Josephine Quagraine

gmca 5  17/04/2023

write up of mysql group tasks

Table of Contents

[1.1 Query 1 2](#_Toc134309644)

[1.1.1 Additional query input to obtain information where customer\_id=1 2](#_Toc134309645)

[1.2 Query 2 3](#_Toc134309646)

[1.2.1 **Task 1:** to change points to read times by +10 and \* 100 3](#_Toc134309647)

[1.2.2 **Task 2:** Write SQL query to return all the products in our database in the result set. Show columns; name, unit price, and new price (unit price \* 1.1) 4](#_Toc134309648)

[1.2.3 **Task 3:** to create new query to find all customers with birth date of > ‘1990-01-01’ 5](#_Toc134309649)

[1.2.4 **Task 4:** Write query to find out name of product with most amount in stock 6](#_Toc134309650)

[1.2.5 **Task 5:** Write query to find out name of most expensive product 6](#_Toc134309651)

[1.2.6 **Task 6:** Write query to find out details of oldest customer 7](#_Toc134309652)

[1.3 To create an enhanced entity-relationship (EER) diagram 8](#_Toc134309653)

List of Figures

[Figure 1 Output of query to select \* from customers table 2](#_Toc132692231)

[Figure 2 Output of query to select for order\_id=1, order by first\_name 2](#_Toc132692232)

[Figure 3 Output of query 2 3](#_Toc132692233)

[Figure 4 Output using query select last\_name,first\_name,points,(points+10)\*100 from customers; 3](#_Toc132692234)

[Figure 5 Result of query “ select last\_name,first\_name,points,(points+10)\*100 as discount\_factor from customers;” 4](#_Toc132692235)

[Figure 6 Results of query select \* from products; 4](#_Toc132692236)

[Figure 7 Result of query select name, unit\_price, (unit\_price \* 1.1) as new\_price from products; 5](#_Toc132692237)

[Figure 8 Result of query select \* from customers where birth\_date>'1990-01-01' ; 5](#_Toc132692238)

[Figure 9 Result of query select name from productswhere quantity\_in\_stock = (select max(quantity\_in\_stock)from products); 6](#_Toc132692239)

[Figure 10 Result of query select name FROM products WHERE unit\_price = (SELECT MAX(unit\_price) FROM products); 6](#_Toc132692240)

[Figure 11 Result of query for select first\_name, last\_name, address, birth\_date from customers where birth\_date = (select min(birth\_date) from customers); 7](#_Toc132692241)

[Figure 12 Diagram showing results of reverse engineering run 8](#_Toc132692242)

[Figure 13 EER diagram generated using the sql\_store schema 9](#_Toc132692243)

## Query 1

Imported create database SQL script into MySQL Workbench using the ‘Open a SQL script file in a new query tab’ after which the script was run to create the database. This generated the sql\_store database. Using this database, my teammates and I followed instructions to write a query to select all from customers as shown below in figure 1 using “**select \* from customers**”.

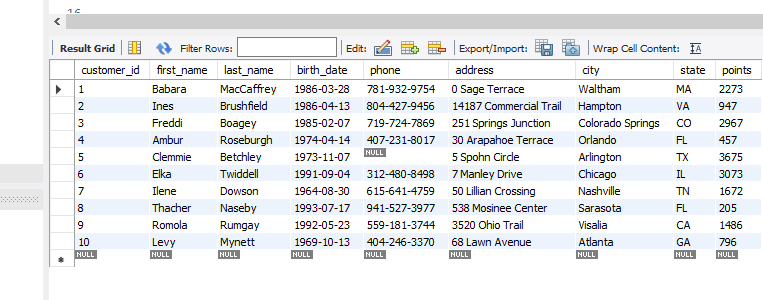


Figure 1 Output of query to select \* from customers table

### Additional query input to obtain information where customer\_id=1

We then added to the query as instructed by using the where clause to only obtain information where order ID was = 1 and ordered by first name. Additional query added to select \* from customers was “**where customer\_id = 1 order by first\_name;**”. Together, this generated the results shown in figure 2.

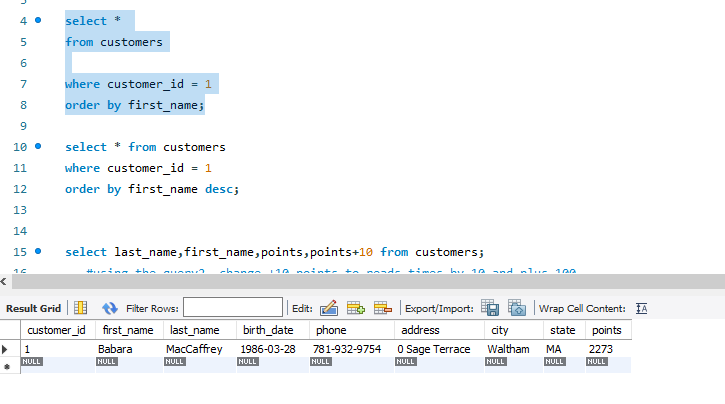


Figure 2 Output of query to select for order\_id=1, order by first\_name

## Query 2

Keyed in the query “**select last\_name, first\_name, points, points+10 from customers;**” as indicated in the powerpoint slide to obtain the results below.

