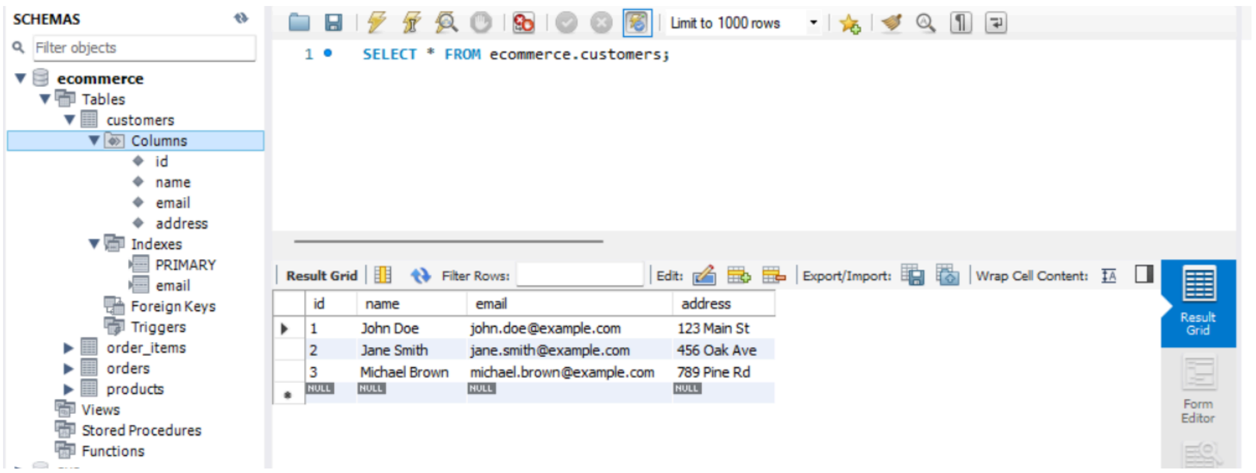
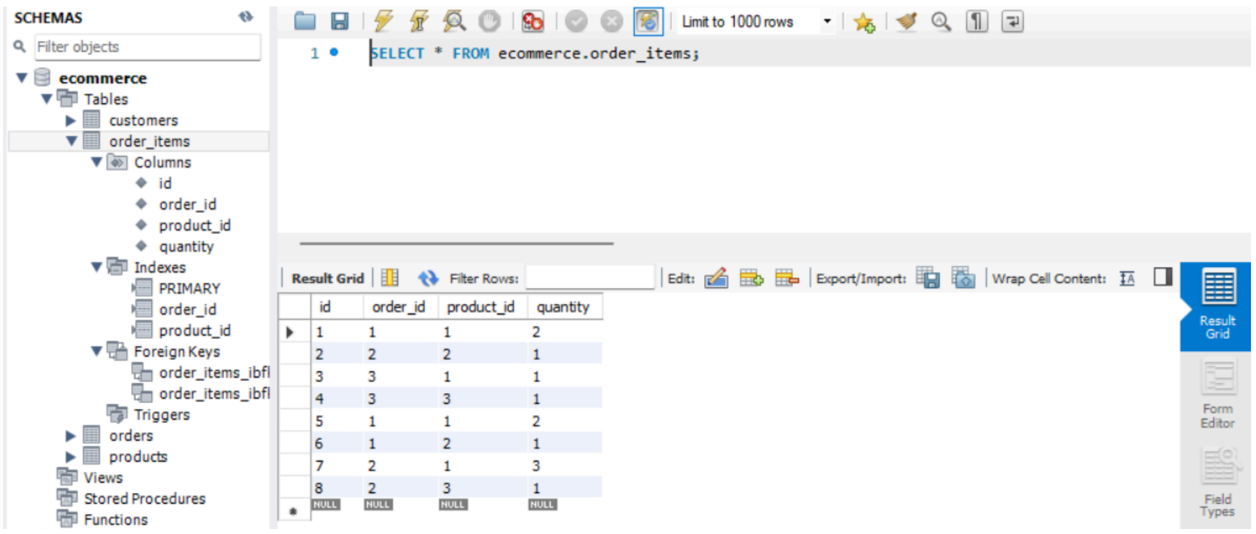
Task:

Create a database and tables to manage a simple e-commerce system.
The system should have three tables: customers, orders, and products.



