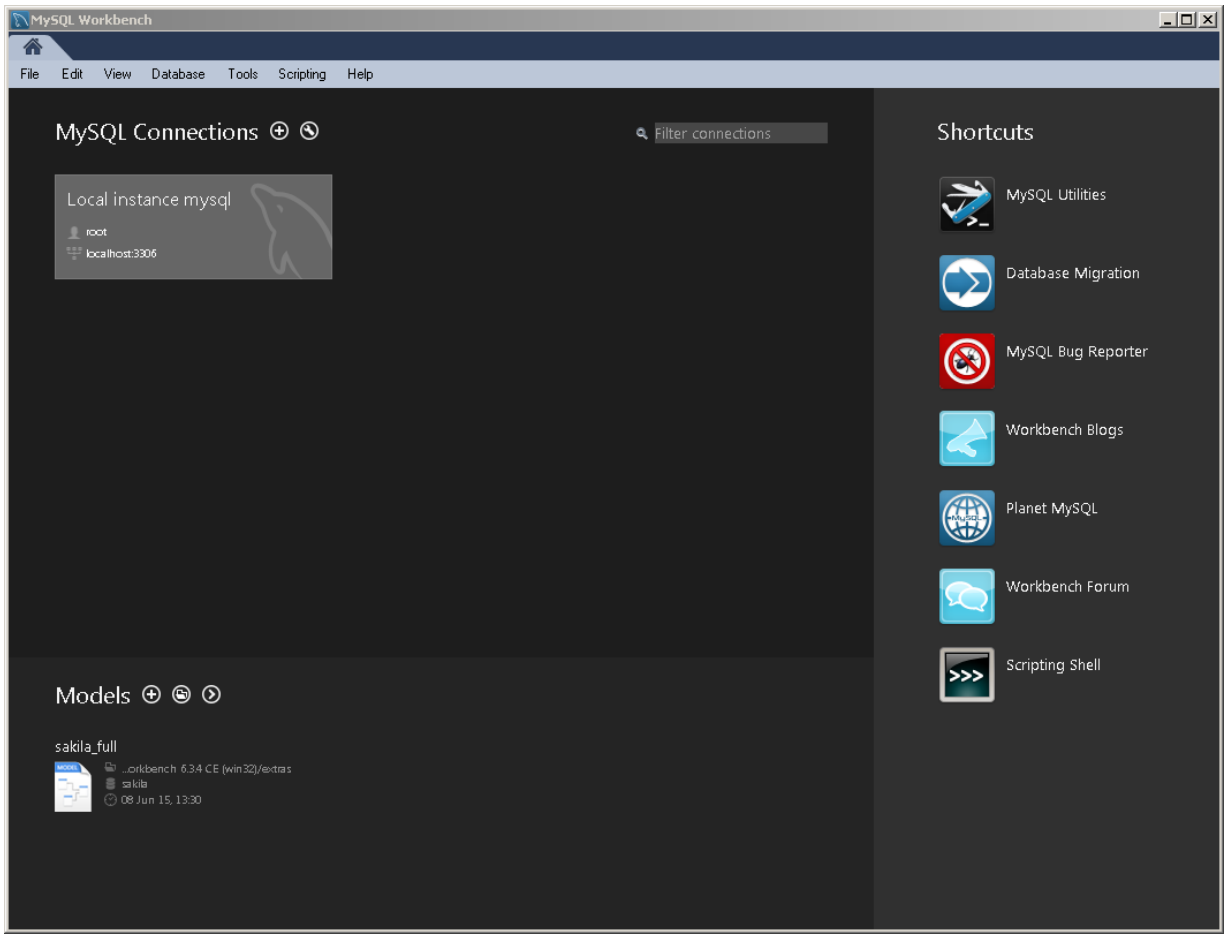
School of Information and Computer Technology
Sirindhorn International Institute of Technology
Thammasat University
ITS351 Database Programming Laboratory

Laboratory #5: Database Design & Administration

Objective:
- To learn how to use MySQL Workbench for Database Development
- To learn how to use MySQL Workbench for Data Modeling (Design)

1 MySQL Workbench

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



Once you start an application, MySQL Workbench The Home Window appears as shown

MySQL Workbench functionality covers five main topics:

- **SQL Development**: Enables you to create and manage connections to database servers. Along with enabling you to configure connection parameters, MySQL Workbench provides the capability to execute SQL queries on the database connections using the built-in SQL Editor.

- **Data Modeling (Design)**: Enables you to create models of your database schema graphically, reverse and forward engineer between a schema and a live database, and edit all aspects of your database using the comprehensive Table Editor. The Table Editor provides easy-to-use facilities for editing Tables, Columns, Indexes, Triggers, Partitioning, Options, Inserts and Privileges, Routines and Views.

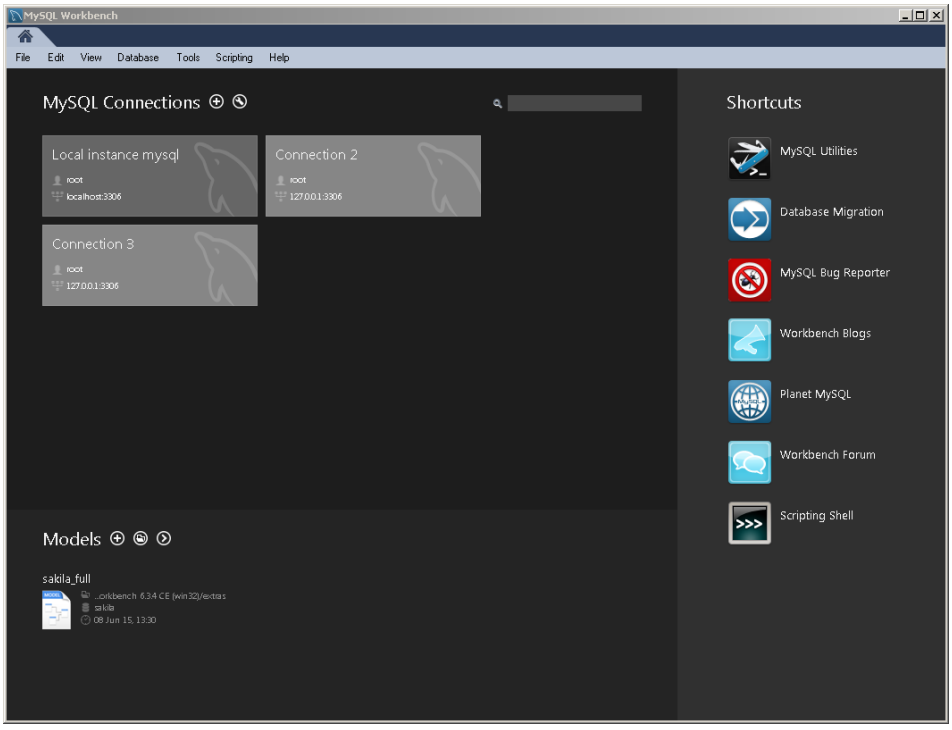
- **Server Administration**: Enables you to administer MySQL server instances by administering users, performing backup and recovery, inspecting audit data, viewing database health, and monitoring the MySQL server performance.

