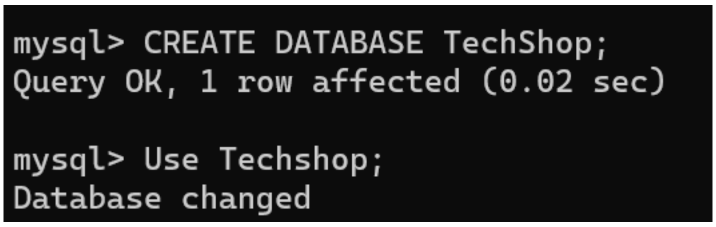
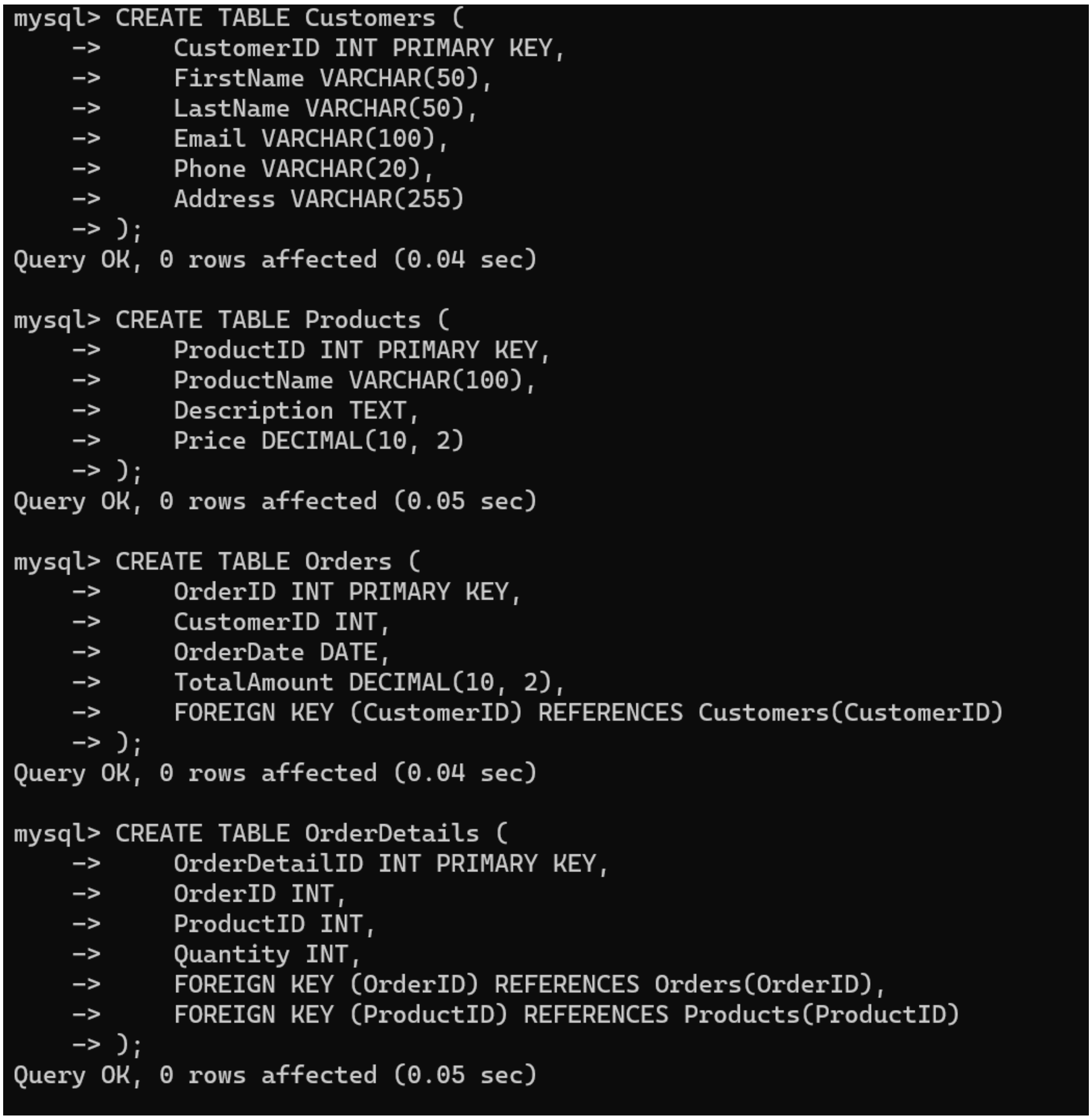
SQL ASSIGNMENT 1

