2023

MySQL

GLA 16

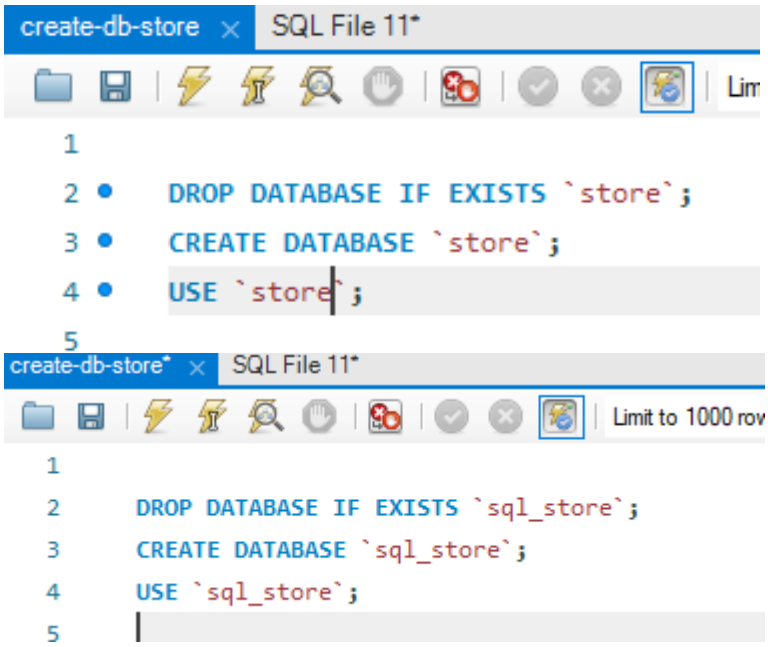
KRISTIN KUNDEVSKA

Contents

PowerPoint Tasks (Group work)

Query 1

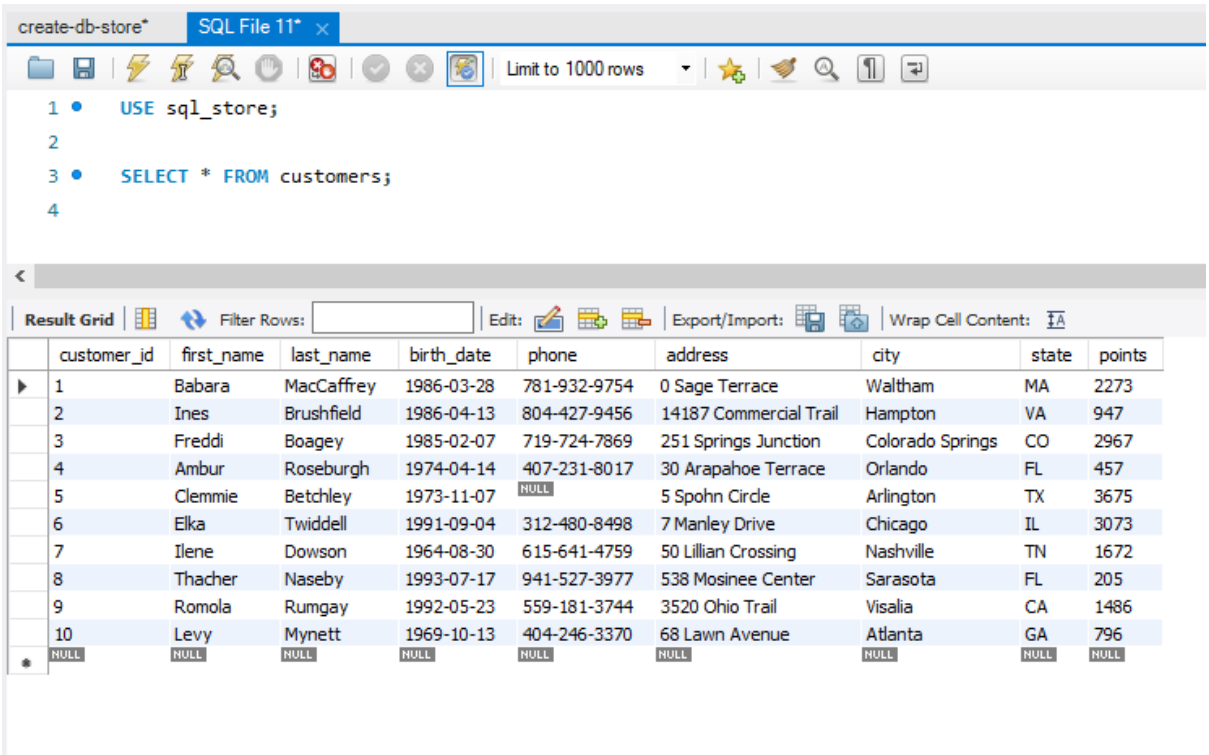
1. First, I changed the database name from *store* to *sql_store* to match the requirements on the task.