- **Data Migration**: Allows you to migrate from Microsoft SQL Server, Microsoft Access, Sybase ASE, SQLite, SQL Anywhere, PostreSQL, and other RDBMS tables, objects and data to MySQL. Migration also supports migrating from earlier versions of MySQL to the latest releases.

- **MySQL Enterprise Support**: Support for Enterprise products such as MySQL Enterprise Backup and MySQL Audit.

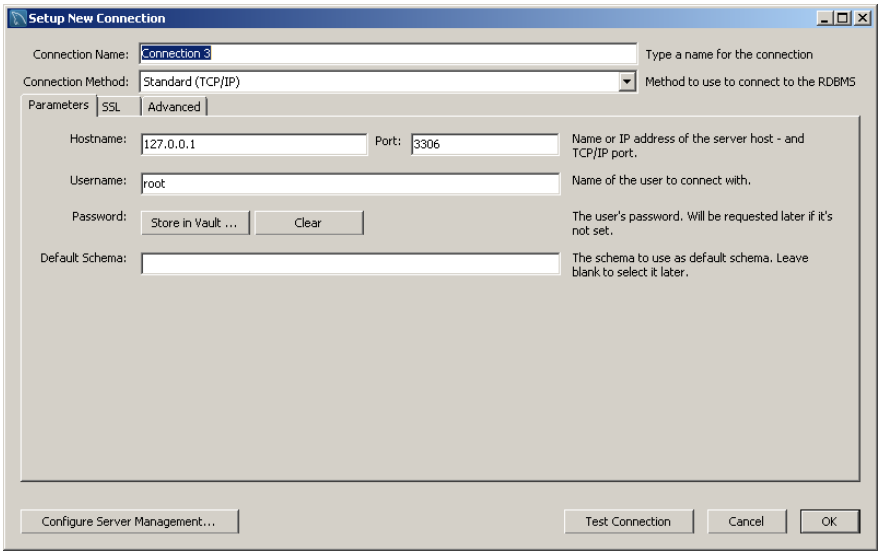
1.1 MySQL Connections

This is the first page you see when opening MySQL Workbench, and it is central to starting MySQL Workbench operations. The three main sections include the MySQL Connections, MySQL Workbench Models, and external Shortcuts.



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

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



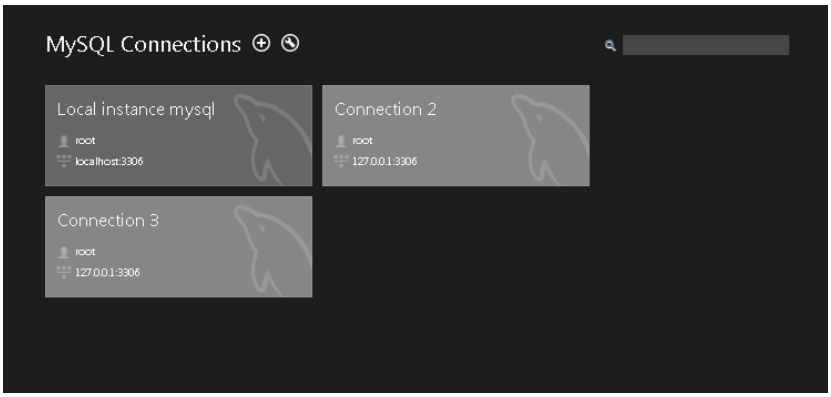
Fill out the connection details. Define the Connection Name value, which we will set to "Connection 3" in this example. The default connection values are for a typical local setup, so check them and enter the appropriate values. If you are unsure, click the Test Connection button to check the connection parameters. Click OK to save the connection. New MySQL connections are added to the Home screen as a tile.

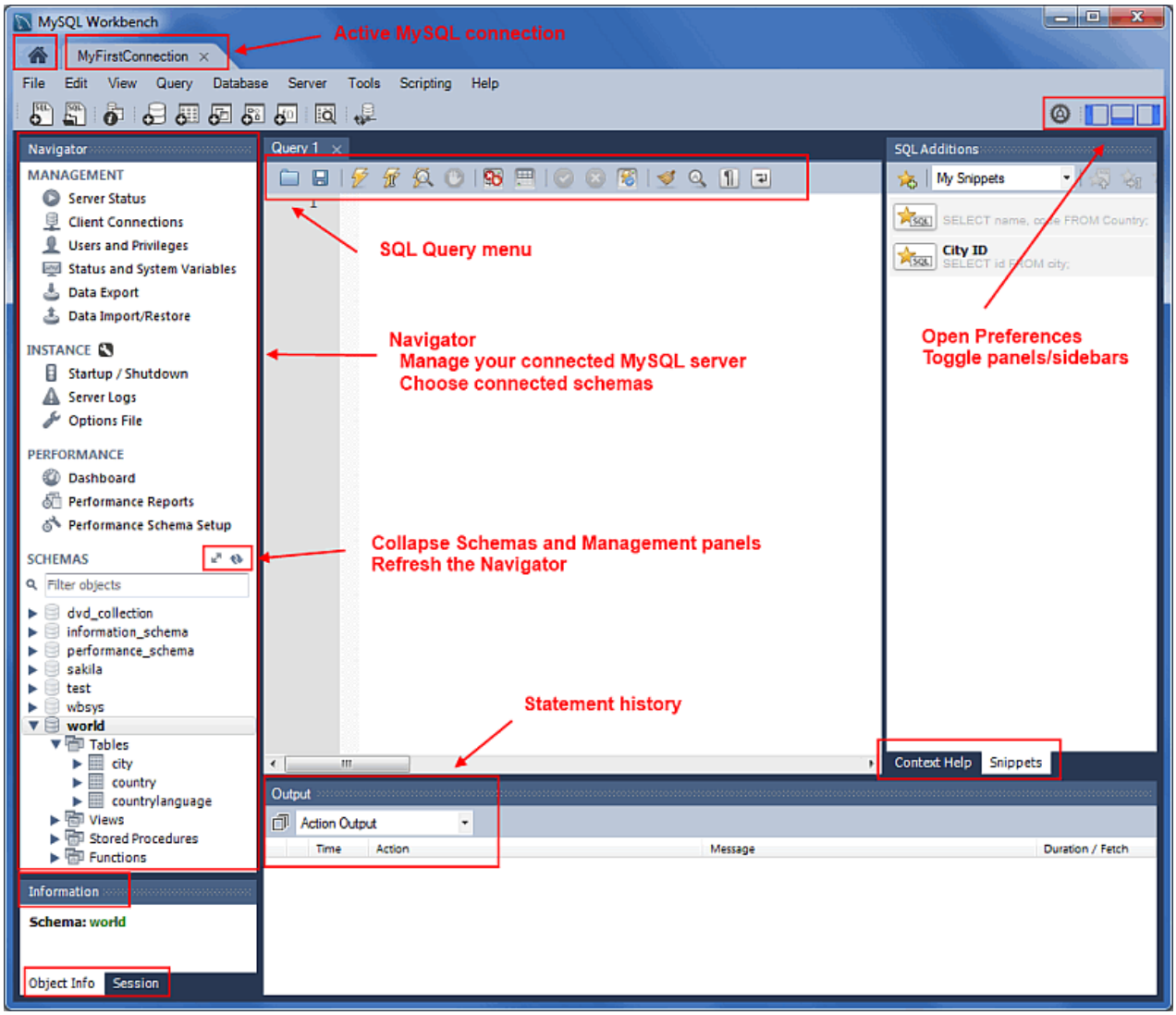
1.2 Database Development (SQL Development)

