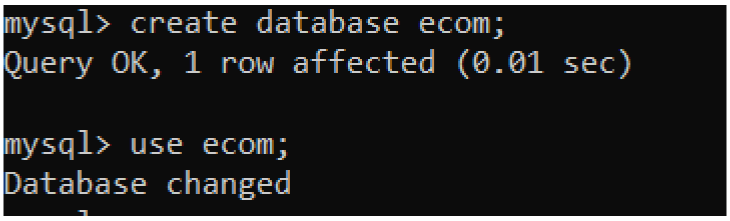
SQL CODING CHALLENGE – SIVAPRAKAS B M – ECOM

-- Create a database “ecom”

- **Create database ecom;**
- **Use ecom;**

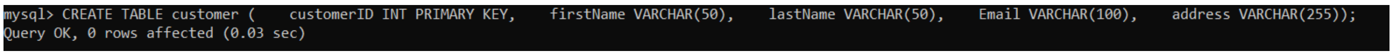


```
mysql> create database ecom;
Query OK, 1 row affected (0.01 sec)

mysql> use ecom;
Database changed
```

-- Create table customer

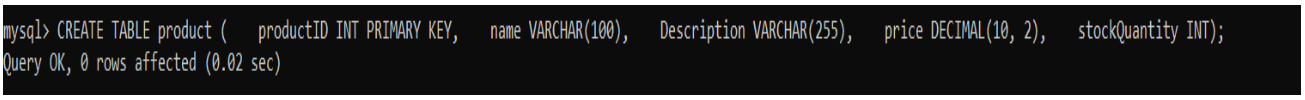
```
CREATE TABLE customer (
    customerID INT PRIMARY KEY,
    firstName VARCHAR(50),
    lastName VARCHAR(50),
    Email VARCHAR(100),
    address VARCHAR(255)
);
```



```
mysql> CREATE TABLE customer (    customerID INT PRIMARY KEY,    firstName VARCHAR(50),    lastName VARCHAR(50),    Email VARCHAR(100),    address VARCHAR(255));
Query OK, 0 rows affected (0.03 sec)
```

-- Create table product

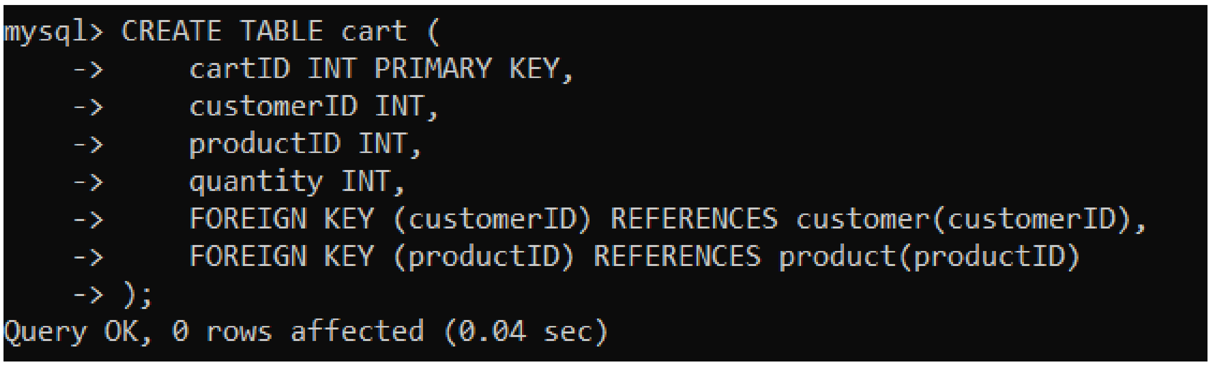
```
CREATE TABLE product (
    productID INT PRIMARY KEY,
    name VARCHAR(100),
    Description VARCHAR(255),
    price DECIMAL(10, 2),
    stockQuantity INT
);
```



```
mysql> CREATE TABLE product (    productID INT PRIMARY KEY,    name VARCHAR(100),    Description VARCHAR(255),    price DECIMAL(10, 2),    stockQuantity INT);
Query OK, 0 rows affected (0.02 sec)
```

-- Create table cart

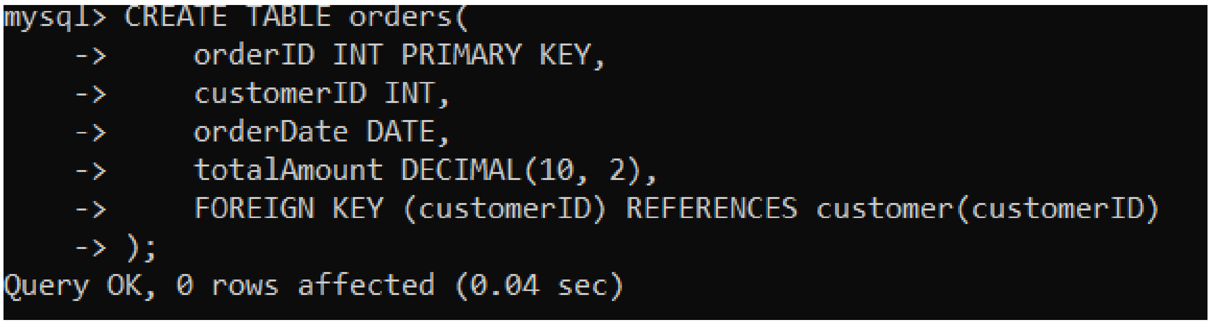
```
CREATE TABLE cart (
    cartID INT PRIMARY KEY,
    customerID INT,
    productID INT,
    quantity INT,
    FOREIGN KEY (customerID) REFERENCES customer(customerID),
    FOREIGN KEY (productID) REFERENCES product(productID)
);
```



```
mysql> CREATE TABLE cart (
    ->     cartID INT PRIMARY KEY,
    ->     customerID INT,
    ->     productID INT,
    ->     quantity INT,
    ->     FOREIGN KEY (customerID) REFERENCES customer(customerID),
    ->     FOREIGN KEY (productID) REFERENCES product(productID)
    -> );
Query OK, 0 rows affected (0.04 sec)
```

-- Create table “orders”

```
CREATE TABLE orders(
    orderID INT PRIMARY KEY,
    customerID INT,
    orderDate DATE,
    totalAmount DECIMAL(10, 2),
    FOREIGN KEY (customerID) REFERENCES customer(customerID)
);
```



```
mysql> CREATE TABLE orders(
    ->     orderID INT PRIMARY KEY,
    ->     customerID INT,
    ->     orderDate DATE,
    ->     totalAmount DECIMAL(10, 2),
    ->     FOREIGN KEY (customerID) REFERENCES customer(customerID)
    -> );
Query OK, 0 rows affected (0.04 sec)
```

-- Create table orderitems

CREATE TABLE orderitems (

orderItemID INT PRIMARY KEY,

orderID INT,

productID INT,

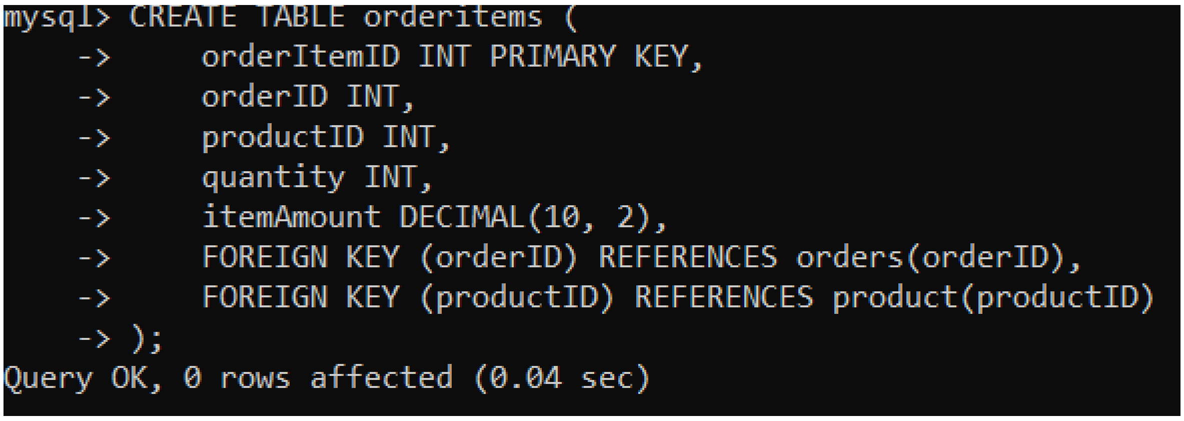
quantity INT,

itemAmount DECIMAL(10, 2),

FOREIGN KEY (orderID) REFERENCES orders(orderID),

FOREIGN KEY (productID) REFERENCES product(productID)

