schema objects will be 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, it will be ignored. Importing a DDL script is discussed in detail in Section 15.1, "Reverse Engineering Using a Create Script".

Under the Import menu option you can also import `DBDesigner4` files.

There are variety of options under the EXPORT menu item. You may generate the SQL statements necessary to create a new database or alter an existing one. These menu items are discussed in detail in Section 16.1, "Forward Engineering Using SQL Scripts".

Using the EXPORT menu item you can also export an EER diagram as a PNG, SVG, PDF or Postscript file. For an example of a PNG file see Figure 5.1, "The `sakila` EER diagram".

The print options are only enabled 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 Chapter 18, *Printing (Commercial*

Version). *Note*: the printing options are only available in commercial versions of MySQL Workbench.

Use the DOCUMENT PROPERTIES menu option to set the following properties of your project:

- `Name` – Defaults to `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.

6.2.2. The Edit Menu

Under this menu item find the options for cutting, copying, and pasting. These actions can also be performed using the **Ctrl X**, **Ctrl C**, and **Ctrl V** key combinations. Undo a deletion using the UNDO DELETE '*`OBJECT_NAME`*' option. The **Ctrl Z** key combination can also be used to undo an operation.

Also find a DELETE '*`OBJECT_NAME`*' menu item for removing the currently selected object. The text description for this menu item changes to reflect the name of the currently selected object. The keyboard command for this action is **Ctrl 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 instance, 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 setting the default behavior when deleting from an EER Diagram see Section 6.2.8.6, “The Advanced Tab”.

Warning

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

Choose RENAME or EDIT SELECTED to edit the currently selected object. You can also perform edits in a new window using the EDIT SELECTED IN NEW WINDOW. The keyboard shortcut for EDIT SELECTED is **Ctrl E** and **Ctrl Shift E** for EDIT SELECTED IN NEW WINDOW.

The SELECT option has the following submenus:

- SELECT ALL (Keyboard shortcut, **Ctrl A**) – Select all the objects on the active EER diagram.
- SIMILAR FIGURES (Objects of the same type) – Use this option to find objects similar to the currently selected object.
- CONNECTED FIGURES – Use this option to find all the objects connected to the currently selected object.

These menu items are only active when an **EER DIAGRAM** tab is selected. The SIMILAR FIGURES and the CONNECTED FIGURES menu options are disabled if no object is currently selected on an EER diagram.

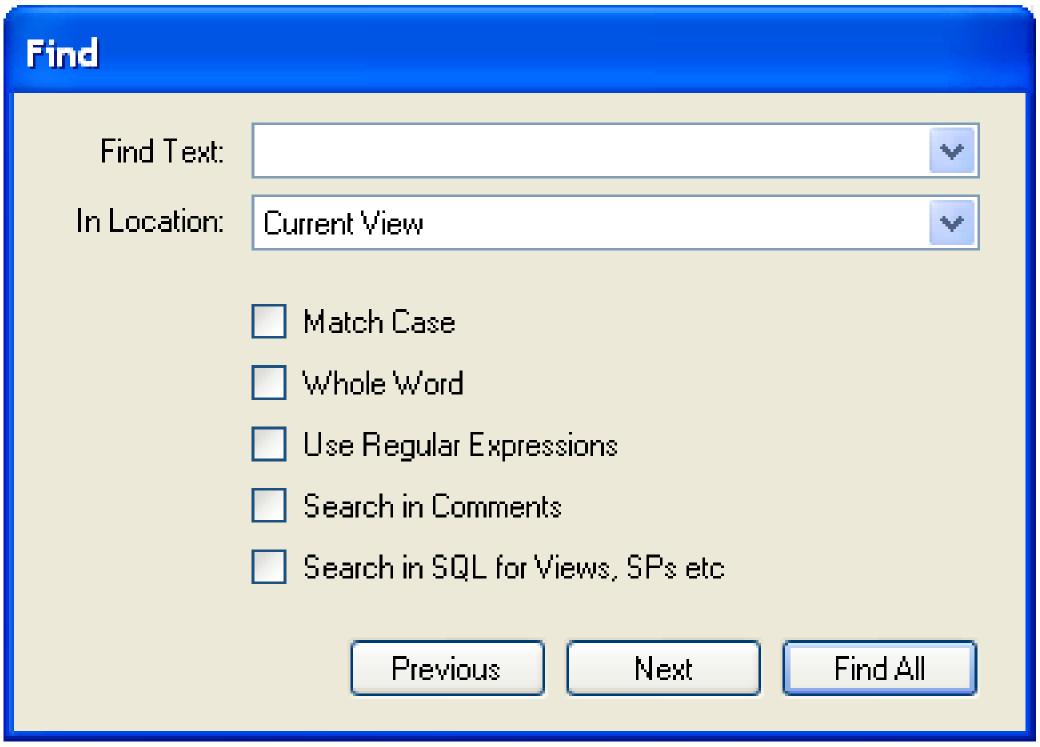
When multiple objects have been selected using one of these menu options, you can navigate between selected items by choosing the GO TO NEXT SELECTED or GO TO PREVIOUS SELECTED menu options.

Selecting items changes some of the EDIT menu options. If only one object is selected, that object's name appears after the CUT, COPY and DELETE menu options. If more than one object is selected, these menu items show the number of objects selected.

6.2.2.1. Find Dialog Window (Commercial Version)

Use the FIND option to open a dialog window used for locating objects.

Figure 6.3. The find window



Note

This menu item is only available in commercial versions of MySQL Workbench.

You can search the following locations:

- Entire Model – Search the entire model.
- Current View – Search the current view only. This may be the `MySQL Model` page.
- All Views – Search the `MySQL Model Page` and all EER diagrams.
- Database Objects – Search database objects only.
- Selected Figures – Search the currently selected objects. This feature only works for EER diagrams.

Enter the text you wish to search for in the **FIND TEXT** drop down list box. 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** drop down list box 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 un-dock 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 **Ctrl F** key combination. Use **Ctrl G** to find the next occurrence and **Ctrl**

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.

6.2.3. The View Menu

The Options available under this menu item are:

- Overview – Under this menu item find the `MySQL Model` option. This option switches to the `MySQL Model` view if it is not currently selected.

  If you have closed the `MySQL Model` window the only way to reopen it is by using this menu option.

- Windows

  - Model Navigator – Open the `Model Navigator` palette
  - Catalog – Open the `Catalog` palette
  - Layers – Open the `Layers` palette
  - Object properties – Open the `Properties` palette
  - Undo History – Open the `History` palette

  These menu options provide a means for opening the windows associated with these options.

- Advanced

  - Output Window – Use this option to display the console output. The keyboard shortcut for this menu item is **Ctrl F2**.
  - GRT Shell – Open the GRT shell. For more information about the GRT shell see Chapter 19, *The Generic Runtime Environment (GRT) Shell*. The keyboard shortcut for opening the GRT shell is **Ctrl F3**.

- Zoom 100% – The default level of detail of an EER diagram.
- Zoom In – Zoom in on an EER diagram.
- Zoom Out – Zoom 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 6.6, "The Model Navigator Palette".

- Set Marker – Use this option to bookmark an object. From the keyboard select the object you wish to bookmark and use the key combination **Ctrl Shift** and the number of the marker (1 through 9). You may create up to nine markers.
- Go To marker – Return to a marker. From the keyboard use the **Ctrl** key and the number of the marker.

6.2.4. The Arrange Menu

The `Arrange` menu option applies only to objects on an EER diagram canvas and is only visible if an EER diagram view is active. The options under this menu item are as follows:

- Align to Grid – Align items on the canvas to the grid lines.
- Bring to Front – Use this option to bring objects to the foreground.
- Send to Back – Use this option to move objects to the background.
- Center Diagram Contents – Use this option to center objects on the canvas.
- Autolayout – Use this option to automatically arrange objects on the canvas.
- Fit Objects to Contents – This option expands an object on an EER diagram. For example, if a table has a long column name that is not fully displayed, using this menu option will expand the table making the column visible. This menu item is not enabled until an object is selected.
- Expand All – Use this option to expand all objects on an EER diagram. This option will display a table's columns if the object notation supports expansion. Some object notations, such as `Classic`, do not allow for expansion or contraction. Indexes will not automatically be expanded unless they were previously expanded and have been collapsed using the Collapse All menu option.

- COLLAPSE ALL – Undo the operation performed by EXPAND ALL.

6.2.5. The Model Menu

The menu options available under the `Model` menu item are as follows:

- **ADD DIAGRAM** – Create a new EER Diagram. The keyboard shortcut is **Ctrl T**.
- **CREATE DIAGRAM FROM CATALOG OBJECTS** – Create an EER diagram from all the objects in the catalog.
- **DBDOC – MODEL REPORTING...**– For information on using this menu option see Section 6.2.5.1, "The DBDoc Model Reporting Dialog Window (Commercial Version)".
- **DIAGRAM SIZE** – Choosing this menu option opens a diagram size dialog box. Use this dialog box 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 option to increase the size of the EER.
- **VALIDATION** – The items available under this option are discussed in Section 6.2.5.2, "The Validation Menu Options (Commercial Version)".
- **VALIDATION (MYSQL)** – The items available under this option are discussed in Section 6.2.5.2, "The Validation Menu Options (Commercial Version)".
- **OBJECT NOTATION** – The items available under this option are discussed in Section 6.2.5.3, "The Object Notation Menu Options (Commercial Version)".
- **RELATIONSHIP NOTATION** – The items available under this option are discussed in Section 6.2.5.4, "The Relationship Notation Menu Option (Commercial Version)".
- **MODEL OPTIONS** – Set options specific to this model. The available options are identical to the options available under the **MODEL**, **MYSQL**, and **DIAGRAM** tabs of the TOOLS, OPTIONS ... menu. For more information, see Section 6.2.8.2, "The Model Tab".

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

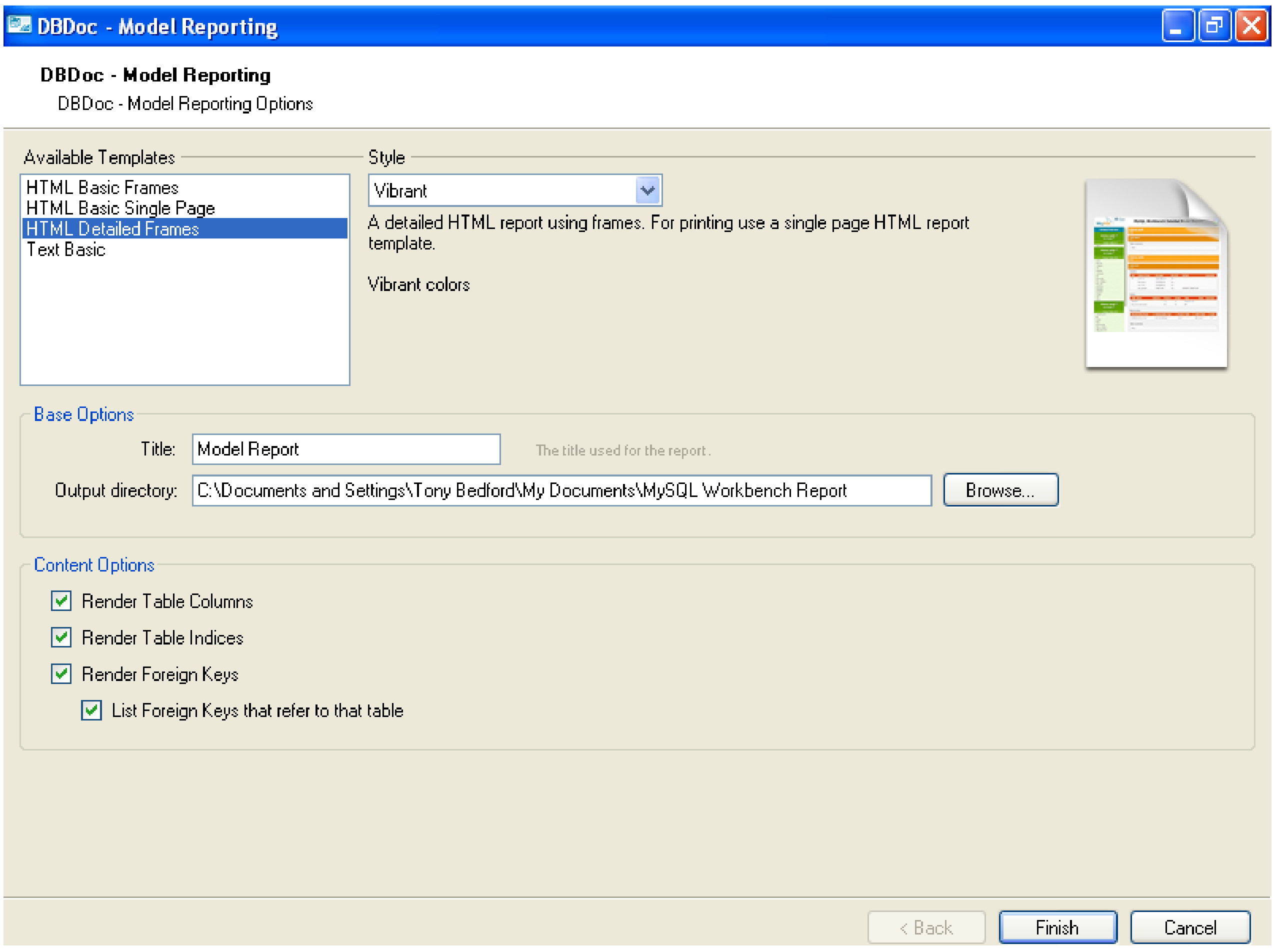
This dialog window is found by navigating to the MODEL menu item and choosing the DBDOC - MODEL REPORTING ... option.

Note

The DBDOC - MODEL REPORTING ... option is not available in the MySQL Workbench OSS version.

Use this dialog window to set the options for creating documentation of your database models.

Figure 6.4. The DBDoc Model Reporting main wizard


DBDoc - Model Reporting

DBDoc - Model Reporting

DBDoc - Model Reporting Options

Available Templates
- HTML Basic Frames
- HTML Basic Single Page
- HTML Detailed Frames
- Text Basic

Style: Vibrant

A detailed HTML report using frames. For printing use a single page HTML report template.

Vibrant colors

Base Options

Title: Model Report — The title used for the report.

Output directory: C:\Documents and Settings\Tony Bedford\My Documents\MySQL Workbench Report

Content Options
- [✓] Render Table Columns
- [✓] Render Table Indices
- [✓] Render Foreign Keys
  - [✓] List Foreign Keys that refer to that table

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.

Click on a template and a preview image displays on the right side of the screen. Choose the `HTML Basic Frames` option and you can select either the `Colorful` or the `Restrained Colors` option from the **STYLE** drop down list box. 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 only offers the `Fixed Size Font` style.

From the **BASE OPTIONS** frame choose the report title and the output directory for the report files.

Content options can also be set. These are:

- `Render Table Columns` – Choose this option to display all the columns.
- `Render Table Indices` – Choose this option to display all the indexes.
- `Render Foreign Keys` – Choose this option to display all the foreign keys.
- `List Foreign Keys that refer to that table` – Choose this option to display the tables that foreign keys reference.

Clicking the FINISH 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.

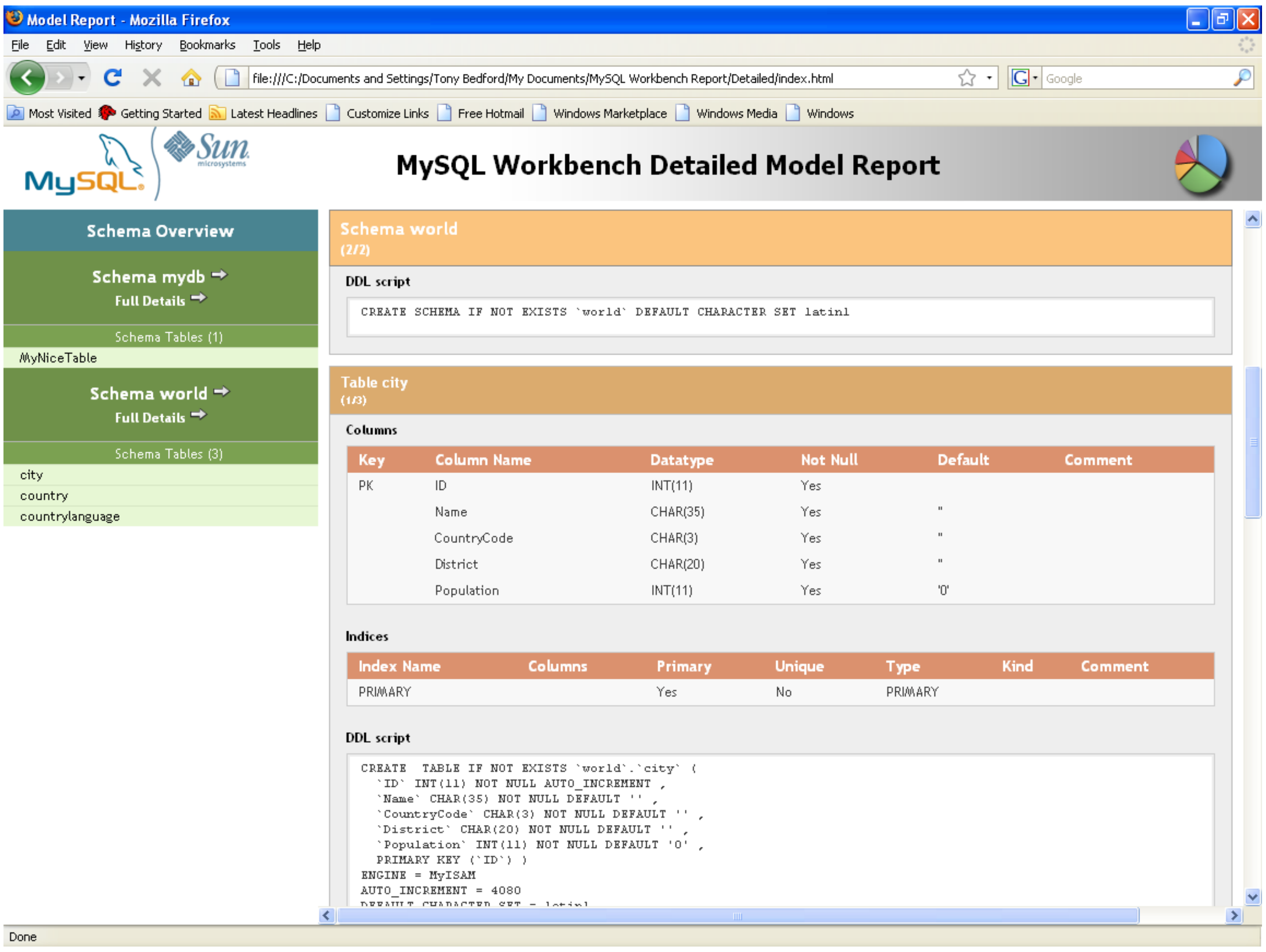
Choosing the `HTML Detailed Frames` option creates the following files:

- `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.
- `details_list.html` – Show a Schema overview along with details of columns, indexes and foreign keys for each schema.
- `full_list.html` – Displays all columns and indexes for the schema.
- `index.html` – The main page.
- `overview.html` – Displays overview information for the report such as report title, project name and author.
- `overview_list.html` – Displays a summary of schema in the model along with a list of tables contained in each schema.
- `table_details.html` – The main report details.
- `top.html` – The top frame of the report.

Choosing the `Text Basic` option creates a directory containing one text file.

You can click on `index.html` to view a report. The following screenshot shows the `HTML Detailed Frames` report being displayed:

Figure 6.5. The DBDoc Model Report


Model Report - Mozilla Firefox

file:///C:/Documents and Settings/Tony Bedford/My Documents/MySQL Workbench Report/Detailed/index.html

MySQL Workbench Detailed Model Report

Schema Overview

Schema mydb
Full Details
Schema Tables (1)
MyNiceTable

Schema world
Full Details
Schema Tables (3)
city
country
countrylanguage

Schema world
(2/2)

DDL script

```
CREATE SCHEMA IF NOT EXISTS `world` DEFAULT CHARACTER SET latin1
```

Table city
(1/3)

Columns

Key	Column Name	Datatype	Not Null	Default	Comment
PK	ID	INT(11)	Yes		
	Name	CHAR(35)	Yes	''	
	CountryCode	CHAR(3)	Yes	''	
	District	CHAR(20)	Yes	''	
	Population	INT(11)	Yes	'0'	

Indices

Index Name	Columns	Primary	Unique	Type	Kind	Comment
PRIMARY		Yes	No	PRIMARY		

DDL script

```
CREATE  TABLE IF NOT EXISTS `world`.`city` (
  `ID` INT(11) NOT NULL AUTO_INCREMENT ,
  `Name` CHAR(35) NOT NULL DEFAULT '' ,
  `CountryCode` CHAR(3) NOT NULL DEFAULT '' ,
  `District` CHAR(20) NOT NULL DEFAULT '' ,
  `Population` INT(11) NOT NULL DEFAULT '0' ,
  PRIMARY KEY (`ID`) )
ENGINE = MyISAM
AUTO_INCREMENT = 4080
DEFAULT CHARACTER SET = latin1
```

Done

If you wish to create custom templates please refer to Chapter 22, *Customizing DBDoc Model Reporting Templates*.

6.2.5.2. The Validation Menu Options (Commercial Version)

Under the MODEL menu option there are two validation options, VALIDATION and VALIDATION (MYSQL). Use these options for general validation and MySQL-specific validation of the objects and relationships defined in your model.

Note

These options are not available in the MySQL Workbench OSS version.

Under the VALIDATION option the menu items are:

- VALIDATE ALL – Perform all the validation options available
- EMPTY CONTENT VALIDATION – Check for objects with no content, for example a table with no columns
- TABLE EFFICIENCY VALIDATION – Check the efficiency of tables, for example a table with no primary key defined
- DUPLICATE IDENTIFIERS VALIDATION – Check for duplicate identifiers, for example two tables with the same name
- CONSISTENCY VALIDATION – Check for consistent naming conventions
- LOGIC VALIDATION – Check, for example, that a foreign key does not reference a non-primary key column in the source table

Under the VALIDATION (MYSQL) option the menu items are:

- VALIDATE ALL – Perform all the validation options available
- INTEGRITY VALIDATION – Check for invalid references, for example, a table name longer than the maximum allowed

- Syntax validation – Check for correct SQL syntax
- Duplicate Identifiers Validation (Additions) – Check for objects with the same name

For detailed information about validation see Chapter 21, *MySQL Workbench Schema Validation Plugins (Commercial Version)*.

6.2.5.3. The Object Notation Menu Options (Commercial Version)

The options under the Object Notation menu apply exclusively to an EER diagram. They are grayed out if an EER diagram tab is not selected.

Note

Alternatives to the default object notation are not available in the MySQL Workbench OSS version.

The menu options are as follows:

- Workbench (Default) – Display table columns, indexes, and triggers.
- Workbench (Simplified) – Show only a table's columns.
- Workbench (PKs only) – Show only columns that are primary and foreign keys.
- 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. To change the default object notation see Section 6.2.8.6, “The Advanced Tab”.

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.

6.2.5.4. The Relationship Notation Menu Option (Commercial Version)

The options under the Relationship Notation menu apply exclusively to an EER diagram. They are grayed out if an EER diagram tab is not selected.

Note

Alternatives to the default object notation (crow's foot) are not available in the MySQL Workbench OSS version.