1.2.1 The visual SQL Editor

The visual SQL Editor lets you build, edit and run queries, create and edit data, and view and export results.

From the Home window, click the new MySQL connection to open the SQL editor for this connection. The SQL editor is the default page.



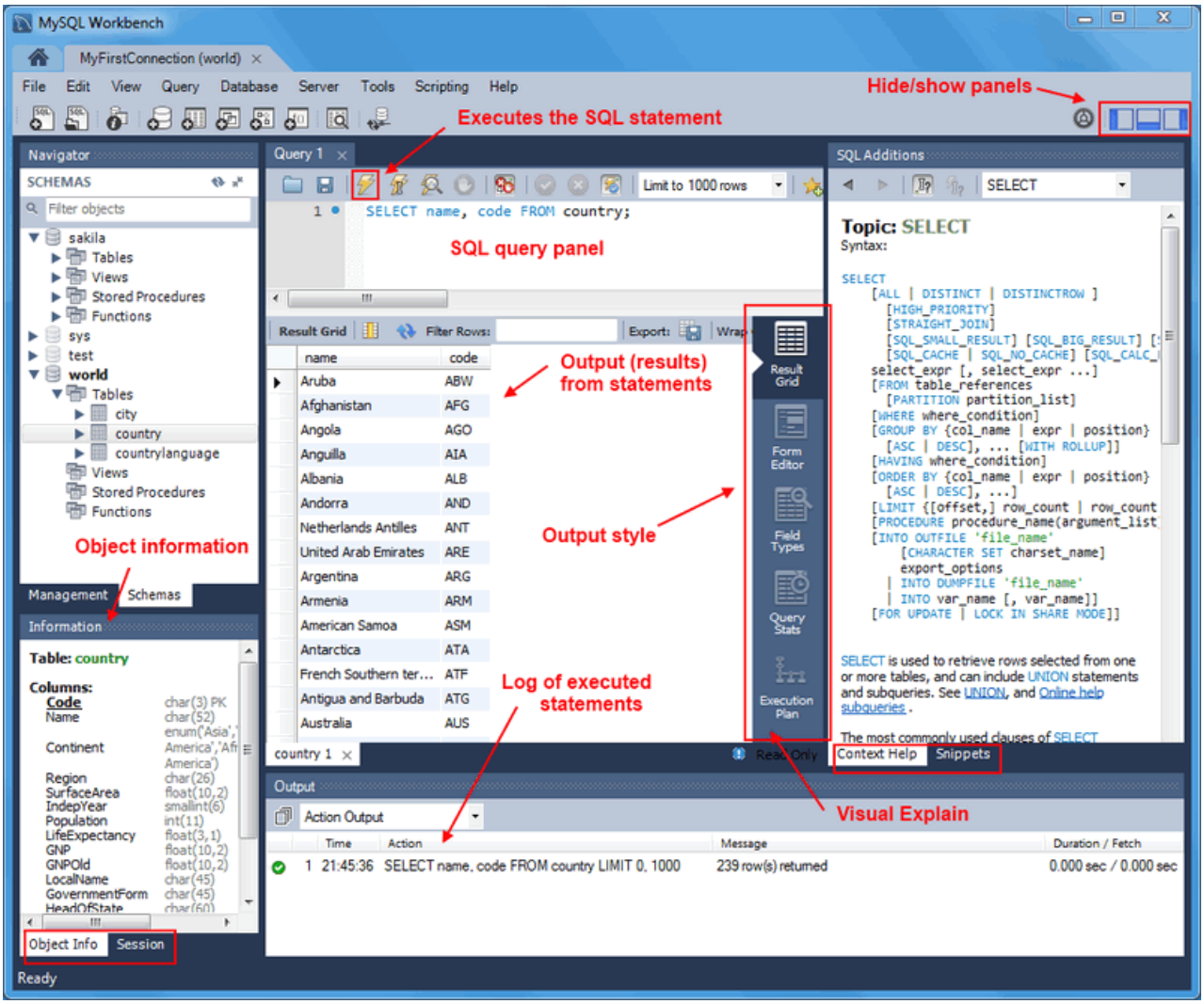


SQL Editor GUI

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

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

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



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

From left to right, these buttons are:

- **Open an SQL Script File**: Loads contents from a saved SQL script into the SQL editor.
- **Save SQL Script to File**: Saves contents from the SQL editor into a file.
- **Execute SQL Script**: Executes the selected portion of the query, or the entire query if nothing is selected.
- **Execute Current SQL script**: Execute the statement under the keyboard cursor.
- **Explain (All or Selection)**: Execute the EXPLAIN command on the query under the keyboard cursor.

- **Stop the query being executed**: Halts execution of the currently executing SQL script.
- **Toggle whether execution of SQL script should continue after failed statements**: If the red "breakpoint" circle is displayed, the script terminates on a statement that fails. If the button is depressed so that the green arrow is displayed, execution continues past the failed code, possibly generating additional result sets. In either case, any error generated from attempting to execute the faulty statement is recorded in the Output tabsheet.
- **Commit**: Commits the current transaction.
- **Rollback**: Rolls back the current transaction.
- **Toggle Auto-Commit Mode**: If selected, each statement will be committed independently.
- **Set Limit for Executed Queries**: The default value is 1000, which appends "LIMIT 0, 1000" to SELECT queries.
- **Save Snippet**: Save the current statement or selection to the active snippet list.
- **Beautify SQL**: Beautify/reformat the SQL script.
- **Find panel**: Show the Find panel for the editor.
- **Invisible characters**: Toggle display of invisible characters, such as newlines, tabs, spaces. A new line is represented as [LF], a space as a single dot (.), and a tab as a right arrow.
- **Wrapping**: Toggles the wrapping of long lines in the SQL editor window.