);



```
mysql> CREATE TABLE orderitems (
    ->     orderItemID INT PRIMARY KEY,
    ->     orderID INT,
    ->     productID INT,
    ->     quantity INT,
    ->     itemAmount DECIMAL(10, 2),
    ->     FOREIGN KEY (orderID) REFERENCES orders(orderID),
    ->     FOREIGN KEY (productID) REFERENCES product(productID)
    -> );
Query OK, 0 rows affected (0.04 sec)
```

INSERT INTO customer (customerID, firstName, lastName, Email, address) 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'),

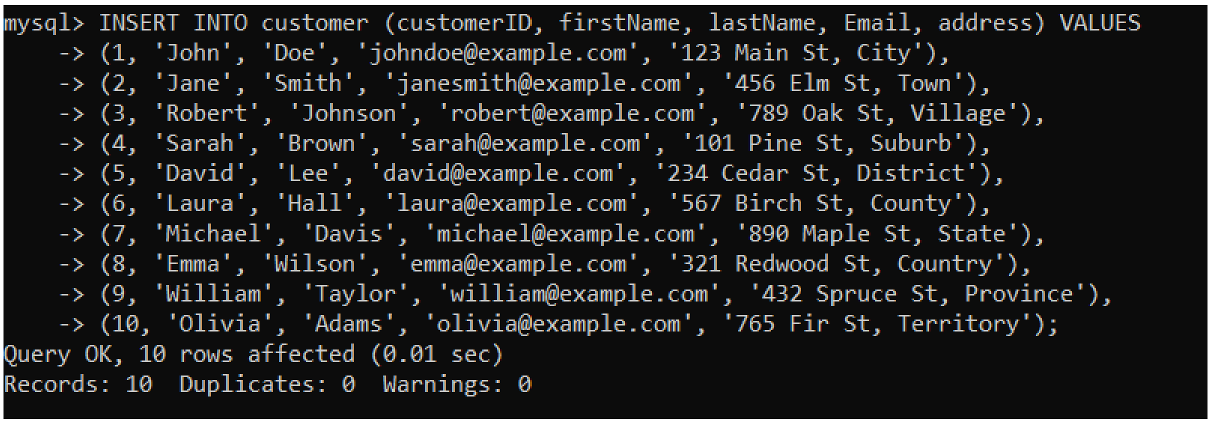
(6, 'Laura', 'Hall', 'laura@example.com', '567 Birch St, County'),

(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');



```
mysql> INSERT INTO customer (customerID, firstName, lastName, Email, address) 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');
Query OK, 10 rows affected (0.01 sec)
Records: 10  Duplicates: 0  Warnings: 0
```

INSERT INTO product (productID, name, Description, price, stockQuantity) 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),

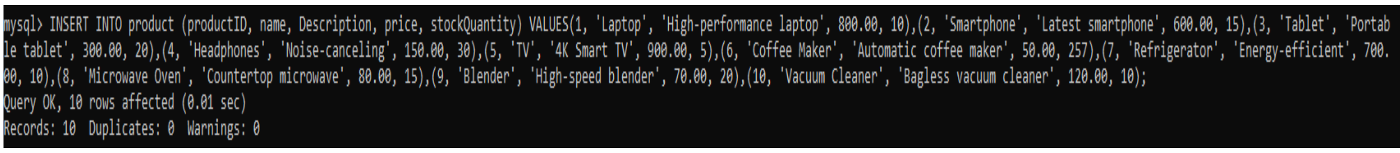
(6, 'Coffee Maker', 'Automatic coffee maker', 50.00, 257),

(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);



```
mysql> INSERT INTO product (productID, name, Description, price, stockQuantity) VALUES(1, 'Laptop', 'High-performance laptop', 800.00, 10),(2, 'Smartphone', 'Latest smartphone', 600.00, 15),(3, 'Tablet', 'Portab
le 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, 257),(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);
Query OK, 10 rows affected (0.01 sec)
Records: 10  Duplicates: 0  Warnings: 0
```

INSERT INTO cart (cartID, customerID, productID, quantity) VALUES

(1, 1, 1, 2),

(2, 1, 3, 1),

(3, 2, 2, 3),

(4, 3, 4, 4),

(5, 3, 5, 2),

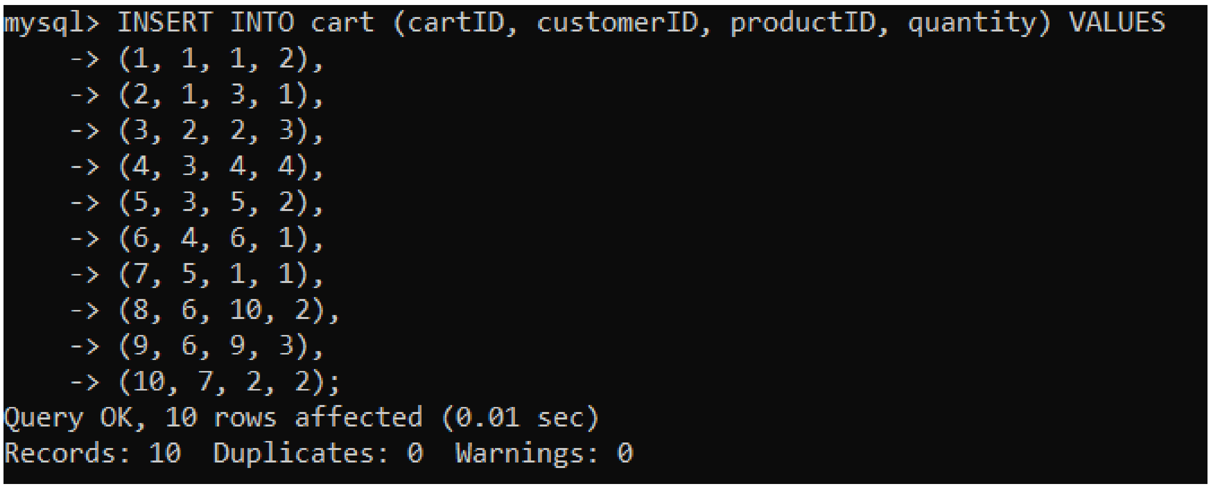
(6, 4, 6, 1),

(7, 5, 1, 1),

(8, 6, 10, 2),

(9, 6, 9, 3),

(10, 7, 2, 2);



```
mysql> INSERT INTO cart (cartID, customerID, productID, quantity) 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, 2, 2);
Query OK, 10 rows affected (0.01 sec)
Records: 10  Duplicates: 0  Warnings: 0
```

INSERT INTO orders (orderID, customerID, orderDate, totalAmount) VALUES

(1, 1, '2023-01-05', 1200.00),

(2, 2, '2023-02-10', 900.00),

(3, 3, '2023-03-15', 300.00),

(4, 4, '2023-04-20', 150.00),

(5, 5, '2023-05-25', 1800.00),

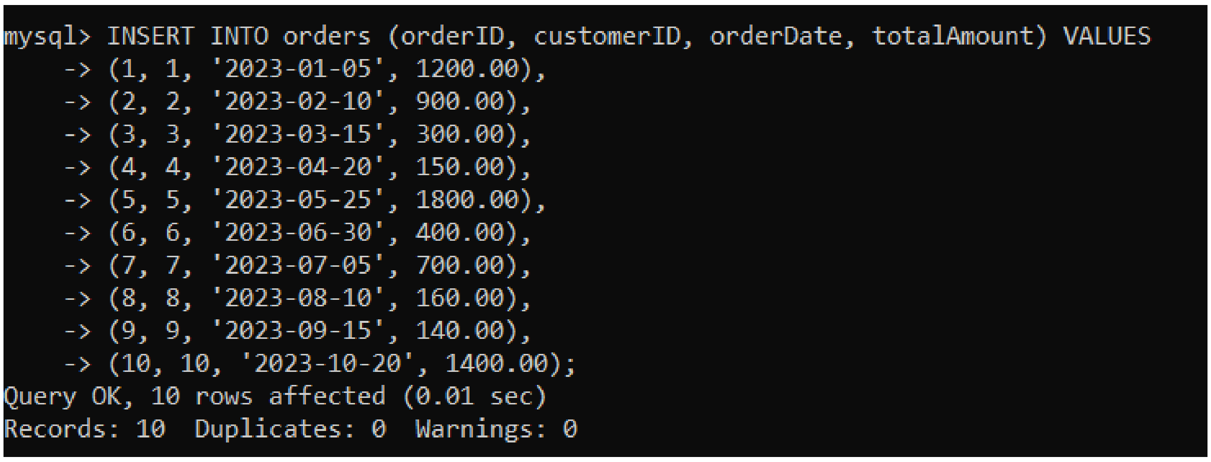
(6, 6, '2023-06-30', 400.00),

(7, 7, '2023-07-05', 700.00),

(8, 8, '2023-08-10', 160.00),

(9, 9, '2023-09-15', 140.00),

(10, 10, '2023-10-20', 1400.00);



```
mysql> INSERT INTO orders (orderID, customerID, orderDate, totalAmount) VALUES
    -> (1, 1, '2023-01-05', 1200.00),
    -> (2, 2, '2023-02-10', 900.00),
    -> (3, 3, '2023-03-15', 300.00),
    -> (4, 4, '2023-04-20', 150.00),
    -> (5, 5, '2023-05-25', 1800.00),
    -> (6, 6, '2023-06-30', 400.00),
    -> (7, 7, '2023-07-05', 700.00),
    -> (8, 8, '2023-08-10', 160.00),
    -> (9, 9, '2023-09-15', 140.00),
    -> (10, 10, '2023-10-20', 1400.00);
Query OK, 10 rows affected (0.01 sec)
Records: 10  Duplicates: 0  Warnings: 0
```

