

2023

# MYSQL Task

BY  
SOWMYA SENTHIL

JUST IT | London

## Query 1

USE sql\_store;

SELECT \*

FROM Customers;

## Query 1 continued

SELECT \*

FROM Customers;

WHERE customer\_id = 1

ORDER BY first\_name;

## Output: Query 1

The screenshot shows the MySQL Workbench interface. The 'Query 1' editor contains the following SQL code:

```
1 USE sql_store;
2
3 SELECT *
4 FROM Customers
5 WHERE customer_id = 1
6 ORDER BY first_name;
```

The 'Result Grid' displays the output of the query, showing columns: customer\_id, first\_name, last\_name, birth\_date, phone, address, and city. The results are ordered by first\_name.

customer_id	first_name	last_name	birth_date	phone	address	city
1	Barbara	MacCaffrey	1986-03-28	781-932-9754	5 Sage Terrace	Waltham
2	Bernie	Donson	1964-08-30	615-641-4758	90 Liban Crossing	Nashville
3	Frank	Rumsey	1965-02-07	735-724-7869	251 Springs Junction	Colerain Spring
4	Arthur	Roseburgh	1974-04-14	407-231-8017	30 Arapahoe Terrace	Orlando
5	Clarence	Butcher	1973-11-07	608	5 Spohn Circle	Arlington
6	Ella	Twissell	1991-09-04	312-480-8498	7 Harley Drive	Chicago
7	Frank	Rumsey	1965-02-07	735-724-7869	251 Springs Junction	Colerain Spring
8	Thatcher	Naseby	1983-07-17	941-527-9977	538 Mosinee Center	Sarasota
9	Rumsey	Rumsey	1962-05-23	555-181-2744	3520 Ohio Trail	Vienna
10	Levy	Mynett	1969-10-13	404-246-3370	68 Leann Avenue	Atlanta

The 'Output' pane shows the execution log for Query 1, including the query text and the number of rows affected.

## Query 2:

SELECT last\_name, first\_name, points, points + 10

FROM customers;

## Output : Query 2

The screenshot shows the MySQL Workbench interface. The 'Query 2' editor contains the following SQL code:

```
1 USE sql_store;
2
3 SELECT *
4 FROM Customers;
5
6 SELECT last_name, first_name, points, points + 10
7 FROM Customers;
```

The 'Result Grid' displays the output of the query, showing columns: last\_name, first\_name, points, and points + 10. The results are ordered by last\_name.

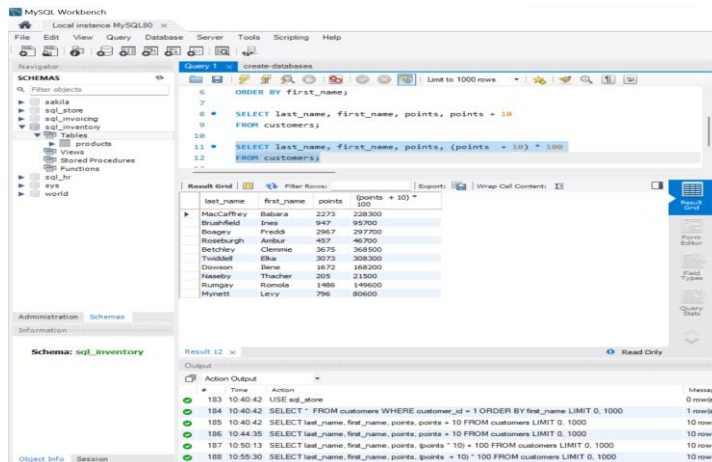
last_name	first_name	points	points + 10
MacCaffrey	Barbara	2273	2283
Butcher	Frank	967	977
Rumsey	Frank	2967	2977
Roseburgh	Arthur	608	618
Butcher	Clarence	3673	3683
Twissell	Ella	3673	3683
Donson	Bernie	1873	1883
Naseby	Thatcher	208	218
Rumsey	Rumsey	1498	1508
Mynett	Levy	796	806

The 'Output' pane shows the execution log for Query 2, including the query text and the number of rows affected.

