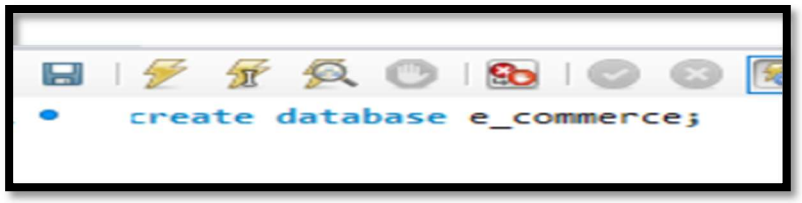
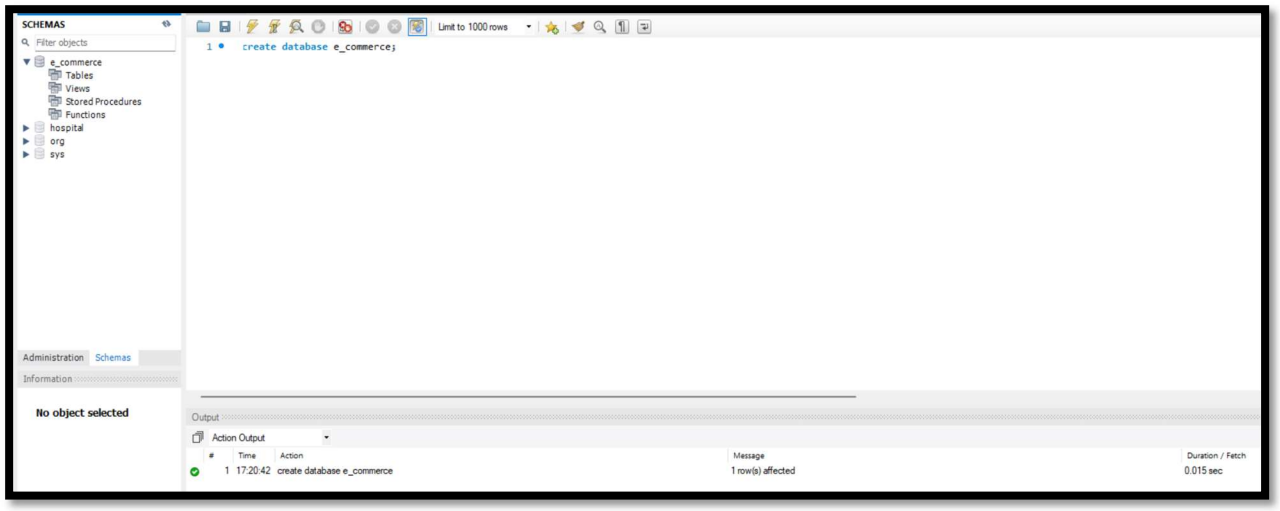
Sara Fatma
NucleusTeq

SARA FATMA **SQL-ASSIGNMENT**

1. Create Database e_commerce





2. Create following Tables:

Customers:

a. customer_id - int auto-increment primary key

b. name - varchar(50)

c. email - varchar(50)

d. mobile - varchar(15)

Products:

a. id - int

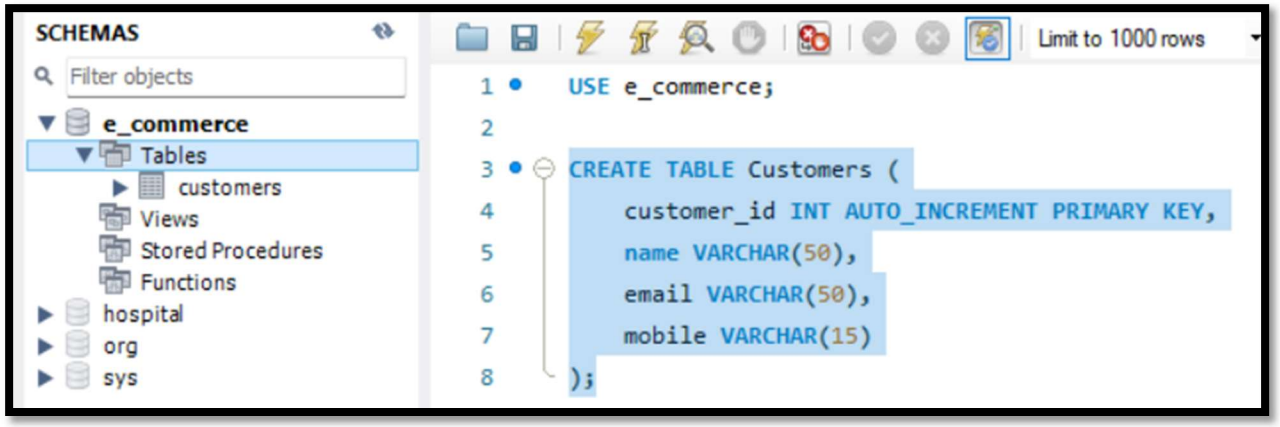
b. name - varchar(50) not null

c. description - varchar(200)

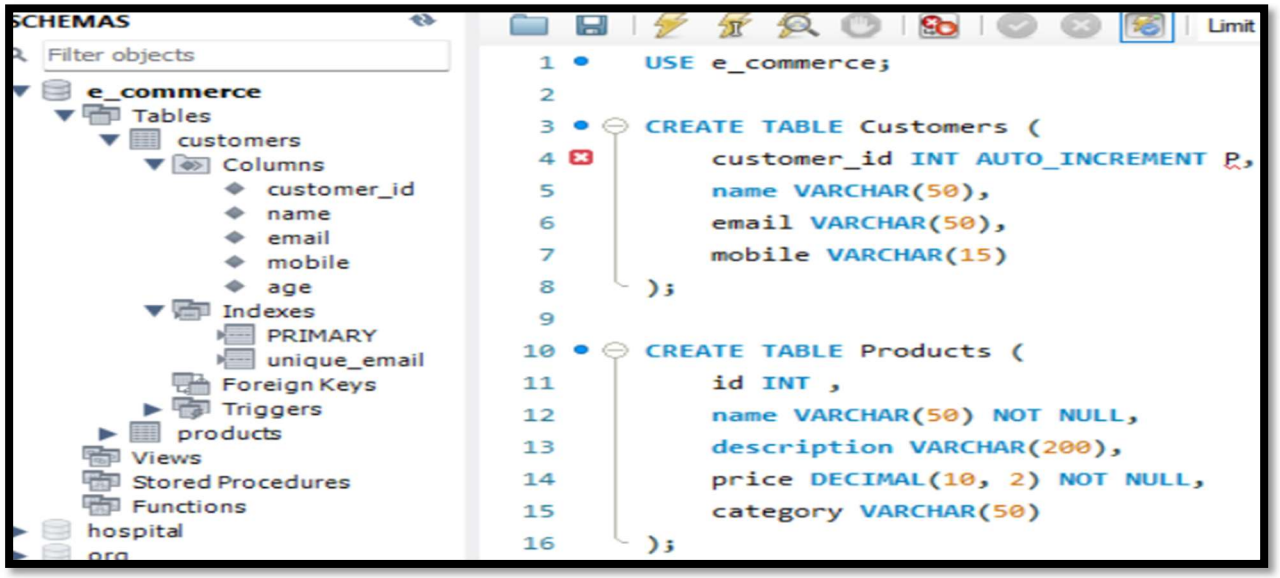
d. price - decimal(10, 2) not null

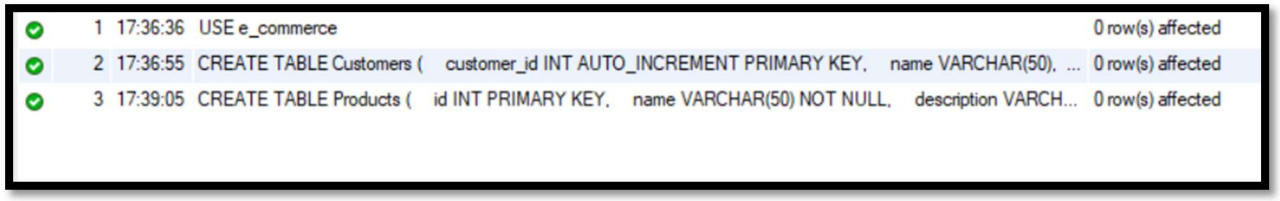
e. category - varchar(50)