INSERT INTO orderitems (orderItemID, orderID, productID, quantity, itemAmount) 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),

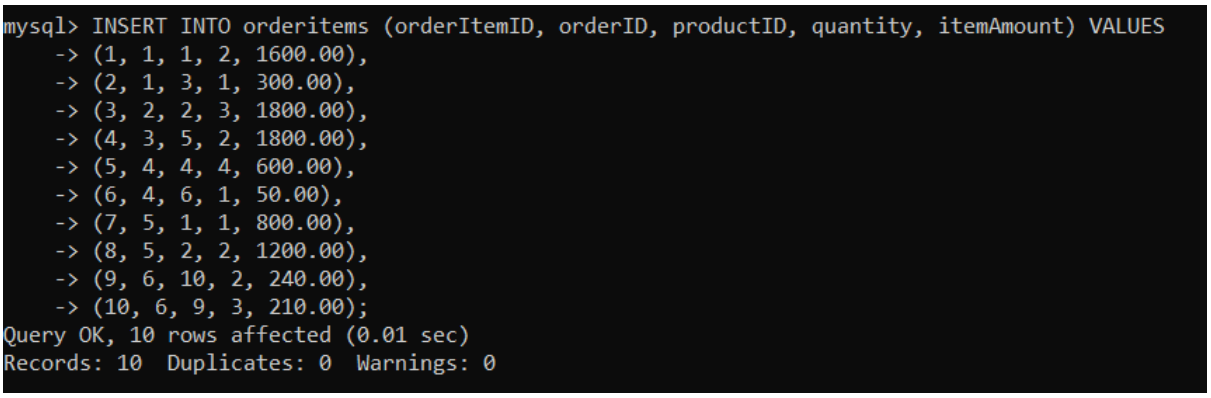
(6, 4, 6, 1, 50.00),

(7, 5, 1, 1, 800.00),

(8, 5, 2, 2, 1200.00),

(9, 6, 10, 2, 240.00),

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

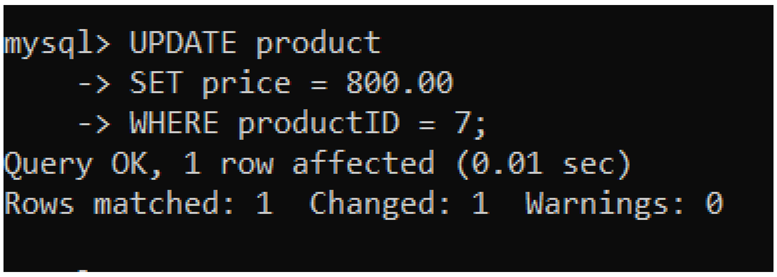

```
mysql> INSERT INTO orderitems (orderItemID, orderID, productID, quantity, itemAmount) VALUES
    -> (1, 1, 1, 2, 1600.00),
    -> (2, 1, 3, 1, 300.00),
    -> (3, 2, 2, 3, 1800.00),
    -> (4, 3, 5, 2, 1800.00),
    -> (5, 4, 4, 4, 600.00),
    -> (6, 4, 6, 1, 50.00),
    -> (7, 5, 1, 1, 800.00),
    -> (8, 5, 2, 2, 1200.00),
    -> (9, 6, 10, 2, 240.00),
    -> (10, 6, 9, 3, 210.00);
Query OK, 10 rows affected (0.01 sec)
Records: 10  Duplicates: 0  Warnings: 0
```

-- 1) Update refrigerator product price to 800.

UPDATE product

SET price = 800.00

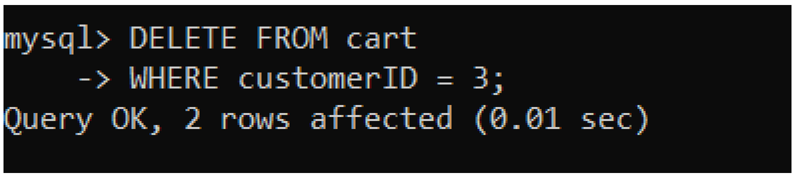
WHERE productID = 7;


```
mysql> UPDATE product
    -> SET price = 800.00
    -> WHERE productID = 7;
Query OK, 1 row affected (0.01 sec)
Rows matched: 1  Changed: 1  Warnings: 0
```

-- 2) **Remove all cart items for a specific customer.**

DELETE FROM cart

WHERE customerID = 3;

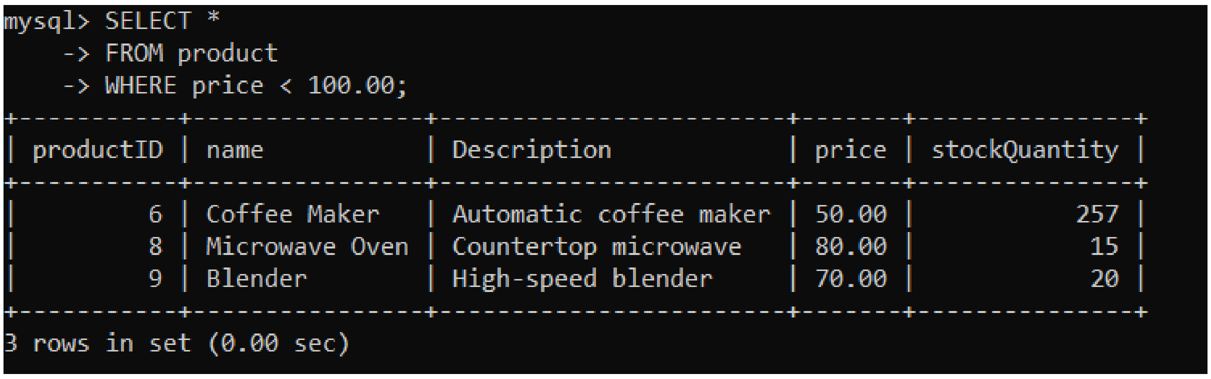

```
mysql> DELETE FROM cart
    -> WHERE customerID = 3;
Query OK, 2 rows affected (0.01 sec)
```

-- 3) Retrieve Products Priced Below $100.

SELECT *

FROM product

WHERE price < 100.00;



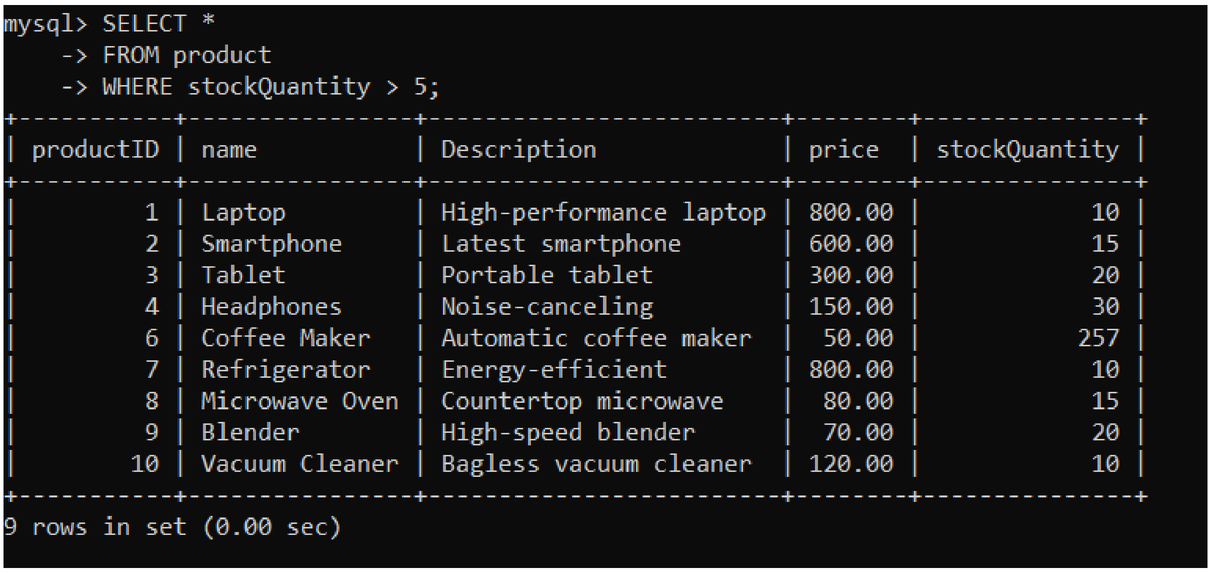
```
mysql> SELECT *
    -> FROM product
    -> WHERE price < 100.00;
+-----------+----------------+-----------------------+-------+---------------+
| productID | name           | Description           | price | stockQuantity |
+-----------+----------------+-----------------------+-------+---------------+
|         6 | Coffee Maker   | Automatic coffee maker | 50.00 |           257 |
|         8 | Microwave Oven | Countertop microwave  | 80.00 |            15 |
|         9 | Blender        | High-speed blender    | 70.00 |            20 |
+-----------+----------------+-----------------------+-------+---------------+
3 rows in set (0.00 sec)
```

-- 4) Find Products with Stock Quantity Greater Than 5.