1.2.2 The MySQL Table Editor

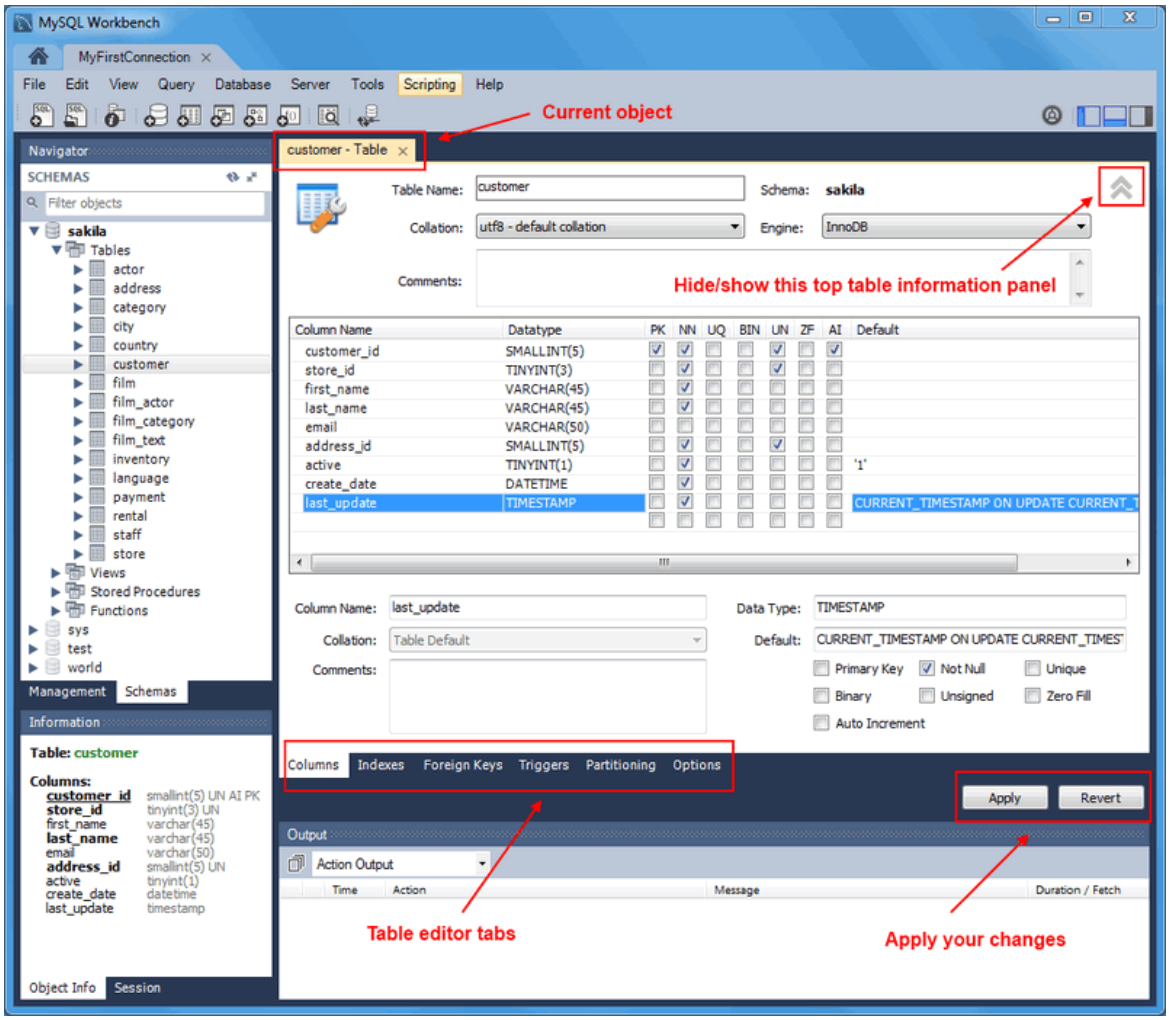
The MySQL Table Editor is a used to create and modify tables. You can add or modify a table's columns or indexes, change the engine, add foreign keys, or alter the table's name.

To create new table, From the Navigator panel on the left, select the table from the database schema. Right-click on the table and choose "Create Table…" from the context menu

To access the MySQL Table Editor, right-click on a table name in the Object Viewer and choose ALTER TABLE.

This opens a new tab within the main SQL Editor window. You can also access the MySQL Table Editor from an EER Diagram by double-clicking on a table object.

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



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

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

- **Columns**: Add or modify columns
  - Use the Columns tab to display and edit all the column information for a table. With this tab, you can add, drop, and alter columns.
  - You can also use the Columns tab to change column properties such as name, data type, and default value.
- **Indexes**: Add or modify indexes
  - The Indexes tab holds all index information for your table. Use this tab to add, drop, and modify indexes.
- **Foreign Keys**: Add or modify foreign keys
  - 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.
- **Triggers**: Add or modify triggers
  - The Triggers tab opens a textbox to create or edit existing triggers.

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

- **Options**: Add or modify other options, divided in categories named general, row, storage, and merge
  - The Options tab enables you to set several types of options.

