Task 4: SQL for Data Analysis

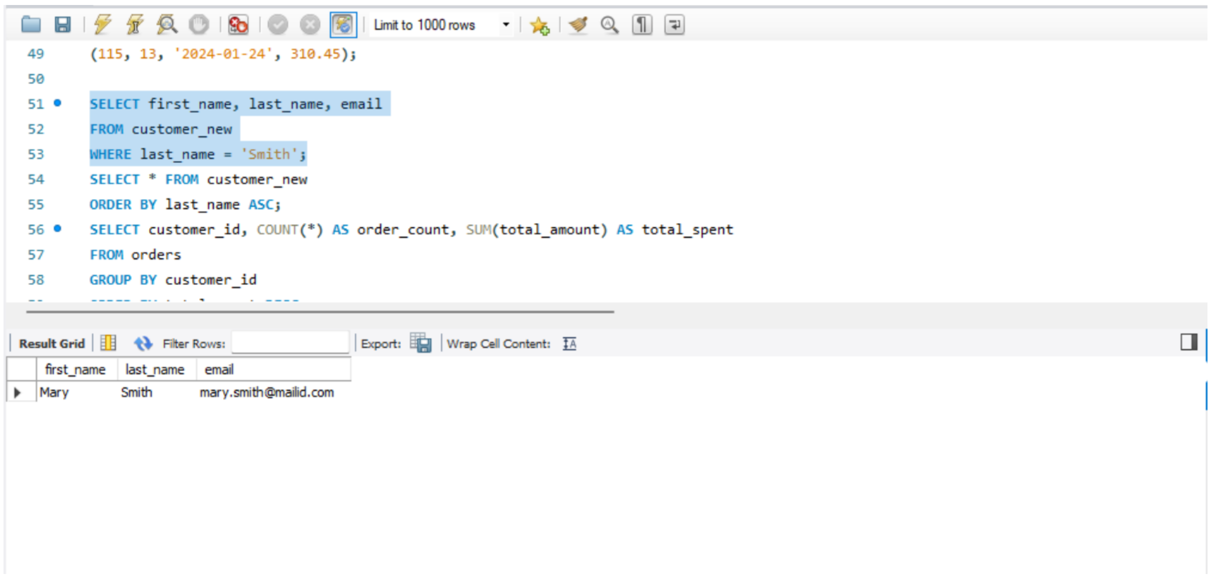
Dataset – customer.csv

Name – Vani Goel

Email ID -vanigoel.110@gmail.com

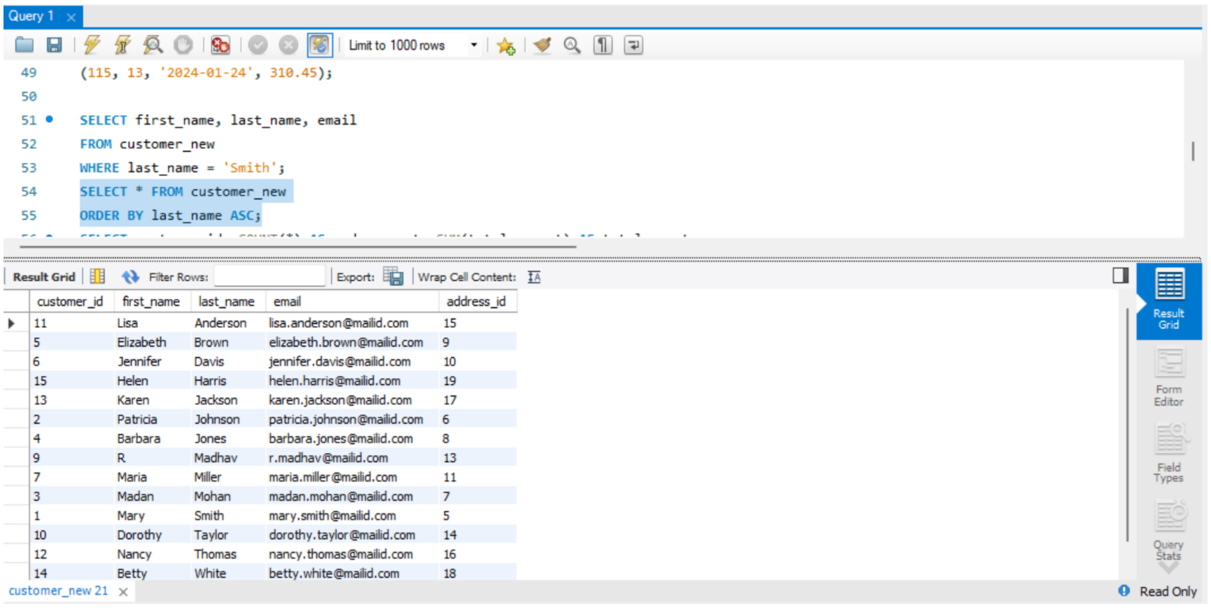
SQL QUERIES :-

1. Use SELECT, WHERE, ORDER BY, GROUP BY



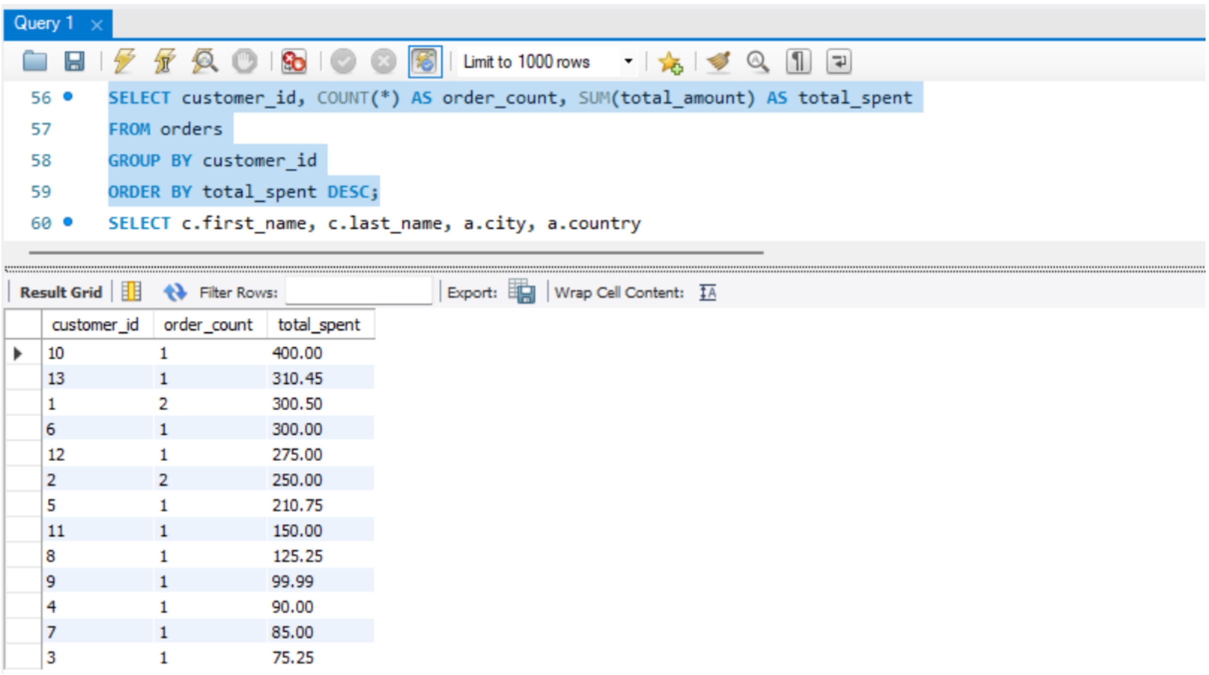
```
(115, 13, '2024-01-24', 310.45);

SELECT first_name, last_name, email
FROM customer_new
WHERE last_name = 'Smith';
SELECT * FROM customer_new
ORDER BY last_name ASC;
SELECT customer_id, COUNT(*) AS order_count, SUM(total_amount) AS total_spent
FROM orders
GROUP BY customer_id
```

first_name	last_name	email
Mary	Smith	mary.smith@mailid.com



```
(115, 13, '2024-01-24', 310.45);

SELECT first_name, last_name, email
FROM customer_new
WHERE last_name = 'Smith';