2. After that I generated the table of customers as reqired in the task.

```
USE sql_store;

SELECT * FROM customers;
```

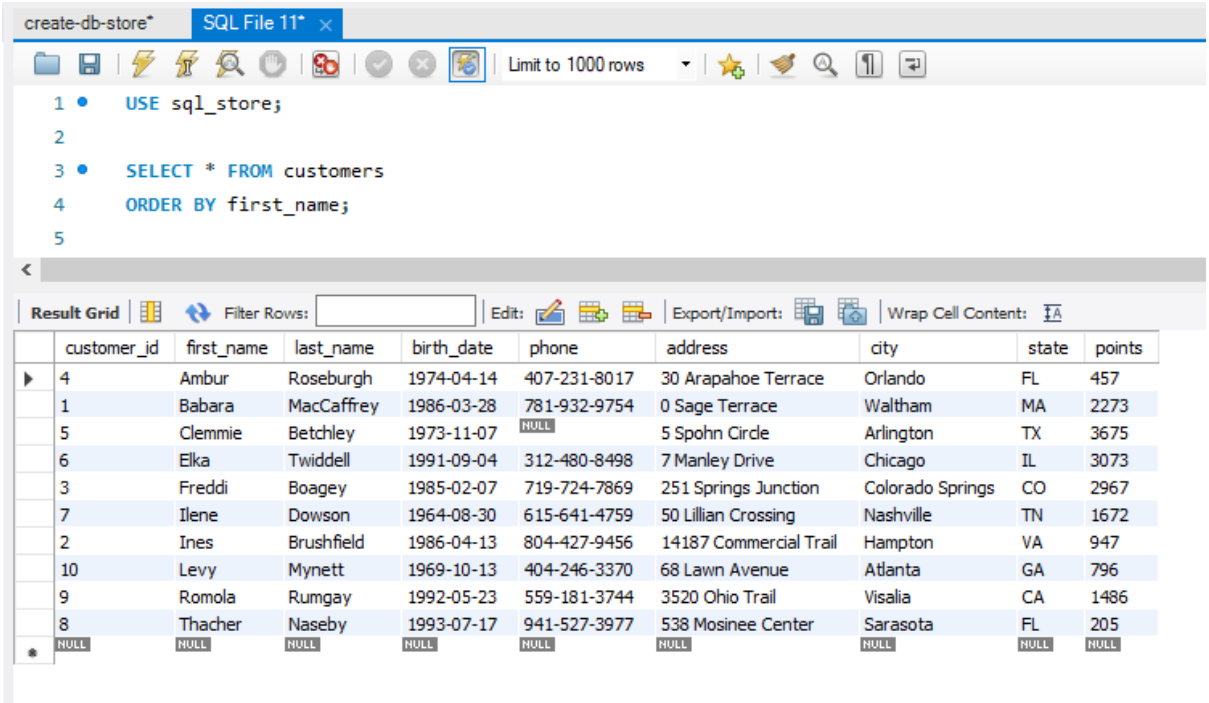


customer_id	first_name	last_name	birth_date	phone	address	city	state	points
1	Babara	MacCaffrey	1986-03-28	781-932-9754	0 Sage Terrace	Waltham	MA	2273
2	Ines	Brushfield	1986-04-13	804-427-9456	14187 Commercial Trail	Hampton	VA	947
3	Freddi	Boagey	1985-02-07	719-724-7869	251 Springs Junction	Colorado 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

3. The next task was to create the same table but this time sort it by the customers first name.

SELECT * FROM customers

ORDER BY first_name;



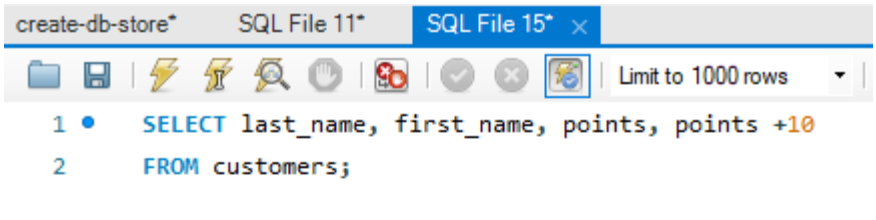
customer_id	first_name	last_name	birth_date	phone	address	city	state	points
4	Ambur	Roseburgh	1974-04-14	407-231-8017	30 Arapahoe Terrace	Orlando	FL	457
1	Babara	MacCaffrey	1986-03-28	781-932-9754	0 Sage Terrace	Waltham	MA	2273
5	Clemmie	Betchley	1973-11-07	NULL	5 Spohn Circle	Arlington	TX	3675
6	Elka	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