## Task 1:

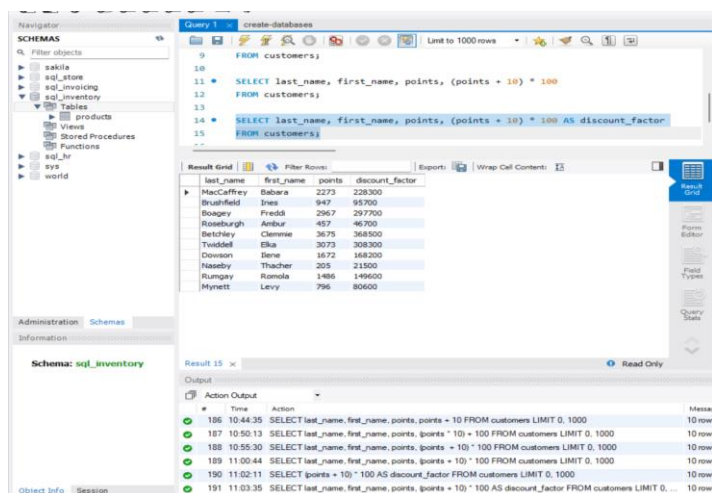
```
SELECT last_name, first_name, points, (points + 10) * 100  
FROM customers;
```

## Output: Task 1



last_name	first_name	points	(points + 10) * 100
MacCaffrey	Barbara	2273	228300
Brushfield	Ines	947	95700
Rongey	Fredd	2967	297700
Rosburgh	Ambar	457	46700
Betchley	Clemmie	3675	368500
Twiddell	Elka	3073	308300
Dowson	Bena	1672	168200
Naseby	Thacher	205	21500
Rumgay	Romola	1486	149600
Mynett	Levy	796	80600

```
SELECT last_name, first_name, points, (points + 10) * 100  
FROM customers;
```



last_name	first_name	points	discount_factor
MacCaffrey	Barbara	2273	228300
Brushfield	Ines	947	95700
Rongey	Fredd	2967	297700
Rosburgh	Ambar	457	46700
Betchley	Clemmie	3675	368500
Twiddell	Elka	3073	308300
Dowson	Bena	1672	168200
Naseby	Thacher	205	21500
Rumgay	Romola	1486	149600
Mynett	Levy	796	80600

## Task 2:

```
SELECT name, unit_price, (unit_price * 1.1)  
FROM products;
```

## Output: Task 2

The screenshot shows the MySQL Workbench interface. On the left, the 'SCHEMAS' pane shows the 'sakila' database selected. The 'Query 1' window contains the following SQL code:

```
12 FROM customers;
13
14 SELECT last_name, first_name, points, (points + 10) * 100 AS discount_factor
15 FROM customers;
16
17 SELECT name, unit_price, (unit_price * 1.1)
18 FROM products;
```

The 'Result Grid' shows the results of the third query, displaying columns 'name', 'unit\_price', and '(unit\_price \* 1.1)'. The results are as follows:

name	unit_price	(unit_price * 1.1)
Foam Dinner Plate	1.21	1.331
Pork - Bacon/Jack Peameal	4.65	5.115
Lettuce - Romaine, Heart	3.35	3.685
Brocolinni - Gaylan, Chinese	4.53	4.983
Sauce - Ranch Dressing	1.63	1.793
Petit Baguette	2.39	2.629
Sweet Pea Sprouts	3.29	3.619
Island Oasis - Raspberry	0.74	0.814
Longan	2.26	2.486
Broom - Push	1.09	1.199

The 'Output' pane shows the execution log with messages for each query.

SELECT name, unit\_price, (unit\_price \* 1.1) AS 'new price'

FROM products;

The screenshot shows the MySQL Workbench interface. On the left, the 'SCHEMAS' pane shows the 'sakila' database selected. The 'Query 1' window contains the following SQL code:

```
15 FROM customers;
16
17 SELECT name, unit_price, (unit_price * 1.1)
18 FROM products;
19
20 SELECT name, unit_price, (unit_price * 1.1) AS 'new price'
21 FROM products;
```

The 'Result Grid' shows the results of the third query, displaying columns 'name', 'unit\_price', and 'new price'. The results are as follows:

name	unit_price	new price
Foam Dinner Plate	1.21	1.331
Pork - Bacon/Jack Peameal	4.65	5.115
Lettuce - Romaine, Heart	3.35	3.685
Brocolinni - Gaylan, Chinese	4.53	4.983
Sauce - Ranch Dressing	1.63	1.793
Petit Baguette	2.39	2.629
Sweet Pea Sprouts	3.29	3.619
Island Oasis - Raspberry	0.74	0.814
Longan	2.26	2.486
Broom - Push	1.09	1.199