SELECT *

FROM product

WHERE stockQuantity > 5;



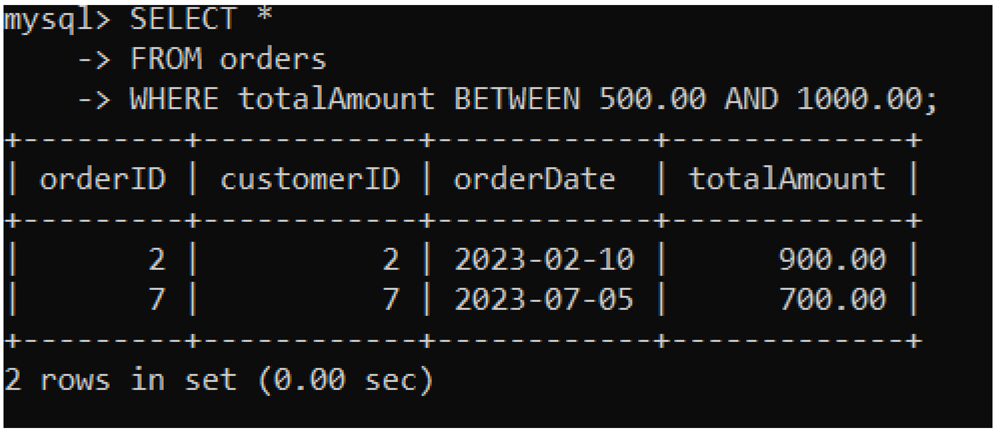
```
mysql> SELECT *
    -> FROM product
    -> WHERE stockQuantity > 5;
+-----------+----------------+-------------------------+--------+---------------+
| productID | name           | Description             | price  | stockQuantity |
+-----------+----------------+-------------------------+--------+---------------+
|         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 |           257 |
|         7 | Refrigerator   | Energy-efficient        | 800.00 |            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 |
+-----------+----------------+-------------------------+--------+---------------+
9 rows in set (0.00 sec)
```

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

SELECT *

FROM orders

WHERE totalAmount BETWEEN 500.00 AND 1000.00;



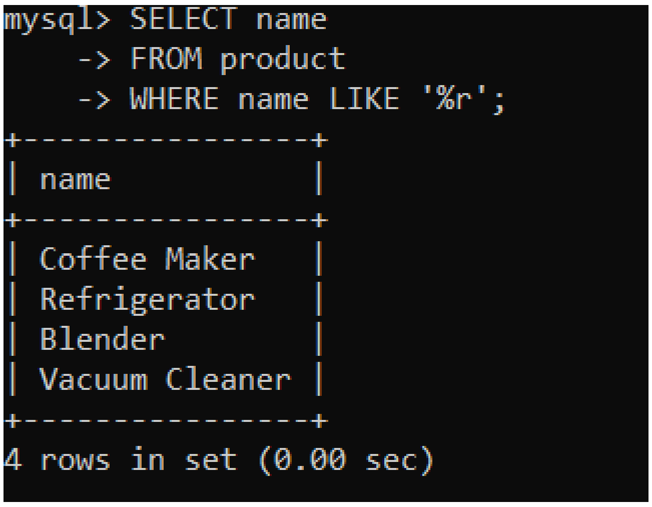
```
mysql> SELECT *
    -> FROM orders
    -> WHERE totalAmount BETWEEN 500.00 AND 1000.00;
+---------+------------+------------+-------------+
| orderID | customerID | orderDate  | totalAmount |
+---------+------------+------------+-------------+
|       2 |          2 | 2023-02-10 |      900.00 |
|       7 |          7 | 2023-07-05 |      700.00 |
+---------+------------+------------+-------------+
2 rows in set (0.00 sec)
```

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

SELECT name

FROM product

WHERE name LIKE '%r';



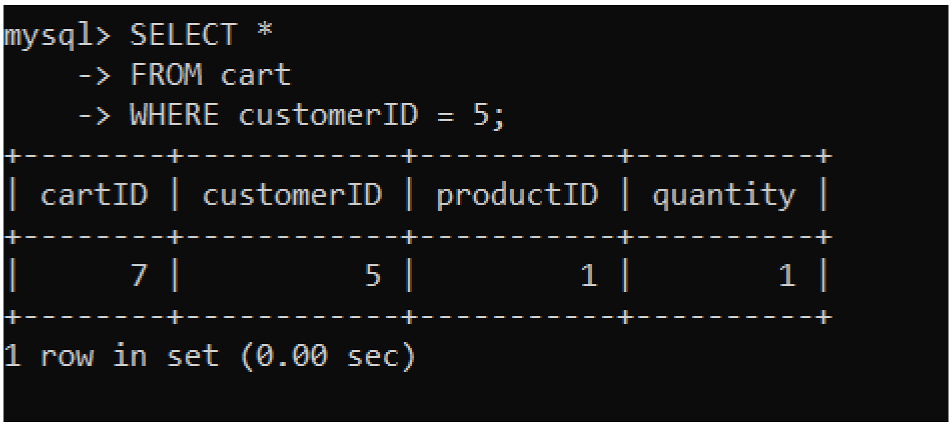
```
mysql> SELECT name
    -> FROM product
    -> WHERE name LIKE '%r';
+----------------+
| name           |
+----------------+
| Coffee Maker   |
| Refrigerator   |
| Blender        |
| Vacuum Cleaner |
+----------------+
4 rows in set (0.00 sec)
```

--7) Retrieve Cart Items for Customer 5.

SELECT *

FROM cart

WHERE customerID = 5;



```
mysql> SELECT *
    -> FROM cart
    -> WHERE customerID = 5;
+--------+------------+-----------+----------+
| cartID | customerID | productID | quantity |
+--------+------------+-----------+----------+
|      7 |          5 |         1 |        1 |
+--------+------------+-----------+----------+
1 row in set (0.00 sec)
```

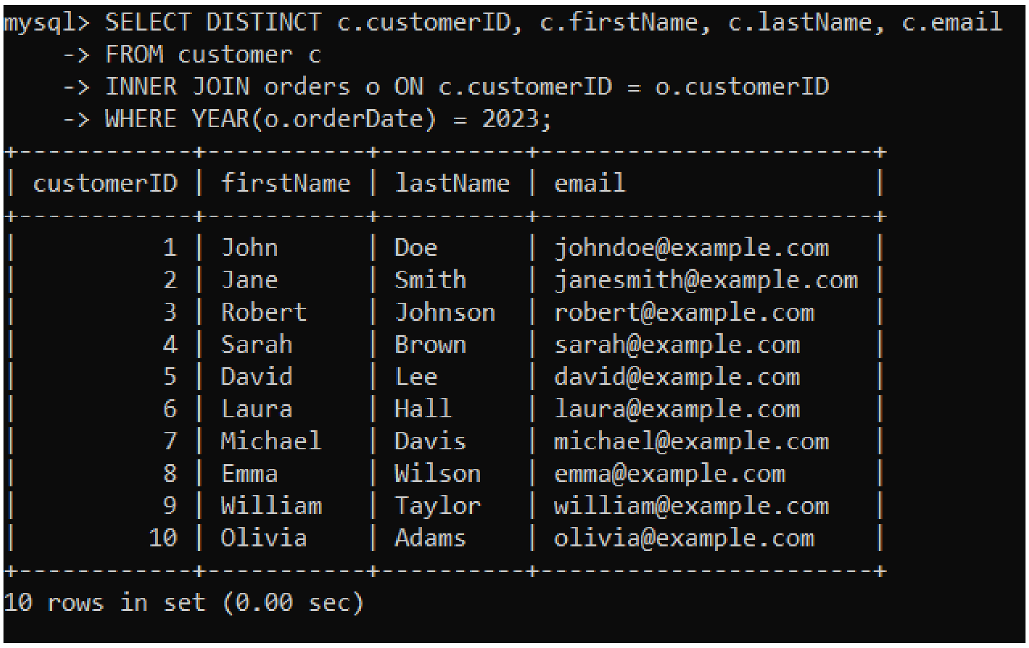
-- 8) Find Customers Who Placed Orders in 2023.

