SQL ASSIGNMENT

ABSTRACT
MySQL syntax and query writing.
Working on schemas.
Understanding PK and FK.
Creating Relational Database Schema.

Busra ArlierMohyuddin
Data Technician Bootcamp Training

Contents

Query 1

Input the following into the file Query 1

USE sql_store;

SELECT * FROM customers;

```
# connect to database
USE store;

# QUERY 1

SELECT * FROM customers;
```

Result Grid

customer_id	first_name	last_name	birth_date	phone	address	city	state	points
1	Babara	MacCaffrey	1986-03-28	781-932-9754	0 Sage Terrace	Waltham	MA	2273
2	Ines	Brushfield	1986-04-13	804-427-9456	14187 Commercial Trail	Hampton	VA	947
3	Freddi	Boagey	1985-02-07	719-724-7869	251 Springs Junction	Colorado Springs	CO	2967
4	Ambur	Roseburgh	1974-04-14	407-231-8017	30 Arapahoe Terrace	Orlando	FL	457
5	Clemmie	Betchley	1973-11-07	NULL	5 Spohn Circle	Arlington	TX	3675
6	Elka	Twiddell	1991-09-04	312-480-8498	7 Manley Drive	Chicago	IL	3073
7	Ilene	Dowson	1964-08-30	615-641-4759	50 Lillian Crossing	Nashville	TN	1672
8	Thacher	Naseby	1993-07-17	941-527-3977	538 Mosinee Center	Sarasota	FL	205
9	Romola	Rumgay	1992-05-23	559-181-3744	3520 Ohio Trail	Visalia	CA	1486
10	Levy	Mynett	1969-10-13	404-246-3370	68 Lawn Avenue	Atlanta	GA	796
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

Add the following into the file Query 1
ORDER BY first_name

```
# QUERY 1

SELECT * FROM customers
ORDER BY first_name;
```

Result Grid

customer_id	first_name	last_name	birth_date	phone	address	city	state	points
4	Ambur	Roseburgh	1974-04-14	407-231-8017	30 Arapahoe Terrace	Orlando	FL	457
1	Babara	MacCaffrey	1986-03-28	781-932-9754	0 Sage Terrace	Waltham	MA	2273
5	Clemmie	Betchley	1973-11-07	NULL	5 Spohn Circle	Arlington	TX	3675
6	Elka	Twiddell	1991-09-04	312-480-8498	7 Manley Drive	Chicago	IL	3073
3	Freddi	Boagey	1985-02-07	719-724-7869	251 Springs Junction	Colorado Springs	CO	2967
7	Ilene	Dowson	1964-08-30	615-641-4759	50 Lillian Crossing	Nashville	TN	1672
2	Ines	Brushfield	1986-04-13	804-427-9456	14187 Commercial Trail	Hampton	VA	947
10	Levy	Mynett	1969-10-13	404-246-3370	68 Lawn Avenue	Atlanta	GA	796
9	Romola	Rumgay	1992-05-23	559-181-3744	3520 Ohio Trail	Visalia	CA	1486
8	Thacher	Naseby	1993-07-17	941-527-3977	538 Mosinee Center	Sarasota	FL	205
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

Query 2

SELECT last_name, first_name, points, points + 10

FROM CUSTOMERS

```
# QUERY 2
SELECT last_name, first_name, points, points + 10
FROM CUSTOMERS;


# TASK 1
```

Result Grid | Filter Rows: | Export: | Wrap Cell Con

last_name	first_name	points	points + 10
MacCaffrey	Babara	2273	2283
Brushfield	Ines	947	957
Boagey	Freddi	2967	2977
Roseburgh	Ambur	457	467
Betchley	Clemmie	3675	3685
Twiddell	Elka	3073	3083
Dowson	Ilene	1672	1682
Naseby	Thacher	205	215
Rumgay	Romola	1486	1496
Mynett	Levy	796	806

TASK 1

Using the Query 2 you created change the points to reads times by 10 and plus 100.

```
# TASK 1

SELECT last_name, first_name, points, points + 10
FROM CUSTOMERS;

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

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

last_name	first_name	points	(points + 10) *100
MacCaffrey	Babara	2273	228300
Brushfield	Ines	947	95700
Boagey	Freddi	2967	297700
Roseburgh	Ambur	457	46700
Betchley	Clemmie	3675	368500
Twiddell	Elka	3073	308300
Dowson	Ilene	1672	168200
Naseby	Thacher	205	21500
Rumgay	Romola	1486	149600
Mynett	Levy	796	80600

Change the Query 2 code to create a discount factor so the table now shows a discount header and change the (point + 10) *100