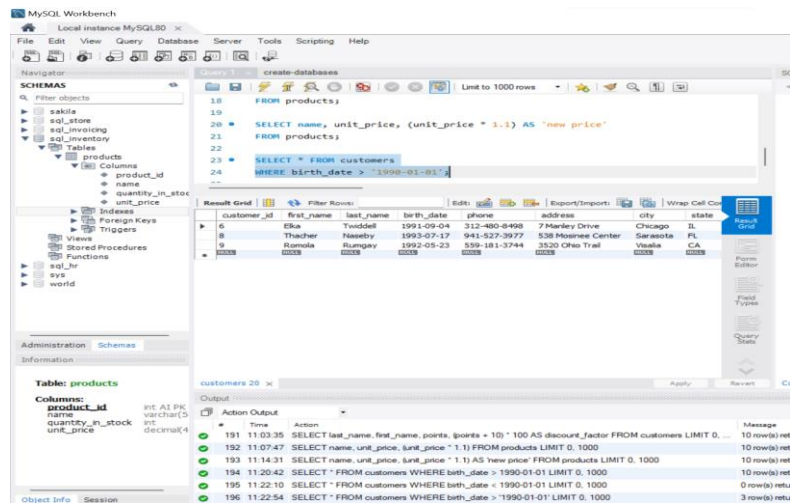
The 'Output' pane shows the execution log with messages for each query.

## Task 3:

SELECT \* FROM customers

WHERE birth\_date > '1990-01-01';

## Output: Task 3



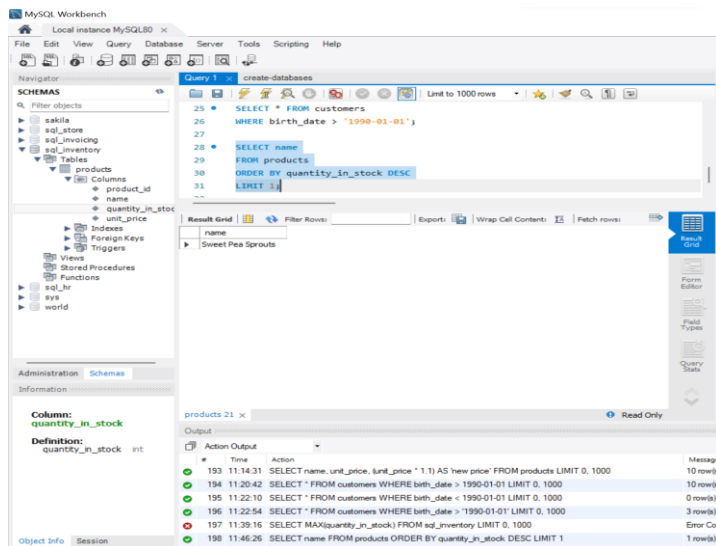
## Task 4:

SELECT name

FROM products

ORDER BY quantity\_in\_stock DESC

LIMIT 1;



## Task 5:

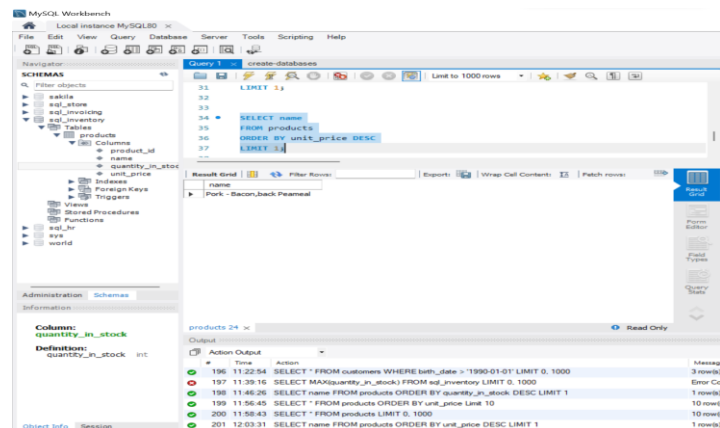
SELECT name

FROM products

ORDER BY unit\_price DESC

LIMIT 1;

## Output: Task 5



MySQL Workbench

Query 1: create-databases

```
SELECT name
FROM products
ORDER BY unit_price DESC
LIMIT 1;
```

Result Grid

name
Pork - Bacon,back,Peameal

products 24 x

Output

Action	Time	Message
196	11:22:54	SELECT * FROM customers WHERE birth_date > '1990-01-01' LIMIT 0, 1000 3 row(s)
197	11:26:16	SELECT MAX(quantity_in_stock) FROM sql_inventory LIMIT 0, 1000 Error Co
198	11:46:26	SELECT name FROM products ORDER BY quantity_in_stock DESC LIMIT 1 1 row(s)
199	11:56:45	SELECT * FROM products ORDER BY unit_price Limit 10 10 row(s)
200	11:58:43	SELECT * FROM products LIMIT 0, 1000 10 row(s)
201	12:03:31	SELECT name FROM products ORDER BY unit_price DESC LIMIT 1 1 row(s)