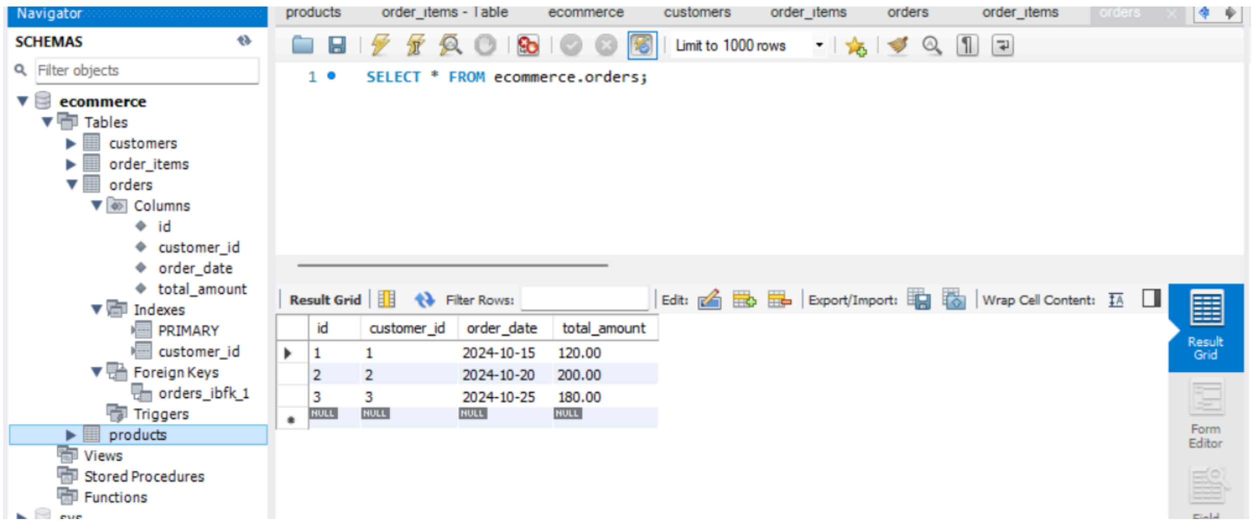
```
SELECT * FROM ecommerce.customers;
```

id	name	email	address
1	John Doe	john.doe@example.com	123 Main St
2	Jane Smith	jane.smith@example.com	456 Oak Ave
3	Michael Brown	michael.brown@example.com	789 Pine Rd
NULL	NULL	NULL	NULL