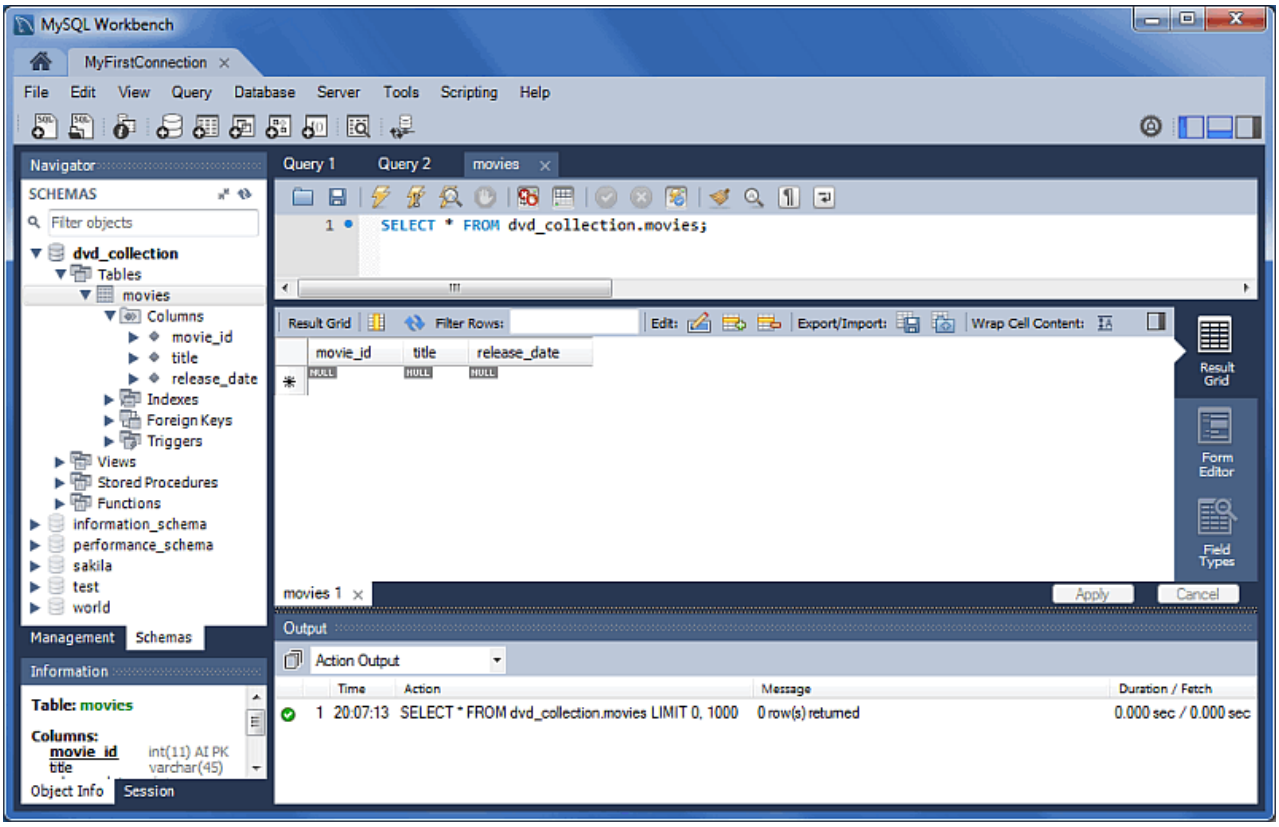
1.2.3 Adding Data

This section uses MySQL Workbench to add data into your MySQL database.

To create new table, from the Navigator panel on the left, select the table from the database schema. Right-click on the table and choose "Create Table…" from the context menu. Create table with the following table information.

Column Name	Data Type	Column Properties
movie_id	INT	PK, NN, AI
movie_title	VARCHAR(45)	NN
release_date	DATE (YYYY-MM-DD)	None

From the Navigator panel on the left, select the existing table from the database schema. Right-click on the table and choose "Select Rows - Limit 1000" from the context menu.



This displays the query and its associated results grid. The table is empty, and data may be added into the results grid

- Input the following sample data into the movies table:

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

Note

The movie_id column is set to AUTO_INCREMENT, so values are not needed for this column.

Do not modify movie_id column values.

- Click Apply to apply these changes to the live MySQL server.
- View the data grid again and observe the generated AUTO_INCREMENT values.
- You can also use MySQL Workbench to perform a similar check. Close the MyFirstConnection tab (or MySQL Workbench) and then open the database connection from the home page. Execute USE dvd_collection; SELECT * FROM movies; to display the newly entered data.

1.3 Database Design / Modeling

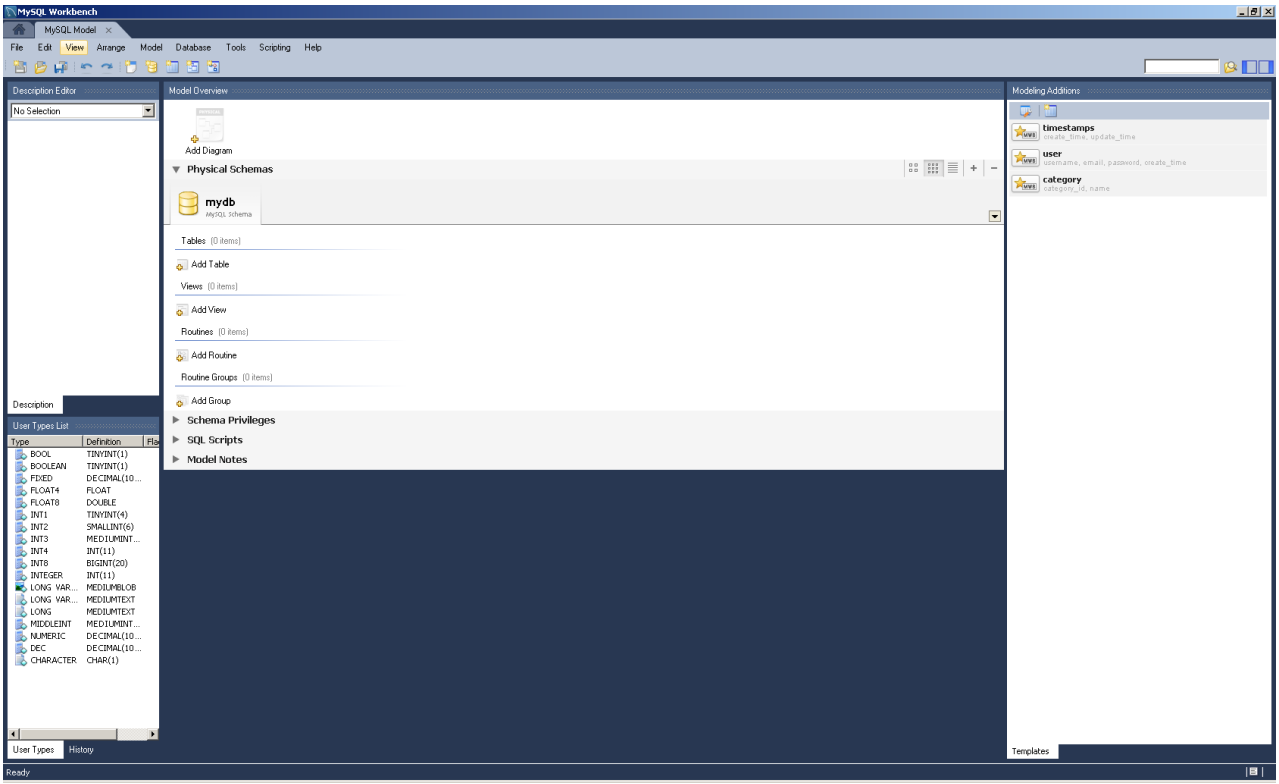
The visual SQL Editor, Modeling simplifies database design and maintenance by enabling you, the data architect, to visualize requirements and resolve design issues. Model-driven database design is an efficient methodology for creating valid and well-performing databases, while providing the flexibility to respond to evolving data requirements. Models are used to build ER diagrams and physical MySQL databases.