SELECT * FROM customer_new
ORDER BY last_name ASC;
```

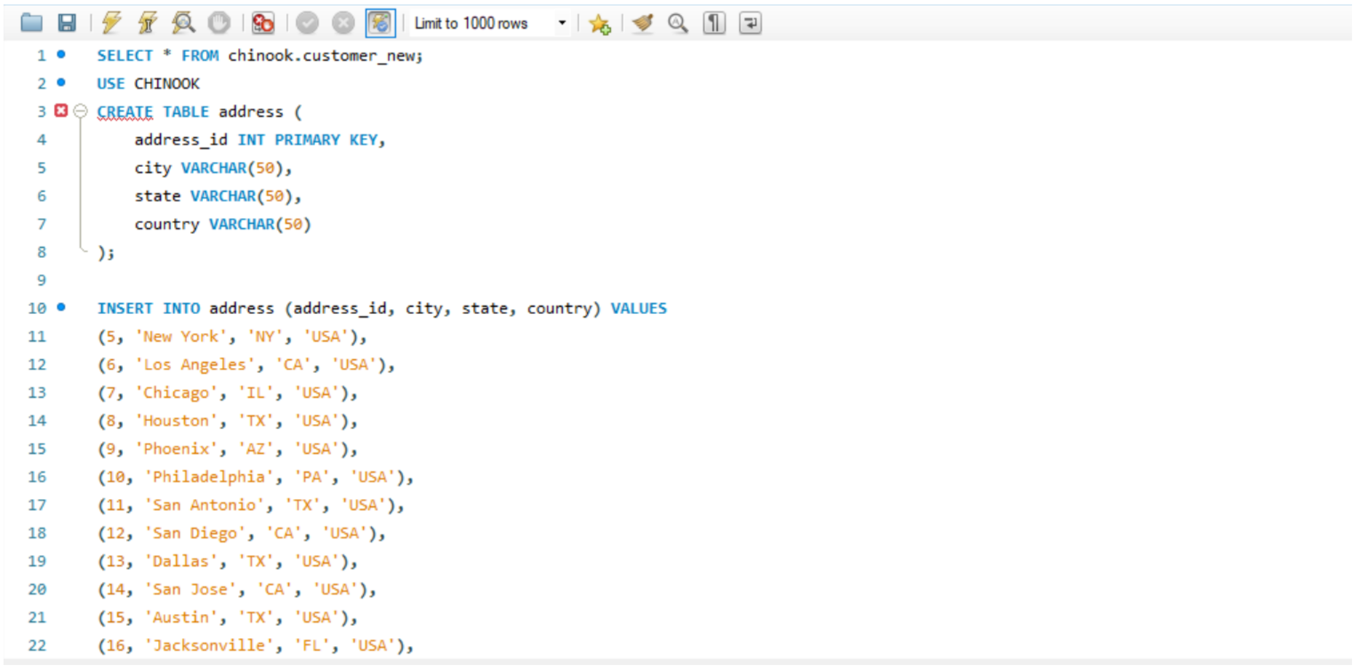
customer_id	first_name	last_name	email	address_id
11	Lisa	Anderson	lisa.anderson@mailid.com	15
5	Elizabeth	Brown	elizabeth.brown@mailid.com	9
6	Jennifer	Davis	jennifer.davis@mailid.com	10
15	Helen	Harris	helen.harris@mailid.com	19
13	Karen	Jackson	karen.jackson@mailid.com	17
2	Patricia	Johnson	patricia.johnson@mailid.com	6
4	Barbara	Jones	barbara.jones@mailid.com	8
9	R	Madhav	r.madhav@mailid.com	13
7	Maria	Miller	maria.miller@mailid.com	11
3	Madan	Mohan	madan.mohan@mailid.com	7
1	Mary	Smith	mary.smith@mailid.com	5
10	Dorothy	Taylor	dorothy.taylor@mailid.com	14
12	Nancy	Thomas	nancy.thomas@mailid.com	16
14	Betty	White	betty.white@mailid.com	18