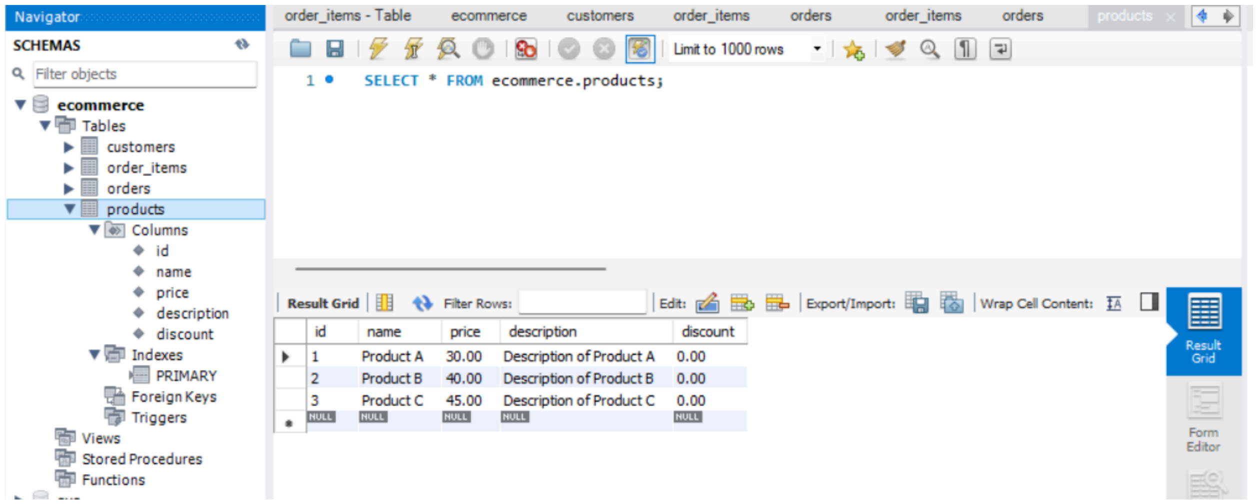
```
SELECT * FROM ecommerce.order_items;
```

id	order_id	product_id	quantity
1	1	1	2
2	2	2	1
3	3	1	1
4	3	3	1
5	1	1	2
6	1	2	1
7	2	1	3
8	2	3	1
NULL	NULL	NULL	NULL



```
SELECT * FROM ecommerce.orders;
```

id	customer_id	order_date	total_amount
1	1	2024-10-15	120.00
2	2	2024-10-20	200.00
3	3	2024-10-25	180.00
NULL	NULL	NULL	NULL



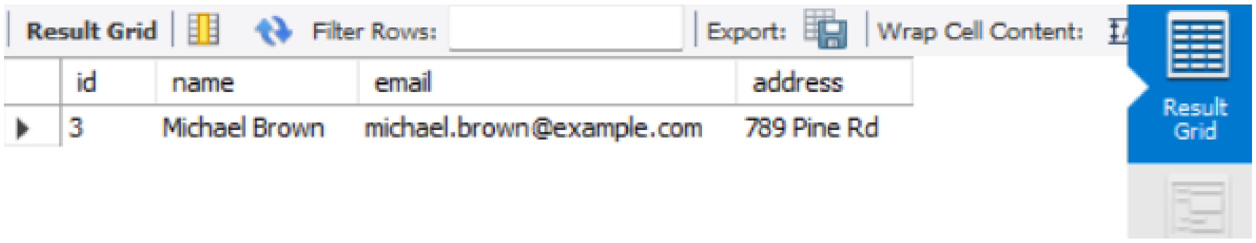
```
SELECT * FROM ecommerce.products;
```

id	name	price	description	discount
1	Product A	30.00	Description of Product A	0.00
2	Product B	40.00	Description of Product B	0.00
3	Product C	45.00	Description of Product C	0.00
NULL	NULL	NULL	NULL	NULL

Query:-

1.Retrieve all customers who have placed an order in the last 30 days.

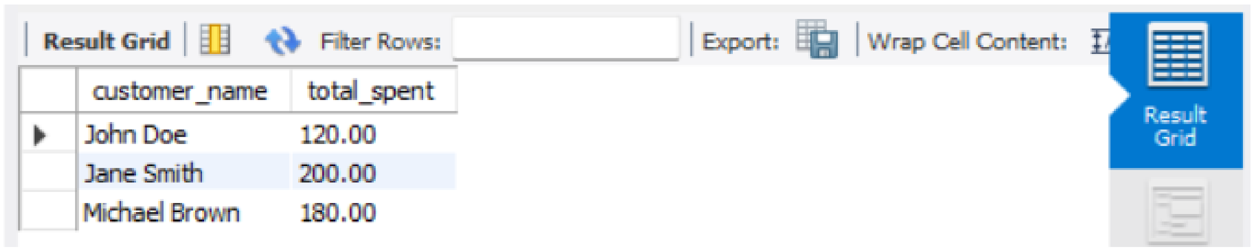
SELECT DISTINCT customers.*
FROM customers
JOIN orders ON customers.id = orders.customer_id
WHERE orders.order_date >= CURDATE() - INTERVAL 30 DAY;