Customers Table



Products Table



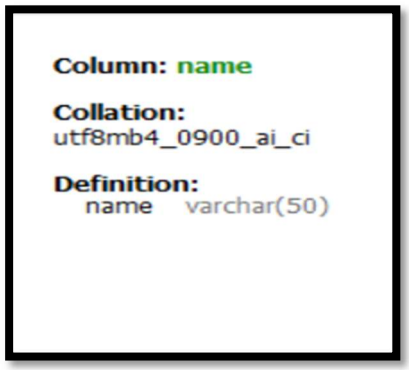


1	17:36:36	USE e_commerce		0 row(s) affected
2	17:36:55	CREATE TABLE Customers (customer_id INT AUTO_INCREMENT PRIMARY KEY, name VARCHAR(50), ...		0 row(s) affected
3	17:39:05	CREATE TABLE Products (id INT PRIMARY KEY, name VARCHAR(50) NOT NULL, description VARCH...		0 row(s) affected

3. Modify Tables(using Alter keyword):

a. Add not null on name and email in the Customers table

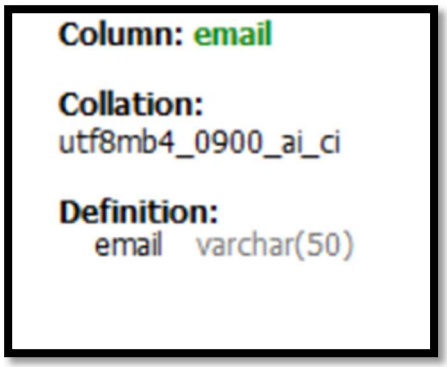
```
17
18 • ALTER TABLE Customers MODIFY COLUMN name VARCHAR(50) NOT NULL, MODIFY COLUMN email VARCHAR(50) NOT NULL;
19
```



Column: name

Collation:
utf8mb4_0900_ai_ci

Definition:
name varchar(50)



Column: email

Collation:
utf8mb4_0900_ai_ci

Definition:
email varchar(50)

Output

Action Output

#	Time	Action
1	17:36:36	USE e_commerce
2	17:36:55	CREATE TABLE Customers (customer_id INT AUTO_INCREMENT PRIMARY KEY, name VARCHAR(50), ...
3	17:39:05	CREATE TABLE Products (id INT PRIMARY KEY, name VARCHAR(50) NOT NULL, description VARCH...
4	17:42:58	ALTER TABLE Customers MODIFY COLUMN name VARCHAR(50) NOT NULL, MODIFY COLUMN email VARC...

b. Add unique key on email in the Customers table

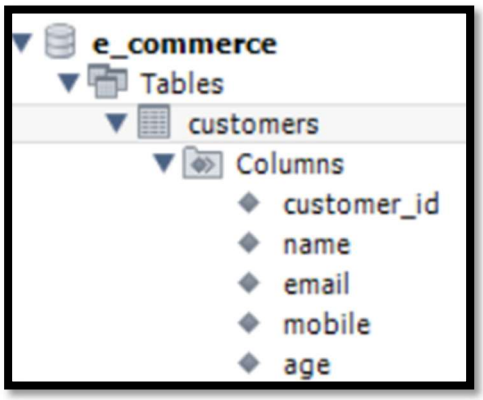
```
19
20 •  ALTER TABLE Customers ADD CONSTRAINT unique_email UNIQUE (email);
21
```

```
4  17:42:58  ALTER TABLE Customers MODIFY COLUMN name VARCHAR(50) NOT NULL, MODIFY COLUM
5  17:46:36  ALTER TABLE Customers ADD CONSTRAINT unique_email UNIQUE (email)
```

c. Add column age in the Customers table

```
22
23      ALTER TABLE Customers ADD COLUMN age INT;
24
```