SELECT DISTINCT c.customerID, c.firstName, c.lastName, c.email

FROM customer c

INNER JOIN orders o ON c.customerID = o.customerID

WHERE YEAR(o.orderDate) = 2023;



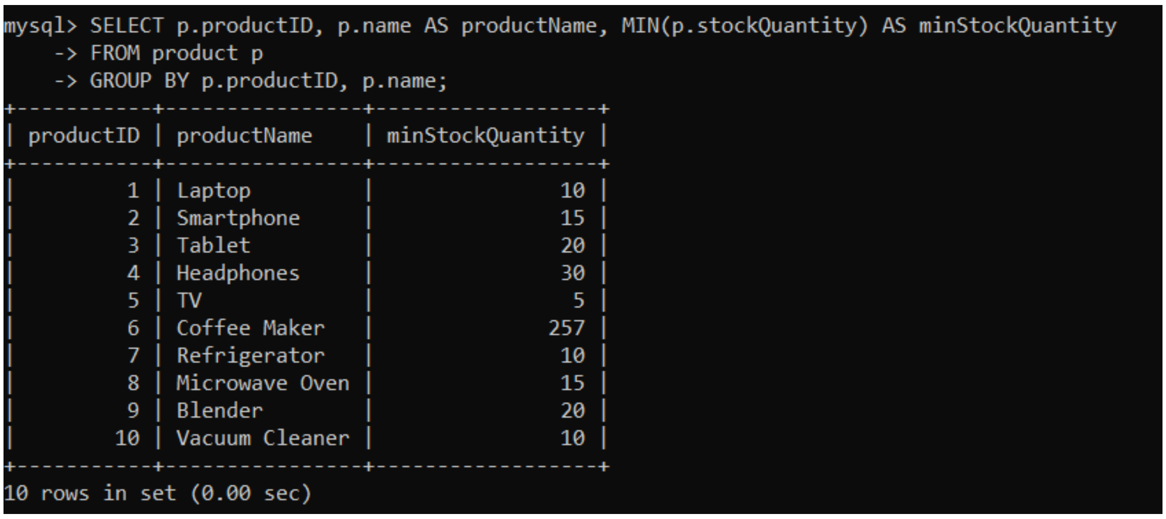
```
mysql> SELECT DISTINCT c.customerID, c.firstName, c.lastName, c.email
    -> FROM customer c
    -> INNER JOIN orders o ON c.customerID = o.customerID
    -> WHERE YEAR(o.orderDate) = 2023;
+------------+-----------+----------+-----------------------+
| customerID | firstName | lastName | email                 |
+------------+-----------+----------+-----------------------+
|          1 | John      | Doe      | johndoe@example.com   |
|          2 | Jane      | Smith    | janesmith@example.com |
|          3 | Robert    | Johnson  | robert@example.com    |
|          4 | Sarah     | Brown    | sarah@example.com     |
|          5 | David     | Lee      | david@example.com     |
|          6 | Laura     | Hall     | laura@example.com     |
|          7 | Michael   | Davis    | michael@example.com   |
|          8 | Emma      | Wilson   | emma@example.com      |
|          9 | William   | Taylor   | william@example.com   |
|         10 | Olivia    | Adams    | olivia@example.com    |
+------------+-----------+----------+-----------------------+
10 rows in set (0.00 sec)
```

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

SELECT p.productID, p.name AS productName, MIN(p.stockQuantity) AS minStockQuantity

FROM product p

GROUP BY p.productID, p.name;



```
mysql> SELECT p.productID, p.name AS productName, MIN(p.stockQuantity) AS minStockQuantity
    -> FROM product p
    -> GROUP BY p.productID, p.name;
+-----------+----------------+------------------+
| productID | productName    | minStockQuantity |
+-----------+----------------+------------------+
|         1 | Laptop         |               10 |
|         2 | Smartphone     |               15 |
|         3 | Tablet         |               20 |
|         4 | Headphones     |               30 |
|         5 | TV             |                5 |
|         6 | Coffee Maker   |              257 |
|         7 | Refrigerator   |               10 |
|         8 | Microwave Oven |               15 |
|         9 | Blender        |               20 |
|        10 | Vacuum Cleaner |               10 |
+-----------+----------------+------------------+
10 rows in set (0.00 sec)
```

-- 10) Calculate the Total Amount Spent by Each Customer.

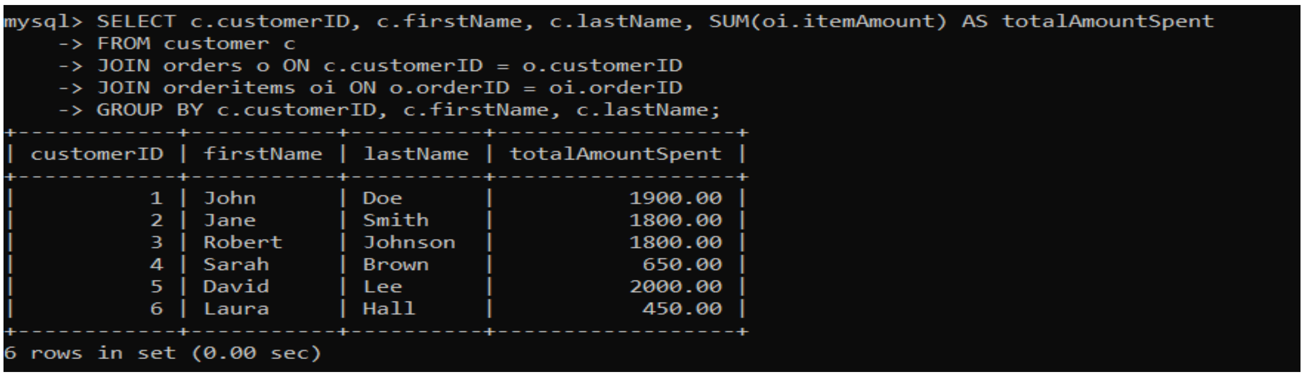
SELECT c.customerID, c.firstName, c.lastName, SUM(oi.itemAmount) AS totalAmountSpent

FROM customer c

JOIN orders o ON c.customerID = o.customerID

JOIN orderitems oi ON o.orderID = oi.orderID

GROUP BY c.customerID, c.firstName, c.lastName;



```
mysql> SELECT c.customerID, c.firstName, c.lastName, SUM(oi.itemAmount) AS totalAmountSpent
    -> FROM customer c
    -> JOIN orders o ON c.customerID = o.customerID
    -> JOIN orderitems oi ON o.orderID = oi.orderID
    -> GROUP BY c.customerID, c.firstName, c.lastName;
+------------+-----------+----------+------------------+
| customerID | firstName | lastName | totalAmountSpent |
+------------+-----------+----------+------------------+
|          1 | John      | Doe      |          1900.00 |
|          2 | Jane      | Smith    |          1800.00 |
|          3 | Robert    | Johnson  |          1800.00 |
|          4 | Sarah     | Brown    |           650.00 |
|          5 | David     | Lee      |          2000.00 |
|          6 | Laura     | Hall     |           450.00 |
+------------+-----------+----------+------------------+
6 rows in set (0.00 sec)
```

-- 11) Find the Average Order Amount for Each Customer.