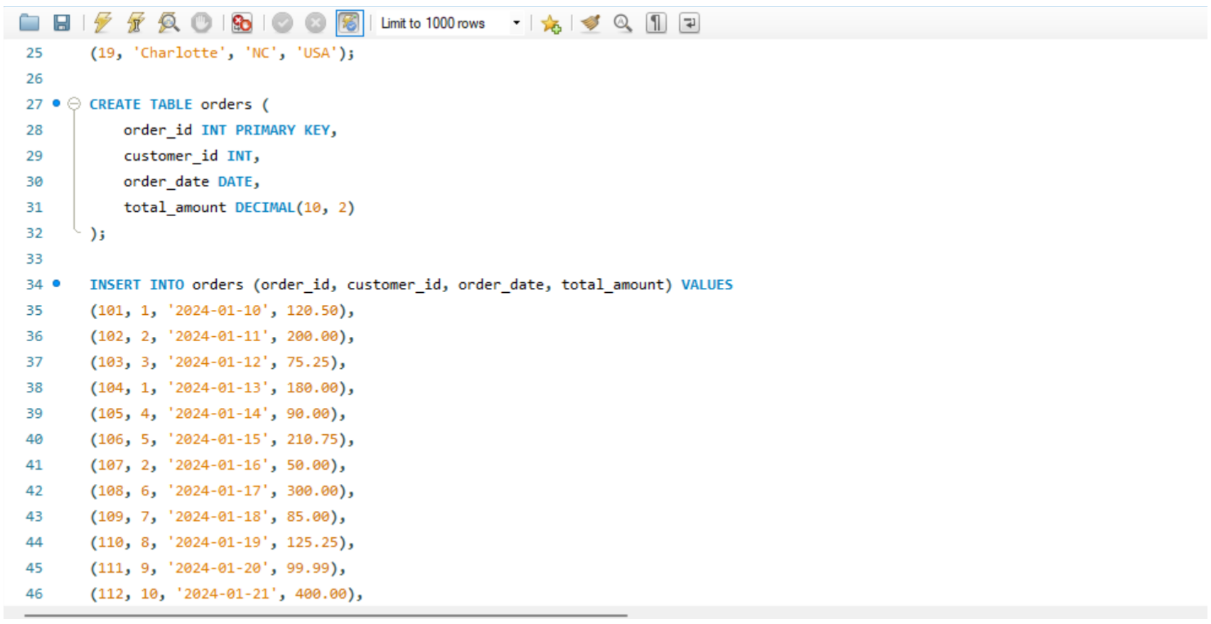
```
SELECT customer_id, COUNT(*) AS order_count, SUM(total_amount) AS total_spent
FROM orders
GROUP BY customer_id
ORDER BY total_spent DESC;
SELECT c.first_name, c.last_name, a.city, a.country
```

customer_id	order_count	total_spent
10	1	400.00
13	1	310.45
1	2	300.50
6	1	300.00
12	1	275.00
2	2	250.00
5	1	210.75
11	1	150.00
8	1	125.25
9	1	99.99
4	1	90.00
7	1	85.00
3	1	75.25

