```
create database Ecommerce;
use Ecommerce;

CREATE TABLE customers (
    customer_id INT PRIMARY KEY,
    name VARCHAR(100),
    email VARCHAR(100),
    address VARCHAR(255)
);

CREATE TABLE products (
    product_id INT PRIMARY KEY,
    name VARCHAR(100),
    description TEXT,
    price DECIMAL(10,2),
    stockQuantity INT
);
CREATE TABLE cart (
    cart_id INT PRIMARY KEY,
    customer_id INT,
    product_id INT,
    quantity INT,
    FOREIGN KEY (customer_id) REFERENCES customers(customer_id),
    FOREIGN KEY (product_id) REFERENCES products(product_id)
);

CREATE TABLE orders (
    order_id INT PRIMARY KEY,
    customer_id INT,
```

```
    order_date DATE,
    total_price DECIMAL(10,2),
    shipping_address VARCHAR(255),
    FOREIGN KEY (customer_id) REFERENCES customers(customer_id)
);

CREATE TABLE order_items (
    order_item_id INT PRIMARY KEY,
    order_id INT,
    product_id INT,
    quantity INT,
    item_amount DECIMAL(10,2),
    FOREIGN KEY (order_id) REFERENCES orders(order_id),
    FOREIGN KEY (product_id) REFERENCES products(product_id)
);
 INSERT INTO customers VALUES
(1, 'John Doe', 'johndoe@example.com', '123 Main St, City'),
(2, 'Jane Smith', 'janesmith@example.com', '456 Elm St, Town'),
(3, 'Robert Johnson', 'robert@example.com', '789 Oak St, Village'),
(4, 'Sarah Brown', 'sarah@example.com', '101 Pine St, Suburb'),
(5, 'David Lee', 'david@example.com', '234 Cedar St, District'),
(6, 'Laura Hall', 'laura@example.com', '567 Birch St, County'),
(7, 'Michael Davis', 'michael@example.com', '890 Maple St, State'),
(8, 'Emma Wilson', 'emma@example.com', '321 Redwood St, Country'),
(9, 'William Taylor', 'william@example.com', '432 Spruce St, Province'),
(10, 'Olivia Adams', 'olivia@example.com', '765 Fir St, Territory');
INSERT INTO products VALUES
(1, 'Laptop', 'High-performance laptop', 800.00, 10),
(2, 'Smartphone', 'Latest smartphone', 600.00, 15),
```

```
(3, 'Tablet', 'Portable tablet', 300.00, 20),
(4, 'Headphones', 'Noise-canceling', 150.00, 30),
(5, 'TV', '4K Smart TV', 900.00, 5),
(6, 'Coffee Maker', 'Automatic coffee maker', 50.00, 25),
(7, 'Refrigerator', 'Energy-efficient', 700.00, 10),
(8, 'Microwave Oven', 'Countertop microwave', 80.00, 15),
(9, 'Blender', 'High-speed blender', 70.00, 20),
(10, 'Vacuum Cleaner', 'Bagless vacuum cleaner', 120.00, 10);
INSERT INTO orders VALUES
(1, 1, '2023-01-05', 1200.00, '123 Main St, City'),
(2, 2, '2023-02-10', 900.00, '456 Elm St, Town'),
(3, 3, '2023-03-15', 300.00, '789 Oak St, Village'),
(4, 4, '2023-04-20', 150.00, '101 Pine St, Suburb'),
(5, 5, '2023-05-25', 1800.00, '234 Cedar St, District'),
(6, 6, '2023-06-30', 400.00, '567 Birch St, County'),
(7, 7, '2023-07-05', 700.00, '890 Maple St, State'),
(8, 8, '2023-08-10', 160.00, '321 Redwood St, Country'),
(9, 9, '2023-09-15', 140.00, '432 Spruce St, Province'),
(10, 10, '2023-10-20', 1400.00, '765 Fir St, Territory');
INSERT INTO order_items VALUES
(1, 1, 1, 2, 1600.00),
(2, 1, 3, 1, 300.00),
(3, 2, 2, 3, 1800.00),
(4, 3, 5, 2, 1800.00),
(5, 4, 4, 4, 600.00),
(6, 4, 6, 1, 50.00),
(7, 5, 1, 1, 800.00),
(8, 5, 2, 2, 1200.00),
(9, 6, 10, 2, 240.00),
```