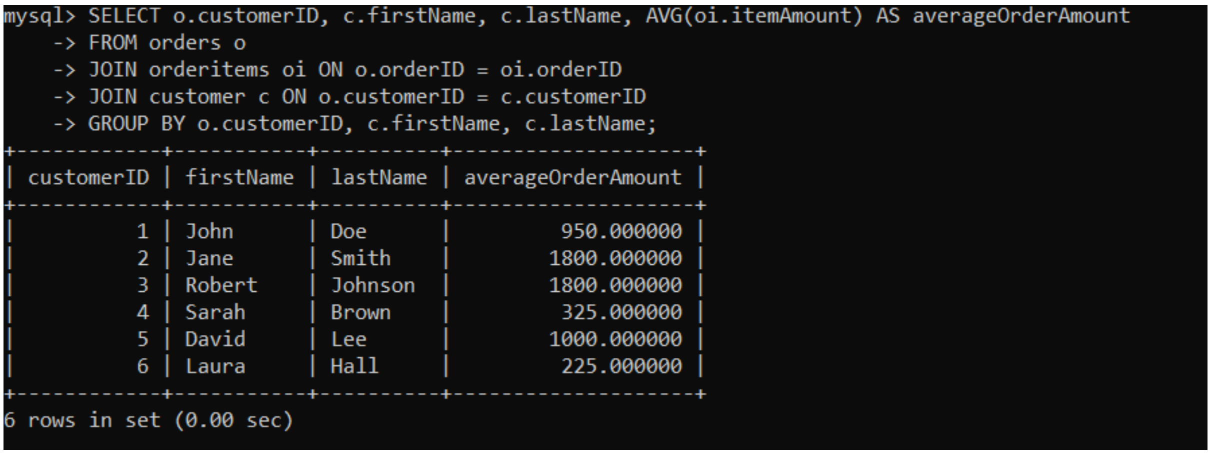
SELECT o.customerID, c.firstName, c.lastName, AVG(oi.itemAmount) AS averageOrderAmount

FROM orders o

JOIN orderitems oi ON o.orderID = oi.orderID

JOIN customer c ON o.customerID = c.customerID

GROUP BY o.customerID, c.firstName, c.lastName;



```
mysql> SELECT o.customerID, c.firstName, c.lastName, AVG(oi.itemAmount) AS averageOrderAmount
    -> FROM orders o
    -> JOIN orderitems oi ON o.orderID = oi.orderID
    -> JOIN customer c ON o.customerID = c.customerID
    -> GROUP BY o.customerID, c.firstName, c.lastName;
+------------+-----------+----------+--------------------+
| customerID | firstName | lastName | averageOrderAmount |
+------------+-----------+----------+--------------------+
|          1 | John      | Doe      |         950.000000 |
|          2 | Jane      | Smith    |        1800.000000 |
|          3 | Robert    | Johnson  |        1800.000000 |
|          4 | Sarah     | Brown    |         325.000000 |
|          5 | David     | Lee      |        1000.000000 |
|          6 | Laura     | Hall     |         225.000000 |
+------------+-----------+----------+--------------------+
6 rows in set (0.00 sec)
```

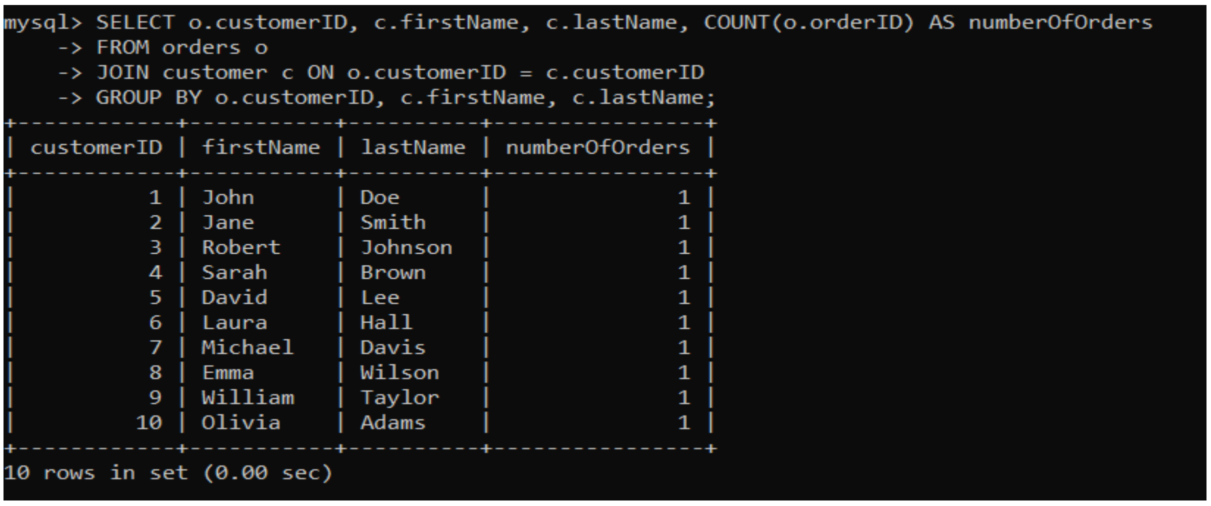
--12) Count the Number of Orders Placed by Each Customer.

SELECT o.customerID, c.firstName, c.lastName, COUNT(o.orderID) AS numberOfOrders

FROM orders o

JOIN customer c ON o.customerID = c.customerID

GROUP BY o.customerID, c.firstName, c.lastName;



```
mysql> SELECT o.customerID, c.firstName, c.lastName, COUNT(o.orderID) AS numberOfOrders
    -> FROM orders o
    -> JOIN customer c ON o.customerID = c.customerID
    -> GROUP BY o.customerID, c.firstName, c.lastName;
+------------+-----------+----------+----------------+
| customerID | firstName | lastName | numberOfOrders |
+------------+-----------+----------+----------------+
|          1 | John      | Doe      |              1 |
|          2 | Jane      | Smith    |              1 |
|          3 | Robert    | Johnson  |              1 |
|          4 | Sarah     | Brown    |              1 |
|          5 | David     | Lee      |              1 |
|          6 | Laura     | Hall     |              1 |
|          7 | Michael   | Davis    |              1 |
|          8 | Emma      | Wilson   |              1 |
|          9 | William   | Taylor   |              1 |
|         10 | Olivia    | Adams    |              1 |
+------------+-----------+----------+----------------+
10 rows in set (0.00 sec)
```

-- 13) Find the Maximum Order Amount for Each Customer.

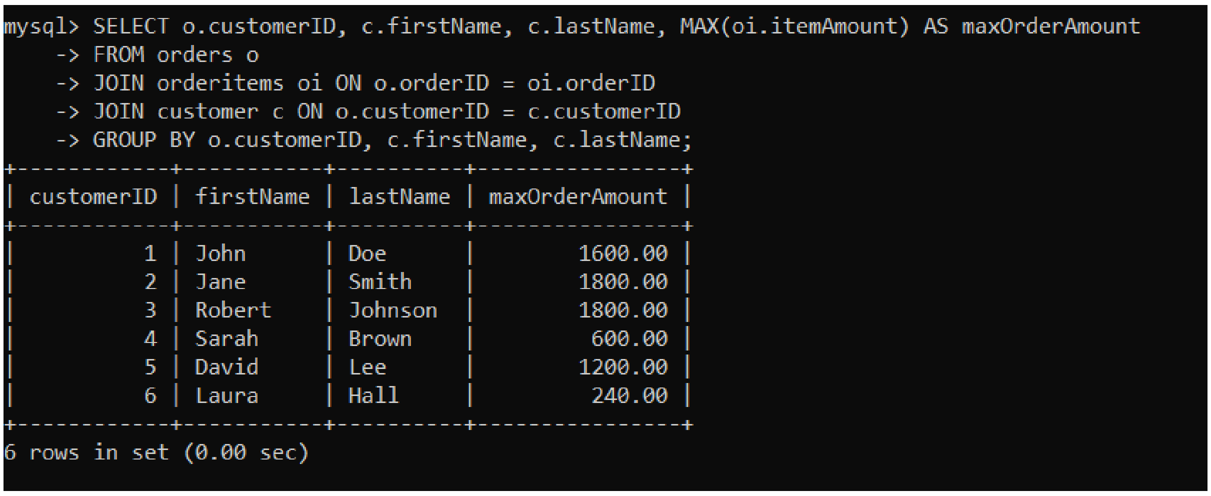
SELECT o.customerID, c.firstName, c.lastName, MAX(oi.itemAmount) AS maxOrderAmount

FROM orders o

JOIN orderitems oi ON o.orderID = oi.orderID

JOIN customer c ON o.customerID = c.customerID

GROUP BY o.customerID, c.firstName, c.lastName;



```
mysql> SELECT o.customerID, c.firstName, c.lastName, MAX(oi.itemAmount) AS maxOrderAmount
    -> FROM orders o
    -> JOIN orderitems oi ON o.orderID = oi.orderID
    -> JOIN customer c ON o.customerID = c.customerID
    -> GROUP BY o.customerID, c.firstName, c.lastName;
+------------+-----------+----------+----------------+
| customerID | firstName | lastName | maxOrderAmount |
+------------+-----------+----------+----------------+
|          1 | John      | Doe      |        1600.00 |
|          2 | Jane      | Smith    |        1800.00 |
|          3 | Robert    | Johnson  |        1800.00 |
|          4 | Sarah     | Brown    |         600.00 |
|          5 | David     | Lee      |        1200.00 |
|          6 | Laura     | Hall     |         240.00 |
+------------+-----------+----------+----------------+
6 rows in set (0.00 sec)
```