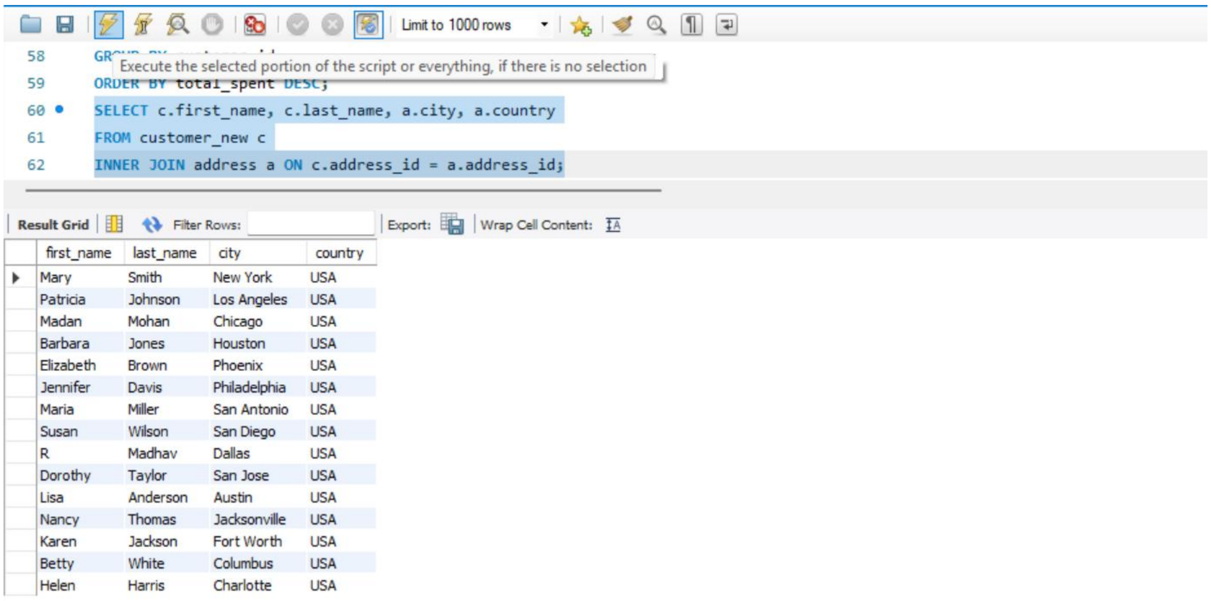
2. Use JOINS (INNER, LEFT, RIGHT)



```
SELECT * FROM chinook.customer_new;
USE CHINOOK
CREATE TABLE address (
    address_id INT PRIMARY KEY,
    city VARCHAR(50),
    state VARCHAR(50),
    country VARCHAR(50)
);

INSERT INTO address (address_id, city, state, country) VALUES
(5, 'New York', 'NY', 'USA'),
(6, 'Los Angeles', 'CA', 'USA'),
(7, 'Chicago', 'IL', 'USA'),
(8, 'Houston', 'TX', 'USA'),
(9, 'Phoenix', 'AZ', 'USA'),
(10, 'Philadelphia', 'PA', 'USA'),
(11, 'San Antonio', 'TX', 'USA'),
(12, 'San Diego', 'CA', 'USA'),
(13, 'Dallas', 'TX', 'USA'),
(14, 'San Jose', 'CA', 'USA'),
(15, 'Austin', 'TX', 'USA'),
(16, 'Jacksonville', 'FL', 'USA'),
```



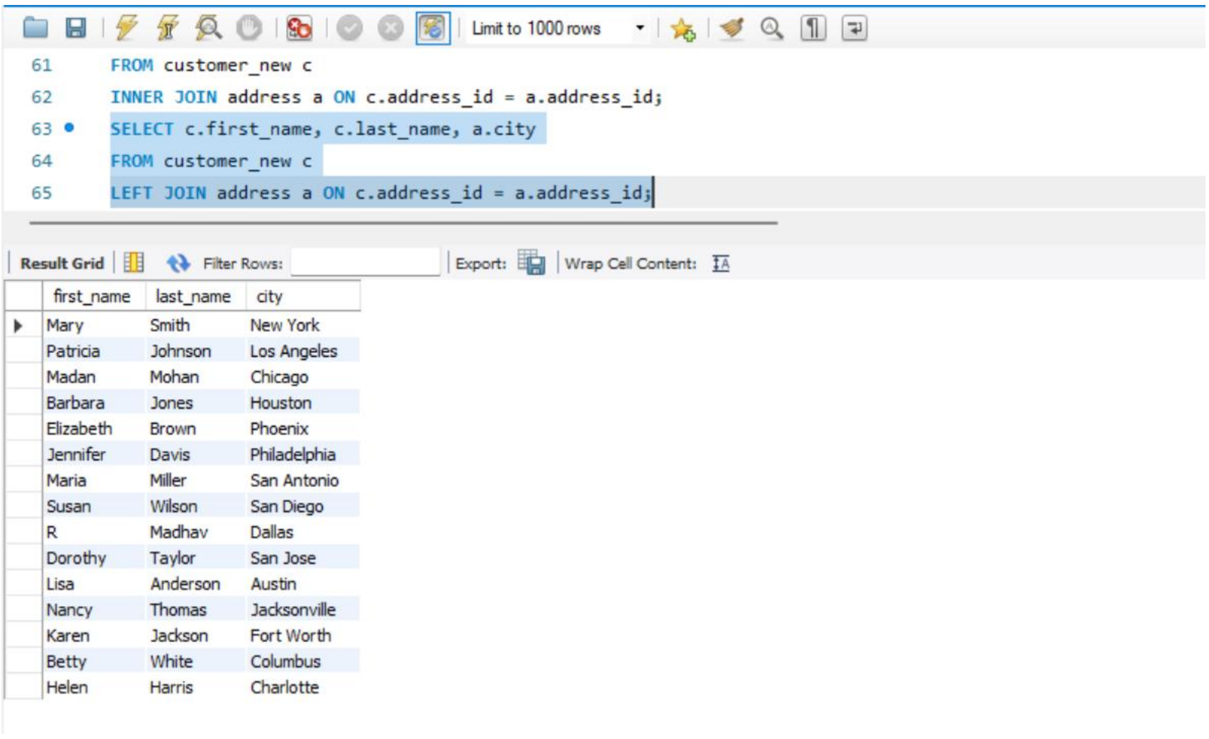
```
(19, 'Charlotte', 'NC', 'USA');

CREATE TABLE orders (
    order_id INT PRIMARY KEY,
    customer_id INT,
    order_date DATE,
    total_amount DECIMAL(10, 2)
);

INSERT INTO orders (order_id, customer_id, order_date, total_amount) VALUES
(101, 1, '2024-01-10', 120.50),
(102, 2, '2024-01-11', 200.00),
(103, 3, '2024-01-12', 75.25),
(104, 1, '2024-01-13', 180.00),
(105, 4, '2024-01-14', 90.00),
(106, 5, '2024-01-15', 210.75),
(107, 2, '2024-01-16', 50.00),
(108, 6, '2024-01-17', 300.00),
(109, 7, '2024-01-18', 85.00),
(110, 8, '2024-01-19', 125.25),
(111, 9, '2024-01-20', 99.99),
(112, 10, '2024-01-21', 400.00),
```