## Task 6:

USE sql\_store;

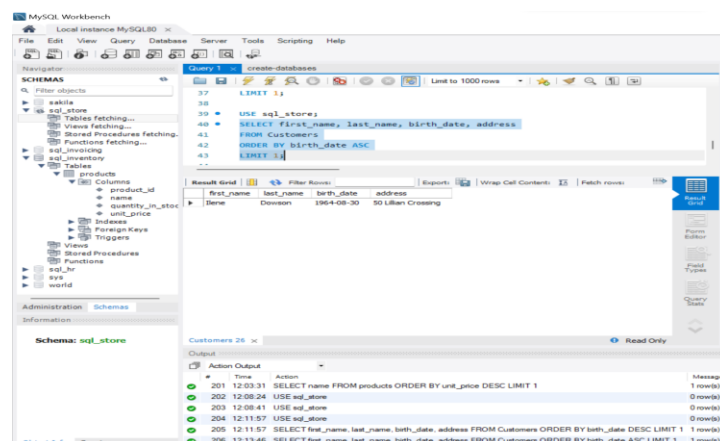
SELECT first\_name, last\_name, birth\_date, address

FROM Customers

ORDER BY birth\_date ASC

LIMIT 1;

## Output: Task 6



MySQL Workbench

Query 1: create-databases

```
USE sql_store;
SELECT first_name, last_name, birth_date, address
FROM Customers
ORDER BY birth_date ASC
LIMIT 1;
```

Result Grid

first_name	last_name	birth_date	address
Rene	Devos	1964-09-30	50 Lilaan Crossing

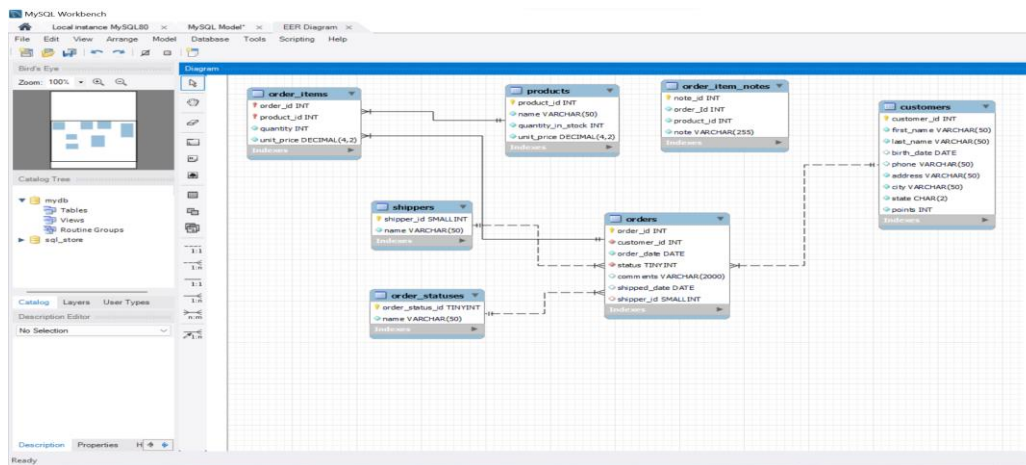
Customers 26 x

Output

Action	Time	Message
201	12:03:31	SELECT name FROM products ORDER BY unit_price DESC LIMIT 1 1 row(s) ne
202	12:08:24	USE sql_store 0 row(s) af
203	12:08:41	USE sql_store 0 row(s) af
204	12:11:57	USE sql_store 0 row(s) af
205	12:11:57	SELECT first_name, last_name, birth_date, address FROM Customers ORDER BY birth_date DESC LIMIT 1 1 row(s) ne
206	12:13:46	SELECT first_name, last_name, birth_date, address FROM Customers ORDER BY birth_date ASC LIMIT 1 1 row(s) ne

## Task 7:

### EER Diagram:



### Relationship between tables:

I just explore the tables in sql store which contains

Order\_items, shippers, products, customers, orders, orders

### Primary Key:

A Primary Key is a column or set of columns that uniquely identifies each row in a table. It is used to enforce the integrity and consistency of the data in the table, and it ensures that each row has a unique identifier.

In this EER diagram sql store table is taken into account;

The primary key for the tables order\_items and products is product\_id INT.

### Foreign Key:

A foreign key (FK) is a column or combination of columns that is used to establish and enforce a link between the data in two tables to control the data that can be stored in the foreign key table.

There is no foreign key between the tables in these SQL store databases.

THANK YOU