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

SELECT c.customerID, c.firstName, c.lastName

FROM customer c

JOIN orders o ON c.customerID = o.customerID

JOIN (

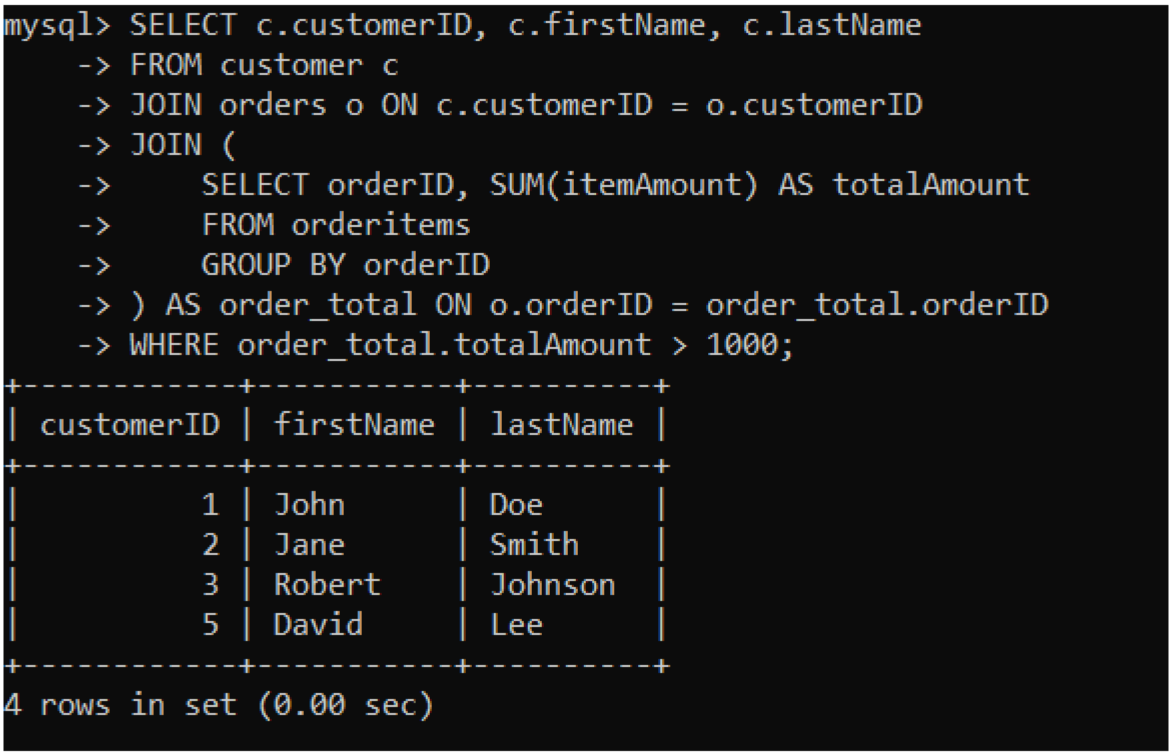
    SELECT orderID, SUM(itemAmount) AS totalAmount

    FROM orderitems

    GROUP BY orderID

) AS order_total ON o.orderID = order_total.orderID

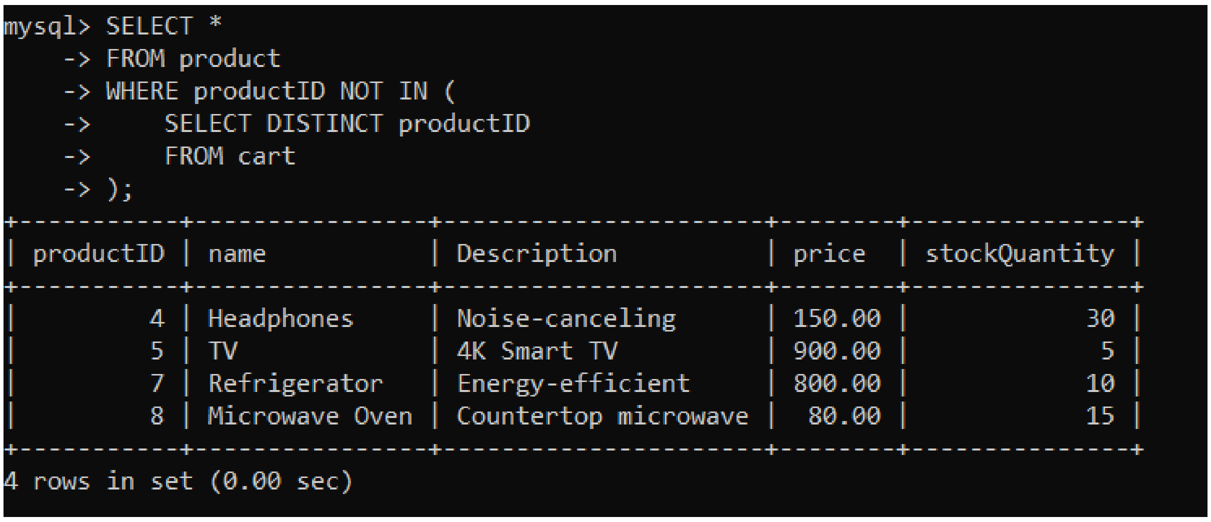
WHERE order_total.totalAmount > 1000;



```
mysql> SELECT c.customerID, c.firstName, c.lastName
    -> FROM customer c
    -> JOIN orders o ON c.customerID = o.customerID
    -> JOIN (
    ->     SELECT orderID, SUM(itemAmount) AS totalAmount
    ->     FROM orderitems
    ->     GROUP BY orderID
    -> ) AS order_total ON o.orderID = order_total.orderID
    -> WHERE order_total.totalAmount > 1000;
+------------+-----------+----------+
| customerID | firstName | lastName |
+------------+-----------+----------+
|          1 | John      | Doe      |
|          2 | Jane      | Smith    |
|          3 | Robert    | Johnson  |
|          5 | David     | Lee      |
+------------+-----------+----------+
4 rows in set (0.00 sec)
```

-- 15) Subquery to Find Products Not in the Cart.

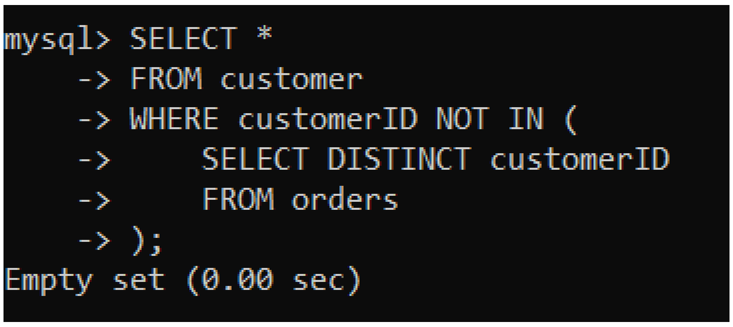
```
SELECT *
FROM product
WHERE productID NOT IN (
   SELECT DISTINCT productID
   FROM cart
);
```



```
mysql> SELECT *
    -> FROM product
    -> WHERE productID NOT IN (
    ->     SELECT DISTINCT productID
    ->     FROM cart
    -> );
+-----------+----------------+----------------------+--------+---------------+
| productID | name           | Description          | price  | stockQuantity |
+-----------+----------------+----------------------+--------+---------------+
|         4 | Headphones     | Noise-canceling      | 150.00 |            30 |
|         5 | TV             | 4K Smart TV          | 900.00 |             5 |
|         7 | Refrigerator   | Energy-efficient     | 800.00 |            10 |
|         8 | Microwave Oven | Countertop microwave |  80.00 |            15 |
+-----------+----------------+----------------------+--------+---------------+
4 rows in set (0.00 sec)
```

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

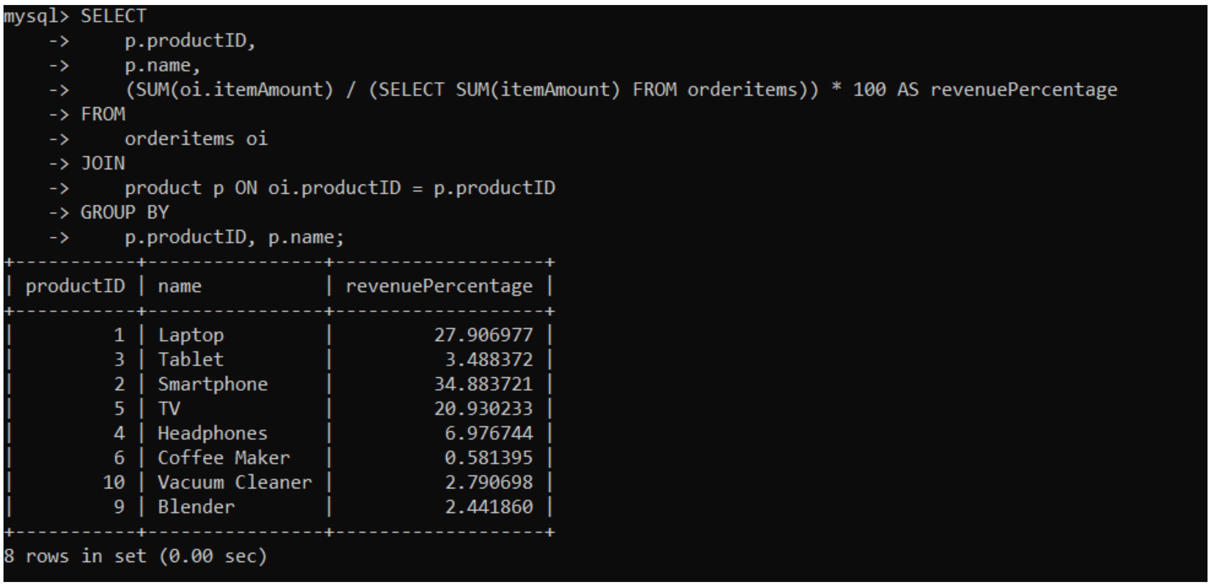
```
SELECT *
FROM customer
WHERE customerID NOT IN (
    SELECT DISTINCT customerID
    FROM orders
);
```



```
mysql> SELECT *
    -> FROM customer
    -> WHERE customerID NOT IN (
    ->     SELECT DISTINCT customerID
    ->     FROM orders
    -> );
Empty set (0.00 sec)
```