Execute the selected portion of the script or everything, if there is no selection

```
GR
ORDER BY total_spent DESC;
SELECT c.first_name, c.last_name, a.city, a.country
FROM customer_new c
INNER JOIN address a ON c.address_id = a.address_id;
```

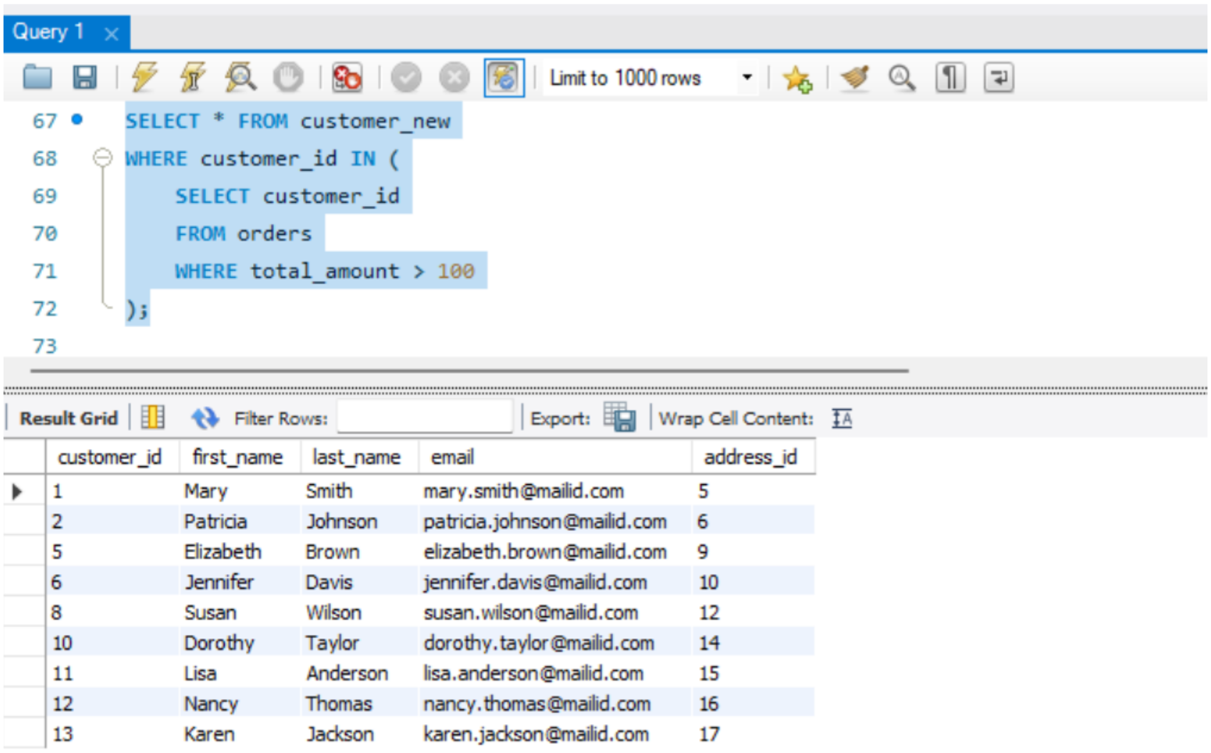
first_name	last_name	city	country
Mary	Smith	New York	USA
Patricia	Johnson	Los Angeles	USA
Madan	Mohan	Chicago	USA
Barbara	Jones	Houston	USA
Elizabeth	Brown	Phoenix	USA
Jennifer	Davis	Philadelphia	USA
Maria	Miller	San Antonio	USA
Susan	Wilson	San Diego	USA
R	Madhav	Dallas	USA
Dorothy	Taylor	San Jose	USA
Lisa	Anderson	Austin	USA
Nancy	Thomas	Jacksonville	USA
Karen	Jackson	Fort Worth	USA
Betty	White	Columbus	USA
Helen	Harris	Charlotte	USA



```
FROM customer_new c
INNER JOIN address a ON c.address_id = a.address_id;
SELECT c.first_name, c.last_name, a.city
FROM customer_new c
LEFT JOIN address a ON c.address_id = a.address_id;
```