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

- Create and manipulate a model graphically
- Reverse engineer a live database to a model
- Forward engineer a model to a script or live database
- Create and edit tables and insert data

1.3.1 Model Editor

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



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

The Model Overview panel includes the following sections:

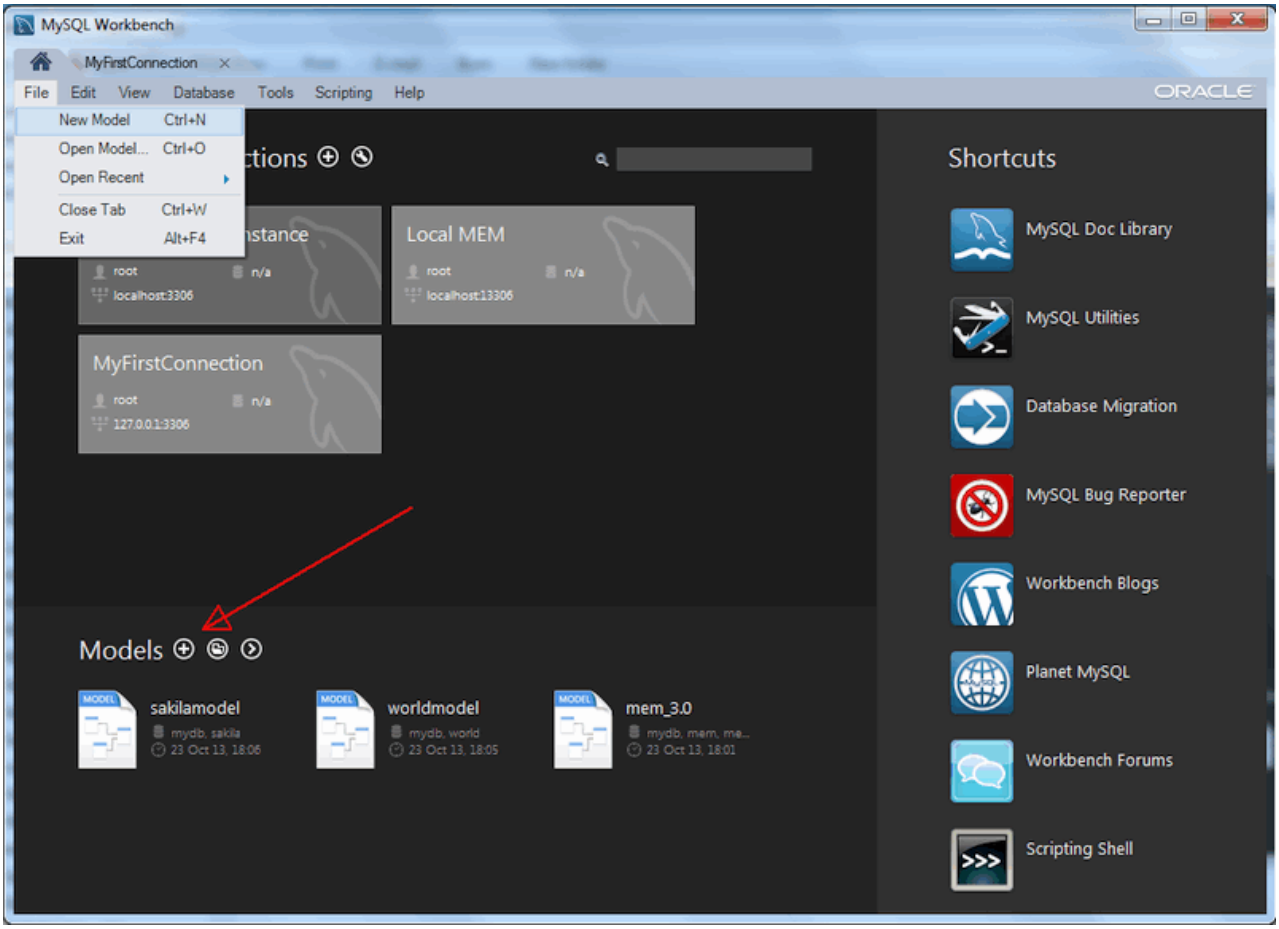
- EER Diagrams
- Physical Schemata
- Schema Privileges
- SQL Scripts
- Model Notes

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

1.3.2 Creating a Model

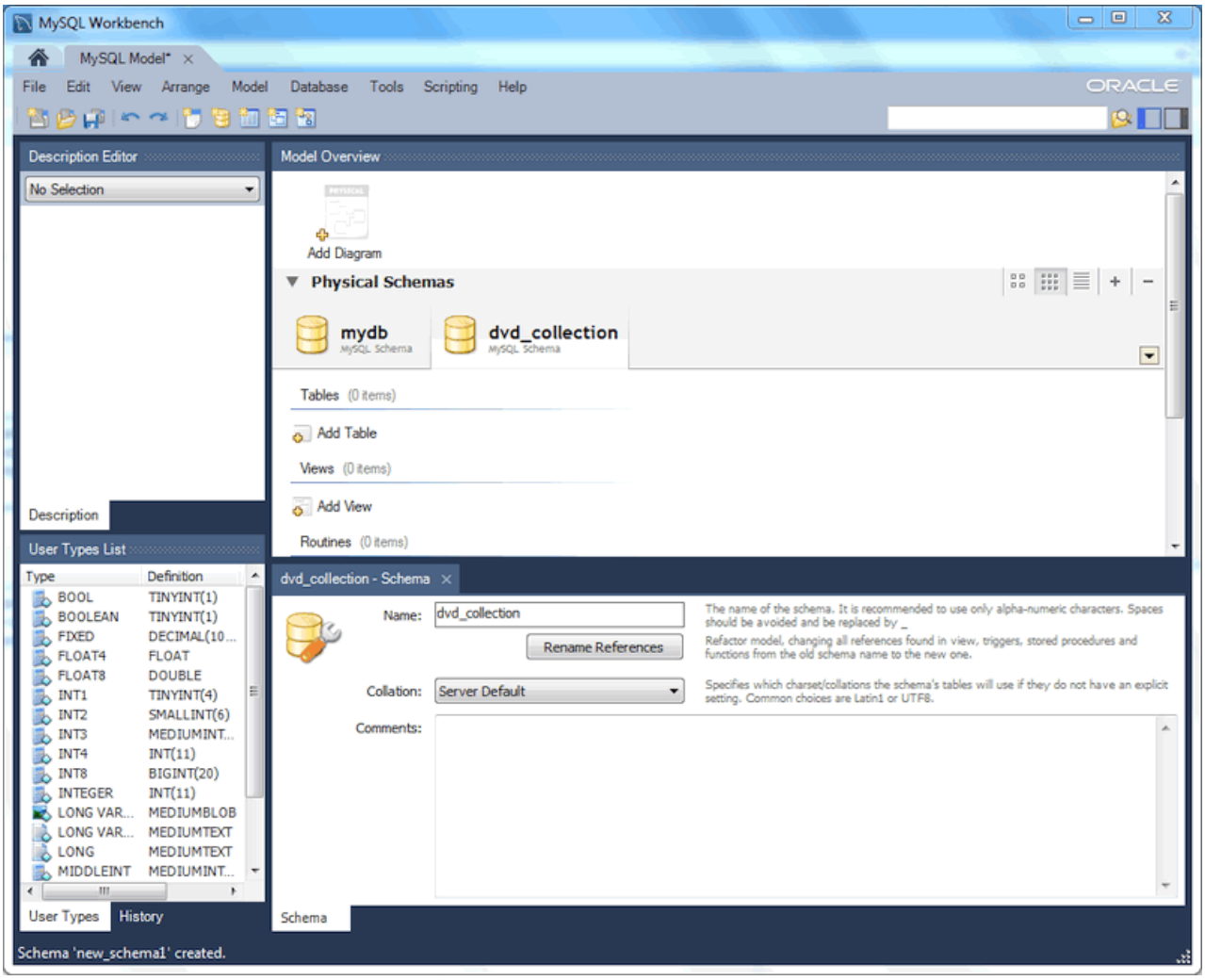
This section provides a tutorial introduction to MySQL models by showing you how to create a new database model, and how to forward engineer a model to a live MySQL server.

