AM05 Data Mgmt – Assignment 1

This assignment has three parts. For the first two parts you will need to download *guitar_shop.zip* from the course website. Extract its contents to a directory you can find again. It contains script files that you need to do the exercises.

Part 1. Create Database – nothing to submit for this part

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. You do not have to submit anything for this.

Make sure the MySQL server is running

1. Start MySQL Workbench and open a connection for the root user.
2. Check whether the MySQL server is running. If it isn't, start it.

Use MySQL Workbench to create the My Guitar Shop database.

3. Open the script file named my_guitar_shop.sql (from *guitar_shop.zip*) 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.
4. Execute the entire script by clicking the Execute SQL Script button in the SQL 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

5. In the Schemas category of the Navigator 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 Schemas tab of the Navigator window, you may need to click on the Refresh button to display it.
6. View the data for the Categories and Products tables.
7. 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

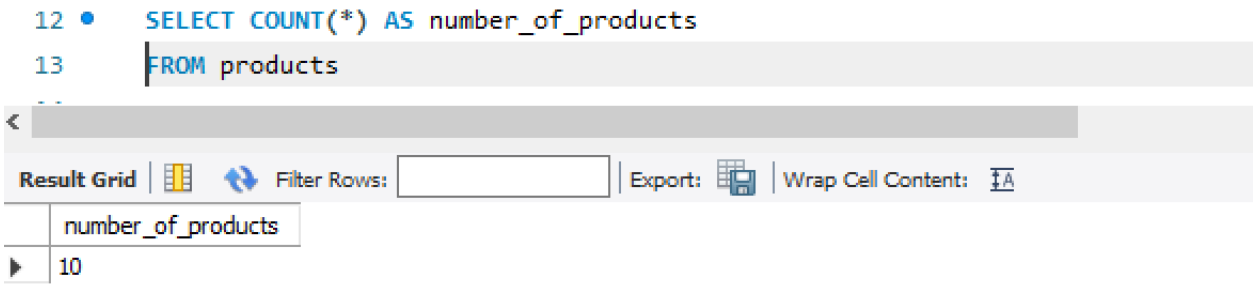
8. Double-click on the my_guitar_shop database to set it as the default database. When you do that, MySQL Workbench should display the database in bold.
9. Open a SQL editor tab. Then, enter and run this SQL statement:

```
SELECT product_name FROM products
```

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

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

    You should get something link this



Use MySQL Workbench to open and run scripts

12. Open the script named product_details.sql (from *guitar_shop.zip*). Note that this script contains just one SQL statement. Then, run the statement.
13. Open the script named product_summary.sql that's in the (from *guitar_shop.zip*). Note that this opens another SQL editor tab.
14. Open the script named product_statements.sql that's in the (from *guitar_shop.zip*). Notice that this script contains two SQL statements that end with semicolons.
15. 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.
16. Move the insertion point into the first statement and press Ctrl+Enter to run just that statement.
17. Move the insertion point into the second statement and press Ctrl+Enter to run just that statement.
18. Exit from MySQL Workbench.

Part 2. SELECT Query Challenges

Now you write and test your own SELECT statements. **When you think you have the query working correctly copy the code to a Word document that will be your assignment submission along with a cropped screen shot that shows the results of the query (see exercise 11 above for an example of a suitable screen shot).**

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 the last_name column in ascending sequence.

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

   NOTE: When comparing strings of characters, 'M' comes before any string of characters that begins with 'M'. For example, 'M' comes before 'Mulligan'.

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 by the date_added column in descending sequence.

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 the discount_price column 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 the item_total column 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.