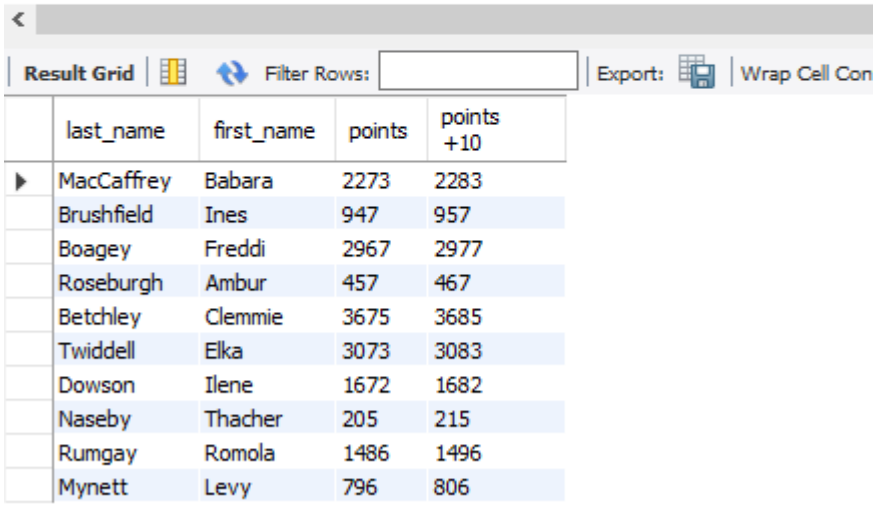
Query 2 SELECT

1. Creating a table with the customers first name, last name, the points they have and the points they have +10

SELECT first_name, last_name, points, points +10

FROM customers;





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

```
1 •  SELECT last_name, first_name, points, points *10 +100
2    FROM customers;
```

1. Multiplying the points by 10 and adding 100

SELECT first_name, last_name, points, points *10 +100

FROM customers;

esult Grid | Filter Rows: | Export: | Wrap Cell Conten

last_name	first_name	points	points *10 +100
MacCaffrey	Babara	2273	22830
Brushfield	Ines	947	9570
Boagey	Freddi	2967	29770
Roseburgh	Ambur	457	4670
Betchley	Clemmie	3675	36850
Twiddell	Elka	3073	30830
Dowson	Ilene	1672	16820
Naseby	Thacher	205	2150
Rumgay	Romola	1486	14960
Mynett	Levy	796	8060

2. In the next step I add 10 to the points first and then multiply the result by 100 and the result represents the discount.

SELECT first_name, last_name, points, (points +10) *100

AS discount

FROM customers;

```
1 •  SELECT last_name, first_name, points, (points +10 )*100
2    AS Discount
3    FROM customers;
4
```

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

last_name	first_name	points	Discount
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

3. This table Represents the increase in the unit_price by 1.1x for the products by name.

SELECT name, unit_price, unit_price * 1.1

AS new_price

FROM Products;

```
SELECT
    name,
    unit_price,
    unit_price * 1.1 AS new_price
FROM
    products;
```