first_name	last_name	city
Mary	Smith	New York
Patricia	Johnson	Los Angeles
Madan	Mohan	Chicago
Barbara	Jones	Houston
Elizabeth	Brown	Phoenix
Jennifer	Davis	Philadelphia
Maria	Miller	San Antonio
Susan	Wilson	San Diego
R	Madhav	Dallas
Dorothy	Taylor	San Jose
Lisa	Anderson	Austin
Nancy	Thomas	Jacksonville
Karen	Jackson	Fort Worth
Betty	White	Columbus
Helen	Harris	Charlotte

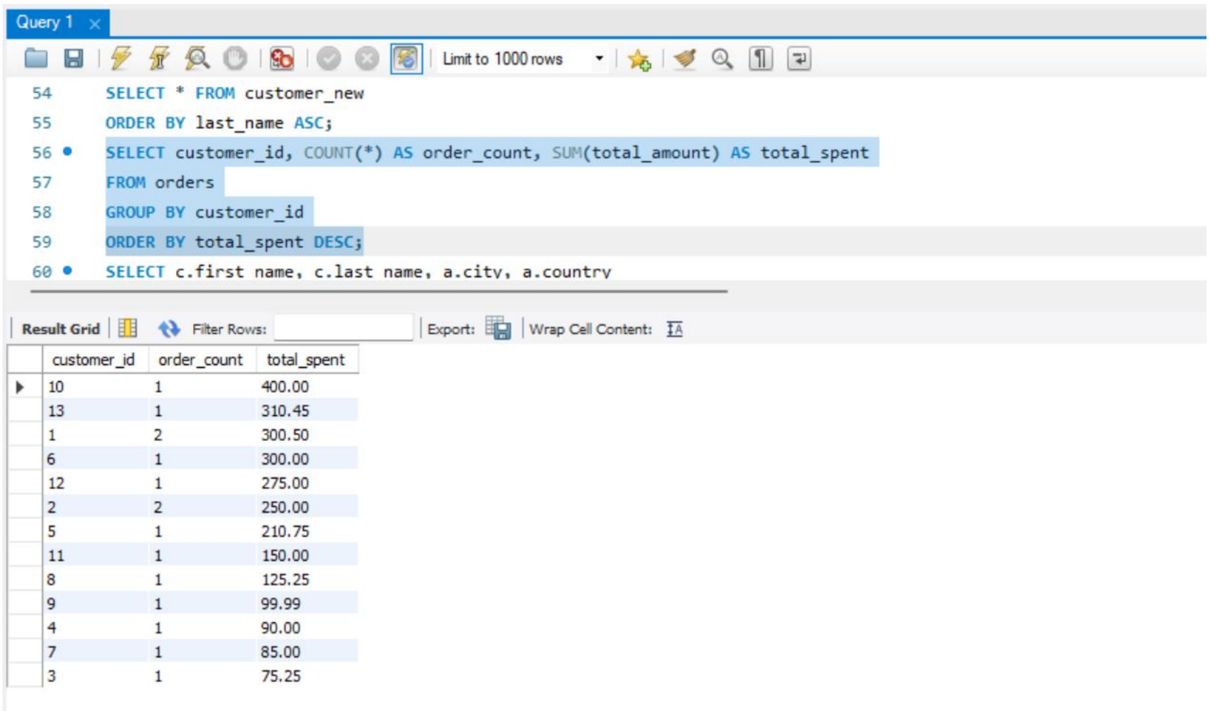
3. Write subqueries



```
SELECT * FROM customer_new
WHERE customer_id IN (
    SELECT customer_id
    FROM orders
    WHERE total_amount > 100
);
```

customer_id	first_name	last_name	email	address_id
1	Mary	Smith	mary.smith@mailid.com	5
2	Patricia	Johnson	patricia.johnson@mailid.com	6
5	Elizabeth	Brown	elizabeth.brown@mailid.com	9
6	Jennifer	Davis	jennifer.davis@mailid.com	10
8	Susan	Wilson	susan.wilson@mailid.com	12
10	Dorothy	Taylor	dorothy.taylor@mailid.com	14
11	Lisa	Anderson	lisa.anderson@mailid.com	15
12	Nancy	Thomas	nancy.thomas@mailid.com	16
13	Karen	Jackson	karen.jackson@mailid.com	17