Creating DataBase

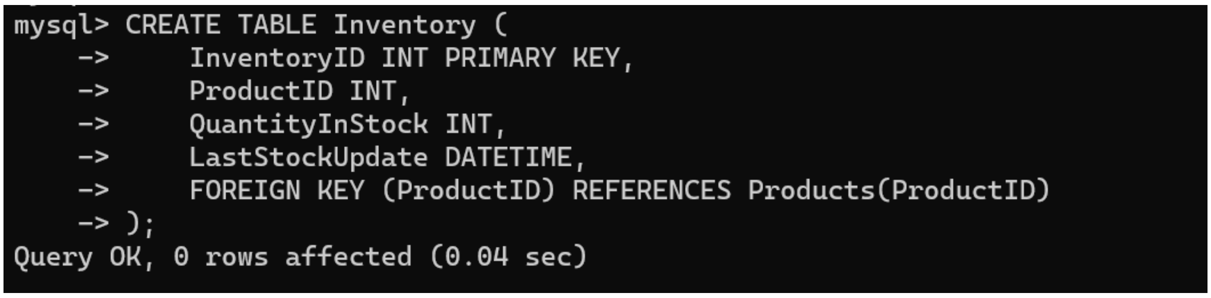


```
mysql> CREATE DATABASE TechShop;
Query OK, 1 row affected (0.02 sec)

mysql> Use Techshop;
Database changed
```

Creating Tables



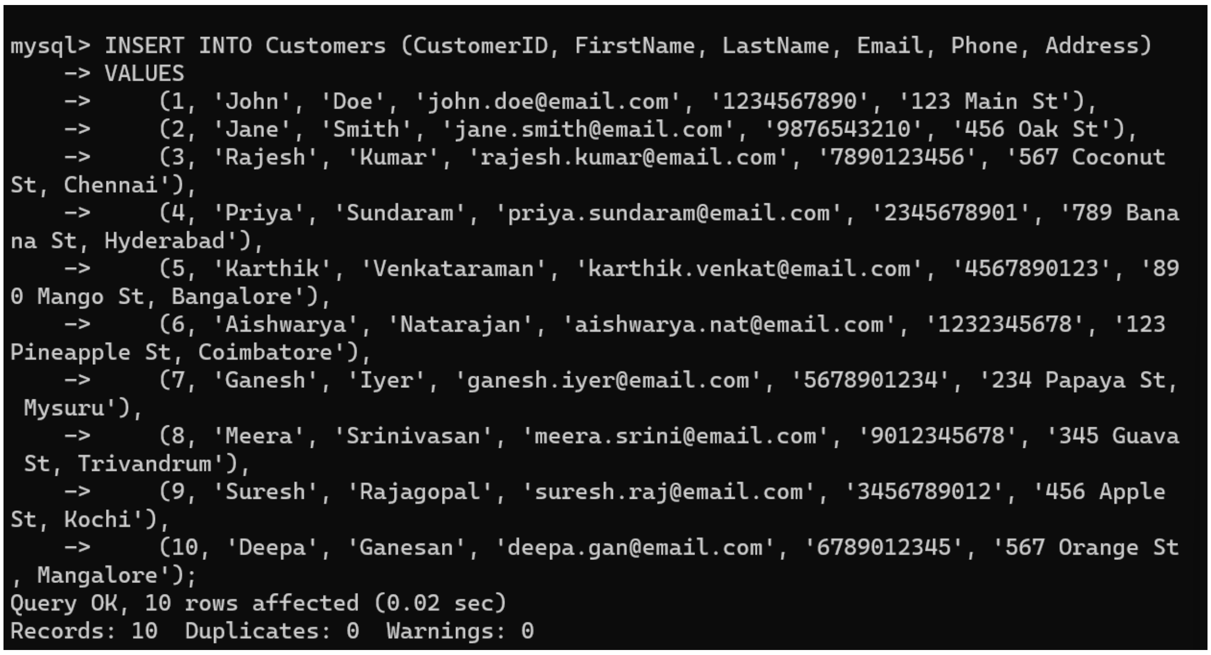
```
mysql> CREATE TABLE Customers (
    ->     CustomerID INT PRIMARY KEY,
    ->     FirstName VARCHAR(50),
    ->     LastName VARCHAR(50),
    ->     Email VARCHAR(100),
    ->     Phone VARCHAR(20),
    ->     Address VARCHAR(255)
    -> );
Query OK, 0 rows affected (0.04 sec)

mysql> CREATE TABLE Products (
    ->     ProductID INT PRIMARY KEY,
    ->     ProductName VARCHAR(100),
    ->     Description TEXT,
    ->     Price DECIMAL(10, 2)
    -> );
Query OK, 0 rows affected (0.05 sec)

mysql> CREATE TABLE Orders (
    ->     OrderID INT PRIMARY KEY,
    ->     CustomerID INT,
    ->     OrderDate DATE,
    ->     TotalAmount DECIMAL(10, 2),
    ->     FOREIGN KEY (CustomerID) REFERENCES Customers(CustomerID)
    -> );
Query OK, 0 rows affected (0.04 sec)

mysql> CREATE TABLE OrderDetails (
    ->     OrderDetailID INT PRIMARY KEY,
    ->     OrderID INT,
    ->     ProductID INT,
    ->     Quantity INT,
    ->     FOREIGN KEY (OrderID) REFERENCES Orders(OrderID),
    ->     FOREIGN KEY (ProductID) REFERENCES Products(ProductID)
    -> );
Query OK, 0 rows affected (0.05 sec)
```