Result Grid | Filter Rows: | Expo

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

4. This table only displays the customers who were born after January 1st 1990

SELECT * FROM customers

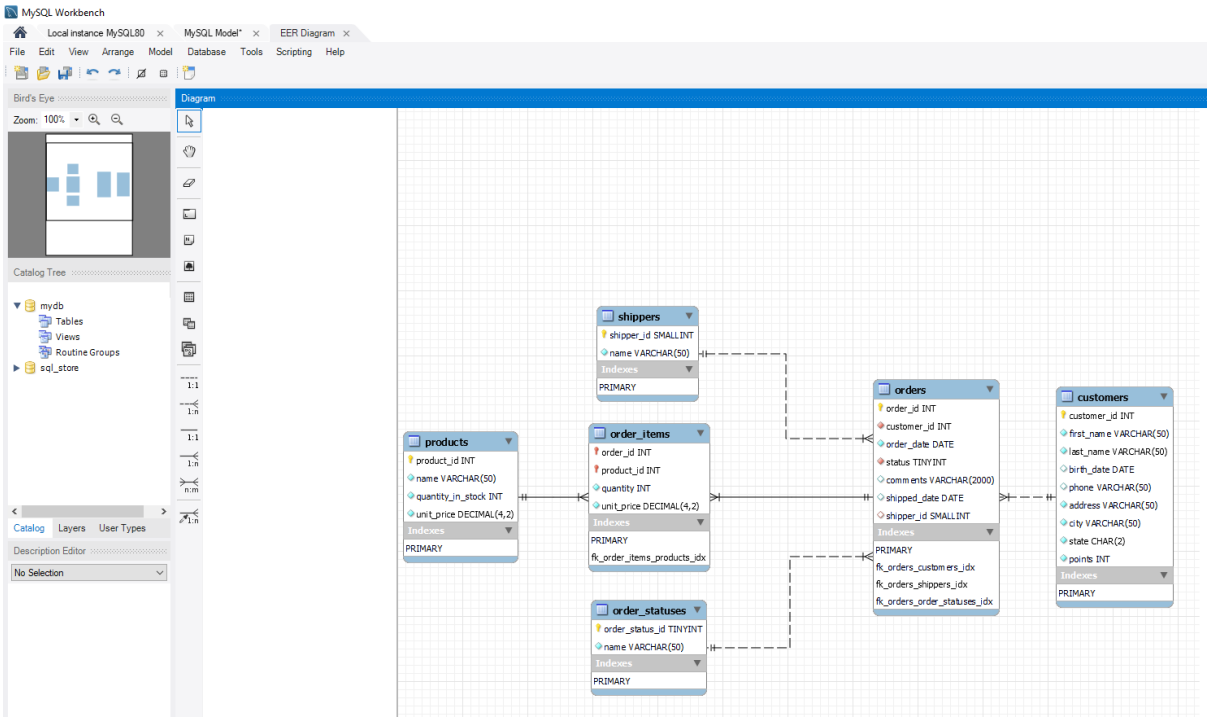
WHERE birth_date > '1990-01-01';

```
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

The schemas



Keys

1. Primary Keys:

- products: product_id
- shippers: shipper_id
- customers: customer_id
- order_statuses: order_status_id
- orders: order_id
- order_items: Composite primary key of (order_id, product_id)
- order_item_notes: note_id

2. Foreign Keys:

- orders: customer_id references customers(customer_id)
- orders: status references order_statuses(order_status_id)
- orders: shipper_id references shippers(shipper_id)
- order_items: order_id references orders(order_id)
- order_items: product_id references products(product_id)
- order_item_notes: order_id references orders(order_id)
- order_item_notes: product_id references products(product_id)

Individual Work

Create sample tables to cross-check data.

I started by creating tables for product, shippers, order_statues, orders, and order_items. I used the generated tables to make sure any calculations performed on the data were correct.

```
USE sql_store;
SELECT * FROM products;
```

```
USE sql_store;
SELECT * FROM shippers;
```

```
USE sql_store;
SELECT * FROM order_statuses;
```

```
USE sql_store;
SELECT * FROM orders;
```

```
USE sql_store;
SELECT * FROM order_items;
```

Retrieve product names and shipper names using joins.

Using joins to connect product names to shippers in order to retrieve which shippers provide which items

```
SELECT p.name AS product_name, s.name
AS shipper_name
FROM products p
JOIN order_items oi
ON p.product_id = oi.product_id
JOIN orders o
ON oi.order_id = o.order_id
```

```
JOIN shippers s
ON o.shipper_id = s.shipper_id;
```

Calculate total revenue for each shipper.

Representing the sum of ordered items unit price, multiplied by ordered items quantity, as total revenue. Then joining shipper's shipper id with the orders shipper id and joining the orders order id with the ordered items order id and then grouping the final result by shipper id to retrieve the total revenue of each shipper.

```
SELECT s.name AS shipper_name, SUM(oi.unit_price * oi.quantity)
AS total_revenue
FROM shippers s
LEFT JOIN orders o
ON s.shipper_id = o.shipper_id
LEFT JOIN order_items oi
ON o.order_id = oi.order_id
GROUP BY s.shipper_id;
```

Calculate the total number of orders and the average order quantity using the total order.

```
SELECT COUNT(*)
AS total_orders,
AVG(quantity)
AS avg_order_quantity
FROM order_items;
```

Retrieve the top 5 customers with the most points.

```
SELECT first_name, last_name, points
FROM customers
ORDER BY points DESC
LIMIT 5;
```

Calculate the number of orders handled by each shipper.

```
SELECT s.name AS shipper_name, COUNT(o.order_id) AS total_orders_handled
FROM shippers s
LEFT JOIN orders o ON s.shipper_id = o.shipper_id
GROUP BY s.shipper_id;
```

Retrieve product names and their corresponding shipper names.

```
SELECT
p.name AS product_name,
s.name AS shipper_name
FROM products p
JOIN order_items oi
ON p.product_id = oi.product_id
JOIN orders o
ON oi.order_id = o.order_id
JOIN shippers s
ON o.shipper_id = s.shipper_id
ORDER BY product_name;
```

Other Tasks

I asked for some feedback from my husband and a friend who are both programmers on how clear my comments were as I was unsure of how clear I was in the comments and if I needed to provide less or more information.