Chapter 2

How to use MySQL Workbench and other development tools

Before you start the exercises…

Before you start these exercises, you need to install the MySQL server and MySQL Workbench. The procedures for doing that are provided in appendix A (PC) and B (Mac).

In addition, you'll need to get the mgs_ex_starts directory from your instructor. This directory contains some script files that you need to do the exercises.

Exercises

In these exercises, you'll use MySQL Workbench to create the My Guitar Shop database, to review the tables in this database, and to enter SQL statements and run them against this database.

Make sure the MySQL server is running

1. Start MySQL Workbench and open a connection for managing the server.
2. Check whether the MySQL server is running. If it isn't, start it. When you're done, close the Admin tab.

Use MySQL Workbench to create the My Guitar Shop database

3. Open a connection to start querying.
4. Open the script file named my_guitar_shop.sql that's in the mgs_ex_starts directory by clicking the Open SQL Script File button in the SQL Editor toolbar. Then, use the resulting dialog box to locate and open the file.
5. Execute the entire script by clicking the Execute SQL Script button in the code editor toolbar or by pressing Ctrl+Shift+Enter. When you do, the Output window displays messages that indicate whether the script executed successfully.

Use MySQL Workbench to review the My Guitar Shop database

6. In the Object Browser window, expand the node for the database named my_guitar_shop so you can see all of the database objects it contains. If it isn't displayed in the Object Browser window, you may need to click on the Refresh button to display it.
7. View the data for the Categories and Products tables.
8. Navigate through the database objects and view the column definitions for at least the Categories and Products tables.

Use MySQL Workbench to enter and run SQL statements

9. Double-click on the my_guitar_shop database to set it as the default database. When you do that, MySQL Workbench should display the database in bold.
10. Open a code editor tab. Then, enter and run this SQL statement:

```
SELECT product_name FROM products
```

11. Delete the *e* at the end of product_name and run the statement again. Note the error number and the description of the error.
12. Open another code editor tab. Then, enter and run this statement:

```
SELECT COUNT(*) AS number_of_products
FROM products
```

Use MySQL Workbench to open and run scripts

13. Open the script named product_details.sql that's in the mgs_ex_starts directory. Note that this script contains just one SQL statement. Then, run the statement.
14. Open the script named product_summary.sql that's in the mgs_ex_starts directory. Note that this opens another code editor tab.
15. Open the script named product_statements.sql that's in the mgs_ex_starts directory. Notice that this script contains two SQL statements that end with semicolons.
16. Press the Ctrl+Shift+Enter keys or click the Execute SQL Script button to run both of the statements in this script. Note that this displays the results in two Result tabs. Make sure to view the results of both SELECT statements.
17. Move the insertion point into the first statement and press Ctrl+Enter to run just that statement.
18. Move the insertion point into the second statement and press Ctrl+Enter to run just that statement.
19. Exit from MySQL Workbench.

Chapter 3

How to retrieve data from a single table

Exercises

Enter and run your own SELECT statements

In these exercises, you'll enter and run your own SELECT statements.

1. Write a SELECT statement that returns four columns from the Products table: product_code, product_name, list_price, and discount_percent. Then, run this statement to make sure it works correctly.

   Add an ORDER BY clause to this statement that sorts the result set by list price in descending sequence. Then, run this statement again to make sure it works correctly. This is a good way to build and test a statement, one clause at a time.

2. Write a SELECT statement that returns one column from the Customers table named full_name that joins the last_name and first_name columns.

   Format this column with the last name, a comma, a space, and the first name like this:

   ```
   Doe, John
   ```

   Sort the result set by last name in ascending sequence.

   Return only the customers whose last name begins with letters from M to Z.

3. Write a SELECT statement that returns these columns from the Products table:

product_name	The product_name column
list_price	The list_price column
date_added	The date_added column

   Return only the rows with a list price that's greater than 500 and less than 2000.

   Sort the result set in descending sequence by the date_added column.

4. Write a SELECT statement that returns these column names and data from the Products table:

product_name	The product_name column
list_price	The list_price column
discount_percent	The discount_percent column
discount_amount	A column that's calculated from the previous two columns
discount_price	A column that's calculated from the previous three columns

   Round the discount_amount and discount_price columns to 2 decimal places.

   Sort the result set by discount price in descending sequence.

   Use the LIMIT clause so the result set contains only the first 5 rows.

5. Write a SELECT statement that returns these column names and data from the Order_Items table:

item_id	The item_id column
item_price	The item_price column
discount_amount	The discount_amount column
quantity	The quantity column
price_total	A column that's calculated by multiplying the item price by the quantity
discount_total	A column that's calculated by multiplying the discount amount by the quantity
item_total	A column that's calculated by subtracting the discount amount from the item price and then multiplying by the quantity

   Only return rows where the item_total is greater than 500.

   Sort the result set by item total in descending sequence.

Work with nulls and test expressions

6. Write a SELECT statement that returns these columns from the Orders table:

order_id	The order_id column
order_date	The order_date column
ship_date	The ship_date column

   Return only the rows where the ship_date column contains a null value.

7. Write a SELECT statement without a FROM clause that uses the NOW function to create a row with these columns:

today_unformatted	The NOW function unformatted
today_formatted	The NOW function in this format: DD-Mon-YYYY

   This displays a number for the day, an abbreviation for the month, and a four-digit year.

8. Write a SELECT statement without a FROM clause that creates a row with these columns:

price	100 (dollars)
tax_rate	.07 (7 percent)
tax_amount	The price multiplied by the tax
total	The price plus the tax

   To calculate the fourth column, add the expressions you used for the first and third columns.