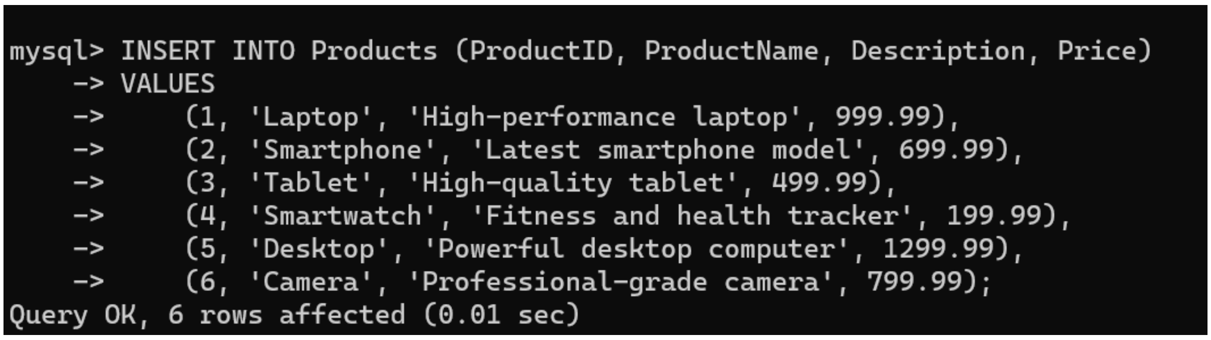
```
mysql> CREATE TABLE Inventory (
    ->     InventoryID INT PRIMARY KEY,
    ->     ProductID INT,
    ->     QuantityInStock INT,
    ->     LastStockUpdate DATETIME,
    ->     FOREIGN KEY (ProductID) REFERENCES Products(ProductID)
    -> );
Query OK, 0 rows affected (0.04 sec)
```

Task 1:

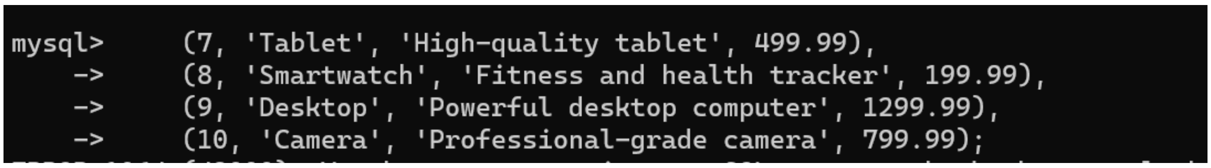
Inserting Values



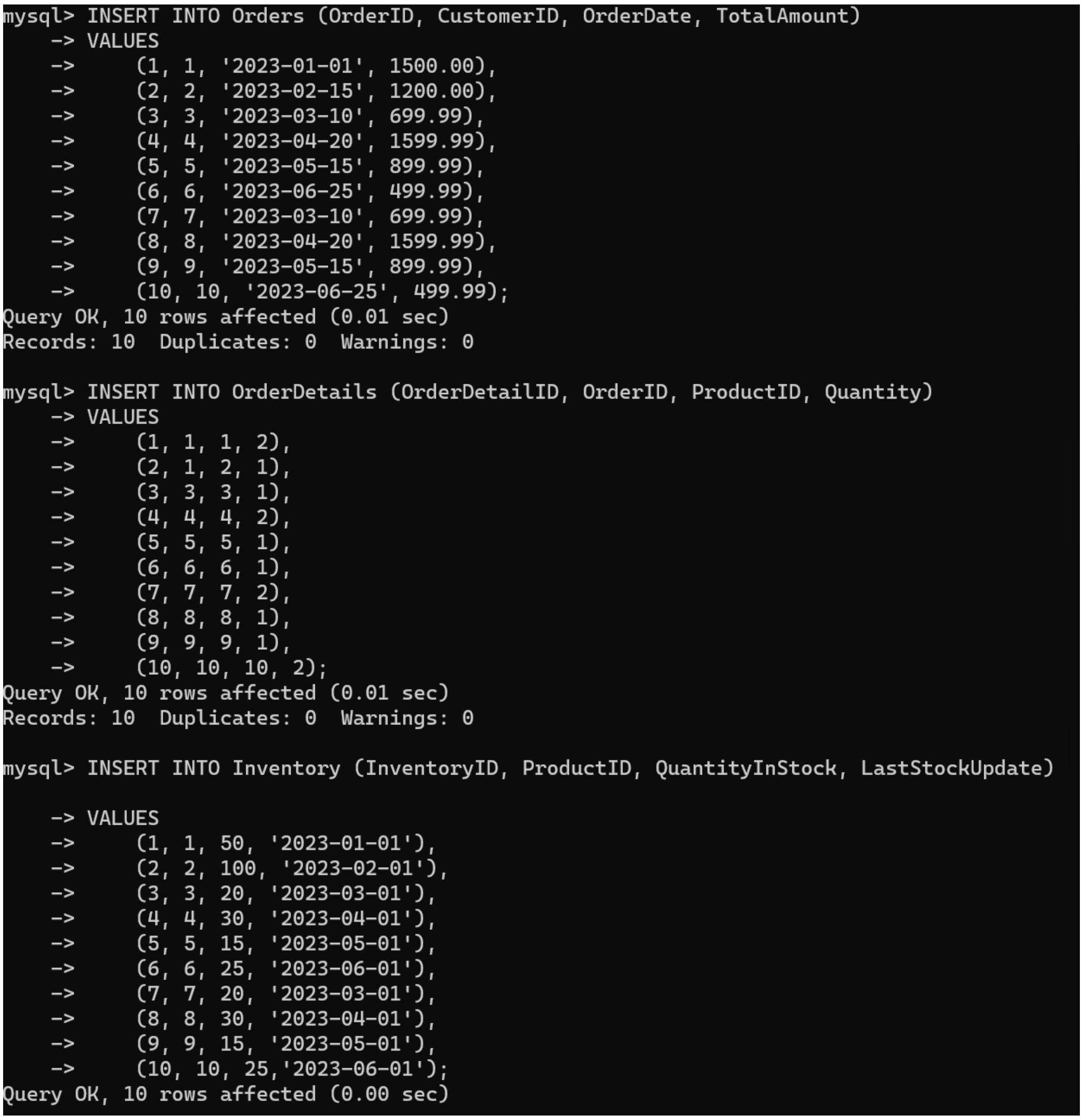
```
mysql> INSERT INTO Customers (CustomerID, FirstName, LastName, Email, Phone, Address)
    -> VALUES
    ->     (1, 'John', 'Doe', 'john.doe@email.com', '1234567890', '123 Main St'),
    ->     (2, 'Jane', 'Smith', 'jane.smith@email.com', '9876543210', '456 Oak St'),
    ->     (3, 'Rajesh', 'Kumar', 'rajesh.kumar@email.com', '7890123456', '567 Coconut
St, Chennai'),
    ->     (4, 'Priya', 'Sundaram', 'priya.sundaram@email.com', '2345678901', '789 Bana
na St, Hyderabad'),
    ->     (5, 'Karthik', 'Venkataraman', 'karthik.venkat@email.com', '4567890123', '89
0 Mango St, Bangalore'),
    ->     (6, 'Aishwarya', 'Natarajan', 'aishwarya.nat@email.com', '1232345678', '123
Pineapple St, Coimbatore'),
    ->     (7, 'Ganesh', 'Iyer', 'ganesh.iyer@email.com', '5678901234', '234 Papaya St,
 Mysuru'),
    ->     (8, 'Meera', 'Srinivasan', 'meera.srini@email.com', '9012345678', '345 Guava
 St, Trivandrum'),
    ->     (9, 'Suresh', 'Rajagopal', 'suresh.raj@email.com', '3456789012', '456 Apple
St, Kochi'),
    ->     (10, 'Deepa', 'Ganesan', 'deepa.gan@email.com', '6789012345', '567 Orange St
, Mangalore');
Query OK, 10 rows affected (0.02 sec)
Records: 10  Duplicates: 0  Warnings: 0
```



```
mysql> INSERT INTO Products (ProductID, ProductName, Description, Price)
    -> VALUES
    ->     (1, 'Laptop', 'High-performance laptop', 999.99),
    ->     (2, 'Smartphone', 'Latest smartphone model', 699.99),
    ->     (3, 'Tablet', 'High-quality tablet', 499.99),
    ->     (4, 'Smartwatch', 'Fitness and health tracker', 199.99),
    ->     (5, 'Desktop', 'Powerful desktop computer', 1299.99),
    ->     (6, 'Camera', 'Professional-grade camera', 799.99);
Query OK, 6 rows affected (0.01 sec)
```