(10, 6, 9, 3, 210.00);

INSERT INTO cart VALUES

(1, 1, 1, 2),

(2, 1, 3, 1),

(3, 2, 2, 3),

(4, 3, 4, 4),

(5, 3, 5, 2),

(6, 4, 6, 1),

(7, 5, 1, 1),

(8, 6, 10, 2),

(9, 6, 9, 3),

(10, 7, 7, 2);

1.Update refrigerator product price to 800.

UPDATE products SET price = 800.00 WHERE name = 'Refrigerator';

2. Remove all cart items for a specific customer.

DELETE FROM cart WHERE customer_id = 3;

3. Retrieve Products Priced Below $100.

SELECT * FROM products WHERE price < 100;

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

product_id	name	description	price	stockQuantity
6	Coffee Maker	Automatic coffee maker	50.00	25
8	Microwave Oven	Countertop microwave	80.00	15
9	Blender	High-speed blender	70.00	20
NULL	NULL	NULL	NULL	NULL

4. Find Products with Stock Quantity Greater Than 5.

SELECT * FROM products WHERE stockQuantity > 5;

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

product_id	name	description	price	stockQuantity
1	Laptop	High-performance laptop	800.00	10
2	Smartphone	Latest smartphone	600.00	15
3	Tablet	Portable tablet	300.00	20
4	Headphones	Noise-canceling	150.00	30
6	Coffee Maker	Automatic coffee maker	50.00	25

products 44 ×

5. Retrieve Orders with Total Amount Between $500 and $1000.

SELECT * FROM orders WHERE total_price BETWEEN 500 AND 1000;

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

order_id	customer_id	order_date	total_price	shipping_address
2	2	2023-02-10	900.00	456 Elm St, Town
7	7	2023-07-05	700.00	890 Maple St, State
NULL	NULL	NULL	NULL	NULL

6. Find Products which name end with letter ‘r’.

SELECT * FROM products WHERE name LIKE '%r';

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

product_id	name	description	price	stockQuantity
6	Coffee Maker	Automatic coffee maker	50.00	25
7	Refrigerator	Energy-efficient	800.00	10
9	Blender	High-speed blender	70.00	20
10	Vacuum Cleaner	Bagless vacuum cleaner	120.00	10
NULL	NULL	NULL	NULL	NULL

products 47

7. Retrieve Cart Items for Customer 5.

SELECT * FROM cart WHERE customer_id = 5;

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

cart_id	customer_id	product_id	quantity
7	5	1	1
NULL	NULL	NULL	NULL

8. Find Customers Who Placed Orders in 2023.

SELECT DISTINCT c.*

FROM customers c

JOIN orders o ON c.customer_id = o.customer_id

WHERE YEAR(order_date) = 2023;

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

customer_id	name	email	address
1	John Doe	johndoe@example.com	123 Main St, City
2	Jane Smith	janesmith@example.com	456 Elm St, Town
3	Robert Johnson	robert@example.com	789 Oak St, Village
4	Sarah Brown	sarah@example.com	101 Pine St, Suburb
5	David Lee	david@example.com	234 Cedar St, District

Result 52

9. Determine the Minimum Stock Quantity for Each Product Category.

SELECT name, MIN(stockQuantity) AS min_stock FROM products GROUP BY name;

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

name	min_stock
Laptop	10
Smartphone	15
Tablet	20
Headphones	30
TV	5

Result 54

10. Calculate the Total Amount Spent by Each Customer.

SELECT c.customer_id, c.name, SUM(o.total_price) AS total_spent

FROM customers c

JOIN orders o ON c.customer_id = o.customer_id

GROUP BY c.customer_id;