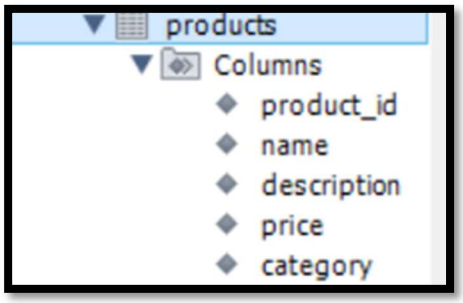
```
5  17:46:36  ALTER TABLE Customers ADD CONSTRAINT unique_email UNIQUE (email)
6  17:48:53  ALTER TABLE Customers ADD COLUMN age INT
```



d. Change column name from id to product_id in the Products table;

```
24
25 •    ALTER TABLE Products CHANGE COLUMN id product_id INT;
26
```

```
6  17:48:53  ALTER TABLE Customers ADD COLUMN age INT
7  17:51:02  ALTER TABLE Products CHANGE COLUMN id product_id INT
```



e. Add primary key and auto increment on product_id in the Products table.

```
41
42 •    ALTER TABLE Products MODIFY COLUMN product_id INT AUTO_INCREMENT PRIMARY KEY;
43
```

```
14  18:04:13  ALTER TABLE Products MODIFY COLUMN product_id INT AUTO_INCREMENT PRIMARY KEY
```

f. Change datatype of description from varchar to text in the Products table

```
29 •    ALTER TABLE Products MODIFY COLUMN description TEXT;
```

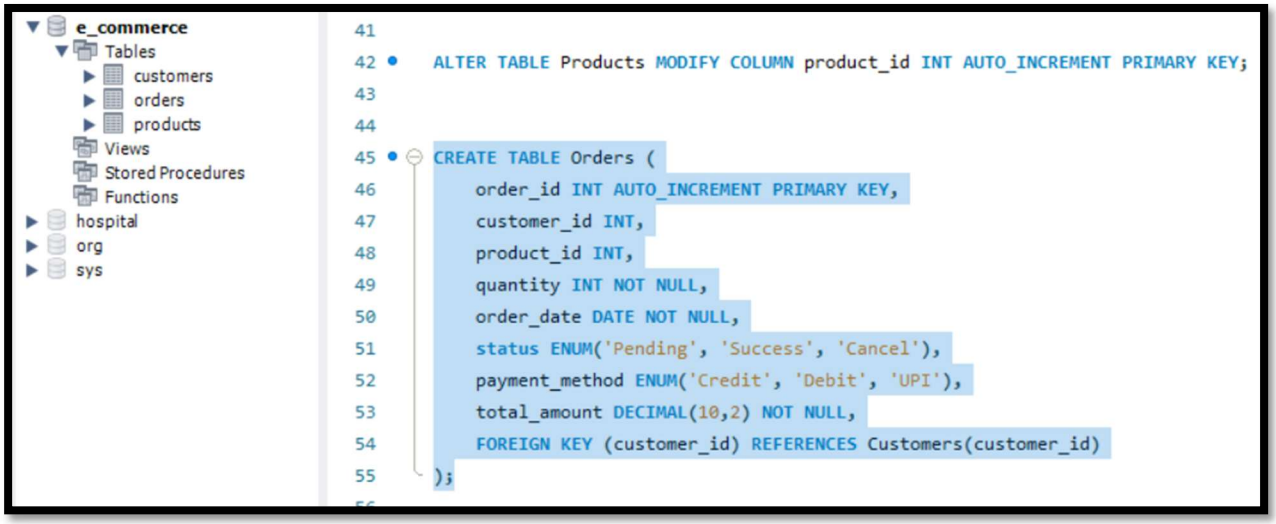
```
9  17:55:45  ALTER TABLE Products MODIFY COLUMN description TEXT
```

4. Create table Order:

a. order_id - int auto-increment primary key

b. **customer_id - int -foreign key**
c. **product_id - int**
d. **quantity - int not null,**
e. **order_date - date not null,**
f. **status - enum(Pending, Success, Cancel),**
g. **payment_method - enum(Credit, Debit, UPI),**
h. **total_amount - decimal(10, 2) not null**

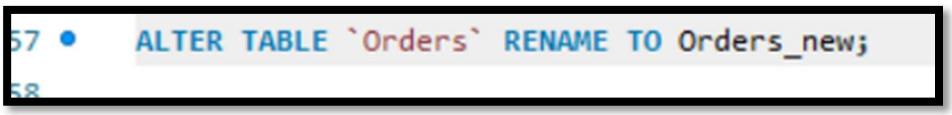
//Since ORDER is a reserved keyword, using it as a table name causes a syntax error therefore used Orders