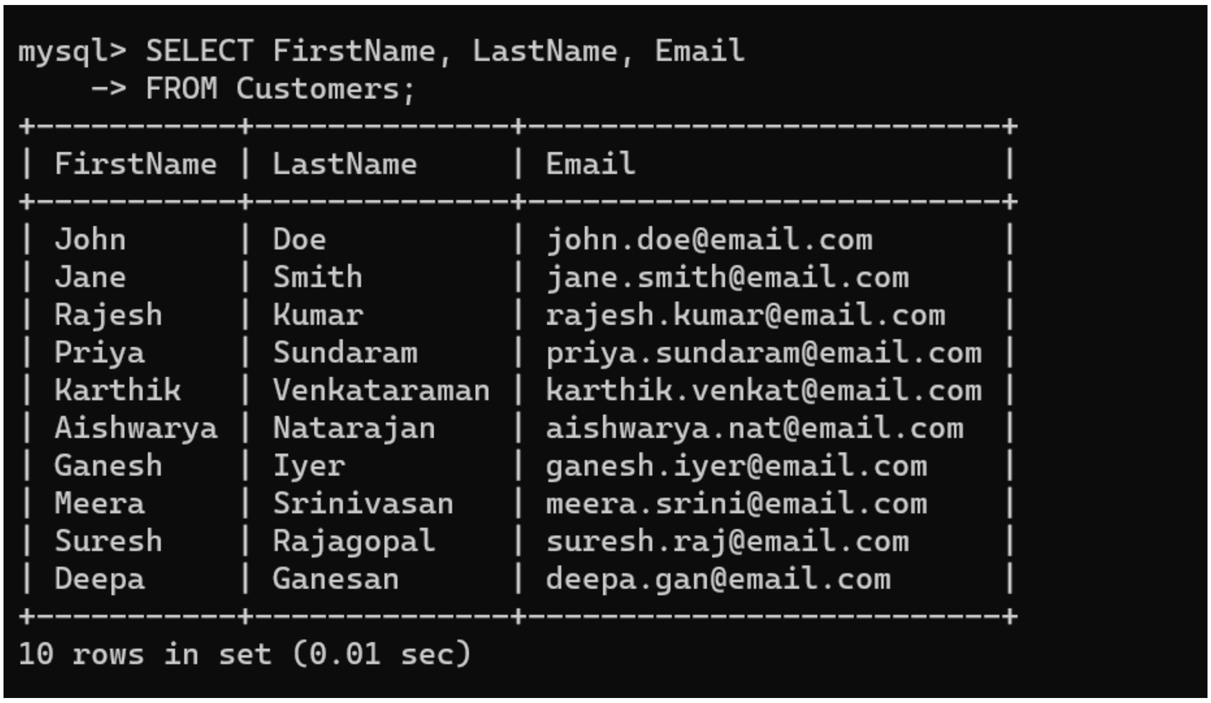
```
mysql>     (7, 'Tablet', 'High-quality tablet', 499.99),
    ->     (8, 'Smartwatch', 'Fitness and health tracker', 199.99),
    ->     (9, 'Desktop', 'Powerful desktop computer', 1299.99),
    ->     (10, 'Camera', 'Professional-grade camera', 799.99);
```



```
mysql> INSERT INTO Orders (OrderID, CustomerID, OrderDate, TotalAmount)
    -> VALUES
    ->     (1, 1, '2023-01-01', 1500.00),
    ->     (2, 2, '2023-02-15', 1200.00),
    ->     (3, 3, '2023-03-10', 699.99),
    ->     (4, 4, '2023-04-20', 1599.99),
    ->     (5, 5, '2023-05-15', 899.99),
    ->     (6, 6, '2023-06-25', 499.99),
    ->     (7, 7, '2023-03-10', 699.99),
    ->     (8, 8, '2023-04-20', 1599.99),
    ->     (9, 9, '2023-05-15', 899.99),
    ->     (10, 10, '2023-06-25', 499.99);
Query OK, 10 rows affected (0.01 sec)
Records: 10  Duplicates: 0  Warnings: 0

mysql> INSERT INTO OrderDetails (OrderDetailID, OrderID, ProductID, Quantity)
    -> VALUES
    ->     (1, 1, 1, 2),
    ->     (2, 1, 2, 1),
    ->     (3, 3, 3, 1),
    ->     (4, 4, 4, 2),
    ->     (5, 5, 5, 1),
    ->     (6, 6, 6, 1),
    ->     (7, 7, 7, 2),
    ->     (8, 8, 8, 1),
    ->     (9, 9, 9, 1),
    ->     (10, 10, 10, 2);
Query OK, 10 rows affected (0.01 sec)
Records: 10  Duplicates: 0  Warnings: 0

mysql> INSERT INTO Inventory (InventoryID, ProductID, QuantityInStock, LastStockUpdate)

    -> VALUES
    ->     (1, 1, 50, '2023-01-01'),
    ->     (2, 2, 100, '2023-02-01'),
    ->     (3, 3, 20, '2023-03-01'),
    ->     (4, 4, 30, '2023-04-01'),
    ->     (5, 5, 15, '2023-05-01'),
    ->     (6, 6, 25, '2023-06-01'),
    ->     (7, 7, 20, '2023-03-01'),
    ->     (8, 8, 30, '2023-04-01'),
    ->     (9, 9, 15, '2023-05-01'),
    ->     (10, 10, 25,'2023-06-01');
Query OK, 10 rows affected (0.00 sec)
```