id	name	email	address
3	Michael Brown	michael.brown@example.com	789 Pine Rd

2.Get the total amount of all orders placed by each customer

```
SELECT customers.name AS customer_name,
    SUM(orders.total_amount) AS total_spent
FROM customers
JOIN orders ON customers.id = orders.customer_id
GROUP BY customers.id;
```

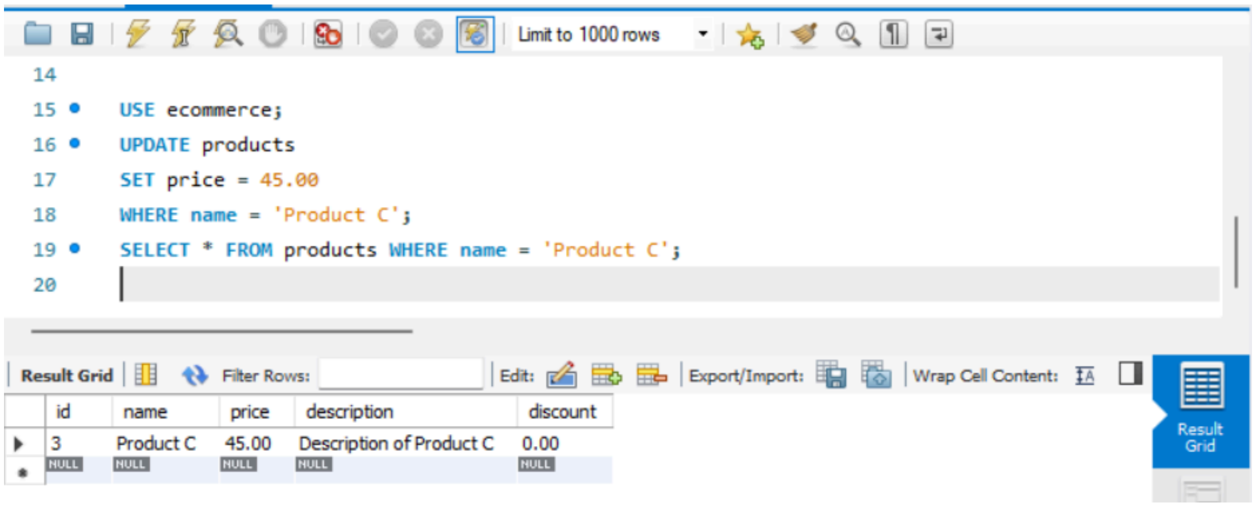


customer_name	total_spent
John Doe	120.00
Jane Smith	200.00
Michael Brown	180.00

3.Update the price of Product C to 45.00.

```
UPDATE products
SET price = 45.00
WHERE name = 'Product C';

SELECT * FROM products WHERE name = 'Product C';
```



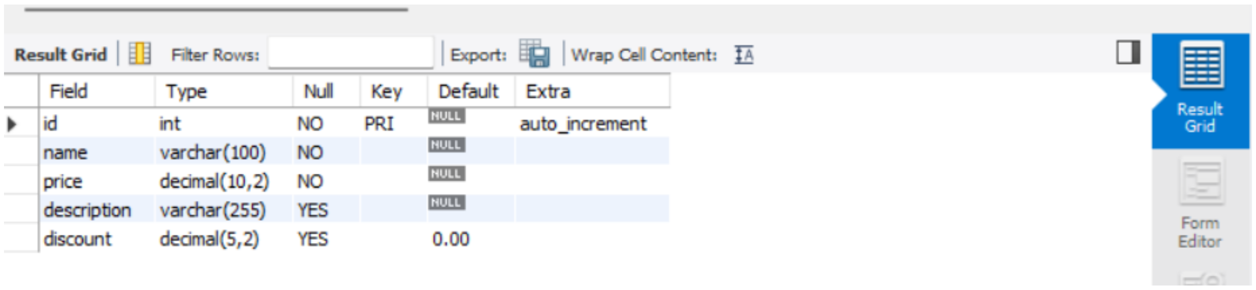
```
USE ecommerce;
UPDATE products
SET price = 45.00
WHERE name = 'Product C';
SELECT * FROM products WHERE name = 'Product C';
```