```
# TASK 1

SELECT last_name, first_name, points, points + 10
FROM CUSTOMERS;

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

SELECT last_name, first_name, points, (points + 10) *100 AS discount_factor
FROM CUSTOMERS;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

last_name	first_name	points	discount_factor
MacCaffrey	Babara	2273	228300
Brushfield	Ines	947	95700
Boagey	Freddi	2967	297700
Roseburgh	Ambur	457	46700
Betchley	Clemmie	3675	368500
Twiddell	Elka	3073	308300
Dowson	Ilene	1672	168200
Naseby	Thacher	205	21500
Rumgay	Romola	1486	149600
Mynett	Levy	796	80600

TASK 2

Write an SQL query to return all the products in our database in the result set.

```
# TASK 2

SELECT * FROM products;
```

Result Grid | Filter Rows: | Edit: | Export/Import:

product_id	name	quantity_in_stock	unit_price
1	Foam Dinner Plate	70	1.21
2	Pork - Bacon,back Peameal	49	4.65
3	Lettuce - Romaine, Heart	38	3.35
4	Brocolinni - Gaylan, Chinese	90	4.53
5	Sauce - Ranch Dressing	94	1.63
6	Petit Baguette	14	2.39
7	Sweet Pea Sprouts	98	3.29
8	Island Oasis - Raspberry	26	0.74
9	Longan	67	2.26
10	Broom - Push	6	1.09
NULL	NULL	NULL	NULL

Show three columns, name, unit price, and new column called new price which is based on this expression, (unit price * 1.1). So, what you are doing is increasing the product price of each by 10%.

```

# TASK 2

SELECT * FROM products;

SELECT name, unit_price, (unit_price * 1.1) AS new_price
FROM products;

```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

name	unit_price	new_price
Foam Dinner Plate	1.21	1.331
Pork - Bacon,back 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

TASK 3

Create a new query to find all the customers with a birth date of > '1990-01-01'