The menu options are as follows:

- Crow's Foot (IE) – The default modeling style. For an example see Figure 4.1, “Adding tables to the canvas”.
- Classic – Uses a diamond shape to indicate cardinality.
- 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 options

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. To change the default relationship notation see Section 6.2.8.6, “The Advanced Tab”.

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.

6.2.6. The Database Menu (Commercial Version)

There are three options under the DATABASE menu option:

- REVERSE ENGINEERING – Create a model from an existing database. For more information, see Section 15.2, "Reverse Engineering a Live Database (Commercial Version)".
- FORWARD ENGINEERING – Create a database from a model. For more information, see Section 16.2, "Forward Engineering to a Live Server (Commercial Version)".
- SYNCHRONIZE ... – Synchronize your database model with an existing database. For more information, see Section 16.3, "Database Synchronization (Commercial Version)".
- GENERATE SCHEMA DIFF REPORT – Compare your schema model with a live database or a script file. Section 16.4, "Creating a Schema Diff Report (Commercial Version)".

Note

These options are not available in the MySQL Workbench OSS version.

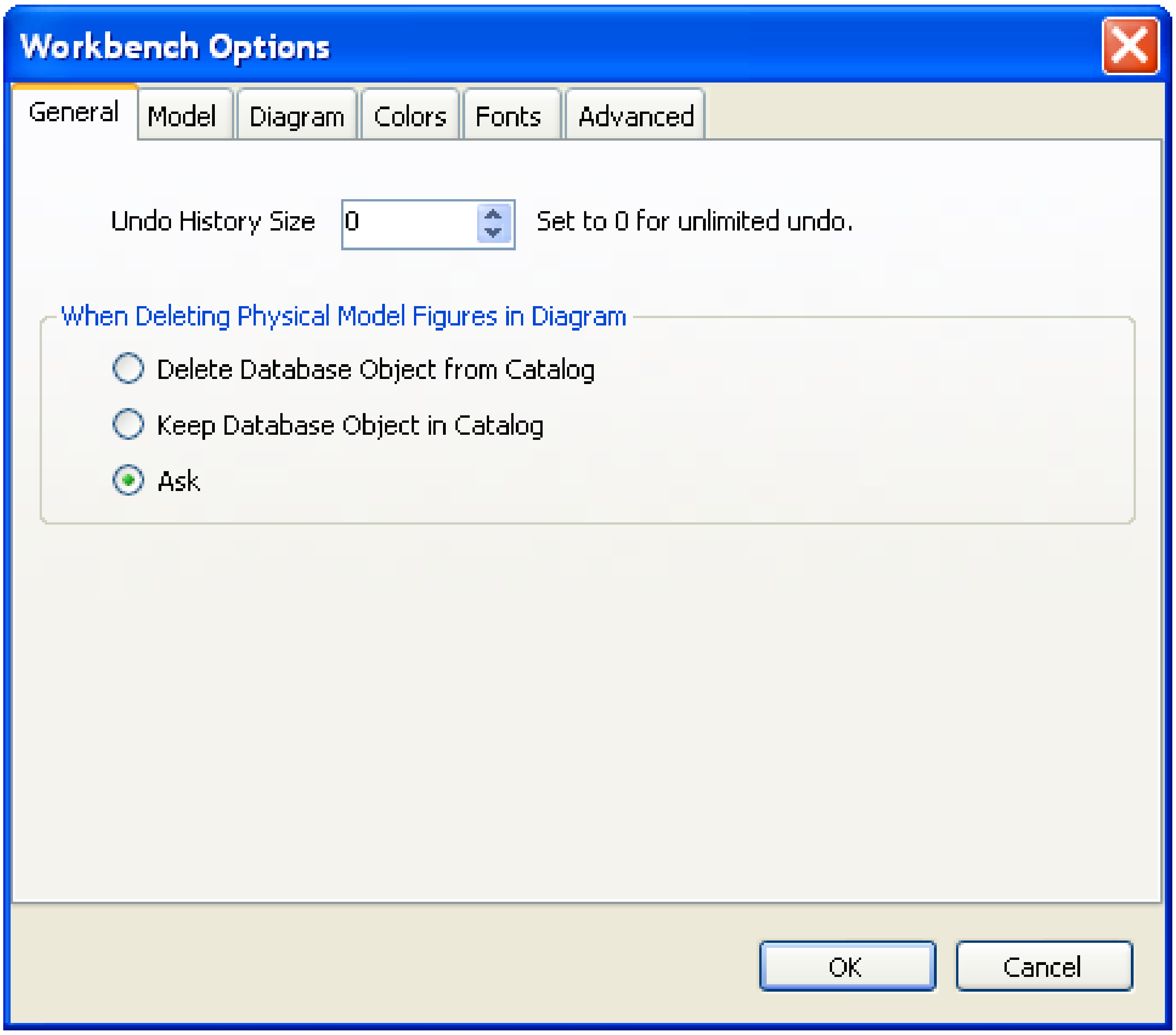
6.2.7. The Plug-ins Menu

The PLUGINS menu option lists any plug-ins that you may have installed. For more information about this menu option see Chapter 20, *Plug-ins*.

6.2.8. The Tools Menu

The OPTIONS menu sets MySQL Workbench defaults. Choosing the OPTIONS menu item opens the following dialog box:

Figure 6.6. The `Options` dialog box



Undo History Size 0 Set to 0 for unlimited undo.

When Deleting Physical Model Figures in Diagram

( ) Delete Database Object from Catalog
( ) Keep Database Object in Catalog
[✓ Ask]

The following list describes the dialog box tabs:

- GENERAL – The delete and undo history options
- MODEL –Default object names
- DIAGRAM – EER diagram settings
- COLORS – Add to or edit the colors that appear in the drop down list boxes
- FONTS – Associate fonts with specific object properties
- ADVANCED – Advanced object configuration

A more detailed discussion of these options follows.

> **Note**
>
> The DEBUG menu option is used for assistance in debugging the beta versions of MySQL Workbench. It will be not appear in the GA release.

6.2.8.1. The General Tab

Under the GENERAL option limit the number of `undos` by setting a value for the **UNDO HISTORY SIZE** text box. For an unlimited number of `undos` set the value to `0`.

Use the **When Deleting Physical Model Figures in Diagram** frame 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 will only open 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 of deleting an object from an EER canvas; using the `eraser` tool, choosing a pop-up menu option, using the delete key, and by choosing the delete option under 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** frame.

If you are using MySQL Workbench OSS, check the check box at the bottom of the page to hide menu options that apply only to the commercial version of MySQL Workbench.

6.2.8.2. The Model Tab

Use the model tab to set the default value for various object names and the primary key data type. A listing of those items with their default values follows:

- Primary Key Column Name – id%table%
- Primary Key Column Type – INT
- Foreign Key Name – fk%stable_%dtable%
- Foreign Key Column Name – %table%_%column%
- Associative Table Name – %stable%_has_%dtable%

The `Primary Key Column Name` is the default primary key column name when using the table editor. Likewise with the default primary key data type. The remaining items are the default names used when using the relationship tools on an EER diagram.

Items enclosed by percentage signs are variables. Their meanings are as follows:

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

6.2.8.3. The Diagram Tab

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

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.

From this tab you can also set the maximum number of characters for:

- Column Names
- Column Types
- Routine Names

Note that this changes the display properties only, not the objects themselves.

6.2.8.4. The Colors 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.

Changes made here affect the drop down list box of colors that appears on the toolbar when adding objects to an EER diagram canvas. For a discussion of using this list box see Section 6.3.1, “Tool-specific Toolbar Items”.

6.2.8.5. The Fonts Tab

Use this tab to set the font face, font size, and the font style for the following list of items:

- Connection Figure Caption
- Layer Title
- Note Figure Text
- Routine Group Figure Item
- Routine Group Figure Title
- Table Figure Items
- Table Figure Section
- Table Figure Title
- View Figure Title

Choose from the drop down list of fonts and font sizes. You may also check the `Bold` and `Italic` check boxes.

6.2.8.6. The Advanced Tab

Use this tab for setting the defaults for various application objects. The options and their default values are shown in the following:

- `DefaultConnectionNotation` – crowsfoot
- `DefaultFigureNotation` – workbench/default
- `FKDeleteRule` – NO ACTION
- `FKUpdateRule` – NO ACTION
- `ProxyServer`
- `ProxyType` – HTTP
- `ProxyUserPwd`
- `VersionsFileURL` – http://dev.mysql.com/workbench/versions.php
- `workbench.model.layer:Color` – #F0F1FE
- `workbench.model.NoteFigure:Color` – #FEFDED
- `workbench.physical.RoutineGroupFigure:Color` – #98D8A5
- `workbench.physical.TableFigure:Color` – #98BFDA
- `workbench.physical.ViewFigure:Color` – #FEDE58

To change the default value simply click the field you wish to change and enter the new value.

Legitimate values for connection notation are:

- `classic`
- `idef1x`
- `crowsfoot` (default)
- `uml`

For more information about these notation styles see Section 6.2.5.4, “The Relationship Notation Menu Option (Commercial Version)”.

Legitimate values for object notation are:

- `workbench/default` (default)
- `workbench/simple`
- `workbench/pkonly`
- `idef1x`
- `classic`

For more information about these notation styles see Section 6.2.5.3, “The Object Notation Menu Options (Commercial Version)”.

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

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

For more information about these actions see Section 7.3.5, “The Foreign Keys Tab”.

The `VersionsFileURL` option is used to support the automatic update feature of MySQL Workbench. The default value is `http://dev.mysql.com/workbench/versions.php`. Normally this value would not be changed by the user.

For more information about the proxy server options see Section 6.2.8.6.1, “Proxy Server Settings”.

Use the remaining options to set the color of various objects. Use hexadecimal values preceded by the ‘`#`’ character.

6.2.8.6.1. Proxy Server Settings

The proxy server group of settings are `ProxyServer`, `ProxyType`, and `ProxyUserPassword`. Use these settings if you are connecting to a database server through a proxy server.

The `ProxyServer` option is used to set the host name and port of a proxy server. This is specified using a host name or IP address followed by a colon and the port number. For example, *`192.168.0.9:8080`* or *`proxy.some-provider.com:80`*. The default value is an empty string, indicating that no proxy server is used.

The `ProxyType` option defines the protocol for the proxy server. Legitimate values are:

- `HTTP` (default)
- `SOCKS4`
- `SOCKS5`

The `ProxyUserPassword` option specifies a user's proxy server credentials. Separate your user name from your password using a colon, for example, *`user_name:password`*. The default value for this option is an empty string.

6.2.9. The Community Menu

The Community menu option offers the following choices:

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

- CONTRIBUTE TO WORKBENCH

Use these menu options to go online and learn more about MySQL Workbench.

6.2.10. The Help Menu

The HELP menu option offers the following choices:

- HELP INDEX – opens a windows showing the MySQL Workbench documentation. Read, search, or print the documentation from this window.
- CHECK FOR UPDATES – open your default browser on the MySQL Workbench website and check for a newer version
- UPDATE ... – update to the latest version
- MYSQL.COM WEBSITE – open your default browser on the MySQL website home page
- WORKBENCH PRODUCT PAGE – open your default browser on the MySQL Workbench product page
- SYSTEM INFO – information about your system, useful when reporting a bug.
- REPORT A BUG – open your default browser on the MySQL bug report page
- ABOUT WORKBENCH – show the MySQL Workbench `About` window

Use these menu options to go online and learn more about MySQL Workbench.

6.2.10.1. System Info

Use the SYSTEM INFO menu option to determine information about your system. This option is especially useful for determining your rendering mode. Sample output follows.

```
Adapter String: VirtualBox Video Adapter
BIOS String: Version 0xB0C2 or later
Memory Size: 65536 KB
MySQL Workbench SE for Windows version 5.0.14
Rendering Mode: Native GDI Rendering
OpenGL Driver Version: Not Detected
OS: Windows XP
CPU: AMD Athlon(tm) 64 X2 Dual Core Processor 4600+, 512 MB RAM
Video adapter info:
Adapter type: VirtualBox Graphics Adapter
Chip Type: VBOX
DAC Type: Integrated RAMDAC
Adapter String:  VirtualBox Video Adapter
BIOS String: Version 0xB0C2 or later
Memory Size: 65536 KB
```

6.3. The Toolbar

The MySQL Workbench toolbar is found immediately below the menu bar. The following tools always appear on the toolbar:

- The folder icon – Click this icon to open a MySQL Workbench file (MWB)
- The save icon – Click this icon to save the current MySQL Workbench project
- The right and left arrows – Not yet activated.

Other tools appear on the toolbar depending upon the context.

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

- The grid icon – Used for aligning objects on the canvas
- The lock icon – Not yet activated.

6.3.1. Tool-specific Toolbar Items

The toolbar also changes depending upon which tool from the vertical toolbar is active. These tools are discussed in Section 6.5.1, "The Vertical Toolbar".

If the `Table` tool is active, drop down list boxes of schemata, engine types and collations appear on the toolbar. There is also a color choice drop down list box.

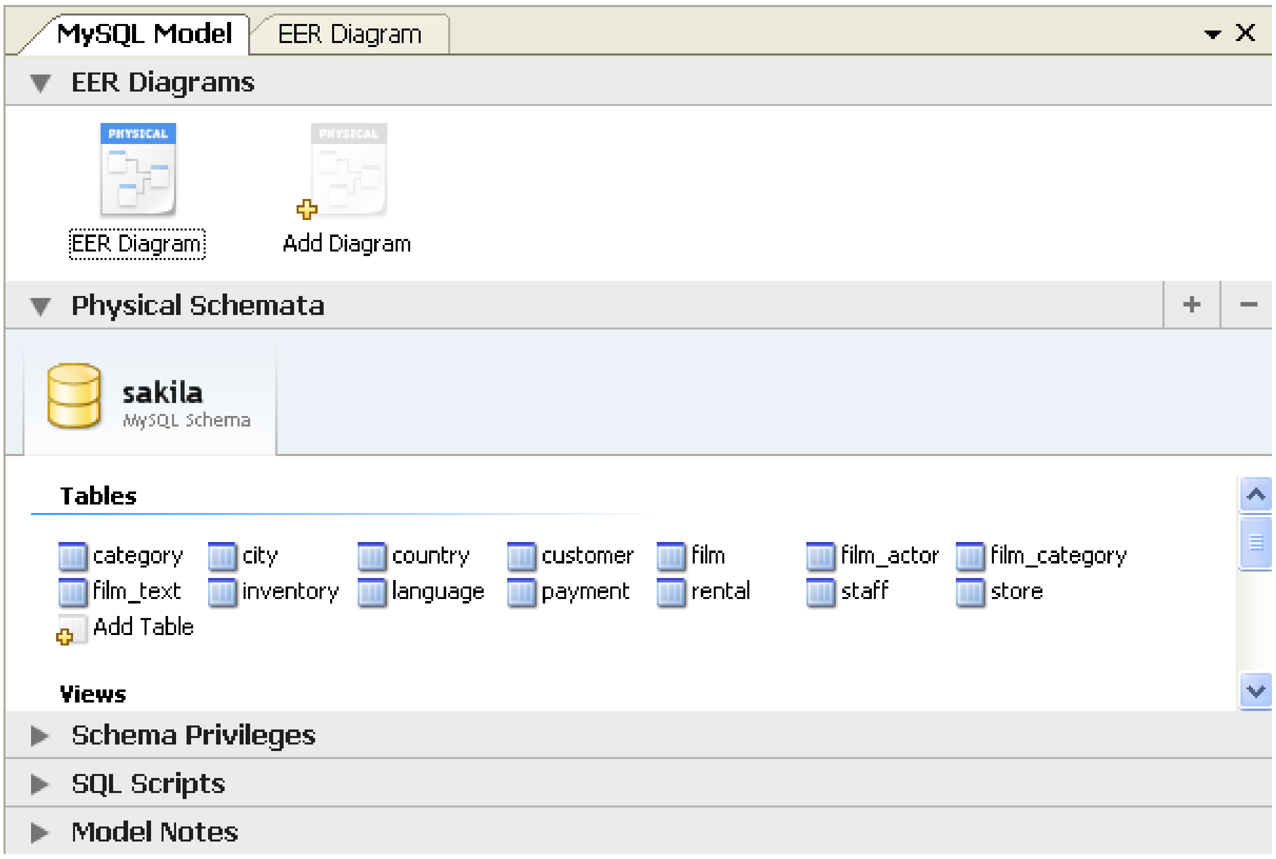
The color choice drop down list box is also displayed when the `Note` or `Layer` tools are active.

When the `View` or `Routine Group` tools are active, the schemata list box and the a color choice drop list box are displayed.

6.4. MySQL Model Page

MySQL Workbench opens on the `MySQL Model` page with the `MySQL Model` tab selected. This page is divided up into five separate sections with space left for docking different windows at the bottom of the page.

Figure 6.7. The `MySQL Model` page



The sections within the MySQL Model page are:

- EER Diagrams
- Physical Schemata
- Schema Privileges (Commercial version)
- SQL Scripts
- Model Notes

For each of these sections objects can be added 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. Doing this opens a pop-up menu.

The sections within the MySQL Model page are discussed in what follows.

6.4.1. EER Diagrams

Use the `Add Diagram` icon in this 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. EER Diagrams are discussed in depth in Section 6.5, “An EER Diagram Canvas”.

If you do not wish to have your EER diagrams as docked or tabbed windows you may change this. For more information, see Section 6.1.2, “Docking and Undocking Windows”.

6.4.2. The Physical Schemata

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, all the schemata that are currently loaded are displayed.

Each schema shows as a tab; a specific schema is selected by clicking 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 Models (MWB) file.

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 the way database object icons are displayed. The left-most button displays database objects as large icons, the next button to the right displays small icons in multiple rows, and the last button displays small icons in a single list.

To expand the Physical Schemata window move the mouse pointer anywhere over the gray area that defines the lower edge of the `Physical Schemata` panel. Hold down the right mouse button and move the mouse to adjust the size of the window.

6.4.2.1. Schema Objects

The `Physical Schemata` panel is divided up into the following sections:

- Tables
- Views
- Routines
- Routine Groups

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

Any objects added to the `Physical Schemata` show up in the `Catalog` palette docked on the right side of the application. 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 6.5, “An EER Diagram Canvas”.

6.4.3. Schema Privileges (Commercial Version)

The `Privileges` panel of the `MySQL Model` page is used to create users for your schemata and also to define roles — making it much easier to assign the same rights to different users or different rights to different users.

Note

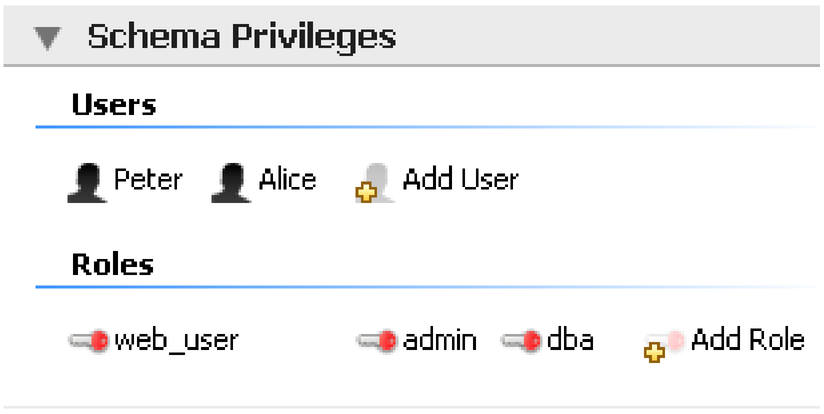
This feature is not available in the MySQL Workbench OSS version.

The `Schema Privileges` panel is divided up into the following sections:

- Users
- Roles

The following image displays the `Schema Privileges` section of the `MySQL Model` tab.

Figure 6.8. Roles and privileges



6.4.3.1. Adding Roles

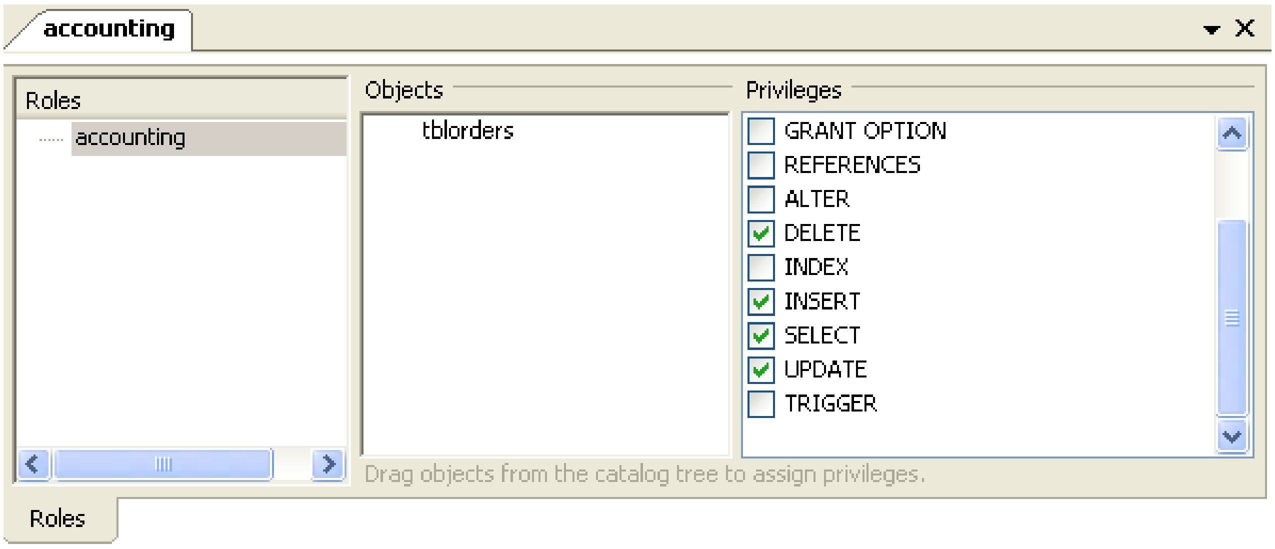
To add a role double click the `Add Role` icon. Doing this creates a role with the default name `role1`. Right clicking a role opens a pop-up menu with the following options:

- RENAME – Rename the role
- CUT '*ROLE_NAME*' – Cut the role
- COPY '*ROLE_NAME*' – Copy the role
- EDIT ROLE ... – Open the role editor.
- EDIT IN NEW WINDOW – Open the role editor in a new editor window.
- DELETE '*ROLE_NAME*' – Remove the role

Use the `Rename` option to give a newly created role a name descriptive of its function.

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 screen.

Figure 6.9. Role editor



Select the role that you wish to add objects to. You may drag and drop objects from the `Physical Schemata` or from the `Catalog` panel to the `Objects` section of the role editor. To assign privileges to a role select a role from the `Roles` section and 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.

6.4.3.2. Adding Users

To add a user double click the `Add User` icon. Doing 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 using the **Name** text box and set the password using the **Password** text box. Assign one role or a number of roles to the user by selecting the desired roles from the text area 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. These options function as described in Section 6.4.3.1, “Adding Roles”.

6.4.4. SQL Scripts and Model Notes

The two remaining panels on the `MySQL Model` page are `SQL Scripts` panel and the `Model Notes` 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 in order to create an `ALTER` script. For more information, see Section 16.1.2, “Altering a Schema”.

Use the `Model Notes` panel to write project notes. Any scripts or notes added will be saved with your project.