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



Creating a Model - Home Window

2. Click the + button on the right side of the Physical Schemata toolbar to add a new schema. The default schema name is "new_schema1", now change it to "dvd_collection" by modifying its Name field. Confirm this change in the Physical Schemata panel. Now you are ready to add a table.



Creating a Model - New Schema

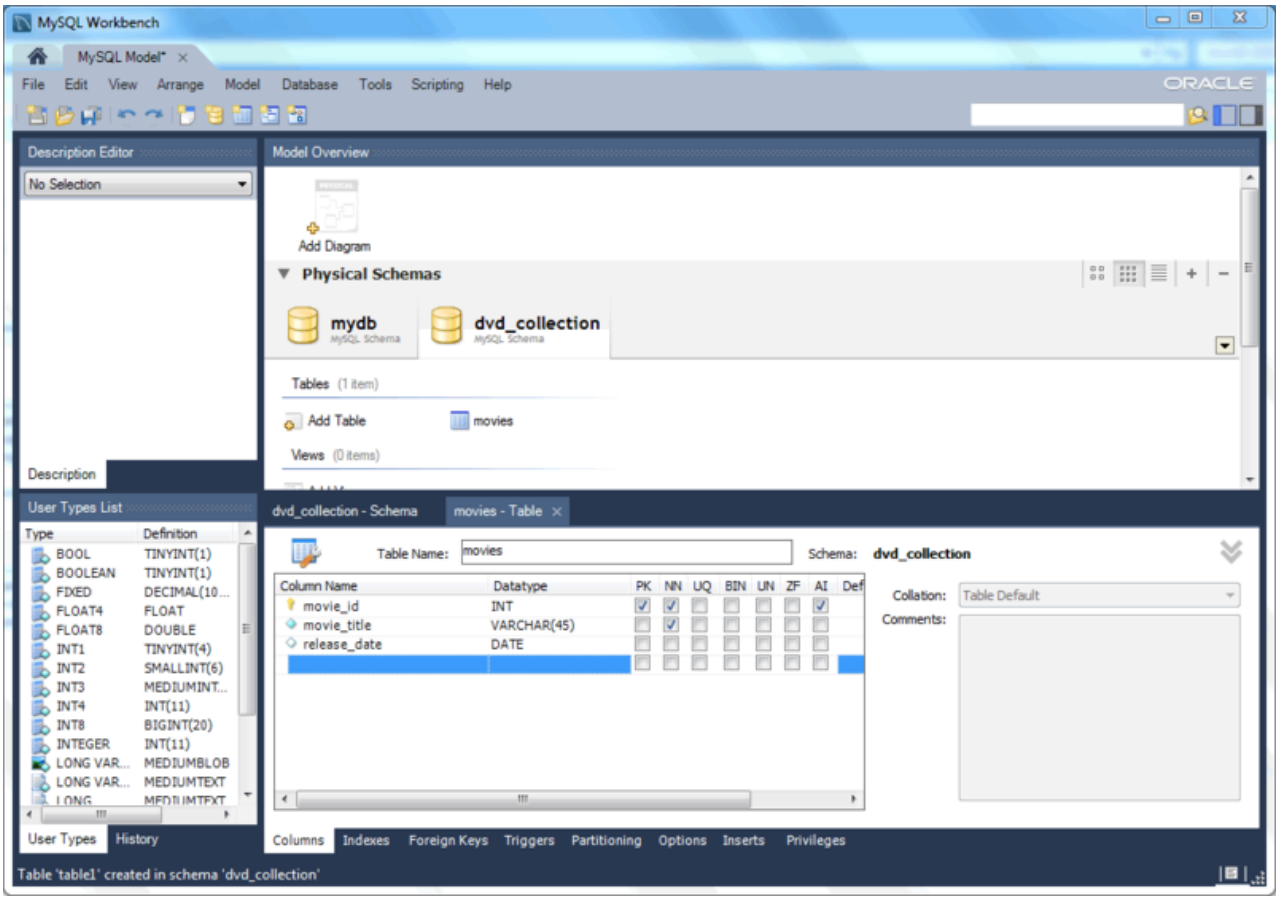
3. Double-click Add Table in the Physical Schemata section.

4. This automatically loads the table editor with the default table name table1. Edit its Table Name field and change the table name from "table1" to "movies".

5. Next, add columns to your table. Double-click a Column Name cell, and the first field defaults to "moviesid" because (by default) MySQL Workbench appends "id" to the table name for the initial field. Change "moviesid" to "movie_id" and keep the Datatype as INT, and also select the PK (PRIMARY KEY), NN (NOT NULL), and AI (AUTO_INCREMENT) check boxes.