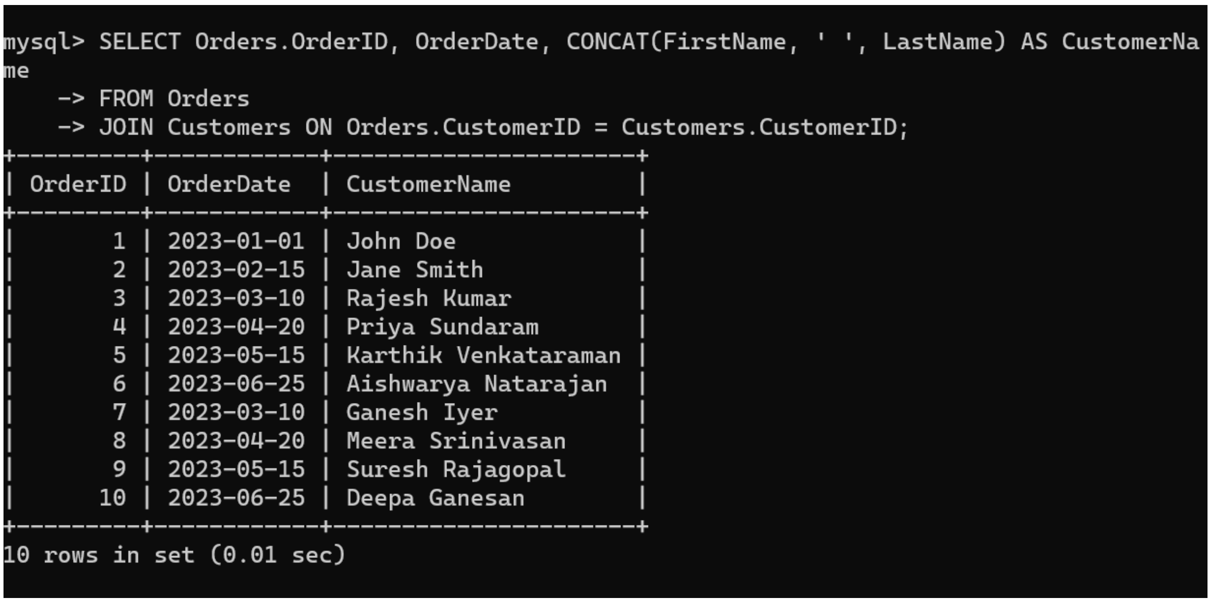
Task 2:

1. Write an SQL query to retrieve the names and emails of all customers



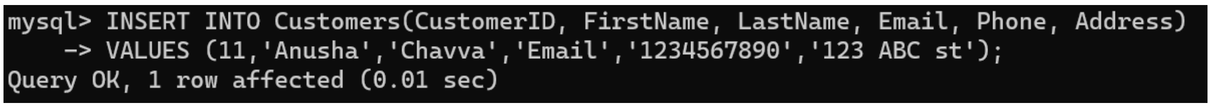
```
mysql> SELECT FirstName, LastName, Email
    -> FROM Customers;
+-----------+--------------+---------------------------+
| FirstName | LastName     | Email                     |
+-----------+--------------+---------------------------+
| John      | Doe          | john.doe@email.com        |
| Jane      | Smith        | jane.smith@email.com      |
| Rajesh    | Kumar        | rajesh.kumar@email.com    |
| Priya     | Sundaram     | priya.sundaram@email.com  |
| Karthik   | Venkataraman | karthik.venkat@email.com  |
| Aishwarya | Natarajan    | aishwarya.nat@email.com   |
| Ganesh    | Iyer         | ganesh.iyer@email.com     |
| Meera     | Srinivasan   | meera.srini@email.com     |
| Suresh    | Rajagopal    | suresh.raj@email.com      |
| Deepa     | Ganesan      | deepa.gan@email.com       |
+-----------+--------------+---------------------------+
10 rows in set (0.01 sec)
```

2. Write an SQL query to list all orders with their order dates and corresponding customer names



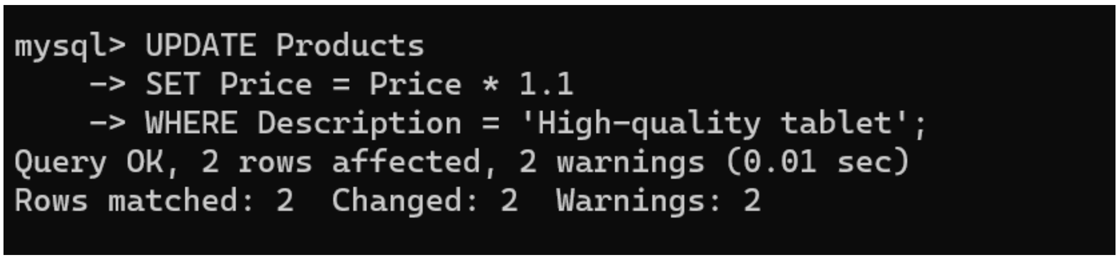
```
mysql> SELECT Orders.OrderID, OrderDate, CONCAT(FirstName, ' ', LastName) AS CustomerNa
me
    -> FROM Orders
    -> JOIN Customers ON Orders.CustomerID = Customers.CustomerID;
+---------+------------+----------------------+
| OrderID | OrderDate  | CustomerName         |
+---------+------------+----------------------+
|       1 | 2023-01-01 | John Doe             |
|       2 | 2023-02-15 | Jane Smith           |
|       3 | 2023-03-10 | Rajesh Kumar         |
|       4 | 2023-04-20 | Priya Sundaram       |
|       5 | 2023-05-15 | Karthik Venkataraman |
|       6 | 2023-06-25 | Aishwarya Natarajan  |
|       7 | 2023-03-10 | Ganesh Iyer          |
|       8 | 2023-04-20 | Meera Srinivasan     |
|       9 | 2023-05-15 | Suresh Rajagopal     |
|      10 | 2023-06-25 | Deepa Ganesan        |
+---------+------------+----------------------+
10 rows in set (0.01 sec)
```

3. Write an SQL query to insert a new customer record into the "Customers" table. Include customer information such as name, email, and address.



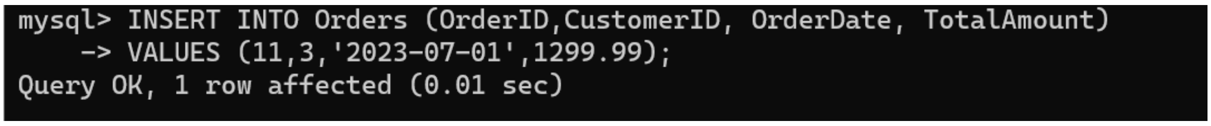
```
mysql> INSERT INTO Customers(CustomerID, FirstName, LastName, Email, Phone, Address)
    -> VALUES (11,'Anusha','Chavva','Email','1234567890','123 ABC st');
Query OK, 1 row affected (0.01 sec)
```

4. Write an SQL query to update the prices of all electronic gadgets in the "Products" table by Increasing them by 10%.



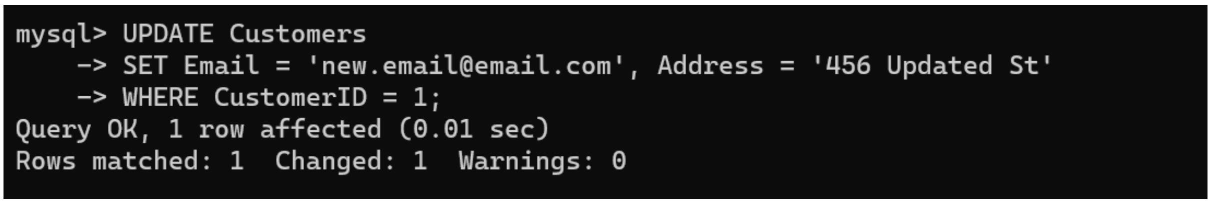
```
mysql> UPDATE Products
    -> SET Price = Price * 1.1
    -> WHERE Description = 'High-quality tablet';
Query OK, 2 rows affected, 2 warnings (0.01 sec)
Rows matched: 2  Changed: 2  Warnings: 2
```

5. Write an SQL query to delete a specific order and its associated order details from the "Orders" and "OrderDetails" tables. Allow users to input the order ID as a parameter.