6.5. An EER Diagram Canvas

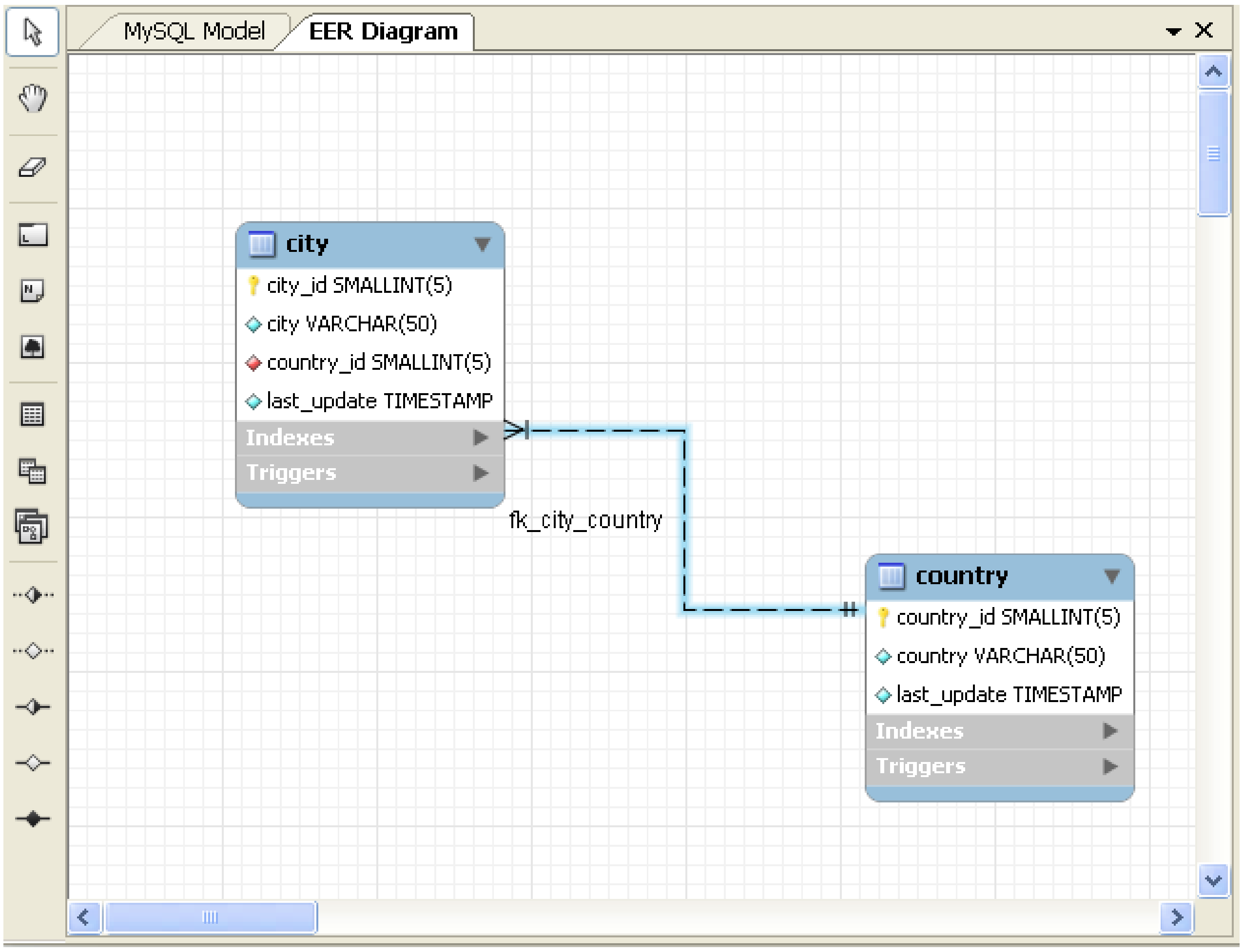
EER diagrams are created by double clicking the `Add Diagram` icon. You may create any number of EER diagrams just as you may create any number of physical schemata. Each EER diagram shows as a tab below the toolbar; a specific EER diagram is selected by clicking its tab.

Clicking an EER diagram tab navigates to the canvas used for graphically manipulating database objects. On the left side of this page is the `Vertical Toolbar`.

6.5.1. 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 6.10. 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. Mousing over a tool displays a description of the tool and its shortcut key.

A more detailed description of each of these tools follows.

6.5.1.1. 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.

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

6.5.1.2. 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 holding down the mouse button. Moving the mouse 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 the VIEW, WINDOWS, MODEL NAVIGATOR to open it.

From the keyboard, use the **H** key to activate this tool.

You can also change the view port of an EER diagram using the `Model Navigator` panel. To do this see Section 6.6, “The Model Navigator Palette”.

6.5.1.3. The Eraser Tool

Use the eraser tool to delete objects from the EER Diagram canvas.

Change the mouse pointer to the eraser tool and 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. Be sure that you understand the available options described in Section 6.2.8.1, "The General Tab" before using the eraser tool

From the keyboard, use the **D** key to activate this tool.

In addition to using the `eraser` tool, you can also delete an object by selecting it and pressing **Ctrl Delete** or right clicking it and choosing DELETE from the pop up menu.

6.5.1.4. The Layer Tool

The layer tool is the rectangular icon with a capital `L` in the lower left corner.

The layer tool is used to organize the objects on an EER Diagram canvas. It is useful for grouping together similar objects. You may, for instance, use it to group all your views together.

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 on 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 one 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.

From the keyboard, use the **L** key to activate this tool. For more information about layers see Chapter 11, *Creating Layers*.

6.5.1.5. 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 and then click the desired location on the canvas. Once 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 either of the pop-up menu options, EDIT NOTE ... or EDIT IN NEW WINDOW ....

You can manipulate the properties of a text object by selecting it and then changing its properties in the `Properties` panel.

From the keyboard, use the **N** key to activate this tool. For more information about text objects see Chapter 13, *Creating Text Objects*.

6.5.1.6. The Image Tool

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

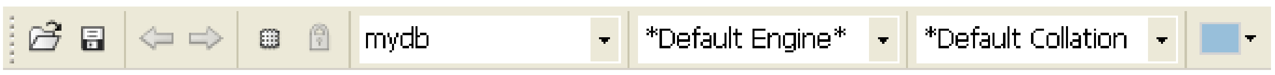
From the keyboard, use the **I** key to activate this tool. For more information about images see Chapter 14, *Creating Images*.

6.5.1.7. The Table Tool

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

When this tool is activated, drop down list boxes appear on the toolbar below the main menu. Use these list boxes to determine the schema, engine type, and collation. All the schemata shown in the `Physical Schemata` section appear in the leftmost drop-down box and all database engines in the engine drop-down box. The collation drop-down box shows all available character set options. There is also a drop down color palette to choose the color accent for your table.

Figure 6.11. The toolbar when the table tool is active



Clicking on the canvas, creates a table. To edit this table, right click it and choose EDIT TABLE or EDIT IN NEW WINDOW from the pop-up menu.

From the keyboard, use the **T** key to activate this tool.

For more information about creating and editing tables see Section 7.3, “The MySQL Table Editor”.

6.5.1.8. The View Tool

Use this tool to create a view on an EER Diagram canvas.

When this tool is activated, a schema drop-down box appears on the toolbar below the main menu, allowing you to associate the new view with a specific schema. You can also select a color for the object by choosing from the color drop down list box to the right of the schema list box.

After selecting this tool, clicking on 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.

From the keyboard, use the **V** key to activate this tool.

For more information about creating and editing views see Chapter 9, *Creating Views*.

6.5.1.9. The Routine Group Tool

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

When this tool is activated, a schema drop-down box appears on the toolbar below the main menu, allowing 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 drop down list box to the right of the schema list box.

After selecting this tool, clicking in 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.

From the keyboard, use the **G** key to activate this tool.

For more information about creating and editing routine groups see Section 10.2, “Routine Groups”.

6.5.1.10. The Relationship Tools

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

- One-to-many non-identifying relationships
- One-to-one non-identifying 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. Mouse over each tool to see a text hint that describes its function.

For more information about relationships see Chapter 8, *Creating Foreign Key Relationships*.

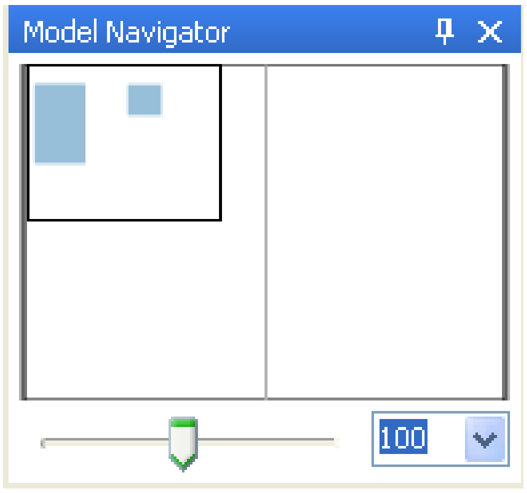
6.6. The Model Navigator Palette

Docked at the top right of the application is the `Model Navigator` palette. This palette gives you 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` palette 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, DIAGRAM SIZE menu option.

Figure 6.12. The Model Navigator palette



Close this window by clicking the X on the top right of the title bar. To reopen the Model Navigator palette choose the VIEW, WINDOWS menu options and then MODEL NAVIGATOR.

You can autohide this window by clicking the push pin icon on the title bar. Doing this displays a vertical tab on the right side of the application. Mouse over this tab to view the Model Navigator.

6.7. The Catalog and Layers Palettes

By default these two palettes are docked on the right, in the middle of the application. You can select one or the other by clicking the tab at the bottom of this palette.

6.7.1. The Catalog Palette

The Catalog 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. Doing 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.

This palette is principally used to drag and drop objects onto an EER diagram canvas.

You can autohide this window by clicking the push pin icon on the title bar. Doing this displays a vertical tab on the right side of the application. Mouse over this tab to view its contents. Autohiding a window when it is a tabbed window, autohides all the tabbed windows at that location.

6.7.2. 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 option. For more information on this topic see Section 6.2.2, “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.

You can autohide this window by clicking the push pin icon on the title bar. Doing this displays a vertical tab on the right side of the application. Mouse over this tab to view its contents. Autohiding a window when it is a tabbed window, autohides all the tabbed windows at that location.

6.7.2.1. Finding Invisible Objects Using the Layers Palette

In some circumstances you may want to make an object on an EER diagram invisible. To do this, select the object and, in the `Properties` palette, set the `visible` property to `False`.

The `Layer` palette provides an easy way to locate an invisible object. Open the `Layers` palette and select the object by double clicking it. Once an object is selected you can reset the `visible` property from the `Properties` palette.

6.8. The Properties and History Palettes

By default these two palettes are docked on the right, at the bottom of the application. You can select one or the other by clicking the tab at the bottom of this palette.

6.8.1. 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. The color of the object is displayed here as is its 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. This opens a color changer dialog box.
- `description` – Applicable to layers only. A means of documenting the purpose of a layer.
- `enabled` – Not yet implemented.
- `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 or not. The value for this attribute is either `true` or `false`.
- `manualSizing` – Whether the object has been manually sized or not. 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` – This property controls whether an object shows up on the canvas or not. Use `‘1’` for true and `‘0’` for false.
- `width` – The width of the object. Depending upon the object, this property may be read only or read/write.

In addition to the properties listed above, tables also have the following properties:

- `indexesExpanded` – This property determines whether indexes are displayed when a table is placed on the canvas. Use `‘1’` for true and `‘0’` for false.
- `triggersExpanded` – This property determines whether triggers are displayed when a table is placed on the canvas. Use `‘1’` for true and `‘0’` for false.

For a discussion of the properties of connections see Section 8.3, “The Properties of a Connection”.

6.8.2. 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 option, COPY HISTORY ENTRIES TO CLIPBOARD. Choose this option 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 non-contiguous entries by using the **Ctrl** key.

Only actions that alter the MySQL model or change an EER diagram are captured by the `History` palette.

Chapter 7. Creating Tables

7.1. 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. Using the table editor is described in Section 7.3, “The MySQL Table Editor”.

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

- RENAME
- CUT '*TABLE_NAME*'
- COPY '*TABLE_NAME*'
- EDIT TABLE...
- EDIT IN NEW WINDOW
- COPY SQL TO CLIPBOARD
- DELETE '*TABLE_NAME*'

If the table editor is not open the EDIT TABLE ... option 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 options are useful for copying tables between different schemata and COPY SQL TO CLIPBOARD copies the `CREATE TABLE` statement to the clipboard.

Warning

Use the DELETE '*TABLE_NAME*' to remove a table from the database. There will be **no** confirmation dialog box.

Any tables 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.

7.2. Adding Tables to an EER Diagram

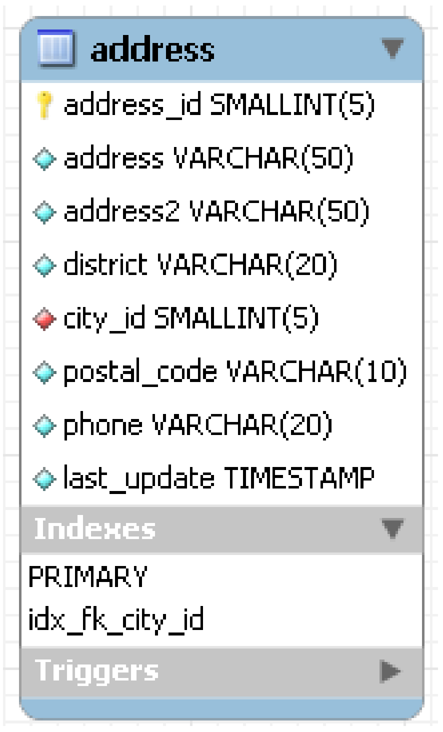
Tables can also be added to an EER Diagram using the `table` tool on the vertical toolbar. To do this make sure that the `EER Diagram` tab is selected, and right click the table icon on the vertical toolbar. The table icon is the rectangular tabular icon.

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 drop down list box of schemata, a drop down list box of engines, a drop down list box of collations, and a drop down color chart. Use these list boxes 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 easily be changed from the table editor and the color of your table can be changed later 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. Doing 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 7.1. 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 options:

- Rename
- Cut '*TABLE_NAME*'
- Copy '*TABLE_NAME*'
- Edit Table...
- Edit in New Window
- Copy SQL to Clipboard
- Delete '*TABLE_NAME*'

With the exception of the deletion option, these menu options function as described in Section 7.1, “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 6.2.8.1, “The General Tab”.

7.3. The MySQL Table Editor

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

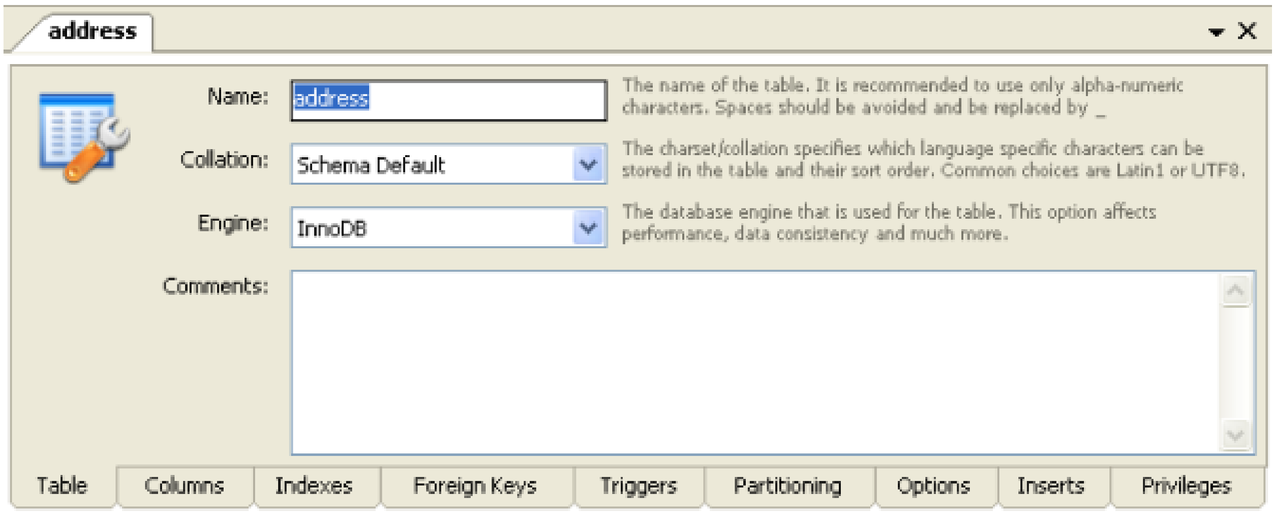
The MySQL Table Editor can be accessed from the MySQL Workbench by first selecting the **MySQL Model** tab and then double clicking a table in the `Physical Schemata` panel. You can also access it from an EER Diagram by double clicking a table object.

7.3.1. The Main Editor Window

Any number of tables may be edited in the MySQL Table Editor at any one time. Adding an additional table creates a new tab at the top of the editor. By default the MySQL Table Editor appears docked at the bottom of the application.

Double clicking the title bar undocks the editor. Do the same to redock it. The MySQL Table Editor is shown in the following figure.

Figure 7.2. The table editor



Name: address
The name of the table. It is recommended to use only alpha-numeric characters. Spaces should be avoided and be replaced by _

Collation: Schema Default
The charset/collation specifies which language specific characters can be stored in the table and their sort order. Common choices are Latin1 or UTF8.

Engine: InnoDB
The database engine that is used for the table. This option affects performance, data consistency and much more.

Comments:

The MySQL Table Editor consists of a work space divided into the following tabs:

- **TABLE:** Use this table to edit features that apply to the table as a whole
- **COLUMNS:** Use this tab to add or modify columns
- **INDEXES:** Use this tab to add or modify indexes
- **FOREIGN KEYS:** Use this tab to add or modify foreign keys
- **TRIGGERS:** Use this tab to add or modify triggers
- **PARTITIONING:** Use this tab to manage partitioning
- **OPTIONS:** Use this tab to add or modify various general, table and row level options
- **INSERTS:** Use this tab for writing INSERT statements
- **PRIVILEGES:** Use this tab to set privileges on the table

Each of these tabs is discussed in further detail in the following sections.

7.3.2. The Table Tab

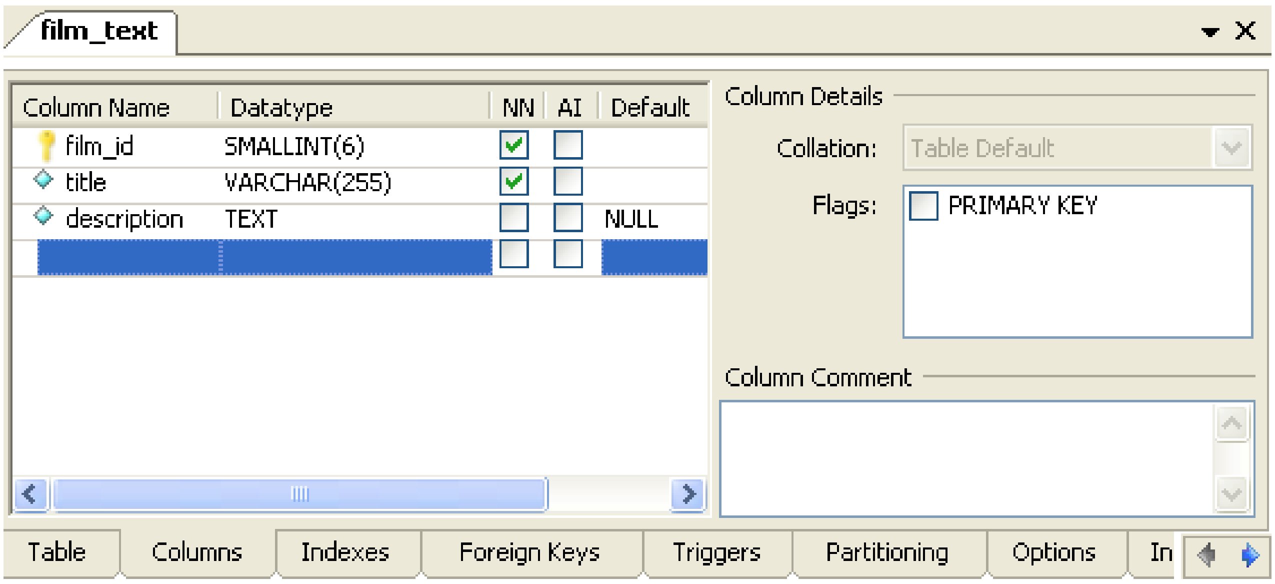
Use this tab to edit the table name or add a comment to the table. Easily change the collation or the table engine using drop down list boxes.

7.3.3. The Columns Tab

The `Columns` tab is used to display and edit all the column information for a table. Using this tab, you can add, drop, and alter columns.

You can also use the column tab to change the name, data type, default value, and other properties of your table's columns.

Figure 7.3. The columns tab



Column Name	Datatype	NN	AI	Default
film_id	SMALLINT(6)	✓	☐	
title	VARCHAR(255)	✓	☐	
description	TEXT	☐	☐	NULL
		☐	☐	

Column Details

Collation: Table Default

Flags: ☐ PRIMARY KEY

Column Comment

To add a column simply click the `Column Name` field in an empty row and enter an appropriate value. Select a data type from the **DATATYPE** drop down list box . Check the check box under the **NN** column to disallow a null value and check the **AI** column if you are defining an autoincrement field.

Right clicking a row under the `Column Name` column opens a pop-up window with the following options:

- MOVE UP – Move the selected column up.
- MOVE DOWN – Move the selected column down.
- DELETE SELECTED COLUMNS – Select multiple contiguous columns by right clicking and pressing the **Shift** key. Use the **Ctrl** key to select non-contiguous columns.
- REFRESH GRID – Update all information in the `Columns` tab.

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

To modify the flags on a column (`PRIMARY KEY`, `UNSIGNED`, `ZEROFILL`) check the desired value in the **FLAGS** frame. Note that values only display in the flags frame when a column is selected.The `ZEROFILL` option only appears if the column type is numeric.

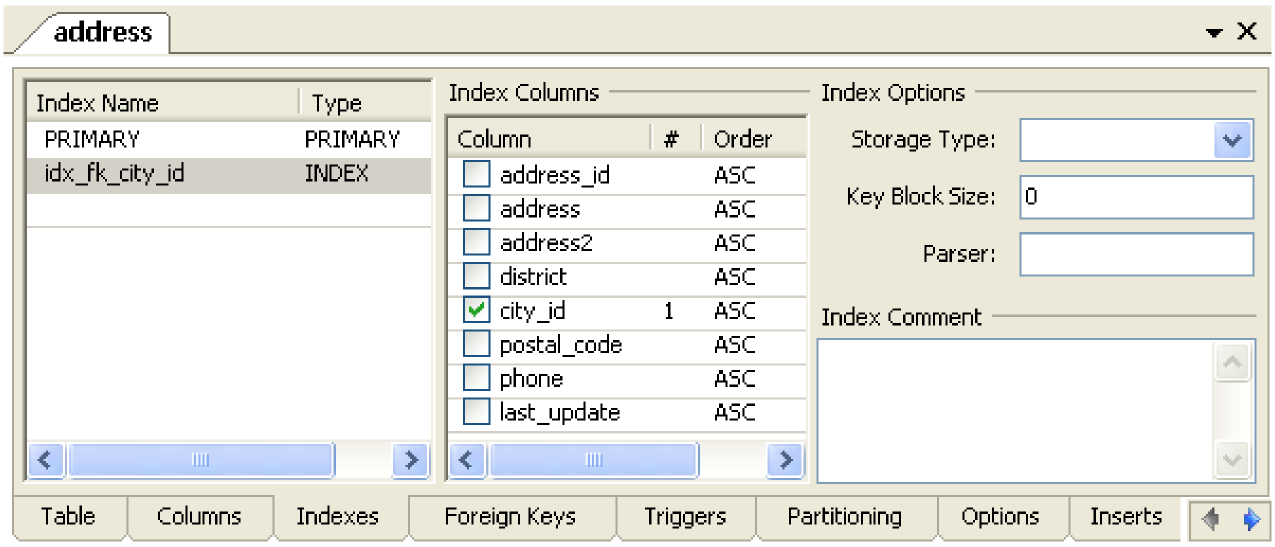
You can also add column comments to the `Column Comment` text area.