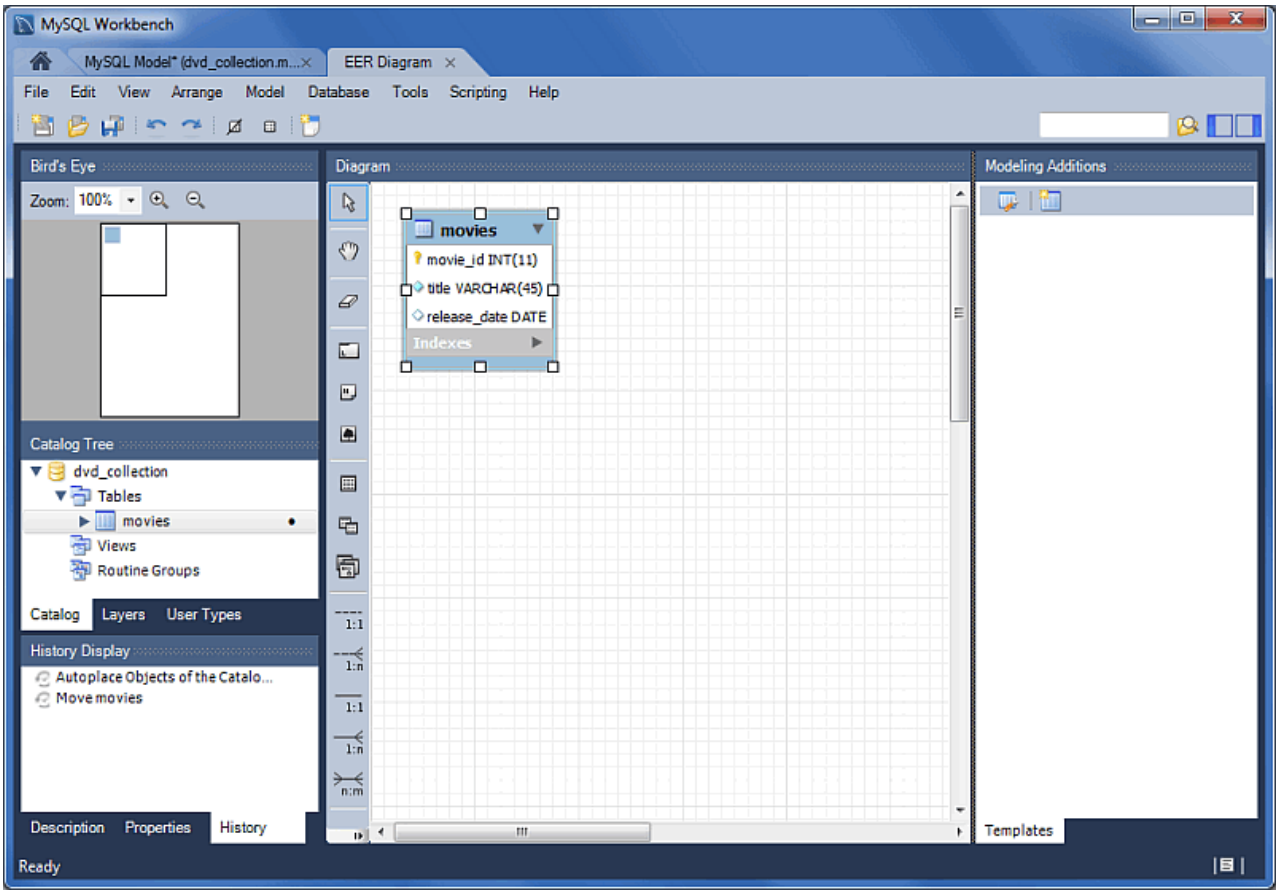
6. Complete all columns using the same method as described above:

Column Name	Data Type	Column Properties
movie_id	INT	PK, NN, AI
movie_title	VARCHAR(45)	NN
release_date	DATE (YYYY-MM-DD)	None



Creating a Model - Editing table columns

7. For a visual representation (EER diagram) of this schema, select Model, Create Diagram from Catalog Objects to create the EER Diagram for the model.



Creating a Model - EER Diagram

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

    a. Note: To open the table editor, either change back to the MySQL Model tab and right-click on the movies table, or right-click on movies in the EER diagram and select an Edit 'movies' option.

9. Save the model by choosing File, Save Model from the main menu, or click Save Model to Current File on the toolbar. Enter a model name at the file prompt. For this tutorial, enter "Home_Media" and then Save the model.

    a. In addition: To synchronizing your new model with the live MySQL server, confirm that you already created a MySQL connection. Now forward engineer your model to the live MySQL server. Select Database, Forward Engineer. From the main menu to open the Forward Engineer to Database wizard.

1.3.3 Creating a Foreign Key

This section provides a tutorial introduction to create a Foreign Key in MySQL models.

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

2. First, click the invoice_item table; notice that a red border indicates that this table is selected. Next, click the invoice table. This creates a foreign key in the invoice_item table, the table on the "many" side of the relationship. This relationship between the two tables is shown graphically in crow's foot notation.

3. 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.

4. Click the Foreign key Name field. The referenced table should show in the Referenced Table column and the appropriate column in the Referenced Column column.

5. To delete the relationship between two tables, click the line joining the tables and then press Control+Delete.

6. Experiment with the other tools on the vertical toolbar. Delete a relationship by selecting the eraser tool and clicking the line joining two tables. Create a view, add a text object, or add a layer.

7. Save your changes to a MySQL Workbench Models file (mwb extension) by choosing Save from the File menu or by using the keyboard command Control+S.

Worksheet

1. Create database named "STAFF" and create two tables along with the specified fields. The following tables show the structure of STAFF database. Note that all fields, except primary key, must be set to allow NULL values.

USERGROUP Table

Field	Type	Length Values	Extra	Primary Key
USERGROUP_ID	INT		Auto_increment	Yes
USERGROUP_CODE	VARCHAR	50		
USERGROUP_NAME	VARCHAR	50		
USERGROUP_REMARK	VARCHAR	255		
USERGROUP_URL	VARCHAR	50		

USER Table

Field	Type	Length Values	Extra	Primary Key
USER_ID	INT		Auto_increment	Yes
USER_TITLE	VARCHAR	25		
USER_FNAME	VARCHAR	50		
USER_LNAME	VARCHAR	50		
USER_GENDER	VARCHAR	25		
USER_EMAIL	VARCHAR	50		
USER_NAME	VARCHAR	25		
USER_PASSWD	VARCHAR	25		
USER_GROUPID	INT			
DISABLE	INT			

2. Complete Foreign Key relation between two tables, each user can belong to only 1 group.

3. Complete database in Database Design (Modeling) as EER Diagram

4. Now forward engineer your model to the live MySQL server.

    a. Hint: Select Database, Forward Engineer... from the main menu to open the Forward Engineer to Database wizard.

5. Add each table with 5 records of data.