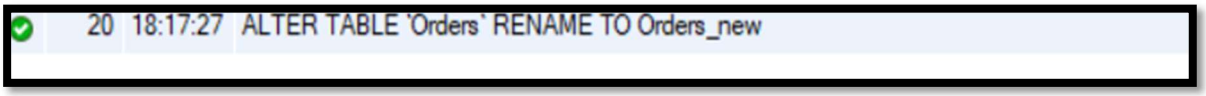


5. Modify Orders Table(using Alter keyword):

a. Change table name Order -> Orders

// Due to the clash of Order table name and Order By keyword , Will alter it to Orders_new





b. Set default value pending in status.

```
59 •  ALTER TABLE Orders_new MODIFY COLUMN status ENUM('Pending', 'Success', 'Cancel') DEFAULT 'Pending';
```

```
22  18:20:11  ALTER TABLE Orders_new MODIFY COLUMN status ENUM('Pending', 'Success', 'Cancel') DEFAULT 'Pending'
```

c. Modify payment_method ENUM to add one more value: 'COD'

```
60 •  ALTER TABLE Orders_new MODIFY COLUMN payment_method ENUM('Credit', 'Debit', 'UPI', 'COD');
61
```

```
25  18:21:26  ALTER TABLE Orders_new MODIFY COLUMN payment_method ENUM('Credit', 'Debit', 'UPI', 'COD')  0 row(s) affected
```

d. Make product id as foreign key

```
61 •  ALTER TABLE Orders_new ADD CONSTRAINT fk_product FOREIGN KEY (product_id) REFERENCES Products(product_id);
62
```

```
26  18:23:14  ALTER TABLE Orders_new ADD CONSTRAINT fk_product FOREIGN KEY (product_id) REFERENCES Produ...  0 row(s) affected Records: 0  Duplic
```

6. Insert 20 sample records in all the tables.

// Inserting into customer table

```
61
62 •    INSERT INTO Customers (name, email, mobile, age) VALUES
63      ('Sara', 'sara@example.com', '9876543210', 21),
64      ('Amit', 'amit@example.com', '9988776655', 30),
65      ('Rehan', 'rehan@example.com', '9876123456', 27),
66      ('Priya', 'priya@example.com', '8765432109', 22),
67      ('Fatima', 'fatima@example.com', '9123456780', 29),
68      ('Rohan', 'rohan@example.com', '9823456710', 31),
69      ('Ali', 'ali@example.com', '9734567891', 28),
70      ('Sneha', 'sneha@example.com', '9988771122', 25),
71      ('Yusuf', 'yusuf@example.com', '9765432189', 26),
72      ('Neha', 'neha@example.com', '9988654321', 23),
73      ('Faisal', 'faisal@example.com', '9786543210', 29),
74      ('Ishita', 'ishita@example.com', '9900123456', 24),
75      ('Zainab', 'zainab@example.com', '9876098765', 27),
76      ('Arjun', 'arjun@example.com', '9754678921', 28),
77      ('Ayesha', 'ayesha@example.com', '9887766554', 30),
78      ('Rajesh', 'rajesh@example.com', '9871234567', 32),
79      ('Nida', 'nida@example.com', '9875076543', 26),
80      ('Sahil', 'sahil@example.com', '9723456789', 25),
81      ('Mariam', 'mariam@example.com', '9827654321', 27),
82      ('Vikram', 'vikram@example.com', '9912345678', 29);
83
```

//Inserting into product table

```
INSERT INTO Products (name, description, price, category) VALUES
('Laptop', 'High-performance ', 65000, 'Electronics'),
('Smartphone', '5G smartphone ', 30000, 'Electronics'),
('Headphones', 'Wireless ', 5000, 'Accessories'),
('Smartwatch', 'Fitness tracker ', 12000, 'Wearables'),
('Tablet', '10-inch tablet', 25000, 'Electronics'),
('Power Bank', 'fast-charging ', 3000, 'Accessories'),
('Mouse', 'wireless mouse', 1500, 'Accessories'),
('Keyboard', 'Mechanical keyboard ', 4000, 'Accessories'),
('LED TV', '4K Smart LED TV ', 50000, 'Electronics'),
('Speaker', 'Portable Bluetooth speaker ', 6000, 'Accessories'),
('Refrigerator', ' double-door refrigerator', 35000, 'Appliances'),
('Microwave', 'Convection microwave ', 10000, 'Appliances'),
('Air Conditioner', 'Split AC ', 45000, 'Appliances'),
('Washing Machine', ' washing machine with smart sensors', 30000, 'Appliances'),
('Gaming Console', '  4K support', 50000, 'Gaming'),
('Camera', ' 24MP sensor', 70000, 'Electronics'),
('Fitness Band', 'Waterproof fitness band', 4000, 'Wearables'),
('Fan', 'Energy-efficient  ', 2500, 'Appliances'),
('Charger', ' Type-C charger', 1200, 'Accessories'),
('Router', 'WiFi 6  ', 7000, 'Electronics');
```