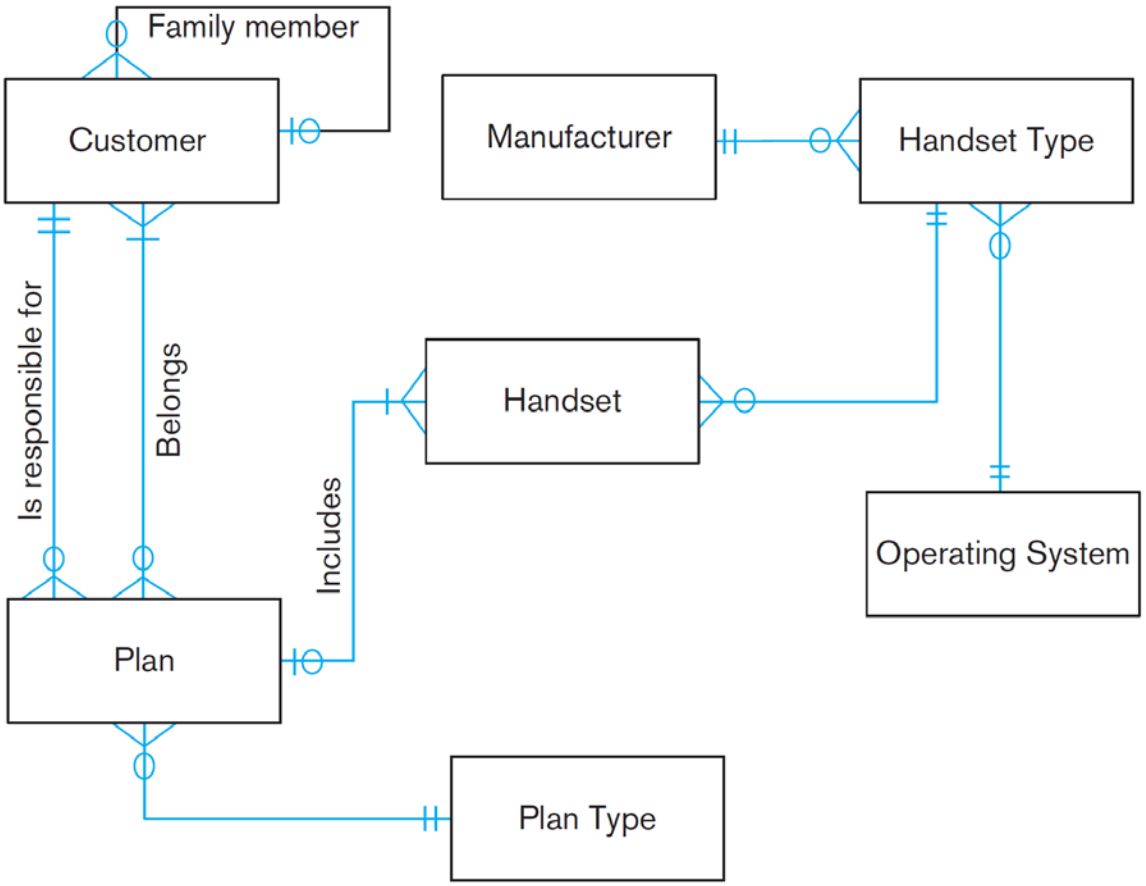
8. Write a SELECT statement without a FROM clause that creates a row with these columns:

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.

Part 3. ERD Challenges

In this part you can demonstrate you understanding of ERDs. Submit your answers in the same Word document you are using for Part 2 of this assignment. You can paste a screenshot of your ERD for question 5.



A cellular operator needs a database to keep track of its customers, their subscription plans, and the handsets (mobile phones) that they are using. The ERD above illustrates the key entities of interest to the operator and the relationships between them. Based on the figure, answer the following questions and explain the rationale for your response.

1. Can a customer exist without a plan?
2. Can a handset be associated with multiple plans?
3. Characterize the degree and the cardinalities of the relationship that connects Customer to itself. Explain its meaning.
4. Assume a handset type exists that can utilize multiple operating systems. Could this situation be accommodated within the model?
5. Prepare an ERD for the following business context. Be sure to express the degree and minimum and maximum cardinalities for each relationship.

- A book is identified by its ISBN number, and it has a title, a price, and a date of publication. It is published by a publisher, which has its own ID number and a name. Each book has exactly one publisher, but one publisher typically publishes multiple books over time.
- A book is written by one or multiple authors. Each author is identified by an author number and has a name and date of birth. Each author has either one or multiple books; in addition, occasionally data are needed regarding prospective authors who have not yet published any books.
- Better information is needed regarding the relationship between a book and its authors. Specifically, it is important to record the percentage of the royalties that belongs to a specific author, whether or not a specific author is a lead author of the book, and each author's position in the sequence of the book's authors.
- A book can be part of a series, which is also identified as a book and has its own ISBN number. One book can belong to several sets, and a set consists of at least one but potentially many books.