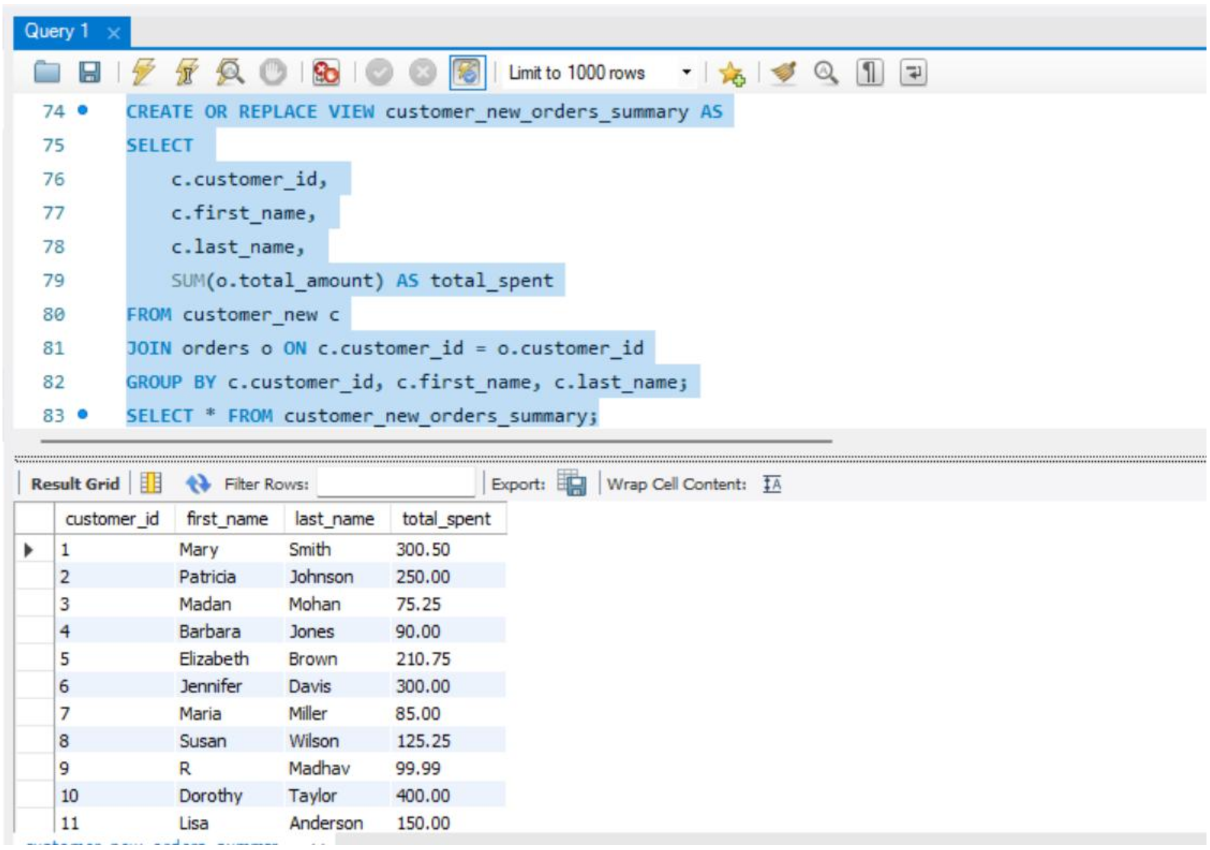
4. Use aggregate functions (SUM, AVG)



```sql
SELECT * FROM customer_new
ORDER BY last_name ASC;
SELECT customer_id, COUNT(*) AS order_count, SUM(total_amount) AS total_spent
FROM orders
GROUP BY customer_id
ORDER BY total_spent DESC;
SELECT c.first name, c.last name, a.city, a.country
```

customer_id	order_count	total_spent
10	1	400.00
13	1	310.45
1	2	300.50
6	1	300.00
12	1	275.00
2	2	250.00
5	1	210.75
11	1	150.00
8	1	125.25
9	1	99.99
4	1	90.00
7	1	85.00
3	1	75.25

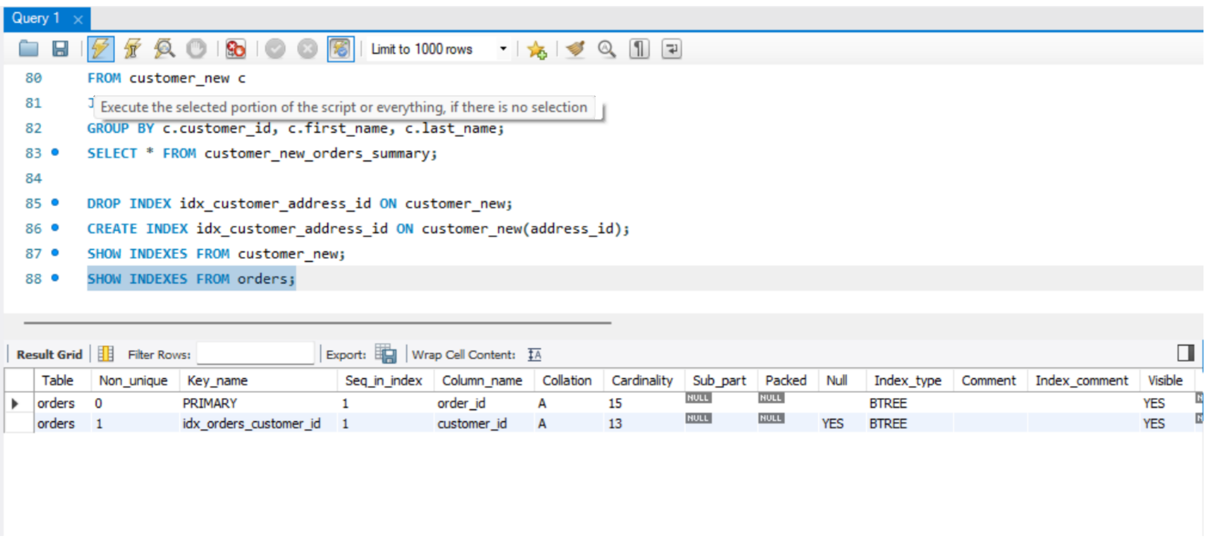
5. Create views for analysis



```sql
CREATE OR REPLACE VIEW customer_new_orders_summary AS
SELECT
    c.customer_id,
    c.first_name,
    c.last_name,
    SUM(o.total_amount) AS total_spent
FROM customer_new c
JOIN orders o ON c.customer_id = o.customer_id
GROUP BY c.customer_id, c.first_name, c.last_name;
SELECT * FROM customer_new_orders_summary;
```