id	name	price	description	discount
3	Product C	45.00	Description of Product C	0.00
NULL	NULL	NULL	NULL	NULL

4.Add a new column discount to the products table.

ALTER TABLE products
ADD COLUMN discount DECIMAL(5, 2) DEFAULT 0.00;

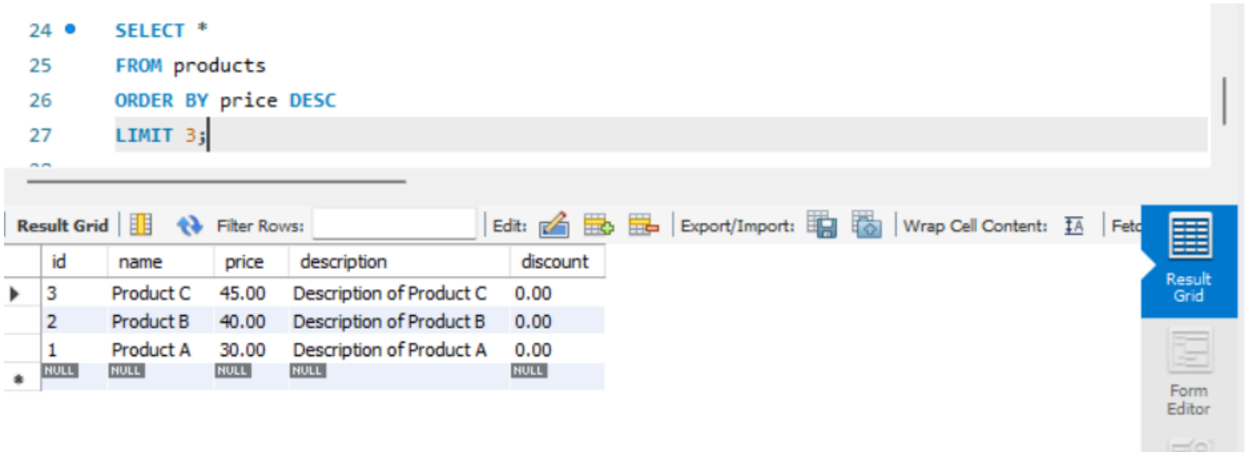
DESCRIBE products;



Field	Type	Null	Key	Default	Extra
id	int	NO	PRI	NULL	auto_increment
name	varchar(100)	NO		NULL	
price	decimal(10,2)	NO		NULL	
description	varchar(255)	YES		NULL	
discount	decimal(5,2)	YES		0.00	

5.Retrieve the top 3 products with the highest price.

SELECT *
FROM products
ORDER BY price DESC
LIMIT 3;