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. To add or remove a column from the primary key, double click on the icon. You can also add a primary key by checking the `PRIMARY KEY` checkbox in the `Column Details` section of the table editor.

7.3.4. The Indexes Tab

The `Indexes` tab holds all index information for your table. You can add, drop, and modify indexes using this tab.

Figure 7.4. The indexes tab


address

Index Name	Type
PRIMARY	PRIMARY
idx_fk_city_id	INDEX

Index Columns

Column	#	Order
☐ address_id		ASC
☐ address		ASC
☐ address2		ASC
☐ district		ASC
✓ city_id	1	ASC
☐ postal_code		ASC
☐ phone		ASC
☐ last_update		ASC

Index Options

Storage Type:

Key Block Size: 0

Parser:

Index Comment

Select an index by right clicking it. Doing this displays information about the index in the **INDEX COLUMNS** section.

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 drop down list box. 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 and then select the DELETE SELECTED INDEXES menu option.

7.3.5. 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.

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 and then select the DELETE SELECTED FKS menu option.

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

7.3.6. The Triggers Tab

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

7.3.7. The Partitioning Tab

If you wish to enable partitioning for your table check the **ENABLE PARTITIONING** check box. Doing this enables the partitioning options.

The **Partition By** drop down list box displays the types of partitions you can create. These are:

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

Use the **Parameters** text box to define the parameter(s) that will be supplied to the partitioning function, an integer column value for example.

Choose the number of partitions from the **Partition Count** drop down list box. If you wish to manually configure your partitions check the **Manual** check box. Doing 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 not yet enabled. For more information about partitioning see Partitioning.

7.3.8. The Options Tab

The **Options** tab allows you to set the general options and row options.

In the **General Options** frame, 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.

To set the row format, choose the desired row format from the drop-down list. See `MyISAM` Table Storage Formats for more information about the different row formats that are available. This only applies to MyISAM tables.

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.

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

The `Merge Table` Options section is used to configure MERGE tables in MyISAM. To create a MERGE table, select MERGE as your storage engine and then specify the tables you wish to MERGE in the **UNION TABLES** dialog.

You can also specify the action the server should take when users attempt to perform INSERT statements on the merge table. See The `MERGE` Storage Engine for more information about MERGE tables. Again, this only applies to MyISAM tables. You may also select the `Merge Method` by selecting from the drop down list box.

7.3.9. The Inserts Tab

Use the `Inserts` tab to insert records into the table. Clicking the OPEN EDITOR ... button on the lower right hand side opens the `Standard Inserts` editor. Use this editor to add records to the table.

When you have finished adding records, press OK. The records you have added are displayed in the `Inserts` tab. Reopening the editor displays all the records shown in the `Inserts` tab. To edit a record simply click on the field you wish to change and enter the new data. To delete a record, click the button on the left beside the desired row and then press the **Delete** key. You can select and delete all records by clicking in the top left hand column of the editor and then pressing the **Delete** key. Your changes will not be applied until you press the OK button. To back out of an operation, press the CANCEL button.

Any records you add will be inserted when you forward engineer the database (if you choose the `Generate INSERT statements for tables` option).

7.3.10. 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 6.4.3.1, “Adding Roles”.

When this tab is first opened, all the roles that have been created are displayed in the list box on the right. Move the roles you wish to associate with this table to the **ROLES** list box 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 **Ctrl** key to select non-contiguous roles.

To assign privileges to a role click on a role in the **ROLES** list box. Doing this displays all available privileges in the **ASSIGNED PRIVILEGES** list box. 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 listed in the preceding. Privileges irrelevant to a specific table, the `FILE` privilege for example, 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 box.

Chapter 8. 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 seeForeign Keys.

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 7.3.5, “The Foreign Keys Tab”.

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

8.1. Adding Foreign Key Relationships Using an EER Diagram

There are five foreign key tools on the vertical toolbar on the left side of an EER Diagram. These tools are:

- The `one-to-many non-identifying relationship` tool
- The `one-to-one non-identifying relationship` tool
- The `one-to-many identifying relationship` tool
- The `one-to-one identifying relationship` tool
- The `many-to-many relationship` tool

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 non-identifying 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 on the table that is on the “many” side of the relationship and then on the table containing the referenced key.

Doing this creates a field in the table on the many side of the relationship. The default name of this field is *`table_name_key_name`* where the table name and the key name are both derived from the table containing the referenced key.

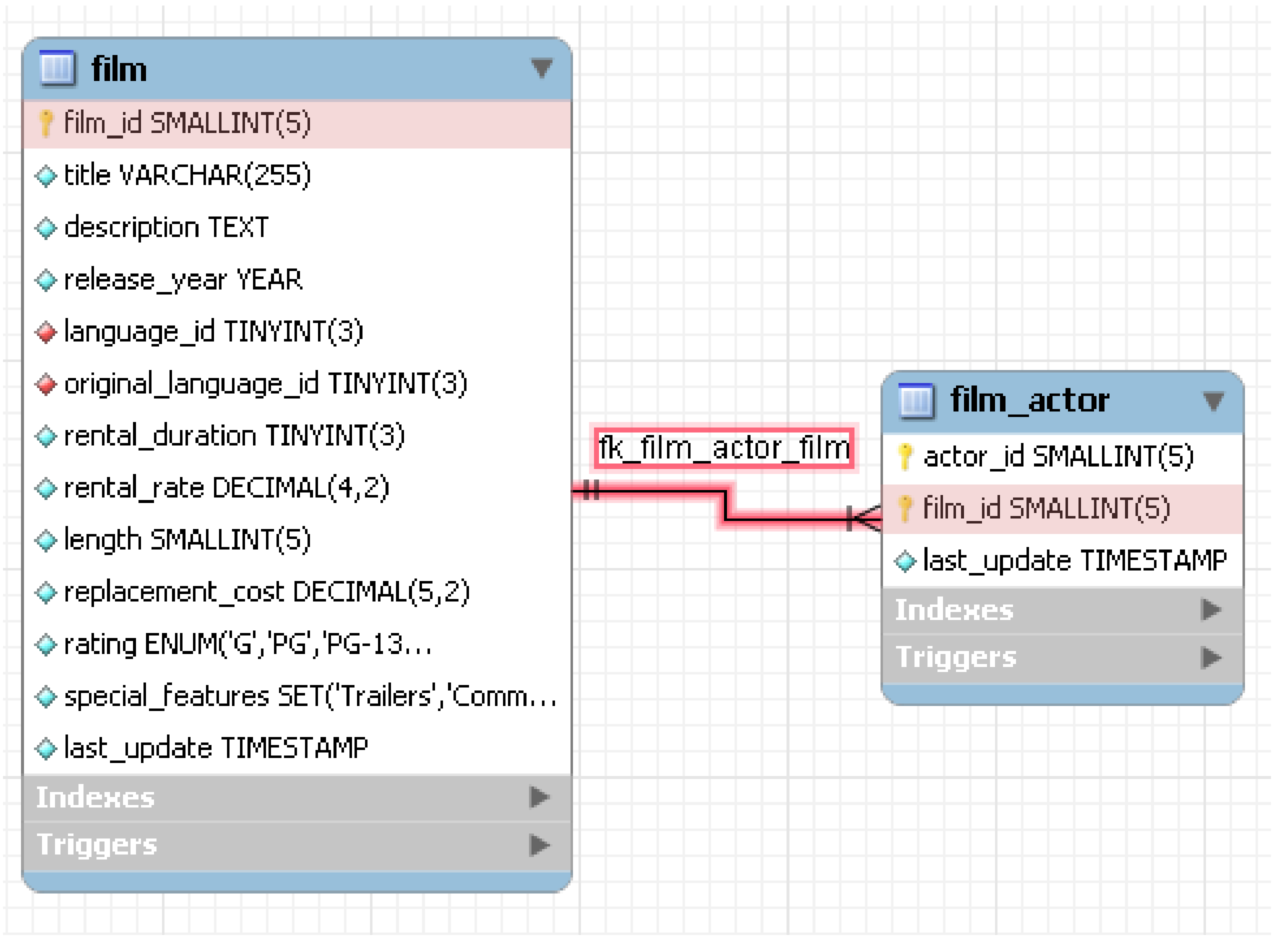
When the many-to-many tool is active, double clicking a table creates an associative table with a many-to-many relationship. For this tool to function there must be a primary key defined in the initial table.

Use the Model, Menu Options menu item to set a project-specific default name for the foreign key column (see Section 6.2.5.4, “The Relationship Notation Menu Option (Commercial Version)”). To change the global default see Section 6.2.8.2, “The Model Tab”.

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

Mousing over a relationship connector highlights the connector and the related keys as shown in the following figure.

Figure 8.1. The relationship connector



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.

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 8.3, “The Properties of a Connection”. Where the notation style allows, `Classic` for instance, the cardinality indicators can also be repositioned.

The relationship notation style in Figure 8.1, “The relationship connector” is the default, crow's foot. If you are using a commercial version of MySQL Workbench you can change this. For more information, see Section 6.2.5.4, “The Relationship Notation Menu Option (Commercial Version)”.

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

8.2. The Relationship Editor

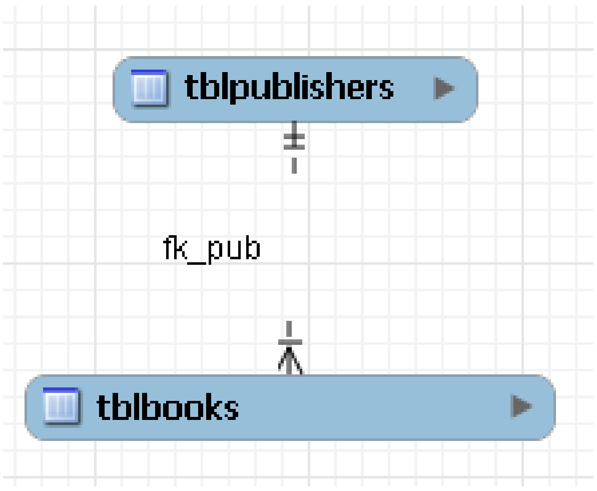
Set the caption of a relationship using the **Caption** text box. This name displays on the canvas and is also the name used for the constraint itself. The default value for this name is *`fk_source_table_destination_table`*. Use the Model, Menu Options menu item to set a project-specific default name for foreign keys. To change the global default see Section 6.2.8.2, “The Model Tab”.

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

In the **Integrity Constraints** frame determine whether the entity and the referred entity are mandatory. The default value for both of these constraints is `true`.

In the **Cardinality** frame, choose whether the relationship is one-to-one or one-to-many. The **Visibility** frame determines 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 8.2. The split connector



Note

A broken line connector is used to indicate a non-identifying relationship. The split line style can be used with either an identifying relationship or a non-identifying 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 go to the MODEL, RELATIONSHIP NOTATION menu item. For more information, see Section 6.2.5.4, "The Relationship Notation Menu Option (Commercial Version)".

8.3. The Properties of a Connection

To select a connection, right click it. When a connection is selected it is highlighted and its properties are displayed in the properties palette. The properties of a connection are quite different from the properties of other objects. These properties are described in the following list:

- `caption` – The name of the object. By default this property is centered above the connection line. Its default value is the name of the foreign key.
- `captionXOffs` – The "x" offset of the caption.
- `captionYOffs` – The "y" offset of the caption.
- `comment` – The comment associated with the relationship.
- `drawSplit` – Whether or not 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. The default location for 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 ot not the entities are mandatory. For more information, see Section 8.2, "The Relationship Editor".
- `many` – False if the relationship is a one-to-one relationship.
- `middleSegmentOffset` – The offset of the middle section of the connector.
- `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 or not 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.

- `visible` – Whether the relationship line is visible or not.

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 the relationship editor is closed, you will no longer be able to select the relationship in order to bring up its relationship editor. To make the relationship visible again you will need to expand the table object relating to the relationship in the **LAYERS** window and select the relationship object. Once selected, you can change the relationship's `visible` property to `True` in the corresponding **PROPERTIES** window. The relationship will then be visible in the **EER DIAGRAM** window.

Chapter 9. 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.

9.1. Adding Views to the Physical Schemata

Double clicking the `Add View` icon in the `Physical Schemata` section of the `MySQL Model` page adds a view with the default name of `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. Using the view editor is described in Section 9.3, "The View Editor".

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

- RENAME
- CUT '*VIEW_NAME*'
- COPY '*VIEW_NAME*'
- EDIT VIEW...
- EDIT IN NEW WINDOW
- COPY SQL TO CLIPBOARD
- DELETE '*VIEW_NAME*'

If the table editor is not open the EDIT VIEW ... option 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 options are useful for copying views between different schemata and COPY SQL TO CLIPBOARD copies the `CREATE VIEW` statement to the clipboard.

Warning

Use the DELETE '*VIEW_NAME*' to remove a view from the database. There will be **no** confirmation dialog box.

Any views 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 Digram by dragging and dropping them from this palette.

9.2. Adding Views to an EER Diagram

Views can also be added to an EER Diagram using the `View` tool on the vertical toolbar. To do this make sure that the `EER Diagram` tab is selected, and right click the view icon on the vertical toolbar. The view icon is the two overlapping rectangles found below the table icon.

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

Choosing the `View` tool changes the contents of the toolbar that appears immediately below the menu bar. When the `Views` pointer is active, this toolbar contains a drop down list box of schemata and a drop down color chart. Use these list boxes 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 easily be changed later 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 option, these menu options function as described in Section 9.1, "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 6.2.8.1, "The General Tab".

9.3. The View Editor

You can invoke the view editor by double clicking a view object on the EER Diagram canvas or by double clicking a view in the

`Physical Schemata` section on the `MySQL Model` page. Doing 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 two tabs at the bottom of the view editor, the **View** and the **Privileges** tabs. Navigate between different tabs using the mouse or from the keyboard by pressing **Ctrl** + **Alt** + **Tab**.

9.3.1. The `View` Tab

From the `View` tab of the view editor you can perform the following tasks:

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

9.3.2. 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 7.3.10, “The Privileges Tab”.

9.3.3. 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 the properties accessible through the `Properties` palette see Section 6.8.1, “The Properties Palette”.

Chapter 10. 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 only be added from the **Physical Schemata** section of the **MySQL Model** page.

To view an existing schema, along with its Routines and Routine Groups, select 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.

10.1. Routines

10.1.1. Adding Routines to the Physical Schemata

Double clicking the `Add Routine` icon in the `Physical Schemata` section of the `MySQL Model` page adds a routine with the default name of `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. Using the routine editor is described in Section 10.1.2, "The Routine Editor".

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

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

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

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

The action of the delete option varies depending upon the way you have configured MySQL Workbench. For more information, see Section 6.2.8.1, "The General Tab".

10.1.2. The Routine Editor

You can invoke the routine editor by double clicking a routine in the `Physical Schemata` section on the `MySQL Model` page. Doing this opens the routine editor docked at the bottom of the application. Double clicking the routine tab undocks the editor. Double click the title bar to redock it. 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,

There are two tabs at the bottom of the routine editor, the **View** and the **Privileges** tabs. Navigate between different tabs using the mouse or from the keyboard by pressing **Ctrl** + **Alt** + **Tab**.

10.1.2.1. The `Routine` Tab

From the `Routine` tab of the routine editor you can perform the following tasks:

- Rename the routine using the **Name** text box.

- Enter the SQL to create a routine using the **SQL** text area.

10.1.2.2. 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 7.3.10, “The Privileges Tab”.

10.2. Routine Groups

10.2.1. 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 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. Using the routine groups editor is described in Section 10.2.3, “The Routine Group Editor”.

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

- 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... option opens the routine group editor. Using the routine group editor is described in Section 10.2.3, “The Routine Group Editor”.

The cut and paste options 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 Digram by dragging and dropping them from this palette.

10.2.2. Adding Routine Groups to an EER Diagram

Routine groups can also be added to an EER Diagram using the `Routine Groups` tool on the vertical toolbar. To do this make sure that the `EER Diagram` tab is selected, and 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 drop down list box of schemata and a drop down color chart. Use these list boxes 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 easily 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 10.2.1, “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 6.2.8.1, “The General Tab”.

10.2.3. The Routine Group Editor

You can invoke the routine group editor by double clicking a routine group object on the EER Diagram canvas or by double clicking a routine group in the `Physical Schemata` section on the `MySQL Model` page. Doing 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,

There are two tabs at the bottom of the routine editor, the **ROUTINE GROUP** and the **PRIVILEGES** tabs. Navigate between different tabs using the mouse or from the keyboard by pressing **Ctrl** + **Alt** + **Tab**.

10.2.3.1. The `Routine Groups`Tab

From the `Routine Groups` tab of the routine groups editor you can perform the following tasks:

- Rename the routine group using the **NAME** text box.
- Add routines to the group by dragging and dropping them.
- Add comments to the routine group.

10.2.3.2. 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 7.3.10, “The Privileges Tab”.

10.2.3.3. 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 the properties accessible through the `Properties` palette see Section 6.8.1, “The Properties Palette”.

Chapter 11. 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.

11.1. Adding Layers to an EER Diagram

Layers are added to an EER Diagram using the `Layer` tool on the vertical toolbar. To do this 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.

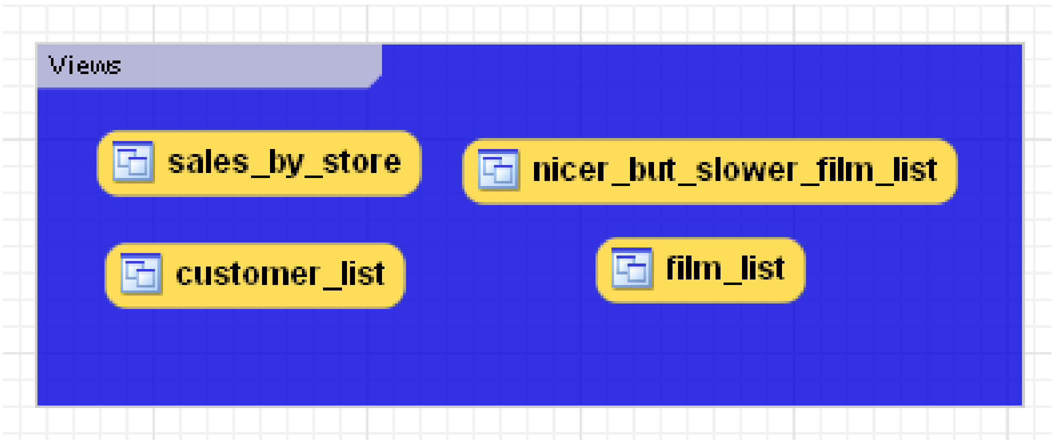
Clicking the mouse on this icon changes the mouse pointer to a layer pointer. You can also change the mouse pointer to a layer pointer by pressing the **L** key.

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 drop down color chart. Use this list box to select the color accent for the new layer. The color of your layer can easily be changed later using the `Properties` palette.

Create a layer by clicking anywhere on the EER Diagram canvas and, 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.

Find below an image of a layer containing a number of views:

Figure 11.1. The Layer object



Use the `name` property of the `Properties` palette to change the name of a layer.

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

- CUT '*LAYER_NAME*'.
- COPY '*LAYER_NAME*'
- DELETE '*LAYER_NAME*'

The cut and copy options 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.

11.1.1. Adding Objects to a Layer

Add an object to a layer by dragging and dropping it directly from the `Catalog` palette onto a layer. If you pick up an object from an EER diagram you need to press **Ctrl** as you drag it on to the layer, otherwise it will not be "locked" inside the layer.

Locking objects to a layer prevents their accidental removal. You cannot remove them simply by clicking and dragging; in order to remove an object, you also need to press the **Ctrl** 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.

11.2. 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. To do this, 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. Once an object is selected you can reset the `visible` property from the `Properties` palette.

For a list of the properties accessible through the `Properties` palette see Section 6.8.1, “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.

Chapter 12. 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.

12.1. 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. Using the note editor is described in Section 12.2, "The Note Editor".

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

- RENAME
- CUT '*NOTE_NAME*'
- COPY '*NOTE_NAME*'
- DELETE '*NOTE_NAME*'

The EDIT NOTE ... option opens the note editor. Using the note editor is described in Section 12.2, "The Note Editor".

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

Notes can only be added on the `MySQL Model` page.

12.2. The Note Editor

You can invoke the note editor by double clicking a note object in the `Model Note` section on the `MySQL Model` page. Doing 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.

Using the editor you can change the name of a note or its contents.

Chapter 13. Creating Text Objects

Text objects are applicable to an EER diagram only. 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.

13.1. Adding Text Objects to an EER Diagram

Text objects can be added to an EER Diagram using the `Text Object` tool on the vertical toolbar. To do this make sure that the `EER Diagram` tab is selected, and 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 drop down color chart. Use this list box to select the color accent for the new text object. The color of your text object can easily 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.

13.2. The Text Object Editor

You can invoke the text object editor by double clicking a text object on the EER Diagram canvas. Doing 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.

Using the editor you can change the name of a text object or its contents.

13.2.1. 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 the properties accessible through the `Properties` palette see Section 6.8.1, “The Properties Palette”.

There is no property in the `Properties` palette for changing the font used by a text object. To change the font used by a text object choose the `Fonts` tab of the MySQL Workbench options window. For more information, see Section 6.2.8.5, “The Fonts Tab”.

Chapter 14. Creating Images

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

14.1. Adding Images to an EER Diagram

Images can be added to an EER Diagram using the `Image` tool on the vertical toolbar. To add an image make sure that the `EER Diagram` tab is selected, and 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 a image by clicking anywhere on the EER Diagram canvas. This opens a file open dialog box. Select the desired image, and 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 options:

- CUT *'IMAGE'*
- COPY *'IMAGE'*
- EDIT IMAGE ...
- EDIT IN NEW WINDOW ...
- DELETE *'IMAGE'*

These menu options 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.

14.2. The Image Editor

You can invoke the image editor by double clicking a image object on an EER Diagram canvas. Doing 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 re-dock 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,

14.2.1. The `Image` Tab

From the `Image` tab of the image editor you can perform the following tasks:

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

Chapter 15. Reverse Engineering

Using 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. Reverse engineering using a MySQL DDL script applies to all versions of MySQL Workbench; reverse engineering a database directly from a MySQL server applies to commercial versions of MySQL Workbench only.

15.1. Reverse Engineering Using a Create Script

Reverse engineering using a create script is done by using the File, Import, Reverse Engineer MySQL Create Script ... menu options. Doing 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 DDL statements only. 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 in order to create an `ALTER` script. For more information, see Section 16.1.2, “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, a new physical schemata tab is created 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 in the same way that any other objects can.

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 4.1, “Importing a Data Definition SQL Script” for a tutorial on reverse engineering the `sakila` database.