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

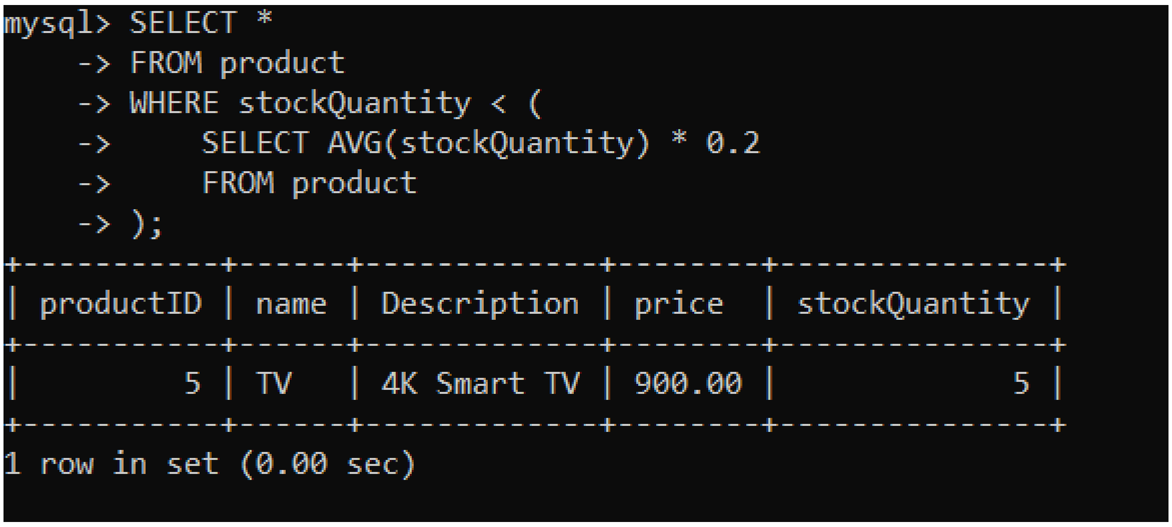
```
SELECT
    p.productID,
    p.name,
    (SUM(oi.itemAmount) / (SELECT SUM(itemAmount) FROM orderitems)) * 100 AS
revenuePercentage
FROM
    orderitems oi
JOIN
    product p ON oi.productID = p.productID
GROUP BY
    p.productID, p.name;
```



```
mysql> SELECT
    ->     p.productID,
    ->     p.name,
    ->     (SUM(oi.itemAmount) / (SELECT SUM(itemAmount) FROM orderitems)) * 100 AS revenuePercentage
    -> FROM
    ->     orderitems oi
    -> JOIN
    ->     product p ON oi.productID = p.productID
    -> GROUP BY
    ->     p.productID, p.name;
+-----------+----------------+-------------------+
| productID | name           | revenuePercentage |
+-----------+----------------+-------------------+
|         1 | Laptop         |         27.906977 |
|         3 | Tablet         |          3.488372 |
|         2 | Smartphone     |         34.883721 |
|         5 | TV             |         20.930233 |
|         4 | Headphones     |          6.976744 |
|         6 | Coffee Maker   |          0.581395 |
|        10 | Vacuum Cleaner |          2.790698 |
|         9 | Blender        |          2.441860 |
+-----------+----------------+-------------------+
8 rows in set (0.00 sec)
```

-- 18) Subquery to Find Products with Low Stock.

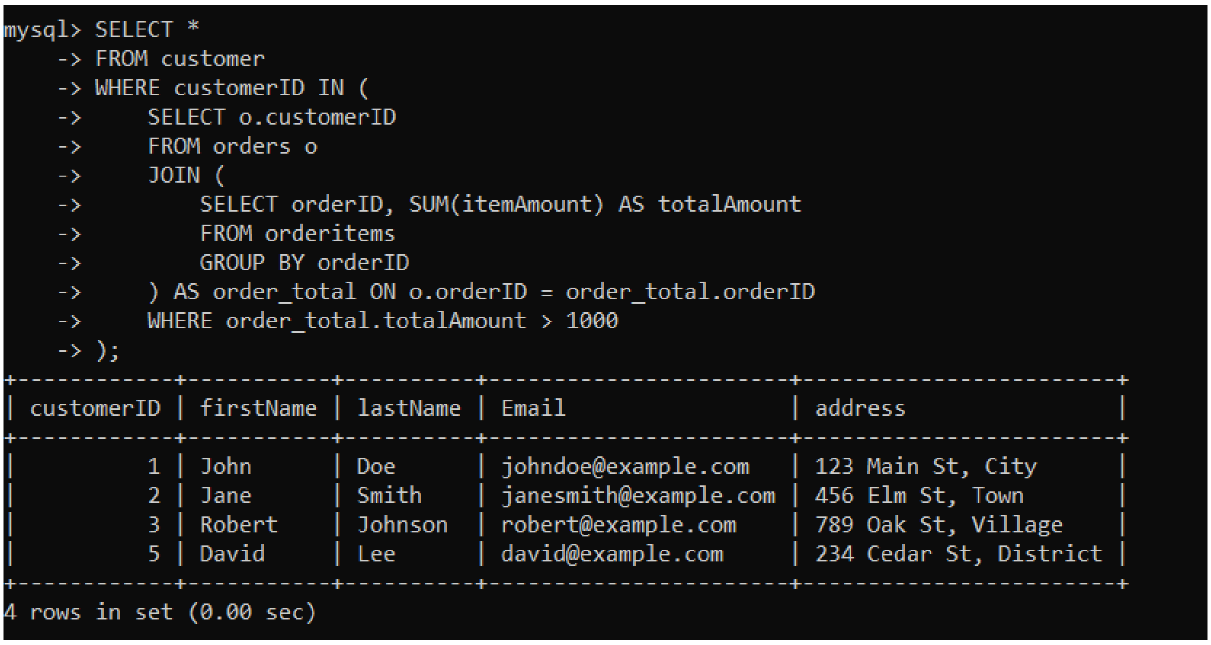
```
SELECT *
FROM product
WHERE stockQuantity < (
   SELECT AVG(stockQuantity) * 0.2
   FROM product
);
```



```
mysql> SELECT *
    -> FROM product
    -> WHERE stockQuantity < (
    ->     SELECT AVG(stockQuantity) * 0.2
    ->     FROM product
    -> );
+-----------+------+-------------+--------+---------------+
| productID | name | Description | price  | stockQuantity |
+-----------+------+-------------+--------+---------------+
|         5 | TV   | 4K Smart TV | 900.00 |             5 |
+-----------+------+-------------+--------+---------------+
1 row in set (0.00 sec)
```

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

```
SELECT *
FROM customer
WHERE customerID IN (
   SELECT o.customerID
   FROM orders o
   JOIN (
      SELECT orderID, SUM(itemAmount) AS totalAmount
      FROM orderitems
      GROUP BY orderID
   ) AS order_total ON o.orderID = order_total.orderID
   WHERE order_total.totalAmount > 1000
);
```



```
mysql> SELECT *
    -> FROM customer
    -> WHERE customerID IN (
    ->     SELECT o.customerID
    ->     FROM orders o
    ->     JOIN (
    ->         SELECT orderID, SUM(itemAmount) AS totalAmount
    ->         FROM orderitems
    ->         GROUP BY orderID
    ->     ) AS order_total ON o.orderID = order_total.orderID
    ->     WHERE order_total.totalAmount > 1000
    -> );
+------------+-----------+----------+-----------------------+------------------------+
| customerID | firstName | lastName | 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    |
|          5 | David     | Lee      | david@example.com     | 234 Cedar St, District |
+------------+-----------+----------+-----------------------+------------------------+
4 rows in set (0.00 sec)
```