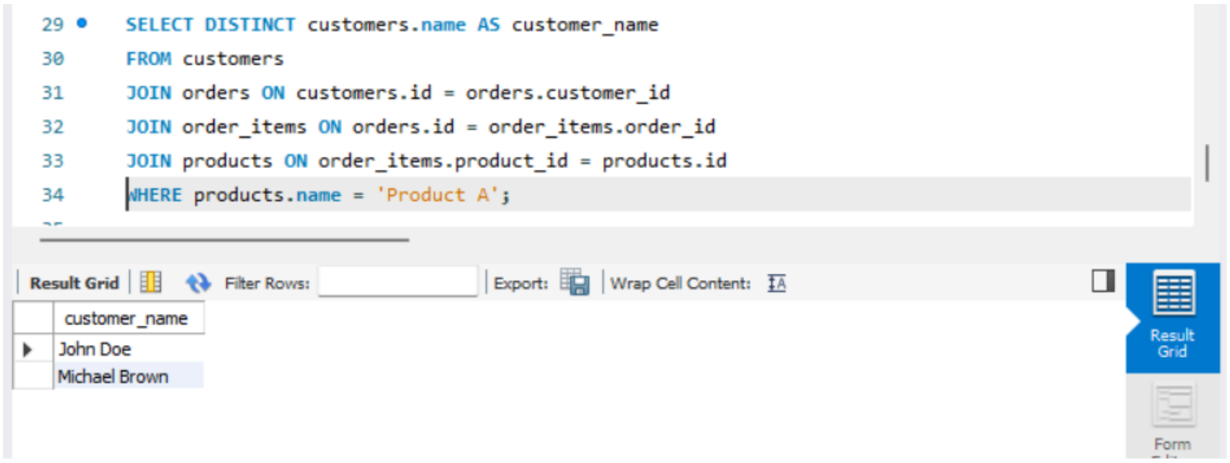


```
SELECT *
FROM products
ORDER BY price DESC
LIMIT 3;
```

id	name	price	description	discount
3	Product C	45.00	Description of Product C	0.00
2	Product B	40.00	Description of Product B	0.00
1	Product A	30.00	Description of Product A	0.00
NULL	NULL	NULL	NULL	NULL

6.Get the names of customers who have ordered Product A.

SELECT DISTINCT customers.name AS customer_name
FROM customers
JOIN orders ON customers.id = orders.customer_id
JOIN order_items ON orders.id = order_items.order_id
JOIN products ON order_items.product_id = products.id
WHERE products.name = 'Product A';



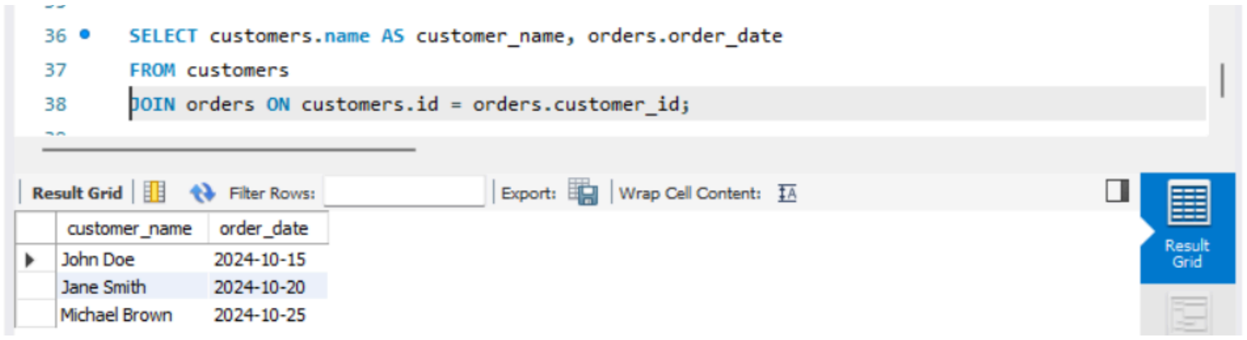
```
SELECT DISTINCT customers.name AS customer_name
FROM customers
JOIN orders ON customers.id = orders.customer_id
JOIN order_items ON orders.id = order_items.order_id
JOIN products ON order_items.product_id = products.id
WHERE products.name = 'Product A';
```

customer_name
John Doe
Michael Brown

7.Join the orders and customers tables to retrieve the customer's name and order date for each order.