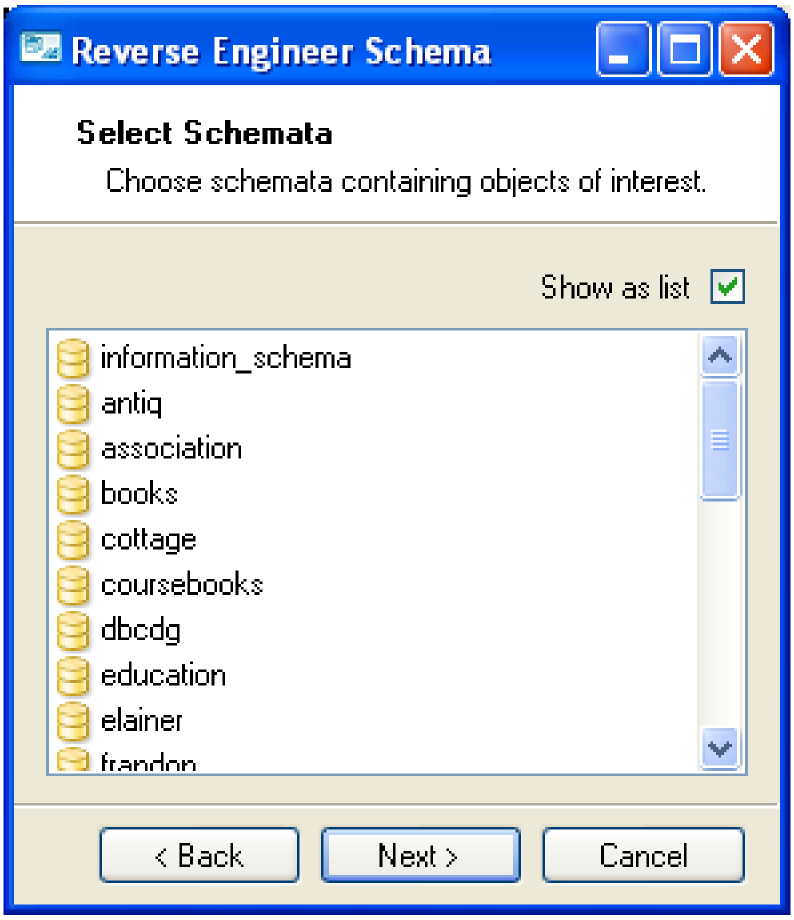
15.2. Reverse Engineering a Live Database (Commercial Version)

This section explains how to reverse engineer a live database using MySQL Workbench. This capability is available in commercial versions of MySQL Workbench only.

Choose the Reverse Engineer ... menu item found under the main menu, Database. Doing this opens the connection dialog window. You may choose to create a new connection or simply locate the server you wish to connect to and enter your credentials. Instructions for connecting to a server are described in detail in Chapter 17, *Server Connections (Commercial Version)*.

After confirmation that you have successfully connected to a MySQL server, clicking Next shows the schemata available on the selected MySQL server.

Figure 15.1. Selecting schemata



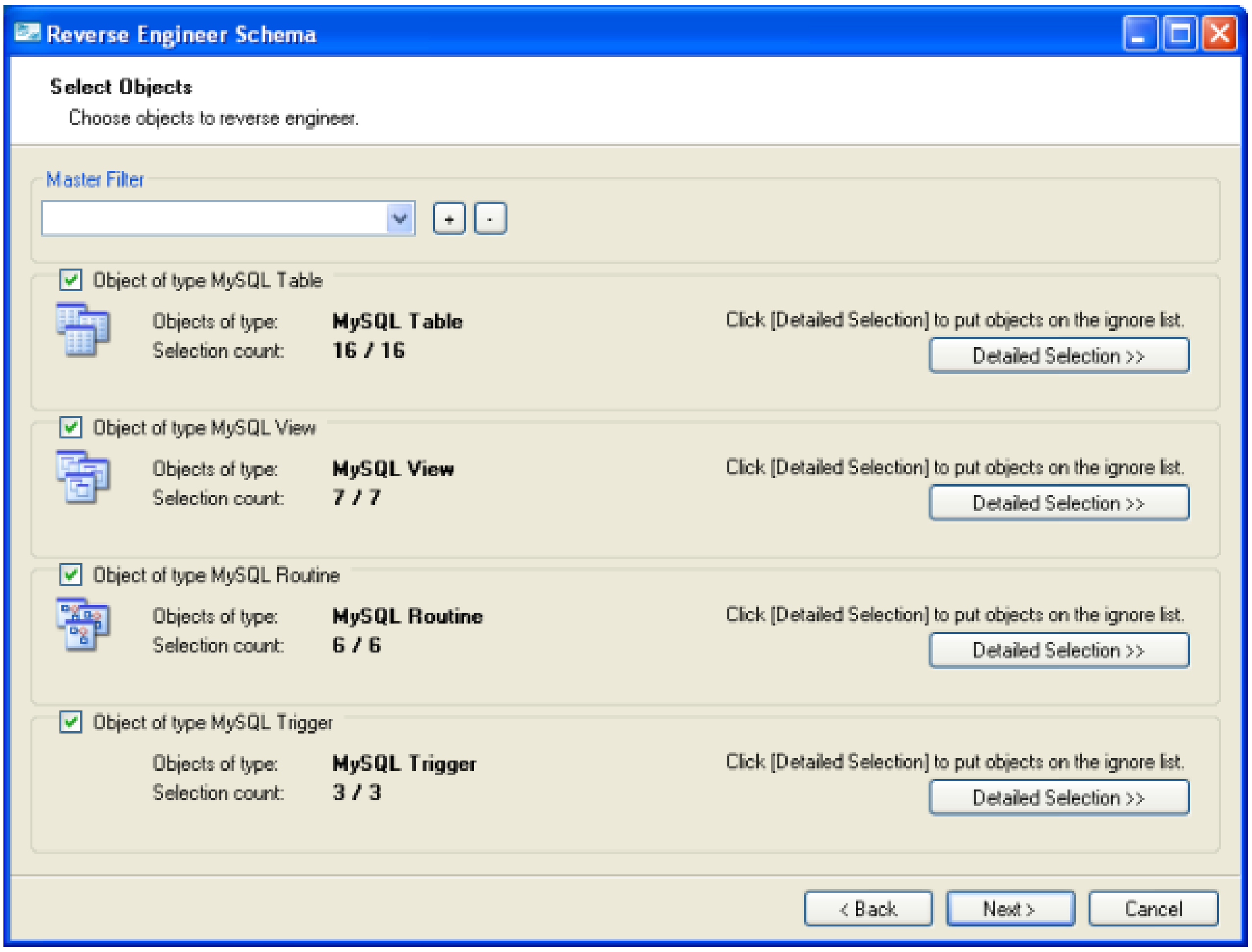
Clicking CANCEL returns you to the previous screen. For any of these screens, you can use the **Enter** key instead of the NEXT button and the **Esc** key instead of the CANCEL button.

Choose a schema or a number of schemata to reverse engineer. To select contiguous schemata press the **Shift** key and click the schemata you wish to select. Non-contiguous schemata are selected using the **Ctrl** key.

The subsequent screen shows a progress bar; reverse engineering will take a shorter or longer time depending upon the number of objects you are importing. You may view a log of the activities by clicking the ADVANCED button.

The next screen is the `Select Objects` screen. It is sectioned off by object type. This screen is of special interest if you do not wish to import all the objects from the existing database — this screen gives you the option of filtering which objects are imported. Each section has a DETAILED SELECTION > > button. Click this button if you do not want to import all the objects of a specific type.

Figure 15.2. Select objects



Reverse Engineer Schema

Select Objects
Choose objects to reverse engineer.

Master Filter

Object of type MySQL Table
Objects of type: MySQL Table
Selection count: 16 / 16
Click [Detailed Selection] to put objects on the ignore list.

Object of type MySQL View
Objects of type: MySQL View
Selection count: 7 / 7
Click [Detailed Selection] to put objects on the ignore list.

Object of type MySQL Routine
Objects of type: MySQL Routine
Selection count: 6 / 6
Click [Detailed Selection] to put objects on the ignore list.

Object of type MySQL Trigger
Objects of type: MySQL Trigger
Selection count: 3 / 3
Click [Detailed Selection] to put objects on the ignore list.

Move the objects you do not wish to import to the **EXCLUSION MASKS** text box by selecting them and clicking the > button.

You may also exclude all objects of a specific type by checking the check box found in the top left corner of each section.

On the next screen you can review the DDL script that will be parsed. After viewing the results of reverse engineering decide whether or not you wish to create an EER diagram of all the objects in the database. Press FINISH to complete the process.

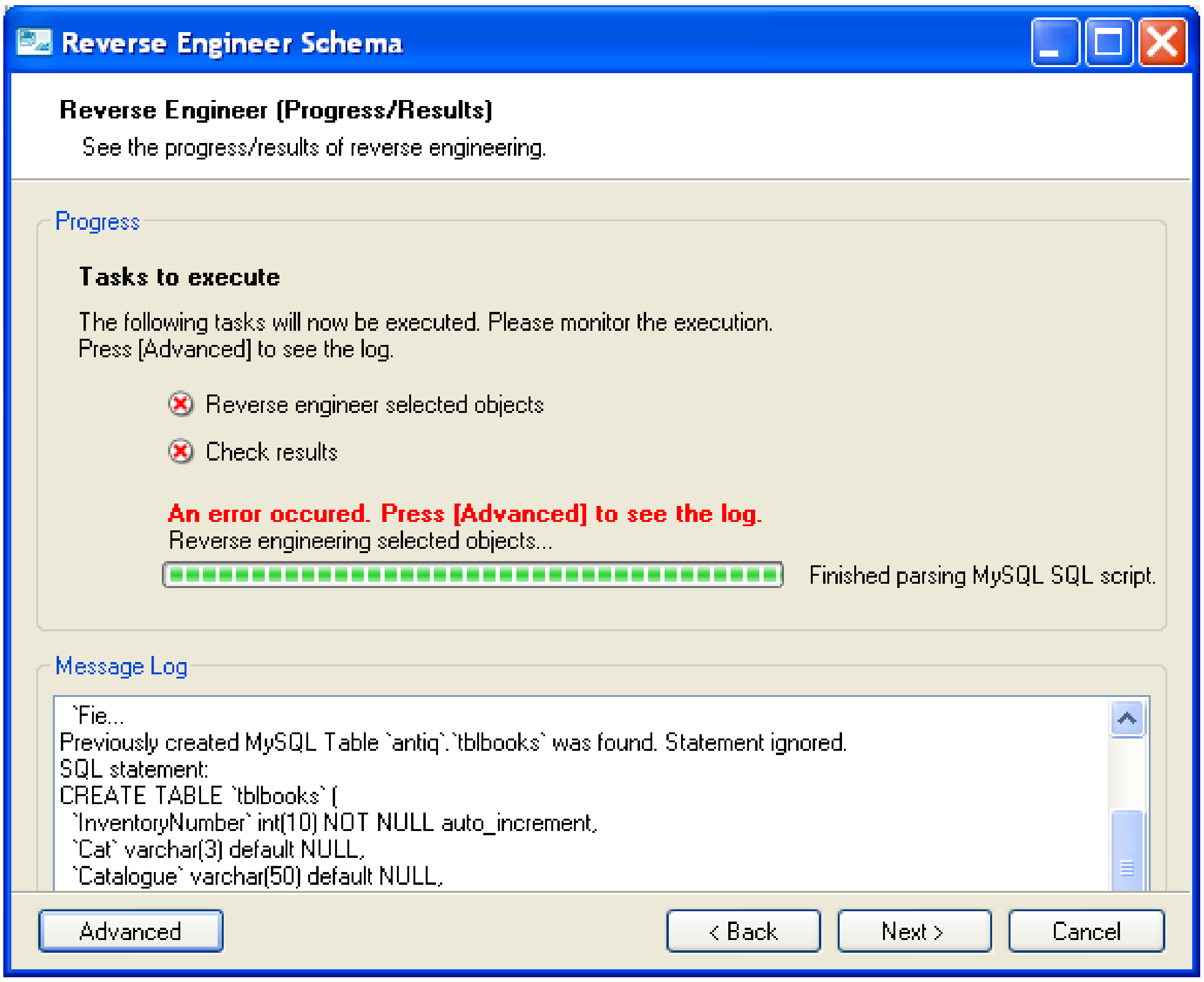
The objects created by reverse-engineering a schema may be manipulated in exactly the same way as any other objects.

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`.

15.2.1. 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 in the following fashion:

Figure 15.3. Duplicating an object name



Reverse Engineer (Progress/Results)
See the progress/results of reverse engineering.

Progress

Tasks to execute

The following tasks will now be executed. Please monitor the execution.
Press [Advanced] to see the log.

Reverse engineer selected objects

Check results

An error occured. Press [Advanced] to see the log.
Reverse engineering selected objects...

Finished parsing MySQL SQL script.

Message Log

```
 `Fie...
Previously created MySQL Table `antiq`.`tblbooks` was found. Statement ignored.
SQL statement:
CREATE TABLE `tblbooks` (
  `InventoryNumber` int(10) NOT NULL auto_increment,
  `Cat` varchar(3) default NULL,
  `Catalogue` varchar(50) default NULL,
```

The message log is displayed by clicking the ADVANCED button.

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.

If you encounter errors during the process of reverse engineering, clicking on the ADVANCED button displays the server error. For instance, forgetting to enter your passsword results in the following error:

```
Opening connection.
Access denied for user 'root'@'localhost' (using password: NO)
```

Chapter 16. Forward Engineering

It is possible to forward engineer a database using an SQL script or by connecting to a live database. Forward engineering to a live database is available for commercial versions of MySQL Workbench only.

16.1. Forward Engineering Using SQL Scripts

To create a script of your database model use the EXPORT option found under 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 a number of export options that you may choose from.

16.1.1. Creating a Schema

The SQL export options are as follows:

- Generate DROP statements (except DROP DATABASE)
- Generate separate CREATE INDEX statements
- Generate SHOW WARNINGS after every DDL statement
- Do not create users, just export privileges
- Generate INSERT statements for tables

Choosing the option `Generate separate CREATE INDEX statements` creates separate statements for index creation instead of creating indexes as part of a `CREATE TABLE` statement. To update the privileges of existing users as opposed to creating new users, select the `Do not create users, just export privileges` check box. Exporting privileges for non-existent users will result in errors when you execute the `CREATE` script. Exporting users that already exist, will also result in an error.

If you have added any records to a table using the `INSERT` tab of the MySQL Table Editor, choose the `Generate INSERT statements for tables` option. For more information about inserting records see Section 7.3.9, “The Inserts Tab”.

Use the **OUTPUT OPTIONS** frame to set the name of your script file. The NEXT button is disabled until a script file name is supplied.

Clicking NEXT takes you to the **SQL EXPORT FILTER** window where you can select the objects you wish to export. For a description of this window see Section 15.2, “Reverse Engineering a Live Database (Commercial Version)”.

The FINISH button saves the script file and exits. You may return to the previous screen using the BACK button.

Use the saved script to create a database.

16.1.2. Altering a Schema

The menu option for altering a schema, FORWARD ENGINEER 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. For instructions on importing a DDL script see Section 15.1, “Reverse Engineering Using a Create Script”.

Choose this menu option and on the first screen, **SQL SYNC OPTIONS**, you are asked to supply an input file. In most cases this will be the file that you imported into MySQL Workbench using the REVERSE ENGINEER MYSQL CREATE SCRIPT menu option found under the FILE, IMPORT options. In any case, this file should represent the state of the re-engineered database prior to alteration.

The differences between the original database and the altered database are displayed on the next screen, the **SQL DIFF TREE** window. To view the details, click the + button to the left of the model database.

Pressing NEXT brings you to the **SQL SCRIPT** screen. 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.

Use the saved script to update the database.

16.2. Forward Engineering to a Live Server (Commercial Version)

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 FORWARD ENGINEER ... option under the DATABASE menu option.

On the first screen select the objects that you wish to export.

The next screen offers SQL export options. These options are described in Section 16.1.1, “Creating a Schema”.

On the `Review Script` page you may review and edit the SQL script that will be executed. You next connect to a MySQL server; this process is described in Chapter 17, *Server Connections (Commercial Version)*.

Confirm the creation of a schema by connecting to the target MySQL server and issuing the **`SHOW DATABASES;`** command.

16.3. Database Synchronization (Commercial Version)

The SYNCHRONIZE ... menu item under the DATABASE menu option synchronizes your database model with an existing database.

Note

New tables or views can be synchronized with an existing database. Currently, changes to an existing table, such as the addition of a column, cannot be synchronized using MySQL Workbench.

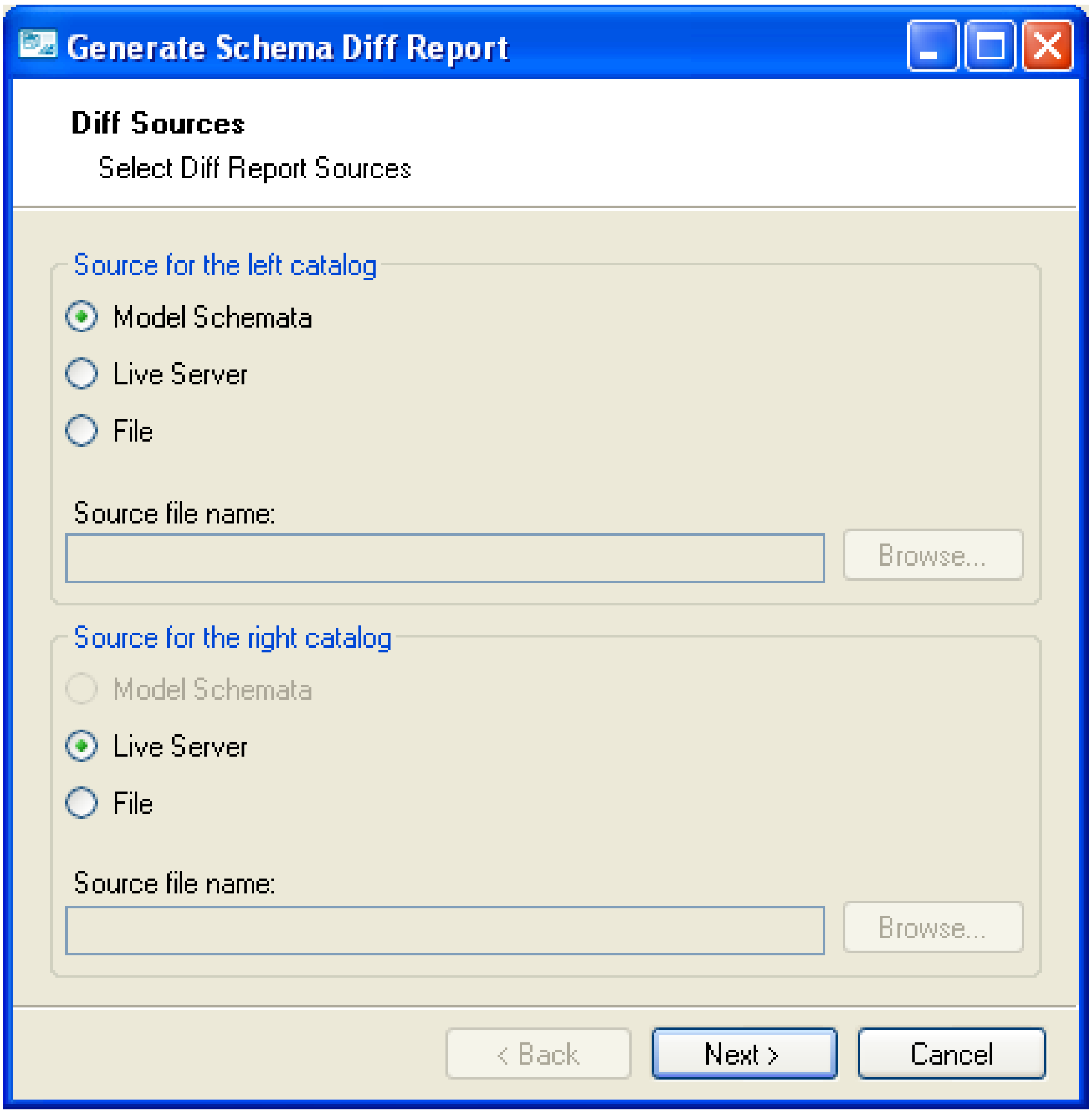
16.4. Creating a Schema Diff Report (Commercial Version)

Use the GENERATE SCHEMA DIFF REPORT menu item under the DATABASE menu option to create a report of the differences between a schema model and a live database or a script.

16.4.1. The Sources for a Diff Report

When creating a Diff report the first task is to specify the source. The various options are shown in the following figure.

Figure 16.1. The sources for a Diff report



Generate Schema Diff Report

Diff Sources

Select Diff Report Sources

Source for the left catalog

[✓ Model Schemata] Live Server File

Source file name:

Source for the right catalog

Model Schemata [✓ Live Server] File

Source file name:

The default value for the source for the left catalog is the model schemata.

If you choose the live server option for the right catalog, you must connect to a database server. For a description of this process see Chapter 17, *Server Connections (Commercial Version)*.

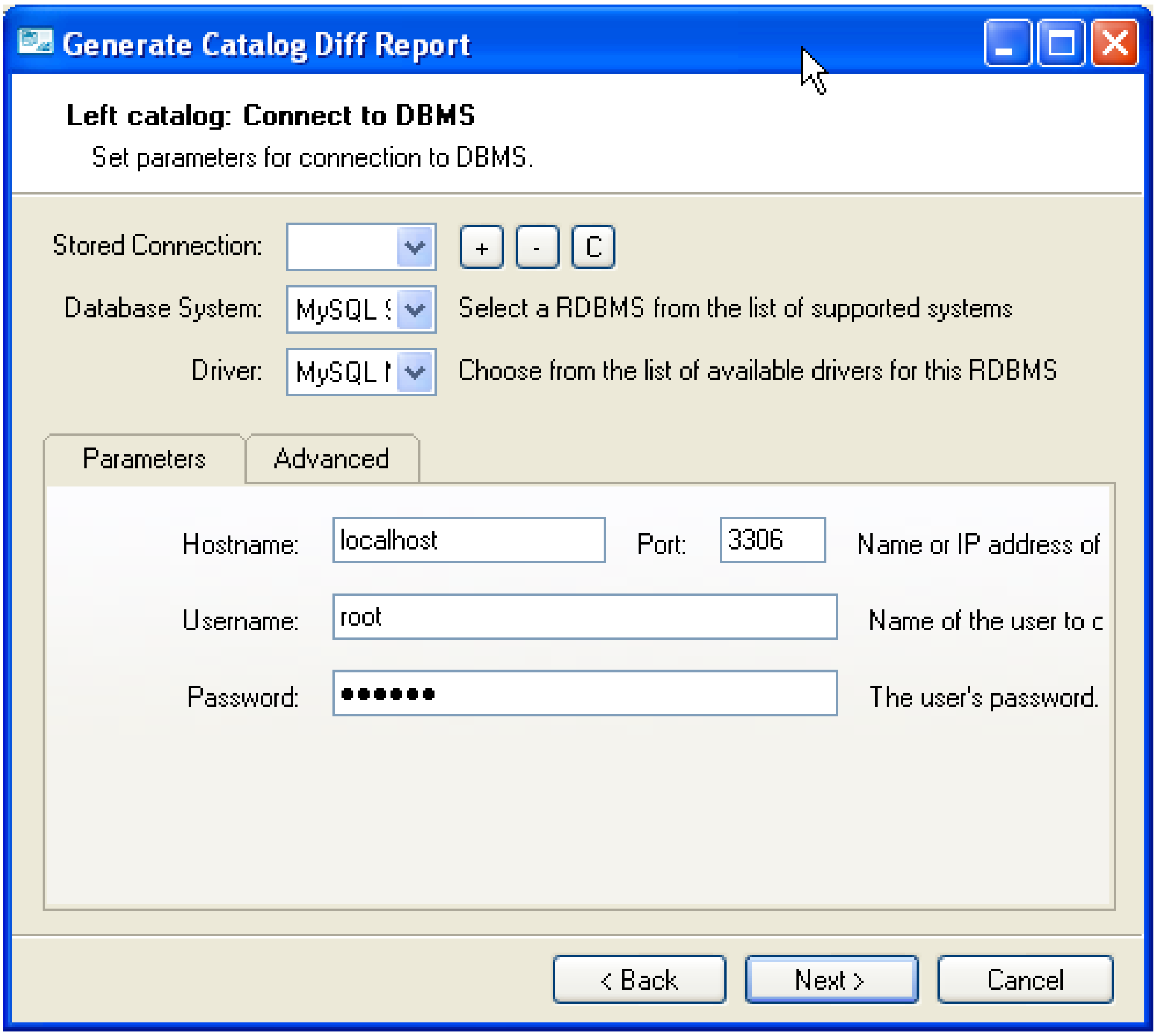
Chapter 17. Server Connections (Commercial Version)

The `Connections` dialog window is used with all the menu options found under the DATABASE menu option. These options are REVERSE ENGINEER ..., FORWARD ENGINEER ..., SYNCHRONIZE ..., and GENERATE CATALOG DIFF REPORT. Use this dialog window to open an ad hoc connection to a server, to open an existing stored connection, or to create new connections.