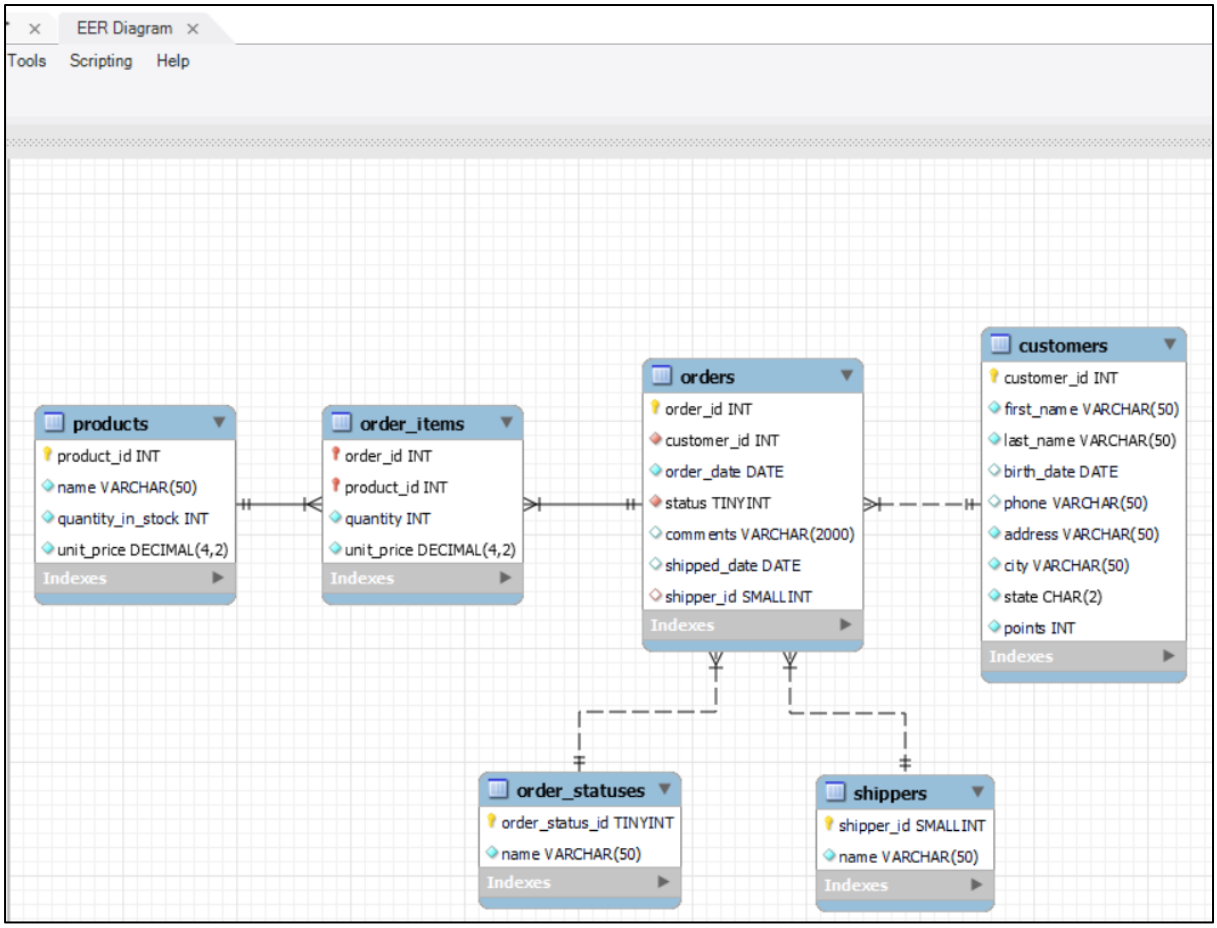
```
# TASK 3

SELECT * FROM customers
WHERE birth_date > '1990-01-01';
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

customer_id	first_name	last_name	birth_date	phone	address	city	state	points
6	Elka	Twiddell	1991-09-04	312-480-8498	7 Manley Drive	Chicago	IL	3073
8	Thacher	Naseby	1993-07-17	941-527-3977	538 Mosinee Center	Sarasota	FL	205
9	Romola	Rumgay	1992-05-23	559-181-3744	3520 Ohio Trail	Visalia	CA	1486
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

EER Diagram

Enhanced Entity-Relationship (EER) diagrams are an essential part of the modeling interface in MySQL Workbench. EER diagrams provide a visual representation of the relationships among the tables in your model.



PRIMARY KEY, FOREIGN KEY AND COMPOSITE KEY

A primary key is a column or a group of columns that uniquely identifies each row in a table. You create a primary key for a table by using the PRIMARY KEY constraint.

Each table can contain only one primary key. All columns that participate in the primary key must be defined as NOT NULL. SQL Server automatically sets the NOT NULL constraint for all the primary key columns if the NOT NULL constraint is not specified for these columns.

A foreign key is a field (or collection of fields) in one table, that refers to the primary key in another table.
The table with the foreign key is called the child table, and the table with the primary key is called the referenced or parent table.

A composite key can be defined as a combination of multiple columns, and these columns are used to identify all the rows that are involved uniquely. Even though a single column can't identify any row uniquely, a combination of over one column can uniquely identify any record.

<u>Customers Entity</u>

- PRIMARY KEY is customer_id

<u>Products Entity</u>

- PRIMARY KEY is product_id

<u>Orders Entity</u>

- PRIMARY KEY is order_id
- FOREIGN KEYS ARE customer_id, shipper_id and order_status_id

<u>Order_items Entity</u>

- COMPOSITE KEY is order_id and product_id

<u>Shippers Entity</u>

- PRIMARY KEY is shipper_id

<u>Order_statuses Entity</u>

- PRIMARY KEY is order_status_id

ADDITIONAL QUERIES

#1 - Fetch Top 5 customers by Points (customer_id, first_name, last_name and points) ORDERED BY POINTS

```
# ADDITIONAL QUERIES

#1 To fetch Top 5 customers by Points (customer_id, first_name, last_name and points) ORDERED BY POINTS

SELECT customer_id, first_name, last_name, points FROM customers
WHERE points > 1000
ORDER BY points
LIMIT 5;
```

customer_id	first_name	last_name	points
9	Romola	Rumgay	1486
7	Ilene	Dowson	1672
1	Babara	MacCaffrey	2273
3	Freddi	Boagey	2967
6	Elka	Twiddell	3073
NULL	NULL	NULL	NULL

#2 FIND MAX(points) , AVERAGE(unit_price) AND MIN(points)

- SELECT MAX(points) FROM customers;

```
#2 SELECT MIN, MAX AND AVERAGE

SELECT MAX(points) FROM customers;
```

Result Grid | Filter Rows: | Exp

MAX(points)
3675

- SELECT AVG(unit_price) FROM order_items;