SELECT customers.name AS customer_name, orders.order_date

FROM customers
JOIN orders ON customers.id = orders.customer_id;

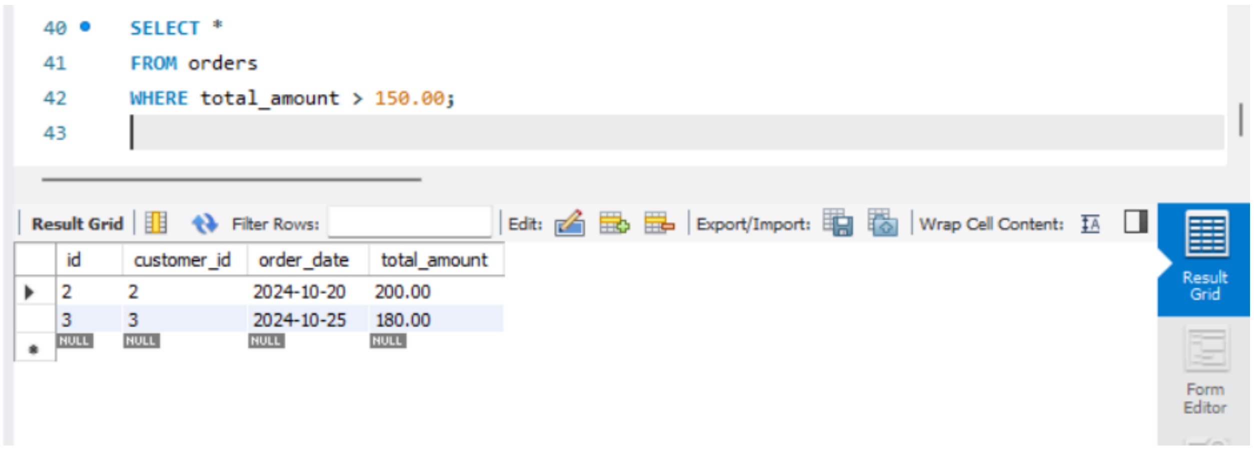


```
SELECT customers.name AS customer_name, orders.order_date
FROM customers
JOIN orders ON customers.id = orders.customer_id;
```

customer_name	order_date
John Doe	2024-10-15
Jane Smith	2024-10-20
Michael Brown	2024-10-25

8.Retrieve the orders with a total amount greater than 150.00.

SELECT *
FROM orders
WHERE total_amount > 150.00;



```
SELECT *
FROM orders
WHERE total_amount > 150.00;
```

id	customer_id	order_date	total_amount
2	2	2024-10-20	200.00
3	3	2024-10-25	180.00
NULL	NULL	NULL	NULL

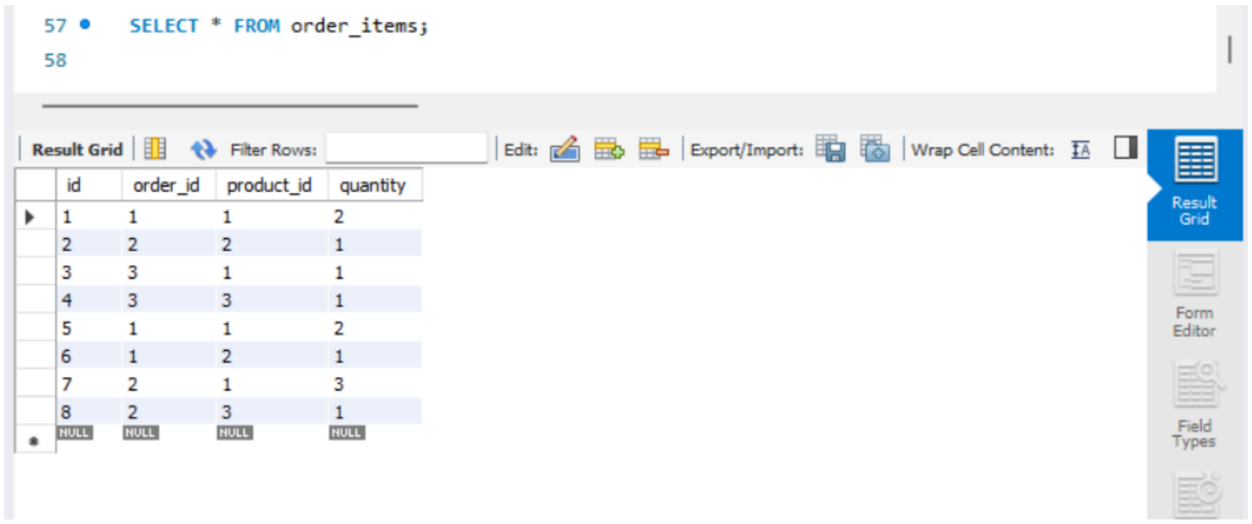
9.Normalize the database by creating a separate table for order items and updating the orders table to reference the order_items table.