17.1. Server Connection Window

In the first window of the server connection wizard you define the properties of the connection.

Figure 17.1. Connect to a server window


Generate Catalog Diff Report

Left catalog: Connect to DBMS

Set parameters for connection to DBMS.

Stored Connection: [ ] + - C

Database System: MySQL S Select a RDBMS from the list of supported systems

Driver: MySQL N Choose from the list of available drivers for this RDBMS

Parameters | Advanced

Hostname: localhost Port: 3306 Name or IP address of

Username: root Name of the user to c

Password: •••••• The user's password.

< Back | Next > | Cancel

Note

The title of the window shown in Figure 17.1, "Connect to a server window" varies depending upon the operation you are performing.

You may choose a stored connection from the drop-down list box to the left of the **STORED CONNECTION:** label or select `<New Connection>` to create a new connection. The + button to the right of the list box is used when creating a new connection. Click it to open a dialog box to input the name of the new connection. If you wish to delete a connection, use the - button. The C button clears all the data related to a connection.

Use the **Database System** list box to select the type of RDBMS server you wish to connect to and select a driver from the **Driver** list box.

When you select a connection profile from the **Stored Connection** list box or when you create a new connection, the `Parameters` tab displays the following fields:

- **Hostname**: The host name or IP address of the MySQL server
- **Port**: The TCP/IP port on which the MySQL server is listening
- **Username**: The user name used to connect to the MySQL server.
- **Password**: The password used to connect to the MySQL server. Note: For security reasons passwords are not stored in the connection profile.

When you create a new connection or select a connection from the `Stored Connection` list box, the `Advanced` tab displays the following checkboxes:

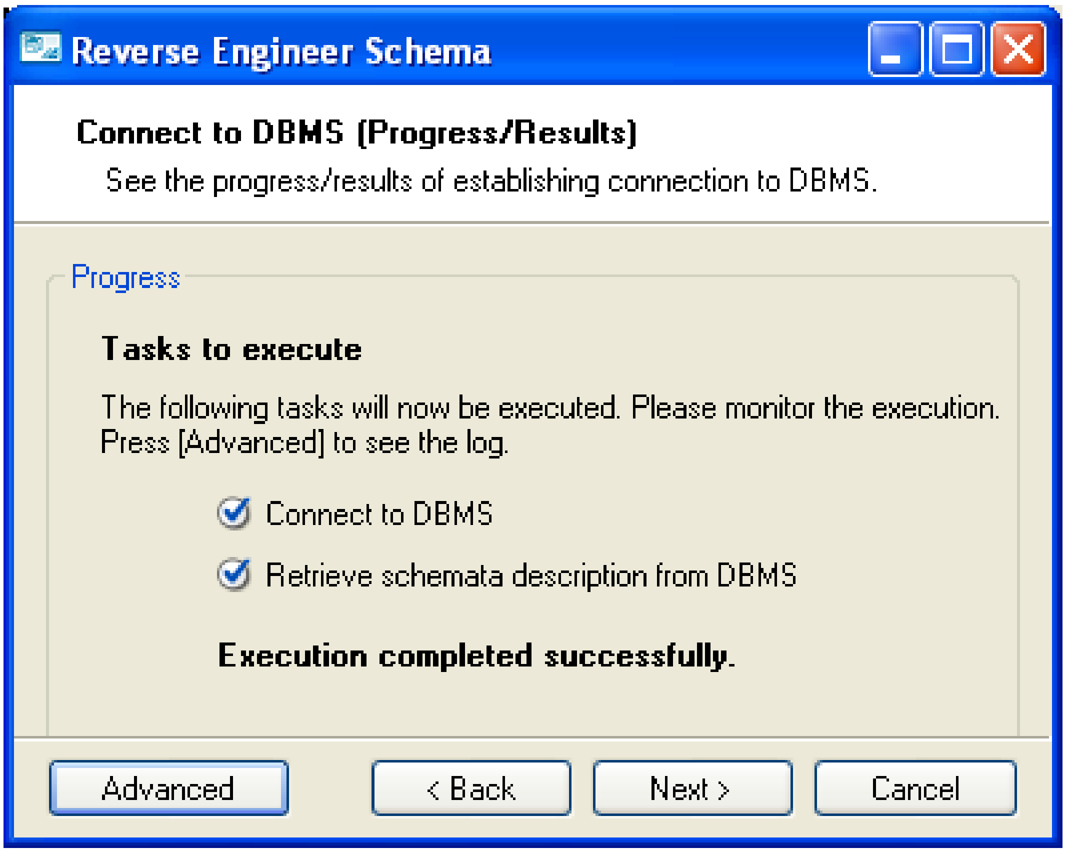
- **Use compressed protocol**: If checked, the communication between the application and the MySQL server will be compressed, which may increase transfer rates. This corresponds to starting a MySQL command-line tool with the `--compress` option.
- **Use SSL if available**: This option turns on SSL encryption.
- **Socket name**: Use this option to connect to a named pipe.

To edit an existing connection profile, first select its name from the **Stored Connection** drop-down listbox. The values that appear in the **Parameters** and **Advanced** tabs, can then be changed as required. Changes are then automatically saved for that connection profile.

17.2. Connection Confirmation

The next window confirms that a connection has been established and that the various schemata have been identified.

Figure 17.2. Connection confirmation


Reverse Engineer Schema

Connect to DBMS (Progress/Results)

See the progress/results of establishing connection to DBMS.

Progress

Tasks to execute

The following tasks will now be executed. Please monitor the execution.
Press [Advanced] to see the log.

✓ Connect to DBMS

✓ Retrieve schemata description from DBMS

Execution completed successfully.

If there are errors connecting to the MySQL server, click the ADVANCED button to view the details.

Subsequent screens vary depending upon the type of operation you are performing.

Chapter 18. Printing (Commercial Version)

The printing options are used to create printouts of your EER Diagrams and are found under the FILE menu. For creating *documentation* of your models see Section 6.2.5.1, “The DBDoc Model Reporting Dialog Window (Commercial Version)”.

Note

The printing options apply to the commercial versions of MySQL Workbench only.

18.1. Printing Options

The printing menu options are grayed if an EER Diagram is not active. The menu options are as follows:

- PAGE SETUP ...

  Use this option to choose the paper size, orientation, and margins.

- PRINT

  Use this option to send 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 ...

  Use this option to create a PDF file of your EER Diagram.

- PRINT TO PS ...

  Use this option to create a PostScript file of your EER Diagram.

Chapter 19. The Generic Runtime Environment (GRT) Shell

19.1. Introduction

The GRT is a thin C layer, inspired by Objective C, which allows for dynamic typing and dynamic data objects. The GRT provides a means for expanding MySQL Workbench. Through the use of the GRT, MySQL Workbench can support new behavior and data sources using code written in languages such as C++ and Lua.

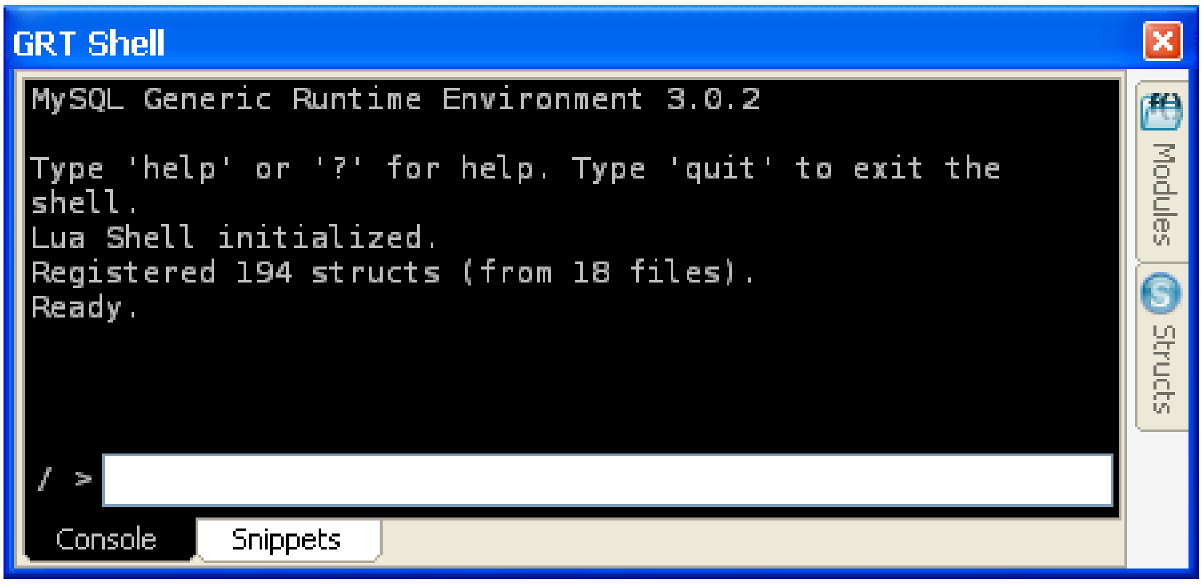
The GRT is not only useful for expanding MySQL Workbench. By using a script file from within the GRT shell you can perform repetitive tasks programmatically from the command line.

The preferred development language is `Lua`, a lightweight scripting language expressly designed for extending applications. For more information about this language see lua.org.

19.2. Exploring the GRT Shell

To open the GRT shell click on the VIEW menu and choose the GRT SHELL option under the ADVANCED submenu. However, the simplest way to open the GRT Shell is to use the **Ctrl F3** key combination. If you haven't docked the GRT shell window you should see something similar to the following:

Figure 19.1. The GRT shell (Windows)



```
MySQL Generic Runtime Environment 3.0.2

Type 'help' or '?' for help. Type 'quit' to exit the
shell.
Lua Shell initialized.
Registered 194 structs (from 18 files).
Ready.
```

By default the GRT shell opens docked at the bottom of the application The GRT shell itself is the default tab labeled **CONSOLE** on the left. Beside it is the `Snippets` tab, used for saving code snippets.

19.3. The Shell

The GRT shell is principally used for running Lua scripts or typing Lua commands directly. However, you can also access the GRT Scripting Library functions and global functions and objects. To see the available commands type “`?`”.

Some OS-specific commands are also available. For instance, under Windows you can clear the screen by typing `cls`. Unlike most shells, you can cut and paste text to and from the shell window.

The `Snippets` tab functions as 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, there may be any number of tabs to the right of the `Snippets` tab. These tabs will be labeled with the names of the script files. 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.

For information on running script files, type `? run` at the GRT shell prompt. The following message is displayed:

```
Shell Command - shell.run
```

```
-------------------------
Load and execute a lua script file.

run filename

Parameters:
filename       File that should be loaded and executed.

Examples:
run scripts/test.lua
Runs the script scripts/test.lua.
```

Opening the GRT shell opens additional palettes docked to the immediate left of the shell window or docked on the left side of the application.

- The `Modules and Structs Palettes`
- The `GRT Tree`
- The `GRT Inspector`

Discussion of these palettes follows.

19.4. The Modules and Structs Palettes

On the right side bar of the GRT shell window are the `Structs` and `Modules` palettes. Expand these views by mousing over them. To make a view fixed, click the push pin icon in the title bar on the left.

Unlike the GRT shell window, the `Modules` and the `structs` windows cannot be be detached from the main application window.

19.4.1. The `Modules` Tab

View the **MODULES** tab by mousing over it. To keep this tab open click the push pin icon in the top right corner.

A module can be either a Python or Lua script or a Java class file. Information about modules appears in the window below the module tree. For example, go to the `Modules` tab and click on the `ReverseEngineeringGeneric` module. Double click a module and you will see its methods.

19.4.2. The `Struct` Tab

View the **STRUCT** tab by mousing over it. To keep this tab open click the push pin icon in the top right corner.

A `struct` is a user-defined data type formed by combining primitive data types. This tab shows the definitions of the structs used by the objects in the `Modules` view.

When the `Structs` tab is selected, right clicking a structure in the list opens a pop-up menu with the options:

- Refresh – refresh the list
- Order by Name – group by the object name
- Order by Hierarchy – group by inheritance
- Order by Package – group by functionality
- Copy Struct Name – copy the struct name to the clipboard

The default view for this tab is by name; this view simply 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 `GRT Object` — the parent object from which all other objects are derived.

19.5. The GRT Tree Palette

When the GRT shell is active, the `Globals Tree` is found on the right side of the screen .

Selecting an object in the GRT tree displays its properties in the `GRT Inspector` panel found immediately below the `GRT Tree Panel`

Right clicking an object in the GRT Tree opens a pop-up window with the menu options, REFRESH and COPY OBJECT PATH.

19.6. The GRT Inspector

The GRT inspector displays the properties of the object that is currently selected in the `GRT Global Tree`.

19.7. Invoking the GRT Shell From the Command Line

In addition to using the GRT shell from within the MySQL Workbench, you can invoke it directly from the command line. If the location of the MySQL Workbench is not included in the `PATH` variable, navigate to the installation directory and find the `MySQL-GrtShell.exe` file.

Execute this file by typing:

```
C:\> MySQLGrtShell.exe
```

Doing this opens the GRT shell window as a stand alone window.

The default shell is the `Lua` shell and is indicated by the `/ >` prompt.

Chapter 20. Plug-ins

Using plugins you can write your own modules to perform a variety of tasks. To do this you need to know the Lua scripting language and be familiar with the various objects used by MySQL Workbench. For more information on this topic see Chapter 19, *The Generic Runtime Environment (GRT) Shell*.

20.1. The `catalog_utils.grt.lua` Module

The best way to learn about plugins is to examine one. An example Lua plugin is located in the `C:\Program Files\MySQL\MySQL Workbench version\modules` directory. In that directory you can find the `catalog_utils.grt.lua` Lua file. This is the plugin for copying SQL to the clipboard.

Any plugin must contain at least three functions:

- `getModuleInfo()` – This function tells the GRT that the file is a module.
- `getPluginInfo()` – This function tells MySQL Workbench that the module is a function and registers that plugin.
- The function or functions that perform the work of the module.

Open the `catalog_utils.grt.lua` file in a text editor. and you can see that it has the required `objectPluginInput()` and `getPluginInfo()` functions, a helper function called `objectPluginInput()` and the function that performs the work of this module, `copySQLToClipboard()`.

20.1.1. Using the GRT Shell

You can use the GRT shell to help understand the `catalog_utils.grt.lua` module. Open the GRT shell by pressing **Ctrl F3** or by selecting the GRT SHELL menu option found under the VIEW, ADVANCED menu options.

Notice that the `getPluginInfo` function makes use of the `grtV` module by invoking its `getObj` function. The `grtV` module is a built-in Lua module that assists with working from the GRT shell. To find out more about the `grtV` module, type `? grtV` in the GRT shell console window. You should see something similar to the following:

```
GRT Value Management Library - grtV
-----------------------------------
A library that contains functions to work with GRT values.

clearList               child               fromXml
getContentType          getKey              getListItemByObjName
getn                    getGlobal           insert
load                    newDict             newList
newObj                  remove              save
setGlobal               toLua               toXml
typeOf
```

Typing `help grtV.<command>` displays help on a specific command. For example, entering **`help grtV.newObj`** displays the following output.

```
GRT Value Management Library - grtV.newObj
------------------------------------------
Creates a new GRT object initialized with the optional given values. All simple values
("int", "string", "real") are initialized and all lists and dicts are created.
Object references are left null.

grtV.newObj (structName[, initValuesDict])

Parameters:
structName         Struct name of the object to create
initValuesDict     A dictionary containing initial values for object fields.


Examples:
rdbmsMgmt= grtV.newObj("db.mgmt.Management",
{name="rdbmsManagement", owner=app})
Create a new object from the struct "db.mgmt.Management"
```

The `grtV.newObj` function is used by both the `objectPluginInput` and the `getPluginInfo` functions of the `catalog_utils.grt.lua` module. A `structName` is the first parameter for the `grtV.newObj` function. In the case of the `objectPluginInput` function in `catalog_utils.grt.lua` module, the `structName` is `app.PluginObjectInput`. To find out more about this struct, locate it in the `Structs` window.

20.2. Accessing Plugins

Plugins can be accessed by calling them from the GRT Shell. For more information on this topic see Section 19.3, “The Shell”. You can also add entries to the `main_menu.xml` file, found in the `C:\Program Files\MySQL\MySQL Workbench` *`version`*`\data` directory.

To determine how to add entries to the `main_menu.xml` file, open it in a text editor and search for `Plugins Menu`. Performing this search takes you to the top level `Plugins` entry. Beneath this entry find a sample submenu entry that has been commented out. This entry should look something like the following:

```
<value type="object" struct-name="app.MenuItem" id="com.mysql.wb.menu.plugins.plugin_name">
  <link type="object" key="owner" struct-name="app.MenuItem">com.mysql.wb.menu.plugins</link>
  <value type="string" key="name">menu_name</value>
  <value type="string" key="caption">menu_caption</value>
  <value type="string" key="itemType">cascade</value>
  <value type="list" key="subItems" content-type="object" content-struct-name="app.MenuItem">
</value>
```

Use this entry as a guideline for creating submenus under the PLUGINS top level menu entry.

Warning

Ensure that any XML added to the `main_menu.xml` file is well-formed, otherwise MySQL Workbench may crash on startup.

Chapter 21. 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 option. 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 tasks performed by the validation modules are outlined in what follows.

21.1. General Validation

The types of validation and examples that violate validation are listed in what follows:

- **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- or write-only by role definition
  - Placeholder objects left over from reverse engineering

21.2. MySQL-Specific Validation

The types of MySQL-specific validation and examples that violate validation are listed in the following.

- **Integrity Violation**
  - An object name longer than the maximum allowed

  - A foreign key defined for an engine type that doesn't support foreign keys (not yet implemented)
  - A view or routine that references a non-existent 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

Chapter 22. Customizing DBDoc Model Reporting Templates

This document aims to provide 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 document does not attempt to explain the Google Template System in detail. The Google document How To Use the Google Template System provides a useful overview of how the Google Template System works.

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:

1. Template Include
2. Comment
3. Set delimiter
4. Pragma
5. Variable
6. Section start and Section end

The last two are the most commonly used in MySQL Workbench templates and these important markers will be briefly described in the following sections.

1. **Variables**

   The use of variables in the templates is straightforward. Any variables denoted by markers in the template file, will be 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 dicionaries 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}} will be 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.

2. **Sections**

   Sections are used to perform iteration in the templates. When MySQL Workbench exchanges the variables in a section for data it will do 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.

   Again, this is best illustrated by example:

   ```
   {{#SCHEMATA}}
   Schema: {{SCHEMA_NAME}}
   {{/SCHEMATA}}
   ```

   In the previous code snippet the section start is indicated by the `{{#SCHEMATA}}` marker. The end of the section is indicated by the `{{/SCHEMATA}}` marker. When the template is processed, MySQL Workbench will note the section and iterate the section 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
   ```

That is, the model contains two schemata, Airlines and 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:

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 then loaded from the root to form the dictionary tree.

> **Note**
>
> If a template has no sections in it, then any variables used in the template will be 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) then no data will be generated in the output file for that marker.

Evaluation of variables

The tree structure of the data dictionaries is important when it comes to evaluation of variables. As variables are defined in data dictionaries, their associated value only has meaning 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 will check 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 then the parent data dictionary will be checked for the variable's value and so on up the tree until the main data dictionary, or root, is reached.

This can best be illustrated by an example. Assume 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 will produce no output, even for a model that contains many columns. In this example the only data dictionary active is the main dictionary. `COLUMN_NAME` however 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. Referring to the table Data Dictionary Hierarchy Tree explains why. 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 in order for a dictionary to be involved in variable lookup, its associated section must currently be active.

So in order to achieve the desired output we would need the template to 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 will be 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 will be iterated over a matching number of times, and `SCHEMA_NAME` will accordingly have the correct value on each iteration.

It's important to always consider which dictionary needs to be active (and which parents) for a variable to be evaluated correctly. In the following section you will find a table that helps you identify section requirements.

22.1. Supported Template Markers

A list of supported markers follows. These markers can be used in any template, including custom templates.

Marker text	**Type**	**Data Dictionary defined in (if variable) or parent dictionary (if section)**
TITLE	Variable	MAIN
GENERATED	Variable	MAIN
STYLE_NAME	Variable	MAIN
SCHEMA_COUNT	Variable	MAIN
PROJECT_TITLE	Variable	MAIN
PROJECT_NAME	Variable	MAIN
PROJECT_AUTHOR	Variable	MAIN

PROJECT_VERSION	Variable	MAIN
PROJECT_DESCRIPTION	Variable	MAIN
PROJECT_CREATED	Variable	MAIN
PROJECT_CHANGED	Variable	MAIN
TOTAL_TABLE_COUNT	Variable	MAIN
TOTAL_COLUMN_COUNT	Variable	MAIN
TOTAL_INDEX_COUNT	Variable	MAIN
TOTAL_FK_COUNT	Variable	MAIN
SCHEMATA	Section	MAIN
SCHEMA_NAME	Variable	SCHEMATA
SCHEMA_ID	Variable	SCHEMATA
TABLE_COUNT	Variable	SCHEMATA
COLUMN_COUNT	Variable	SCHEMATA
INDICES_COUNT	Variable	SCHEMATA
FOREIGN_KEYS_COUNT	Variable	SCHEMATA
TABLES	Section	SCHEMATA
TABLE_NAME	Variable	TABLES
TABLE_ID	Variable	TABLES
COLUMNS_LISTING	Section	TABLES
COLUMNS	Section	COLUMNS_LISTING
COLUMN_KEY	Variable	COLUMNS
COLUMN_NAME	Variable	COLUMNS
COLUMN_DATATYPE	Variable	COLUMNS
COLUMN_NOTNULL	Variable	COLUMNS
COLUMN_DEFAULTVALUE	Variable	COLUMNS
COLUMN_COMMENT	Variable	COLUMNS
COLUMN_ID	Variable	COLUMNS
COLUMN_KEY_PART	Variable	COLUMNS (if detailed)
COLUMN_NULLABLE	Variable	COLUMNS (if detailed)
COLUMN_AUTO_INC	Variable	COLUMNS (if detailed)
COLUMN_CHARSET	Variable	COLUMNS (if detailed)
COLUMN_COLLATION	Variable	COLUMNS (if detailed)
COLUMN_IS_USERTYPE	Variable	COLUMNS (if detailed)
INDICES_LISTING	Section	TABLES
INDICES	Section	INDICES_LISTING
INDEX_NAME	Variable	INDICES
INDEX_PRIMARY	Variable	INDICES
INDEX_UNIQUE	Variable	INDICES
INDEX_TYPE	Variable	INDICES
INDEX_KIND	Variable	INDICES
INDEX_COMMENT	Variable	INDICES
INDEX_ID	Variable	INDICES
INDEX_COLUMNS	Section	INDICES
INDEX_COLUMN_NAME	Variable	INDEX_COLUMNS
INDEX_COLUMN_ORDER	Variable	INDEX_COLUMNS
INDEX_COLUMN_COMMENT	Variable	INDEX_COLUMNS
INDEX_KEY_BLOCK_SIZE	Variable	INDEX_COLUMNS (if detailed)
REL_LISTING	Section	TABLES

REL	Section	REL_LISTING
REL_NAME	Variable	REL, FOREIGN_KEYS
REL_TYPE	Variable	REL, FOREIGN_KEYS
REL_PARENTTABLE	Variable	REL, FOREIGN_KEYS
REL_CHILDTABLE	Variable	REL, FOREIGN_KEYS
REL_CARD	Variable	REL, FOREIGN_KEYS
FOREIGN_KEY_ID	Variable	REL
FOREIGN_KEYS	Section	SCHEMATA
FK_DELETE_RULE	Variable	FOREIGN_KEYS
FK_UPDATE_RULE	Variable	FOREIGN_KEYS
FK_MANDATORY	Variable	FOREIGN_KEYS
TABLE_COMMENT_LISTING	Section	TABLES
TABLE_COMMENT	Variable	TABLE_COMMENT_LISTING
DDL_LISTING	Section	TABLES
DDL_SCRIPT	Variable	DDL_LISTING

Using the table

The table shows which variables are defined in which sections. The variable should be used in its correct section, otherwise its value will not be displayed.

Note

It should be remembered though that the data dictionaries used to perform the lookup form a hierarchical tree, so it is possible to use a variable defined in a parent section, in a child section.

22.2. 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, that 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, you make a custom template based on the `Text Basic`. The following procedure demonstrates this.

1. First you need to make a copy of the template on which you are going to base your custom template. To do this navigate to the folder where the templates are stored. Assuming 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. Then make a copy of the template folder you wish to base your new template on. In this case a copy of the `Text_Basic.tpl` folder is made. The copy can be given any suitable name, for example, `Custom_Basic.tpl`.

3. Now the `info.xml` file needs to be edited, 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-AAEE49B2261E}" 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>
```

Two objects are defined in the file. 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 screen.