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

customer_id	name	total_spent
1	John Doe	1200.00
2	Jane Smith	900.00
3	Robert Johnson	300.00
4	Sarah Brown	150.00
5	David Lee	1800.00

Result 55

11. Find the Average Order Amount for Each Customer.

SELECT c.customer_id, c.name, AVG(o.total_price) AS avg_order_amount

FROM customers c

JOIN orders o ON c.customer_id = o.customer_id

GROUP BY c.customer_id;

143

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

customer_id	name	avg_order_amount
1	John Doe	1200.000000
2	Jane Smith	900.000000
3	Robert Johnson	300.000000
4	Sarah Brown	150.000000
5	David Lee	1800.000000

Result 56

12. Count the Number of Orders Placed by Each Customer.

```
SELECT customer_id, COUNT(*) AS order_count
FROM orders
GROUP BY customer_id;
```

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

customer_id	order_count
1	1
2	1
3	1
4	1
5	1

13. Find the Maximum Order Amount for Each Customer

```
SELECT customer_id, MAX(total_price) AS max_order
FROM orders
GROUP BY customer_id;
```

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

customer_id	max_order
1	1200.00
2	900.00
3	300.00
4	150.00
5	1800.00

Result 59

14. Get Customers Who Placed Orders Totaling Over $1000.

```
SELECT c.customer_id, c.name, SUM(o.total_price) AS total
FROM customers c
JOIN orders o ON c.customer_id = o.customer_id
GROUP BY c.customer_id
HAVING total > 1000;
```

```
164 •   SELECT * FROM products
```

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

customer_id	name	total
1	John Doe	1200.00
5	David Lee	1800.00
10	Olivia Adams	1400.00

15. Subquery to Find Products Not in the Cart

WHERE product_id NOT IN (SELECT product_id FROM cart);

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

product_id	name	description	price	stockQuantity
4	Headphones	Noise-canceling	150.00	30
5	TV	4K Smart TV	900.00	5
8	Microwave Oven	Countertop microwave	80.00	15
NULL	NULL	NULL	NULL	NULL

products 62 ×

16. Subquery to Find Customers Who Haven't Placed Orders.

SELECT * FROM customers

WHERE customer_id NOT IN (SELECT customer_id FROM orders);

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

customer_id	name	email	address
NULL	NULL	NULL	NULL

17. Subquery to Calculate the Percentage of Total Revenue for a Product.

SELECT

product_id,

ROUND(SUM(item_amount) / (SELECT SUM(item_amount) FROM order_items) * 100, 2) AS revenue_percentage

FROM order_items

WHERE product_id = 1

GROUP BY product_id;

```
177
178 •  SELECT * FROM products
179    WHERE stockQuantity < (
```

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

product_id	revenue_percentage
1	27.91

18. Subquery to Find Products with Low Stock.

```
SELECT * FROM products

WHERE stockQuantity < (

    SELECT AVG(stockQuantity) FROM products

);
```

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

product_id	name	description	price	stockQuantity
1	Laptop	High-performance laptop	800.00	10
2	Smartphone	Latest smartphone	600.00	15
5	TV	4K Smart TV	900.00	5
7	Refrigerator	Energy-efficient	800.00	10
8	Microwave Oven	Countertop microwave	80.00	15

19. Subquery to Find Customers Who Placed High-Value Orders.

```
SELECT * FROM customers

WHERE customer_id IN (

    SELECT customer_id

    FROM orders

    WHERE total_price > 1000

);
```

.94

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

customer_id	name	email	address
1	John Doe	johndoe@example.com	123 Main St, City
5	David Lee	david@example.com	234 Cedar St, District
10	Olivia Adams	olivia@example.com	765 Fir St, Territory
NULL	NULL	NULL	NULL

stomers 67