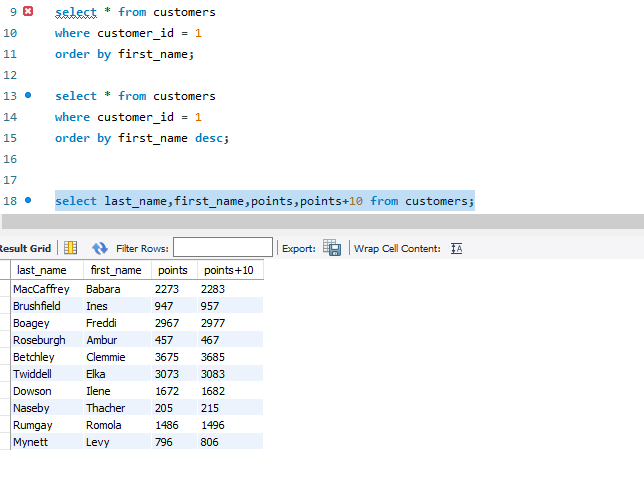


Figure 3 Output of query 2

### **Task 1:** to change points to read times by +10 and \* 100

To do this task, we wrote the query “**select last\_name, first\_name, points, (points+10) \* 100 from customers;**” This resulted in the output shown in figure 4 below.

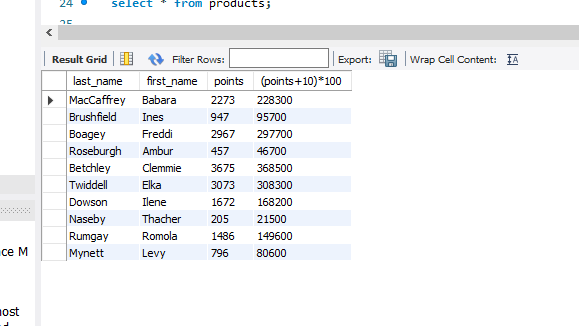


Figure 4 Output using query select last\_name,first\_name,points,(points+10)\*100 from customers;

#### To change query 2 code to create a discount factor which shows in the header section

This task was to rename the “(points +10) \* 100” header generated in the previous task to “discount\_factor”. To do this, we wrote the query “**select last\_name, first\_name, points, (points+10) \* 100 as discount\_factor from customers;**”. Result is as indicated in figure 5 below.

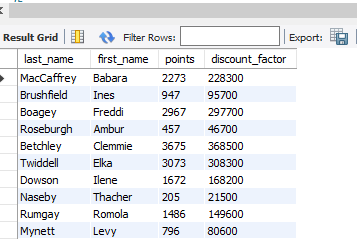


Figure 5 Result of query “ select last\_name,first\_name,points,(points+10)\*100 as discount\_factor from customers;”

### **Task 2:** Write SQL query to return all the products in our database in the result set. Show columns; name, unit price, and new price (unit price \* 1.1)

To do this task, we wrote a query “**select \* from products;**” to select all from products which brought up the results below in figure 6.

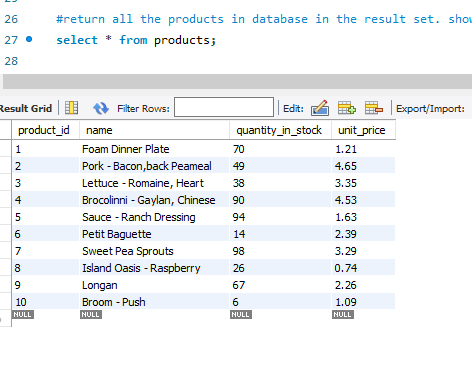


Figure 6 Results of query select \* from products;

Next, we wrote a query to only show the columns name, unit price and new price which was generated based on the expression unit price \* 1.1. which effectively increased product price by 10%. Query used was “**select name, unit\_price, (unit\_price \* 1.1) as new\_price from products;**”