```
SELECT AVG(unit_price) FROM order_items;
```

Result Grid | Filter Rows: | Export:

AVG(unit_price)
6.386111

- SELECT MIN(points) FROM customers;

Limit to 500 rows

```
SELECT MIN(points) FROM customers;
```

Result Grid | Filter Rows: | Export: | Wra

MIN(points)
205

#3 Find CITY names that starts with the letter ‘c’

```
#3 Finding string starting with with a specific value (city name starts with 'C')
SELECT * FROM customers
WHERE city LIKE 'c%';
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

customer_id	first_name	last_name	birth_date	phone	address	city	state	points
3	Freddi	Boagey	1985-02-07	719-724-7869	251 Springs Junction	Colorado Springs	CO	2967
6	Elka	Twiddell	1991-09-04	312-480-8498	7 Manley Drive	Chicago	IL	3073
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

#4 Retrieve the first four characters of the city names from customers table

```
#4 Retrieve the first four characters of city name from customers table
SELECT SUBSTRING(city, 1, 4)
FROM customers;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

SUBSTRING(city, 1, 4)
Walt
Hamp
Colo
Orla
Arli
Chic
Nash
Sara
Visa
Atla

#5 Find customer_id from orders, order_date between 01/01/2017 – 01/02/2017

```
#5 BETWEEN Clause

SELECT customer_id FROM orders
WHERE order_date BETWEEN '2017-01-01' AND '2017-02-01';
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

customer_id
2

#6 HAVING Clause

```
#6 HAVING Clause
SELECT COUNT(customer_id), state
FROM Customers
GROUP BY state
HAVING COUNT(customer_id) > 0;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

COUNT(customer_id)	state
1	MA
1	VA
1	CO
2	FL
1	TX
1	IL
1	TN
1	CA
1	GA

#7 JOIN and ALIAS

```sql
#7 JOIN and ALIAS
SELECT c.customer_id, c.last_name, c.city, c.points,
o.status, o.shipped_date
FROM customers AS c
JOIN orders AS o
ON c.customer_id = o.customer_id;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

customer_id	last_name	city	points	status	shipped_date
6	Twiddell	Chicago	3073	1	NULL
7	Dowson	Nashville	1672	2	2018-08-03
8	Naseby	Sarasota	205	1	NULL
2	Brushfield	Hampton	947	1	NULL
5	Betchley	Arlington	3675	2	2017-08-26
10	Mynett	Atlanta	796	1	NULL
2	Brushfield	Hampton	947	2	2018-09-23
5	Betchley	Arlington	3675	1	NULL
10	Mynett	Atlanta	796	2	2017-07-06
6	Twiddell	Chicago	3073	2	2018-04-23

#8 LEFT JOIN AND GROUP BY

```sql
#8 LEFT JOIN AND GROUP BY
SELECT o.order_id, SUM(o.quantity) AS 'Total Quantity',
SUM(p.quantity_in_stock) AS 'Total Stock Quantity'
FROM order_items AS o
LEFT JOIN products AS p
ON o.product_id=p.product_id
GROUP BY o.order_id;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

order_id	Total Quantity	Total Stock Quantity
1	4	90
2	8	174
3	10	38
4	14	44
5	3	49
6	13	251
7	7	38
8	4	120
9	5	14
10	19	137

#9 COALESCE (For rows with empty phone numbers, we can fill it in with the value 'Unknown'.)

```
#9 COALESCE (For rows with empty phone numbers, we can fill it in with the value 'Unknown'.)
SELECT phone,
COALESCE(phone, 'Unknown') AS phone_coalesced
FROM customers;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

phone	phone_coalesced
781-932-9754	781-932-9754
804-427-9456	804-427-9456
719-724-7869	719-724-7869
407-231-8017	407-231-8017
NULL	Unknown
312-480-8498	312-480-8498
615-641-4759	615-641-4759
941-527-3977	941-527-3977
559-181-3744	559-181-3744
404-246-3370	404-246-3370

#10

```
#10
SELECT customer_id, order_id, status, order_date
FROM orders
WHERE (customer_id BETWEEN 1 AND 5 OR customer_id = 8)
AND status = 2;
```

Result Grid | Filter Rows: | Edit: | Export/Import:

customer_id	order_id	status	order_date
5	5	2	2017-08-25
2	7	2	2018-09-22
NULL	NULL	NULL	NULL