//Inserting into Orders_new table

```
INSERT INTO Orders_new (customer_id, product_id, quantity, order_date, status, payment_method, total_amount) VALUES
(1, 2, 1, '2025-02-15', 'Pending', 'UPI', 30000),
(2, 5, 2, '2025-02-14', 'Success', 'Credit', 50000),
(3, 1, 1, '2025-02-13', 'Pending', 'Debit', 65000),
(4, 8, 1, '2025-02-12', 'Success', 'COD', 4000),
(5, 7, 3, '2025-02-11', 'Cancel', 'UPI', 4500),
(6, 3, 2, '2025-02-10', 'Success', 'Debit', 10000),
(7, 9, 1, '2025-02-09', 'Pending', 'Credit', 50000),
(8, 12, 1, '2025-02-08', 'Success', 'UPI', 10000),
(9, 14, 2, '2025-02-07', 'Success', 'Debit', 60000),
(10, 16, 1, '2025-02-06', 'Pending', 'COD', 70000),
(11, 18, 1, '2025-02-05', 'Success', 'UPI', 2500),
(12, 19, 3, '2025-02-04', 'Pending', 'Credit', 3600),
(13, 11, 1, '2025-02-03', 'Success', 'Debit', 35000),
(14, 13, 2, '2025-02-02', 'Cancel', 'COD', 90000),
(15, 4, 1, '2025-02-01', 'Pending', 'UPI', 12000),
(16, 6, 2, '2025-01-31', 'Success', 'Credit', 6000),
(17, 10, 1, '2025-01-30', 'Success', 'Debit', 6000),
(18, 15, 1, '2025-01-29', 'Pending', 'UPI', 50000),
(19, 17, 3, '2025-01-28', 'Success', 'COD', 12000),
(20, 20, 1, '2025-01-27', 'Cancel', 'Debit', 7000);
```

7. Perform following queries:

a. Count the number of products as product_count in each category.

```
129
130 •   SELECT category, COUNT(*) AS product_count FROM Products GROUP BY category;
```

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

category	product_count
Electronics	6
Accessories	6
Wearables	2
Appliances	5
Gaming	1

b. Retrieve all products that belong to the 'Electronics' category, have a price between $50 and $500, and whose name contains the letter 'a'.

```
131
132 •  SELECT * FROM Products WHERE category = 'Electronics' AND price BETWEEN 50 AND 500 AND name LIKE '%a%';
133
```

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

product_id	name	description	price	category
21	Laptop	High-performance	400.00	Electronics
NULL	NULL	NULL	NULL	NULL

c. Get the top 5 most expensive products in the 'Electronics' category, skipping the first 2.

```
133
134 •  SELECT * FROM Products WHERE category = 'Electronics' ORDER BY price DESC LIMIT 5 OFFSET 2;
135
136
```

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

product_id	name	description	price	category
9	LED TV	4K Smart LED TV	50000.00	Electronics
2	Smartphone	5G smartphone	30000.00	Electronics
5	Tablet	10-inch tablet	25000.00	Electronics
20	Router	WiFi 6	7000.00	Electronics
21	Laptop	High-performance	400.00	Electronics
NULL	NULL	NULL	NULL	NULL

d. Retrieve customers who have not placed any orders.