4. The first thing you need to change are 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. The `info.xml` file should then be edited 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. The next step is to create the new template file. Again this 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_NOTNULL}}{{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.

8. The above 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 will also be displayed as a result of the use of the `{{GENERATED}}` variable.

9. 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 then click FINISH to generate the report. Finally, navigate to the output directory to view the finished report.

Chapter 23. MySQL Workbench FAQ

Questions

- 23.1: MySQL Workbench 5.0 appears to run slowly. How can I increase performance?

Questions and Answers

23.1: 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 that is 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.

Appendix A. MySQL Workbench Change History

The following sections outline the changes between versions for MySQL Workbench.

A.1. Changes in Release 5.1

A.1.1. Changes in MySQL Workbench 5.1.7 (Not yet released)

This section documents all changes and bug fixes that have been applied since the release of 5.1.6.

Bugs fixed:

- MySQL Workbench 5.1.7 for MacOSX crashed on startup. The reason was that it was looking for `libmysqlclient.15`, which was not found in `/usr/local/mysql/lib`. The error generated was:

  ```
  Process:         MySQLWorkbench [14915]
  Path:            /Applications/MySQLWorkbench.app/Contents/MacOS/MySQLWorkbench
  Identifier:      com.sun.MySQLWorkbench
  Version:         ??? (???)
  Code Type:       X86 (Native)
  Parent Process:  launchd [95]

  Date/Time:       2009-02-02 18:53:52.120 +0100
  OS Version:      Mac OS X 10.5.6 (9G55)
  Report Version:  6

  Exception Type:  EXC_BREAKPOINT (SIGTRAP)
  Exception Codes: 0x0000000000000002, 0x0000000000000000
  Crashed Thread:  0

  Dyld Error Message:
    Library not loaded: /usr/local/mysql/lib/libmysqlclient.15.dylib
    Referenced from: /Applications/MySQLWorkbench.app/Contents/MacOS/MySQLWorkbench
    Reason: image not found
  ```

  Note that MySQL was installed, but the specific version of client library required was not present. (Bug#42550)

A.1.2. Changes in MySQL Workbench 5.1.6 (Not yet released)

This section documents all changes and bug fixes that have been applied since the release of 5.1.5.

Bugs fixed:

- A dialog displayed a message with a missing filename. The message displayed was:

  ```
  Import of SQL script file '' has finished successfully.
  ```

  Note the filename is missing from the message.

  This dialog is located in the File, Import, Reverse Engineer SQL CREATE Script wizard. It is displayed on the page after importing the file, clicking Next and then Execute. (Bug#39922)

- The Advanced button displayed the text label &Advanced.

  This button is located in the File, Import, Reverse Engineer SQL CREATE Script wizard. It is displayed on the page after importing the file. (Bug#39921)

A.1.3. Changes in MySQL Workbench 5.1.4 (Not yet released)

This section documents all changes and bug fixes that have been applied since the release of 5.1.3.

Functionality added or changed:

- There was a problem where relationships that were hidden could then not be selected in order to bring up their relationship editor. Relationships can now be selected as objects in the Layer window. Once selected, the relationship's `visible` property can be set to `True` in the Properties window, thus making the relationship visible again. (Bug#40167)

Bugs fixed:

- Loading a model using the Linux version of Workbench resulted in a crash. However, the model loaded correctly with the Windows versions of Workbench. (Bug#39992)
- A model created using the Windows version of Workbench caused the Linux version of Workbench to crash on loading the model. (Bug#39983)

A.2. Changes in Release 5.0

A.2.1. Changes in MySQL Workbench 5.0.30 (18th February 2009)

This section documents all changes and bug fixes that have been applied since the release of 5.0.29.

Bugs fixed:

- MySQL Workbench crashed when the mouse wheel was used. If you scrolled the **Options** tab of the **Table Editor**, closed the **Table Editor** and then used the mouse wheel again on the **MySQL Model** page, MySQL Workbench crashed. (Bug#42847)
- Introducing a UserType into a model caused the File, Export, Forward Engineer SQL CREATE Script wizard to crash. Further, peforming a Plugins, Objects, Copy SQL to Clipboard operation also caused MySQL Workbench to crash. (Bug#42085)
- The Forward Engineer SQL CREATE Script wizard failed to generate a script correctly.

  This happened when using the File, Export, Forward Engineer SQL CREATE Script facility. If, in the wizard, **Object of type MySQL table** was selected, and then all tables added to the **Exclusion Masks** pane, before moving back the required table to the **Objects to Process** pane, the script was generated for the entire database rather than the selected table. (Bug#41475)
- When a diagram was renamed, the history displayed:

  ```
  Rename 'new name' to 'new name
  ```

  It should have instead displayed:

  ```
  Rename 'old name' to 'new name'
  ```

  (Bug#41355)
- If a model contained a View that was using a Function, and an attempt was made to Synchronize the database, then an error was generated such as:

  ```
  Error 1305: FUNCTION `bleble` does not exist
  ```

  A similar error was also generated if the Forward Engineer SQL CREATE Script wizard was used. (Bug#40846)
- The viewport, which is the combobox in the top right corner of Workbench, did not scale to less than 40%. However, resizes above 40% worked fine. (Bug#39607)
- The Forward Engineer SQL ALTER Script wizard produced an erroneous script.

  If Forward Engineer SQL CREATE Script was used to generate a script and this was then used as an input to Forward Engineer SQL ALTER Script, without having made any changes to the model, then an ALTER script with no changes should be produced. However, the ALTER script showed many changes, even though no changes had been made to the model. (Bug#37709)

A.2.2. Changes in MySQL Workbench 5.0.29 (12th December 2008)

This section documents all changes and bug fixes that have been applied since the release of 5.0.28.

Bugs fixed:

- Workbench crashed when objects other than tables were moved out of a layer. (Bug#41358)
- In the EER Diagram view an icon was not displayed for Not-NULL items. (Bug#41326)
- When a diagram was renamed, the label of the corresponding tab was not automatically updated. However, when the focus was changed, the text was correctly updated. (Bug#38867)
- The table figures in the Diagram view had insufficient information. They did not display information such as constraints or default values. (Bug#38553)
- When the grid was activated, dragged objects on layers were incorrectly placed with an offset of -1,-1. (Bug#35989)
- The last column in a table disappeared in the table editor, and it was not possible to add further columns. (Bug#35905)

A.2.3. Changes in MySQL Workbench 5.0.28 (6th December 2008)

This section documents all changes and bug fixes that have been applied since the release of 5.0.27.

Bugs fixed:

- If you attempted to select several tables in the table list of the **MySQL Model** view, and you accidentally included the Add Table button in your selection, then a message box appeared warning of an unknown exception:

  ```
  "Unknown Exception caught in: c:\documents and settings\mysqldev\my documents\visual
  studio 2005\projects\workbench\backend\windows\wb.wr\src\Wb.h at line 1010"
  ```

  The program did not crash. Only the messagebox appeared. (Bug#41201)

- If two foreign keys were created in a table that referenced a second table and then an attempt was made to delete the relations and the referenced table, MySQL Workbench crashed. (Bug#41025)
- When clicking the + and - buttons in the **Physical Schemata** pane of the **MySQL Model** tab, an `Unhandled Exception` was generated:

  ```
  System.Runtime.InteropServices.SEHException: External component has thrown an exception.
  ```

  (Bug#40971)

- The **Referenced Column** pane of the **Foreign Key** tab became cleared if the foreign key was renamed. Subsequently, attempting to choose a **Referenced Column** did not display a link in the **EER Diagram** view. In order to get foreign key relationships working again it was necessary to de-select the checkboxes from the **Columns** pane, re-select them, and then select the **Referenced Column** pane. (Bug#40649)
- When a table was renamed the inserted data was lost. (Bug#40327)
- A complex EER diagram threw an exception whenever an action was peformed on it. However, other diagrams in the same MWB file functioned correctly.

  The exception generated was:

  ```
  System.Runtime.InteropServices.SEHException: Un composant externe a levé une exception.
     à wb.ModelViewForm.handle_mouse_button(ModelViewForm* , MouseButton , Boolean , Int32
  , Int32 , EventState )
     à MySQL.Workbench.ModelViewForm.OnMouseUp(MouseEventArgs e, Int32 X, Int32 Y, Keys
  keystate, MouseButtons buttons)
     à MySQL.GUI.Workbench.ModelViewForm.CanvasPanel_MouseUp(Object sender, MouseEventArgs
  e)
     à System.Windows.Forms.Control.OnMouseUp(MouseEventArgs e)
     à MySQL.Utilities.WindowsCanvasViewerPanel.OnMouseUp(MouseEventArgs e)
     à System.Windows.Forms.Control.WmMouseUp(Message& m, MouseButtons button, Int32
  clicks)
     à System.Windows.Forms.Control.WndProc(Message& m)
     à System.Windows.Forms.ScrollableControl.WndProc(Message& m)
     à System.Windows.Forms.Control.ControlNativeWindow.OnMessage(Message& m)
     à System.Windows.Forms.Control.ControlNativeWindow.WndProc(Message& m)
     à System.Windows.Forms.NativeWindow.Callback(IntPtr hWnd, Int32 msg, IntPtr wparam,
  IntPtr lparam)
  ```

  (Bug#39360)

A.2.4. Changes in MySQL Workbench 5.0.27 (7th November 2008)

This section documents all changes and bug fixes that have been applied since the release of 5.0.26.

Functionality added or changed:

- There was a problem where relationships that were hidden could then not be selected in order to bring up their relationship editor. Relationships can now be selected as objects in the Layer window. Once selected, the relationship's `visible` property can be set to `True` in the Properties window, thus making the relationship visible again. (Bug#40167)

Bugs fixed:

- When a stored routine was edited, the edit cursor jumped back to the start of the page unless typing was constant. (Bug#40426)
- When using the Copy Insert to Clipboard menu item the generated SQL code was incorrect. The "S" was missing from "VALUES" and the data was not included. This resulted in SQL code such as:

  ```
  INSERT INTO `table1` (`table1_id`, `descr`) VALUE ();
  ```

  (Bug#40041)
- If a trigger was renamed, and the design then synched with a database instance, the generated SQL created a trigger with the new name and then dropped the trigger with the old name. This resulted in the following error:

  ```
  Error 1235: This version of MySQL doesn't yet support
  'multiple triggers with the same action time and event for one table'
  ```

  (Bug#39989)
- The Copy SQL to Clipboard action (right click menu on table) did not use Windows-compatible line endings. (Bug#39476)
- When a column had a datatype `BOOLEAN` and it was exported using Forward Engineer SQL ALTER, the exported type was `BOOLEAN(2)` instead of `BOOLEAN`. (Bug#39257)
- Workbench application performance was poor, with slow loading times and excessive memory usage. (Bug#38439)
- When a DBDesigner model with 333 tables was imported into Workbench the RAM usage went up to approximately 1GB. Workbench then crashed with the following exception:

  ```
    Error creating cairo context: out of memory
  ```

  (Bug#37178)

A.2.5. Changes in MySQL Workbench 5.0.26 (16th October 2008)

This section documents all changes and bug fixes that have been applied since the release of 5.0.25.

Bugs fixed:

- When attempting to export a model using the File, Export, Forward Engineer SQL CREATE Script menu item, Workbench crashed on clicking the wizard's Finish button. (Bug#39578)
- The **Copy Insert to Clipboard** action generated SQL with lower case keywords. This was not consistent with the behavior of the **Copy SQL to Clipboard** action. (Bug#39477)
- Renaming a table and then selecting Forward Engineer SQL ALTER Script did not result in a `RENAME` statement. Instead, `DROP` and `CREATE` statements were generated. (Bug#39256)
- The script generated by the Export, Forward Engineer SQL CREATE Script menu item contained invalid statements when using two schemata. (Bug#39211)
- Exported SQL code containing a trigger that called a procedure would fail when an `INSERT` activated the trigger. (Bug#39088)

A.2.6. Changes in MySQL Workbench 5.0.25 (12th September 2008)

This section documents all changes and bug fixes that have been applied since the release of 5.0.24.

Bugs fixed:

- If the user closed all tabs and then quit, Workbench crashed. (Bug#39346)
- Foreign keys referencing a deleted table were not removed. (Bug#39150)
- FORWARD ENGINEER SQL CREATE SCRIPT and FORWARD ENGINEER SQL ALTER SCRIPT generated scripts that did not put index names in quotes. (Bug#39140)
- When Workbench was started with the GRT Shell tab opened, the object tree in the **GRT TREE** pane was not displayed. (Bug#39122)
- When triggers were exported with the **GENERATE DROP TABLES STATEMENTS** option checked, `DROP TRIGGER IF EXISTS` did not appear in the exported SQL. (Bug#39119)
- The **TRIGGERS** tab would always enable Insert mode when opened. (Bug#39118)
- In the **FOREIGN KEY** tab of the **TABLE EDITOR**, the dropdown menu that is displayed on clicking in the **REFERENCED TABLE** column, listed table names by creation date, rather than by sorted name. (Bug#38944)
- If any `DEFAULT` properties were defined for a model, they appeared to be lost after saving the model and restarting Workbench. (Bug#38825)
- When you loaded a UTF-8 encoded script file into Workbench, the embedded SQL editor replaced international characters with the `?` symbol. (Bug#38783)
- When creating Views and Routines, the entry in the **UNDO HISTORY** window showed “Parse MySQL View” instead of “View Created”, and “Parse MySQL Routine” instead of “Routine Created”.

  When subsequently undoing this operation the correct text was displayed. Performing a redo then resulted in the incorrect text being displayed again.

  Additionally, when undoing a Routine Group, the previous undo action in the history was incorrectly renamed and the last entry in the history was deleted. (Bug#36047)
- In the **TABLE EDITOR** tab, wherever data could be entered, such as in the **FOREIGN KEY NAME** entry field, the default wrap protocol was to go to a new line. This resulted in text that was only partially visible. (Bug#34510)
- The synchronization wizard could show a diff tree for schemata different from those that had been selected. (Bug#32365)

A.2.7. Changes in MySQL Workbench 5.0.24 (12th August 2008)

This section documents all changes and bug fixes that have been applied since the release of 5.0.23.

Functionality added or changed:

- In the `MySQLGrtShell.exe` program the **VALUES** tab has been renamed to **GRT TREE**. However, the **GRT TREE** tab only shows a root node because there is no GRT Tree loaded when the `Shell` is started in standalone mode. (Bug#35052)

Bugs fixed:

- Indexes listed when the **INDEX** tab was selected could not be deleted if the index type was `FOREIGN`. (Bug#38639)
- When the menu item MODEL, VALIDATION, VALIDATE ALL was selected, and an error dialog subsequently displayed, the dialog error message had a missing dot separator between the database name and table name. (Bug#38632)
- When a DBDesigner 4 model that contained duplicate relationships was imported into Workbench, and then exported, the resultant script would fail when executed on MySQL server. (Bug#38488)
- It was not possible to synchronize a model to an external database, if the model contained triggers. (Bug#38436)
- When resizing the comment column under **PHYSICAL SCHEMATA** view in column format, the column resize was reverted when switching between schemas. (Bug#38431)

- An attempt to copy a table and then paste it into a new schema resulted in an `Unknown Exception` being generated. (Bug#38429)
- If you created a new view with an `OR REPLACE` clause, the Forward Engineer SQL CREATE Script output contained the `OR REPLACE` clause twice. (Bug#38337)
- When a DBDesigner 4 XML file was imported into Workbench the `INSERT` statements were incorrectly converted. (Bug#38196)
- Importing a script that specified an incorrect data type required Workbench to close. (Bug#38146)
- Workbench crashed when using the Model, Validation(MySQL), Validate All menu item on a model that contained a dangling foreign key index. (Bug#38115)
- Foreign key options (`onDelete`, `onUpdate`) are not imported from DBDesigner schema. (Bug#37794)
- In the `mysql-workbench-oss-5.0.23-win32-noinstall` version of Workbench the menu item Plugins, Objects, Copy SQL to Clipboard did not work. (Bug#37736)
- When synchronizing the database, table comments were not updated. However, column comments worked as expected. (Bug#37686)
- Running Help, Update... crashes Workbench when the wizard comes to the point where it is trying to close Workbench. (Bug#37665)
- Database, Synchronize did not update the model view when the table was changed in the database, until after Workbench was restarted. (Bug#37634)
- Forward Engineer SQL CREATE Script did not reflect changes made to the model. (Bug#37574)
- When using the Forward Engineer SQL CREATE Script, columns marked as `NOT NULL` were generated as `NOT NULL DEFAULT NULL`. (Bug#37385)
- Errors were generated in SQL code during Forward Engineer Schema for Inserts data in `TIMESTAMP` columns. (Bug#37059)
- If a database was imported using Reverse Engineer SQL ALTER Script and the database name changed in Workbench, the script then generated by Forward Engineer SQL ALTER Script was incorrect. (Bug#36178)
- The auto-increment flag was not cleared internally for a column, when the type of that column was changed to one for which auto-increment is invalid e.g. `char`. When the model was exported using Export, Forward Engineer SQL Create Script, the resulting script incorrectly retained the auto-increment flag for the changed column. (Bug#36085)

A.2.8. Changes in MySQL Workbench 5.0.23 (25th June 2008)

This section documents all changes and bug fixes that have been applied since the release of 5.0.22.

Functionality added or changed:

- It was not clear how a stored connection profile could be edited and the changes saved. Tooltips have been added to the relevant buttons and the main documentation clarified. (Bug#37061)

Bugs fixed:

- The File, Export, Forward Engineer SQL CREATE Script menu item exports a script it is then unable to import using the File, Export, Reverse Engineer MySQL Create Script menu item, as it incorrectly imports comments containing special characters. (Bug#37563, Bug#37562)
- Workbench was failing to correctly export Trigger DDLs. (Bug#37432)
- Using Generate Schema Diff Report resulted in a crash. The crash was caused by improper handling of an invalid FK in a table. While this issue is correctly reported by a validation module, in Standard Edition Generate Schema Diff Report didn't handle that correctly. (Bug#37393)
- When a new column was added to a table `Inserts` data was deleted. (Bug#37192)
- Trying to edit a table in a new window displays an error message dialog:

```
plugin:wb.edit.editSelectedInNewWindow
Invalid plugin
Invalid plugin wb.edit.editSelectedInNewWindow
```

(Bug#37180)

- If you try to place a new image into an EER Diagram and select an invalid filetype, you get a error message dialog with the following text:

  `CAIRO ERROR: INVALID MATRIX (NOT INVERTIBLE)`

  If you then click OK to clear the dialog and then try to select Place a New Table, the error message dialog is displayed again. (Bug#37079)

- The Forward Engineer wizard did not report connection status correctly. If invalid database credentials were entered, the wizard reported success, even though the connection failed. (Bug#37060)

- Incorrect behaviour when editing a table. When the **columns** tab is selected, if you want to delete multiple selected tables at once, Workbench removes the wrong columns. (Bug#37045)

- The script generated by the File, Export, Forward Engineer SQL ALTER Script menu item contains syntax errors. (Bug#36889)

- The export filter did not properly filter tables. (Bug#36739)

- Workbench generated incorrect syntax when attempting to synchronize with a live server. The resultant code was missing commas which resulted in a syntax error. (Bug#36674)

- After reverse engineering an SQL create script and drawing some EER diagrams, a subsequent import of the same script destroys the EER diagrams. All tables in the catalog are updated, but the reference of the table in the diagram to the table in the catalog is lost. The tables in the diagram are still visible, but do not correspond to the table in the catalog.

  After closing and re-opening the file, all diagrams are empty and it is impossible to delete the diagrams. However, in the overview in the upper right corner, the tables placed in the diagram are still visible. (Bug#36381)

- Mouse wheel does not work when you double-click a table and select the **Options** tab. (Bug#36374)

- When File, Export, Forward Engineer SQL ALTER Script menu item is selected it causes an `ALTER Script Generation (Script Synchronization)` error. (Bug#36355)

- The behaviour of the Synchronize wizard was inconsistent when cancelled and re-run. (Bug#36177)

- Several windows and tabs have fields which are either not completely visible or are obscured by labels that overlap the field. (Bug#36115)

- When creating a `Schema Diff Report` from the local model to a live database, the wizard crashed with an unhandled exception. (Bug#35878)

- Collapsing of the EER Diagram section of the **MySQL Model** tab is not retained after program relaunch. (Bug#35717)

- In the **MySQL Model** tab, in the summary line for **Physical Schemata**, there are three icons, one for large icon view, one for small icon view, one for list view. Changing the view is not saved between application launches. (Bug#35716)

- Performing a Database Synchronization resulted in erroneous `ALTER` statements being generated. (Bug#34812)

- Menu item was incorrectly named Generate Schema Diff Report, when it should have been called Generate Catalog Diff Report. (Bug#34398)

- Workbench failed to restore window states, window positions and side-panel sizes from the previous execution of the application. (Bug#32442)

- The `AUTO_INCREMENT` attribute is now ignored on import for column types that do not support it. (Bug#31986)

A.2.9. Changes in MySQL Workbench 5.0.22 (27th May 2008)

This section documents all changes and bug fixes that have been applied since the release of 5.0.21.