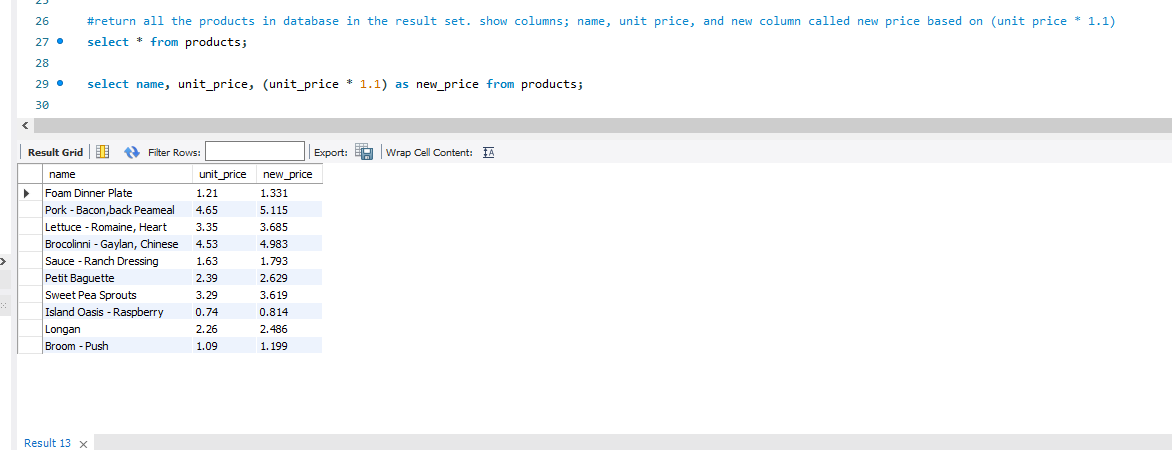


Figure 7 Result of query select name, unit\_price, (unit\_price \* 1.1) as new\_price from products;

### **Task 3:** to create new query to find all customers with birth date of > ‘1990-01-01’

To do the task, we wrote a query to only select for customers who had birthdays from after 1990-01-01 using the query “**select \* from customers where birth\_date>'1990-01-01';”** This query brought out data for 3 people as indicated below.

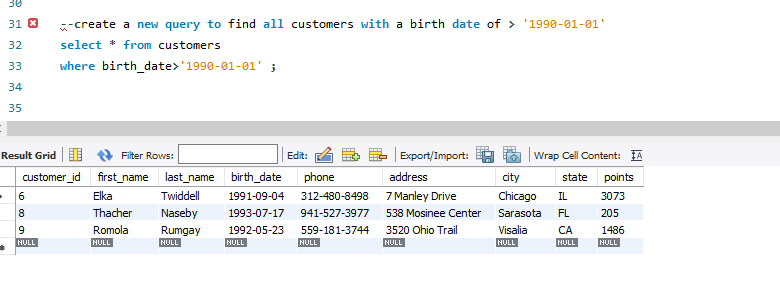


Figure 8 Result of query select \* from customers where birth\_date>'1990-01-01' ;

### **Task 4:** Write query to find out name of product with most amount in stock

For this task, wrote query to use sql inventory “**use sql\_inventory**;”. We then selected all from products (“**select \* from products**”) to find out what was in the table after which we wrote a query to only determine the name of product with most amount in stock. The query we generated was “**select name from products where quantity\_in\_stock = (select max(quantity\_in\_stock) from products);**”. This query showed that the name of the most abundant product in stock was Sweet Pea Sprouts as indicated in figure 9.

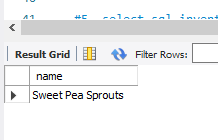


Figure 9 Result of query select name from productswhere quantity\_in\_stock = (select max(quantity\_in\_stock)from products);

### **Task 5:** Write query to find out name of most expensive product

This task still utilized the sql-inventory database. We wrote the query “**select name FROM products WHERE unit\_price = (SELECT MAX(unit\_price) FROM products)**;”. This query told us that pork- bacon, back peameal was the most expensive product as indicated below.

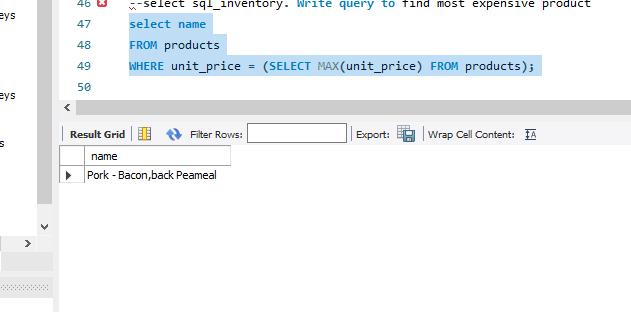


Figure 10 Result of query select name FROM products WHERE unit\_price = (SELECT MAX(unit\_price) FROM products);

### **Task 6:** Write query to find out details of oldest customer

For this task, we used the sql\_store database by keying in the query **use sql\_store;.** To check what was in the customer tables, we used **select \* from customers;.** We then wrote the query “**select \* from customers; select first\_name, last\_name, address, birth\_date from customers where birth\_date = (select min(birth\_date) from customers);**” to find out the first name, last name, address and birthdate of the oldest customer on file.