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

customer_id	name	email	mobile	age
21	Fatma	Fatma@example.com	9874443210	21

e. Find the average total amount spent by each customer.

```
137
138 •  SELECT customer_id, AVG(total_amount) AS avg_spent FROM Orders_new GROUP BY customer_id;
139
```

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

customer_id	avg_spent
1	30000.000000
2	50000.000000
3	65000.000000
4	4000.000000
5	4500.000000
6	10000.000000
7	50000.000000
8	10000.000000
9	60000.000000
10	70000.000000
11	2500.000000
12	3600.000000
13	35000.000000
14	90000.000000
15	12000.000000
16	6000.000000
17	6000.000000
18	50000.000000
19	12000.000000
20	7000.000000

f. Get the products that have a price less than the average price of all products.

```
139
140 •  SELECT * FROM Products WHERE price < (SELECT AVG(price) FROM Products);
141
142
```

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

product_id	name	description	price	category
3	Headphones	Wireless	5000.00	Accessories
4	Smartwatch	Fitness tracker	12000.00	Wearables
6	Power Bank	fast-charging	3000.00	Accessories
7	Mouse	wireless mouse	1500.00	Accessories
8	Keyboard	Mechanical keyboard	4000.00	Accessories
10	Speaker	Portable Bluetooth speaker	6000.00	Accessories
12	Microwave	Convection microwave	10000.00	Appliances
17	Fitness Band	Waterproof fitness band	4000.00	Wearables
18	Fan	Energy-efficient	2500.00	Appliances
19	Charger	Type-C charger	1200.00	Accessories
20	Router	WiFi 6	7000.00	Electronics
21	Laptop	High-performance	400.00	Electronics
NULL	NULL	NULL	NULL	NULL

g. Calculate the total quantity of products ordered by each customer:

```
141
142 •  SELECT customer_id, SUM(quantity) AS total_quantity_ordered FROM Orders_new GROUP BY customer_id;
143
```

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

customer_id	total_quantity_ordered
1	1
2	2
3	1
4	1
5	3
6	2
7	1
8	1
9	2
10	1
11	1
12	3
13	1
14	2
15	1
16	2
17	1
18	1
19	3
20	1

h. List all orders along with customer name and product name.

```
144 •  SELECT o.order_id, c.name AS customer_name, p.name AS product_name, o.quantity, o.order_date, o.status, o.payment_method, o.total_amount
145    FROM Orders_new o JOIN Customers c ON o.customer_id = c.customer_id JOIN Products p ON o.product_id = p.product_id;
146
```

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

order_id	customer_name	product_name	quantity	order_date	status	payment_method	total_amount
1	Sara	Smartphone	1	2025-02-15	Pending	UPI	30000.00
2	Amit	Tablet	2	2025-02-14	Success	Credit	50000.00
3	Rehan	Laptop	1	2025-02-13	Pending	Debit	65000.00
4	Priya	Keyboard	1	2025-02-12	Success	COD	4000.00
5	Fatima	Mouse	3	2025-02-11	Cancel	UPI	4500.00
6	Rohan	Headphones	2	2025-02-10	Success	Debit	10000.00
7	Ali	LED TV	1	2025-02-09	Pending	Credit	50000.00
8	Sneha	Microwave	1	2025-02-08	Success	UPI	10000.00
9	Yusuf	Washing Machine	2	2025-02-07	Success	Debit	60000.00
10	Neha	Camera	1	2025-02-06	Pending	COD	70000.00
11	Faisal	Fan	1	2025-02-05	Success	UPI	2500.00
12	Ishita	Charger	3	2025-02-04	Pending	Credit	3600.00
13	Zainab	Refrigerator	1	2025-02-03	Success	Debit	35000.00
14	Arjun	Air Conditioner	2	2025-02-02	Cancel	COD	90000.00
15	Ayesha	Smartwatch	1	2025-02-01	Pending	UPI	12000.00
16	Rajesh	Power Bank	2	2025-01-31	Success	Credit	6000.00
17	Nida	Speaker	1	2025-01-30	Success	Debit	6000.00
18	Sahil	Gaming Console	1	2025-01-29	Pending	UPI	50000.00
19	Mariam	Fitness Band	3	2025-01-28	Success	COD	12000.00
20	Vikram	Router	1	2025-01-27	Cancel	Debit	7000.00

i. Find products that have never been ordered.

```
149
150 •  SELECT p.* FROM Products p LEFT JOIN Orders_new o ON p.product_id = o.product_id WHERE o.order_id IS NULL;
151
```

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

product_id	name	description	price	category
21	Laptop	High-performance	400.00	Electronics
22	Kettle	Electric	2500.00	Appliances
23	HardDrive	External	6000.00	Electronics