Bugs fixed:

- Can not add values for `TIMESTAMP` columns in the **Inserts** editor. (Bug#37009)
- When columns are added to, or removed from a table, Workbench deletes all **Inserts** data. (Bug#37008)
- Trigger definition auto-formatting resulted in malformed code. (Bug#36815, Bug#37685)
- The script generated by the File, Export, Forward Engineer SQL CREATE Script menu item contains a spurious quote mark. (Bug#36753)
- For `CREATE TABLE` statements, `TIME` column default values were not quoted properly. (Bug#36669)
- Print preview in landscape orientation did not work correctly. (Bug#36647)
- When opening a model created with an earlier version of Workbench, the **Indexes** tab displayed indexes of type `FOREIGN` as type `INDEX`, and it was not possible to change them back to `FOREIGN`. (Bug#36453)
- If a table column definition allows `NULL` and has been set with a default of `NULL`, integrity validation operations complained that the default value for the column is invalid. (Bug#36397)
- After use of Control-X to cut text from a text-edit box and Control-Z to undo the operation, the canvas was updated correctly but not the text box. (Bug#36358)
- Shifted content could not be scrolled or navigated. (Bug#36328)
- The mousewheel scrolled the overview pane when it was open behind the insert-editor. (Bug#36253)
- View renaming in overview did not work properly and has been disabled. (Bug#36202)
- The Copy to SQL operation caused a crash. (Bug#36184)
- Dragging objects out of a layer did not work properly. (Bug#36053)
- The enabled/disabled status of items in the Edit menu was not updated properly. (Bug#35962)
- Relationships were drawn over tables. (Bug#35867)
- The script generated by database synchronize contained errors. (Bug#35644)
- Setting up foreign key relationships across multiple schemas did not work. (Bug#34546)
- Scrollbars now appear correctly when editor windows are reduced in height. (Bug#32454)
- Table partitioning information was not exported properly. (Bug#32226)

A.2.10. Changes in MySQL Workbench 5.0.21 (27 April 2008)

This section documents all changes and bug fixes that have been applied since the release of 5.0.20.

Bugs fixed:

- The undo operation did not completely undo a relationship between two tables. It removed only the line drawn between two tables, but did not undo the fields and keys. (Bug#36645)
- Double clicking a column-heading separator in Find results caused a crash. (Bug#36266)
- The scripts generated by the File, Export, Forward Engineer SQL ALTER Script and File, Export, Forward Engineer SQL CREATE Script include unnecessary SQL code. (Bug#36170)
- The Copy Connection nn menu item on the context menu of a connection does not have a complementary Paste Connection menu item. The Edit menu has a greyed-out Paste Connection menu item. (Bug#36166)
- Forward Engineer wizard failed to create a table, but did not show any error messages. (Bug#35874)
- Saving a file restores the column widths of the list view to default under **Physical Schemata**. (Bug#35718)
- When making a column a primary key and this column has `NULL` as default value, this default value is not changed. When the table gets synchronized back to the database Workbench creates a statement such as:

```
ALTER TABLE `test_defhan`.`table1` CHANGE COLUMN `id_table1` `id_table1` INT(11) NOT NULL
DEFAULT NULL, ...
```

This leads to an error:

`ERROR 1067: INVALID DEFAULT VALUE FOR 'ID_TABLE1'` (Bug#32972)

A.2.11. Changes in MySQL Workbench 5.0.20 (26 April 2008)

This section documents all changes and bug fixes that have been applied since the release of 5.0.19.

Bugs fixed:

- Re-creating a deleted relationship caused a crash. (Bug#36385)
- The message log on the Forward Engineer Progess/Results Advanced dialog had no scroll bar. (Bug#36192)

A.2.12. Changes in MySQL Workbench 5.0.19 (15 April 2008)

This section documents all changes and bug fixes that have been applied since the release of 5.0.18rc.

Bugs fixed:

- The HTML Basic Single Page DBDoc report from the Model -> DBDoc -> Model Reporting menu option was missing the schema and table numbers. (Bug#36060)

A.2.13. Changes in MySQL Workbench 5.0.18rc (not released)

This section documents all changes and bug fixes that have been applied since the release of 5.0.17rc.

Functionality added or changed:

- Foreign key labels could not be hidden, and displayed labels were not centered. There are now options to hide all connection captions, and to center captions. (Bug#30902)

Bugs fixed:

- In the table editor, setting the input focus by clicking the mouse did not work. (Bug#35969)
- The Reference Column dropdown used during foreign key creation was slow to display. (Bug#35948)
- In the table editors foreign key Tab, when a column for the foreign key is checked (right pane), the Referenced Column drop-down opens. Pressing Escape at this point caused a crash. (Bug#35926)
- After changing the Row Format option, closing the table editor and opening a new document caused a crash. (Bug#35925)
- Synchronizing the data model with a live database from the SQL Diff Tree dialog resulted in a crash. (Bug#35884)
- Creating a Schema Diff Report from the local model to a live database caused a crash. (Bug#35878)
- The Pack Keys option could not be saved. (Bug#35872)
- Some menus or submenus had items enabled when the corresponding features were disabled. (Bug#35870)
- The Connection Caption option did not work properly. (Bug#35859)
- The status of a connection line in a table diagram was not updated when a foreign key relationship between tables was changed. (Bug#35800)
- The File -> Export -> Export as PNG menu item was enabled under some circumstances in which it should have been disabled. (Bug#35746)
- Scrolling was slow for table models with large numbers of tables. (Bug#35655)

- Pressing Ctrl-Z to undo the last change in an SQL Script text box deleted the entire script. (Bug#35649)
- Workbench is unable to read files such as Workbench Model Files from a non-English directory. (Bug#35547)
- Workbench allowed table comments to be entered longer than the maximum length of 60 characters. (Bug#34507)
- A crash could occur during foreign key creation. (Bug#33545)
- Autoplacing for display of complex schemas has been improved. (Bug#32888)
- Typing `q` in the GRT Shell caused a crash. (Bug#32755)

A.2.14. Changes in MySQL Workbench 5.0.17rc (07 April 2008)

This section documents all changes and bug fixes that have been applied since the release of 5.0.16rc.

Bugs fixed:

- Creating a new view and then deleting it caused a `System.AccessViolationException`. (Bug#35840)
- Editing a stored procedure within Workbench could cause an exception. (Bug#35828)
- The modified timestamp for an existing model was not correctly updated for all changes. (Bug#35719)
- Identifiers for field names in DML SQL statements would not be quoted correctly, allowing for reserved words to be included in the SQL statements. (Bug#35710)
- Workbench would crash repeatedly when drawing the diagram for a table where the referenced column in a foreign key relationship was blank. (Bug#35677)
- Identifiers using uppercase characters for stored procedures would automatically be modified to lowercase. (Bug#35650)
- When working with the **SQL Script** editor, it was not possible to select all the text in the display when using **Ctrl A**. (Bug#35646)
- The **Model Navigation** window could not be collapsed like other palettes. (Bug#35642)
- Modifying the primary key index definition for within the table view would not update the entity relationship diagram. (Bug#35639)
- When validating an existing model using the Forward Engineer Wizard, MySQL-specific validation would fail. (Bug#35604)
- Deleting an existing layer on a diagram and then editing other objects on the same canvas could generate a number of exceptions, and could corrupt the Workbench file. (Bug#35603)
- Switching to the Connect to Columns notation with an existing model would cause an exception. (Bug#35601)
- Data in `BLOG` and `TEXT` columns defined using the **Inserts** tab would not be quoted correctly in the resulting SQL. (Bug#35525)
- Opening an existing Workbench model with an invalid foreign key definition would cause an exception. (Bug#35501)
- Moving multiple tables on the same diagram, and then using Undo to revert the model to the original layout, only the first table selected be returned to its original position. (Bug#35465)
- When adding a foreign key relationship within a catalog with an existing entity relationship diagram, the foreign key relationship is not added to the existing diagram. (Bug#35429)
- The precise position of individual connections would not be retained when the schema was saved. (Bug#35397)
- Opening a GRT shell while the table editor is open would raise an exception. (Bug#35349)
- When modifying an existing foreign key relationship, the generated `ALTER` script did not reflect the modification. (Bug#35265, Bug#35830)
- When creating foreign key relationships that point to more than one table, the same foreign key identifier for the same table could be created. This would create invalid SQL code for creating the table. (Bug#35262)
- When importing an existing DB Designer schema, Workbench could crash. (Bug#35123)

- Setting up indexes in both the index and foreign key list views, the mouse pointer would dissappear while the entry box was in use. (Bug#35062)
- Double clicking the Catalog title bar undocked the GRT Tree window. (Bug#34856)
- The font for views and routines was not monospace by default. (Bug#34537)
- When using the Forward Engineer Wizard, if an error occurred, the dialog showing the error detail would be incomplete, and determining the reason for the error would be masked because the end of log message would be hidden. (Bug#34509)
- When using the **Hide Menu Items Not Applicable to this Edition** option, a simplified version of the **Find** dialog box was not available. (Bug#34493)
- Editing the text of the Trigger portion of an existing schemata would introduce additional text into the Trigger definition. (Bug#34397)
- Creating more than five stored procedures or views in a model would cause the dialog box for the operation to move to a different layer, making it inaccessible when using the mouse. (Bug#34153)
- Selecting Export, Forward Engineer ALTER script from the File would open a **SQL Script Syncrhronization** dialog, rather than export dialog. (Bug#34099)
- When moving more than layer in Model Navigator, only the first layer's position would be reflected correctly in the output. (Bug#33627)

A.2.15. Changes in MySQL Workbench 5.0.16rc (26 March 2008)

This section documents all changes and bug fixes that have been applied since the release of 5.0.15rc.

Functionality added or changed:

- Options and configuration options that affect models can now be set on a model by model basis. Choose Options from the Model menu and choose the **Diagram** tab. (Bug#34610)

Bugs fixed:

- When double clicking on the row in a column as a primary key, the primary key property would be toggled. The editor will now allow you to edit the value when clicking on a data row on the table. (Bug#35613)
- Opening the **Indexes** portion of a table would generate a unhandled exception error. (Bug#35598)
- When disabling global options on an individual model would fail to honor the model specific options would be ignored. (Bug#35516)
- When placing a 1:n relation, an `index out of range` error could be raised. This could further result in `operation on NULL object: Invalid value` errors when trying to edit the relation. (Bug#35447)
- Setting the value of a numeric column to a negative value was not supported. (Bug#35442)
- Printing an HTML version of the schema would produce a fatal error. (Bug#35400)
- The OK and Cancel buttons for the **Diagram Size** dialog would not be initialized properly. (Bug#34808)
- When using print preview on a diagram, clicking the Print button would send a blank page to the printer. (Bug#34630)
- When copying multiple table definitions from one schema to another, only the first table in the selection would be pasted into the new schema. (Bug#34483)
- The **Draw Line Crossing** option would fail to be recognized correctly. You can also now set this on an individual model basis using the Options optin in the Model menu. (Bug#34248)
- Copying an existing module to the plugins directory would trigger a double registration of the modulem, and produce an error. (Bug#34134)
- When exporting a diagram to PDF, some additional lines would be added to the generated PDF. (Bug#33586)
- Placing an image on to the canvas could crash the application. For images larger than the canvas, the image is automatically re-

duced so that it is properly visible on the canvas for editing. (Bug#33179)

- A 1:m relation in a diagram would fail to be generated properly when exported as a PDF. (Bug#32882)
- The Undo and Redo options would not be applied properly when making modifications to partition definitions. (Bug#32279)

A.2.16. Changes in MySQL Workbench 5.0.15rc (17 March 2008)

This section documents all changes and bug fixes that have been applied since the release of 5.0.14abeta.

Bugs fixed:

- Using **Undo** on a relationship within a model would cause an exception. (Bug#35243)
- A foreign key relationship to the source table (a reflexive relationship) gives a bad representation in the entity model diagram. (Bug#35237, Bug#34810)
- Generating an `ALTER SCRIPT` or using the synchronize functionality on a model with entity relationships, the relationship lines within the diagram would be generated twice. (Bug#35213)
- Boolean values were unsupported when trying to insert values into a table, the `TRUE` would instead be replaced by a textual, quoted version `'TRUE'`. (Bug#35205)
- Printing a model diagram to PDF or Postscript, results in a corrupt file PDF or Postscript file that does not match the model. (Bug#35197)
- Deleting objects within the overview pane when the corresponding editor pane for those objects is open would cause a crash. (Bug#35186)
- When entering data into the **Default** column of the table editor, the use of the **Return** key for saving the information about the default value was not supported. (Bug#35127)
- There was a typographical error in the help message for the GRT command `cd`. The word `Absolute` was missing the final `e`. This has been corrected. (Bug#35119)
- When changing the name within a foreign key relationship, the modified name is not reflected in the tables to which the foreign key is related. (Bug#35093)
- Scrollbar navigation did not work after importing a DB Designer schema with a large canvas size. However, you could still navigate using the `Model Navigator` palette. (Bug#34988)
- After importing a DB Designer schema, the following error occurred: "Cairo error: input string not valid UTF-8." (Bug#34987)
- Creating a new file after changing an existing file with modifications could lead to the original being deleted without prompting to save the changes. (Bug#34976)
- When saving an existing model, the **MySQL Model** overview panel would scroll to the top of the model definition. (Bug#34975)
- Changing the `drawSplit` property of a connection from the `Properties` palette did not updated the `Visibility` section of the connection editor. (Bug#34934)
- Editing a primary key column within a model on Microsoft Vista could cause a crash. (Bug#34922)
- On the `MySQL Model` page, when the large icons view was selected, the `Add Table` icon disappeared. (Bug#34904)
- Incorrect `ALTER` statements are created during the synchronization process if you add foreign keys to an existing or imported model. (Bug#34897)
- A new **GRT Inspector** tab would be created every time the **GRT Shell** was opened. In addition, manually closing the **GRT Inspector** and **GRT Shell** components would cause an exception. (Bug#34857)
- Opening an existing MySQL Workbench file after associating the `.mwb` extension with the application leads to a crash when you open a MySQL Workbench file. (Bug#34849)
- When editing a model, the windows and toolbars would realign themselves during selection. This was related to the configured font sizer the DPI setting of the monitor, causing the application to redraw the windows to account for the configuration combination. (Bug#34822)

- Attempting to move a table on an EER diagram after deleting a relationship, caused the application to crash. (Bug#34816)
- The New File dialog is non-modal, and could be hidden by other windows. The dialog is now always drawn on top of other windows. (Bug#34784)
- Changing the column name of a table when you have pending inserts to the table did not change the column name in the corresponding `INSERT` statements. (Bug#34500)
- The `Properties` palette was not cleared when a new project was started. It retained the properties of the last selected object. (Bug#34433)
- Deleting an existing schema with an open table editor would not close the table editor window. The window is now closed when the schema is deleted. (Bug#34345)
- Searching a project specifying `Entire Model` in the `In Location:` drop down list box did not return any results. This applied to the Standard Edition only. (Bug#34170)
- When the page size was changed from A4 to B4 it was not possible to move objects on an EER diagram beyond the old page boundaries. (Bug#34148)
- When editing comments, the **Return** key would move to the next column, which prevented the use of newlines within the comment information. Workbench now allows use of the **Return** key within the comment field. (Bug#33980)
- Where relationship lines crossed, and one of the connectors was changed to `Hidden` or `Draw Split`, the semi circle that indicated the previous intersection was still shown on the remaining connector. (Bug#33818)
- Editing an existing diagram could cause an unhandled exception on Windows Vista. (Bug#33477)
- Identifiers (tables, column, index, triggers and other data types) could be created with names longer than the maximum support by MySQL Server. (Bug#33265)
- The application crashed when attempting to export an SQL CREATE script. (Bug#33263)
- Placing an object on the canvas of an EER diagram where you have reverse engineered an existing database, would lead to multiple copies of the object appearing on the diagram. (Bug#32891)
- When scrolling through a schema, the tables in the schema were not redrawn correctly. (Bug#32835)
- On an EER digram you could not select a relationship if the connection line wasn't stepped. You can now select a connector even if it is not stepped. (Bug#32734)
- Printing a model when there is no printer connected could result in an application exception. (Bug#32320)

A.2.17. Changes in MySQL Workbench 5.0.14abeta (28 February 2008)

This unscheduled beta release fixes Bug#34847 Other bug fixes that have been applied since the release of 5.0.14beta are also documented.

Bugs fixed:

- MWB files were not saved properly if Workbench crashed. Reopening such files caused Workbench to crash. (Bug#34848)

  See also Bug#34847.
- Workbench models created in version 5.0.13 crashed when used with version 5.0.14. The unscheduled Beta release, 5.0.14a fixes this bug. (Bug#34847)
- When clicking the Browse button in the image editor, the default file name was `openFileDialog1`. This now defaults to an empty string. (Bug#34622)
- Repeatedly changing the object notation crashed Workbench. This is no longer repeatable. (Bug#34499)
- Importing a DBDesigner file immediately threw an exception. This happened even when software rendering was used. DBDesigner files can now be imported without incident. (Bug#33588)

A.2.18. Changes in MySQL Workbench 5.0.14beta (25 February 2008)

This section documents all changes and bug fixes that have been applied since the release of 5.0.13beta.

The following improvements have been added to this version of Workbench:

- EXPAND ALL and COLLAPSE ALL menu options have been added under the ARRANGE menu. The EXPAND ALL option expands all objects on an EER. This option will display a table's columns if the object notation supports expansion. Indexes will not automatically be expanded unless they were previously expanded and have been collapsed using the COLLAPSE ALL menu option. Some object notations, such as `Classic`, do not allow for expansion or contraction. COLLAPSE ALL undoes the operation performed by EXPAND ALL.
- A FIT OBJECTS TO CONTENTS option has been added under the ARRANGE menu option. This option expands an object on an EER diagram. For example, if a table has a long column name that is not fully displayed, using this menu option will expand the table making the column name visible.
- A SYSTEM INFORMATION menu option has been added to the HELP menu. This option displays information about your system that is useful when reporting a big.
- An EXPORT AS SVG menu option has been added under the FILE, EXPORT menu option.
- Because of serious performance and display issues Workbench no longer uses Mesa. For those users who don't have native OpenGL support, Workbench now uses the Windows GDI API. The command line switch for using this mode is `-swrendering`. For more information about running Workbench from the command line see Section 3.2, "Starting MySQL Workbench".
- The GRT inspector has been improved to support new types. Namely:
  - text
  - longtext
  - bool
  - color
  - file

  This makes it much easier to change object properties manually. Multiple selection support has also been improved — you can easily change a value for several selected objects at once.

Bugs fixed:

- When exporting an SQL CREATE script it was possible to create two tables in the same schema with the same name. (Bug#34668)
- After placing related tables on an EER diagram and then removing them using the UNDO menu option, the connection lines between related tables no longer showed up. (Bug#34601)
- When choosing the EXPORT AS PNG menu option the file dialogue box file type was `All Files`instead of `PNG`. The same was true for EXPORT AS SINGLEPAGE PDF and EXPORT AS SINGLEPAGE PS. The default is now the appropriate file type. (Bug#34548)
- If there was a relationship between table `A` and table `B` and also one between table `B` and table `A`, the connection lines appeared on top of each other. Connection lines now appear attached at the related columns. (Bug#34543)
- When there were multiple tables with long identifiers the `Physical Schemata` section of the `MySQL Model` page was messy. Table names were obscured and sometimes overlapped. Also, the position of the `Add Table` icon was not optimal. Now the space between table names is adjusted to the largest entry and the `Add Table` icon is fixed in the upper left corner. (Bug#34536)
- When returning to the `SQL Export Filter` page after using the BACK button, filters were no longer selected. Selections now persist. (Bug#34503)
- The export filters were applied more than once when forward engineering an SQL `CREATE` script. This happened if you exported the script after using the BACK button on the `SQL Export Filter` page. (Bug#34501)
- When the object notation was Workbench Classic the width of a table on an EER diagram could not be less than the widest column. If there was an enum column with many options, this made for a disproportionately wide table. Table width can now be less than the widest column. (Bug#34496)

- When multiple objects on an EER diagram were selected and deleted, Workbench crashed. This happened when both connections and tables were selected. (Bug#34434)
- Setting a column to `AUTO_INCREMENT` caused the application to crash. (Bug#34418)
- It was reported that you could not add a primary key to a table imported from a MySQL `CREATE` script. This was not true but did highlight the fact that the method for adding a primary key was not obvious. Now, in addition to adding a primary key by double clicking the icon to the left of a column in the table editor view, you can also add a primary key by checking the `PRIMARY KEY` checkbox in the `Column Details` section of the table editor. (Bug#34408)
- When using the menu option GENERATE SCHEMA DIFF REPORT an exception was thrown. A new tree-less version of the Diff report plugin resolves this problem. (Bug#34396)
- Users failed to be created when exporting an SQL `CREATE` script. (Bug#34342)
- When a table's `expanded` property was set to `0`, the connection line between related tables, appeared at a diagonal orientation. A connection line is now docked on the sides of a table even when the `expanded` property is set to `0`. (Bug#34249)
- Copying a table from the `MySQL Model` page to an EER diagram canvas created a duplicate table with the same name as the original. This table did not show up in the `Catalog` palette or in the appropriate schema in the`Physical Schemata` section of the `MySQL Model` page. (Bug#34230)
- Creating a new foreign key did not update an EER diagram. An EER diagram is now updated immediately. (Bug#34206)
- When there were many tables on an EER diagram, constant screen refreshing made the application unusable. The performance of the software rendering mode has been improved. (Bug#33646)
- A table with many columns did not display properly. When the table was expanded on an EER diagram it was impossible to scroll down and view all the columns. Improved rendering has helped solve this problem. However, for very large tables you may have to increase the size of an EER. To do this use the MODEL, DIAGRAM SIZE ... menu option. (Bug#33367)
- When changing the foreign key column of a table on an EER diagram, the foreign key did not change color and Workbench crashed when attempting to save the MWB file. The application no longer crashes and the foreign key is updated. (Bug#33139)
- It was not possible to resize a table that used the `Workbench (Default)` object notation. This was problematic for a number of reasons:
  - Long table names make the table very wide.
  - Column definitions that are long relative to the table name, are truncated.
  - Even if you trimmed column names using the **DIAGRAM** tab of the `Workbench Options` the names were sometimes truncated bled over the table border.

  This has been corrected. (Bug#32981)
- When there were two schemata and two EER diagrams tables did not show up on the EER diagram if tables from different schemata were added to different EER diagrams. This was caused by defective software rendering. (Bug#32588)
- When forward engineering to a live database, objects not selected on the `Select Objects` page were still created. This applied to tables, routines, and users. (Bug#32578)
- It was not possible to drag or resize tables on an EER diagram. Tables can now be manually resized. To revert a table to automatic sizing use the `Property` palette and set `manualSizing`to True. (Bug#32549)
- The display turned black when the application was resized. This happened when viewing the `MySQL Model` page or when viewing an EER diagram. (Bug#23959)