CREATE TABLE order_items (
    id INT AUTO_INCREMENT PRIMARY KEY,
    order_id INT NOT NULL,
    product_id INT NOT NULL,
    quantity INT NOT NULL,
    FOREIGN KEY (order_id) REFERENCES orders(id),
    FOREIGN KEY (product_id) REFERENCES products(id)

);

INSERT INTO order_items (order_id, product_id, quantity)
VALUES
(1, 1, 2), -- Order 1, Product A, Quantity 2
(1, 2, 1), -- Order 1, Product B, Quantity 1
(2, 1, 3), -- Order 2, Product A, Quantity 3
(2, 3, 1); -- Order 2, Product C, Quantity 1

SELECT * FROM order_items;

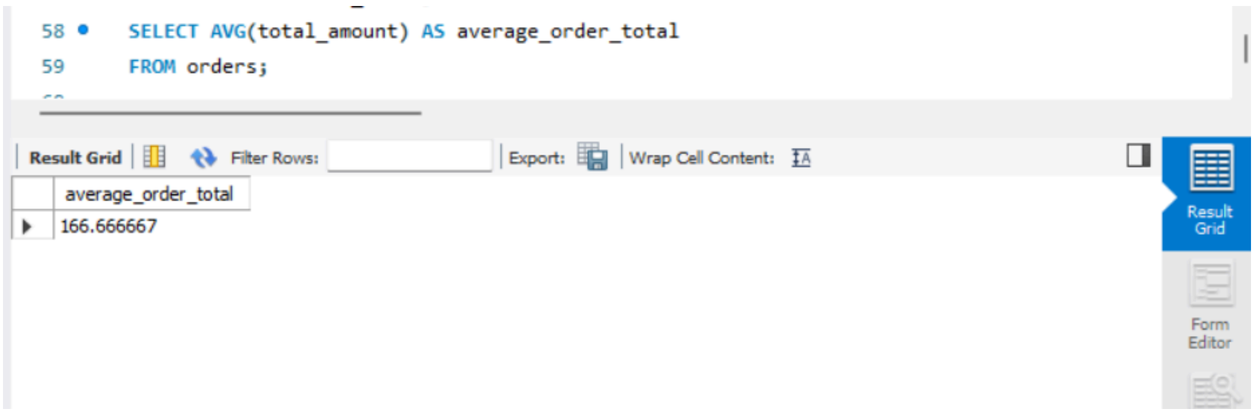


```
SELECT * FROM order_items;
```

id	order_id	product_id	quantity
1	1	1	2
2	2	2	1
3	3	1	1
4	3	3	1
5	1	1	2
6	1	2	1
7	2	1	3
8	2	3	1
NULL	NULL	NULL	NULL

10.Retrieve the average total of all orders.

SELECT AVG(total_amount) AS average_order_total
FROM orders;



```
SELECT AVG(total_amount) AS average_order_total
FROM orders;
```

average_order_total
166.666667