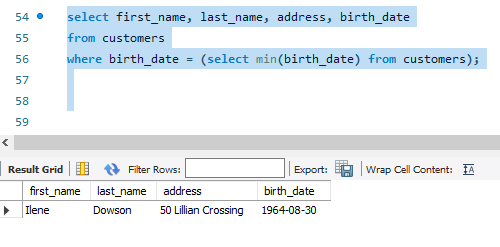


Figure 11 Result of query for select first\_name, last\_name, address, birth\_date from customers where birth\_date = (select min(birth\_date) from customers);

## To create an enhanced entity-relationship (EER) diagram

To do this, we clicked on the database tab and selected reverse engineer which retrieved all schemas within the database system and checked configuration. After clicking next, selected the **sql\_store schema** which we were interested in. Clicked next to enable retrieval of objects from schema. After successful execution, 7 objects (tables) were retrieved in total as shown below.

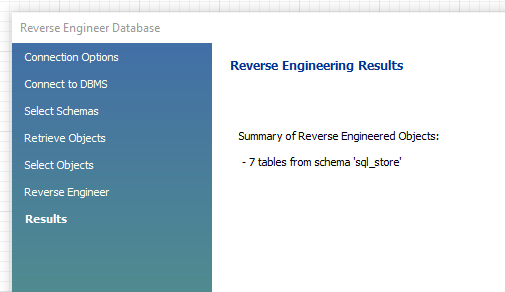


Figure 12 Diagram showing results of reverse engineering run

After obtaining the EER diagram, we noticed that the table ‘order items notes’ was not connected to any other table. As such, it was irrelevant to our EER diagram so that was deleted. We then rearranged the diagram to look like how it was shown in the PowerPoint slide as indicated below in figure 13.

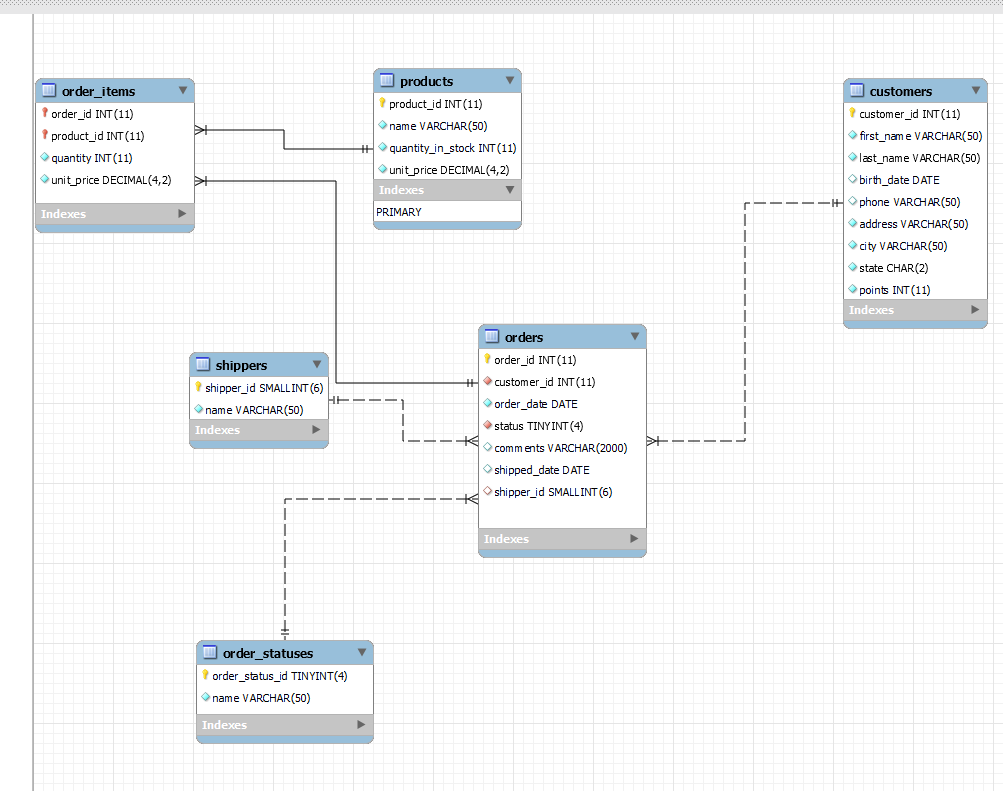


Figure 13 EER diagram generated using the sql\_store schema