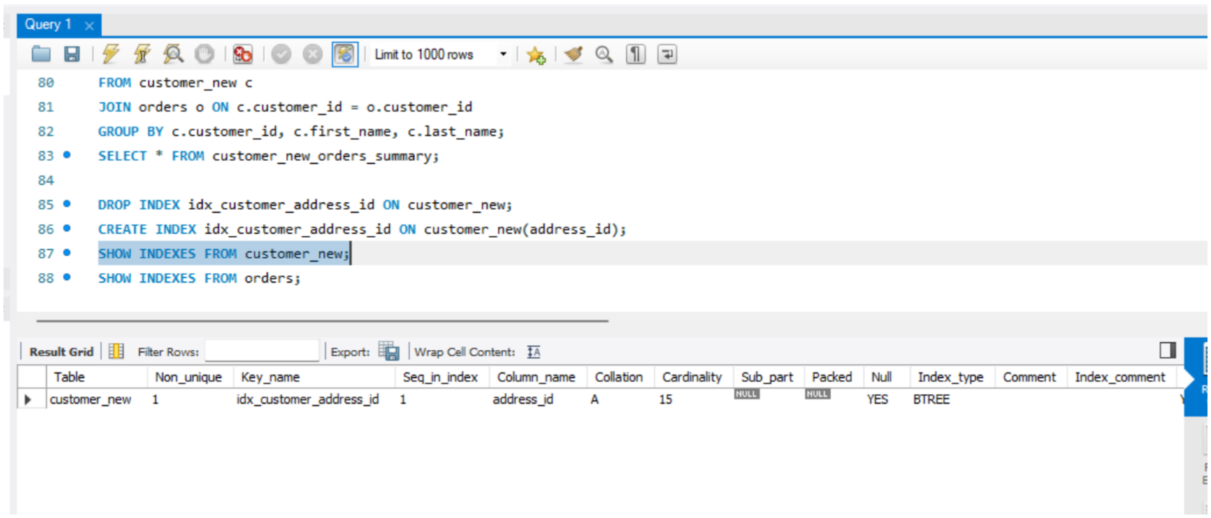
customer_id	first_name	last_name	total_spent
1	Mary	Smith	300.50
2	Patricia	Johnson	250.00
3	Madan	Mohan	75.25
4	Barbara	Jones	90.00
5	Elizabeth	Brown	210.75
6	Jennifer	Davis	300.00
7	Maria	Miller	85.00
8	Susan	Wilson	125.25
9	R	Madhav	99.99
10	Dorothy	Taylor	400.00
11	Lisa	Anderson	150.00

6. Optimize queries with indexes



```sql
FROM customer_new c
J Execute the selected portion of the script or everything, if there is no selection
GROUP BY c.customer_id, c.first_name, c.last_name;
SELECT * FROM customer_new_orders_summary;

DROP INDEX idx_customer_address_id ON customer_new;
CREATE INDEX idx_customer_address_id ON customer_new(address_id);
SHOW INDEXES FROM customer_new;
SHOW INDEXES FROM orders;
```

Table	Non_unique	Key_name	Seq_in_index	Column_name	Collation	Cardinality	Sub_part	Packed	Null	Index_type	Comment	Index_comment	Visible
orders	0	PRIMARY	1	order_id	A	15	NULL	NULL		BTREE			YES
orders	1	idx_orders_customer_id	1	customer_id	A	13	NULL	NULL	YES	BTREE			YES



```sql
FROM customer_new c
JOIN orders o ON c.customer_id = o.customer_id
GROUP BY c.customer_id, c.first_name, c.last_name;
SELECT * FROM customer_new_orders_summary;

DROP INDEX idx_customer_address_id ON customer_new;
CREATE INDEX idx_customer_address_id ON customer_new(address_id);
SHOW INDEXES FROM customer_new;
SHOW INDEXES FROM orders;
```

Table	Non_unique	Key_name	Seq_in_index	Column_name	Collation	Cardinality	Sub_part	Packed	Null	Index_type	Comment	Index_comment
customer_new	1	idx_customer_address_id	1	address_id	A	15	NULL	NULL	YES	BTREE		