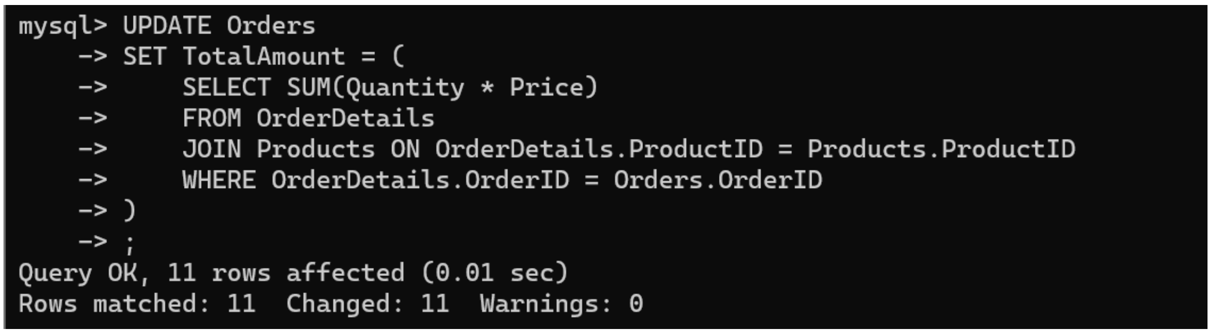
```
mysql> INSERT INTO Orders (OrderID,CustomerID, OrderDate, TotalAmount)
    -> VALUES (11,3,'2023-07-01',1299.99);
Query OK, 1 row affected (0.01 sec)
```

6. Write an SQL query to update the contact information (e.g., email and address) of a specific customer in the "Customers" table. Allow users to input the customer ID and new contact information.



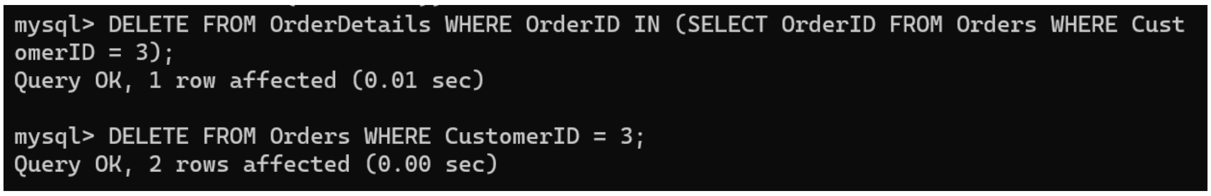
```
mysql> UPDATE Customers
    -> SET Email = 'new.email@email.com', Address = '456 Updated St'
    -> WHERE CustomerID = 1;
Query OK, 1 row affected (0.01 sec)
Rows matched: 1  Changed: 1  Warnings: 0
```

7. Write an SQL query to recalculate and update the total cost of each order in the "Orders" table based on the prices and quantities in the "OrderDetails" table.



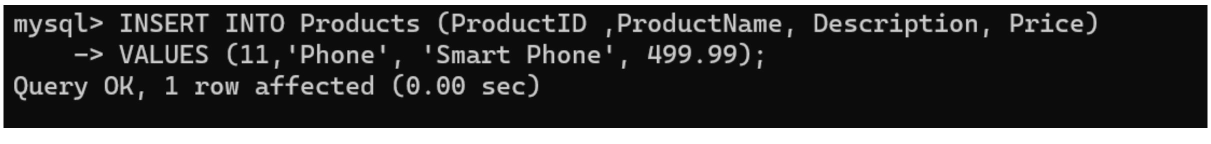
```
mysql> UPDATE Orders
    -> SET TotalAmount = (
    ->     SELECT SUM(Quantity * Price)
    ->     FROM OrderDetails
    ->     JOIN Products ON OrderDetails.ProductID = Products.ProductID
    ->     WHERE OrderDetails.OrderID = Orders.OrderID
    -> )
    -> ;
Query OK, 11 rows affected (0.01 sec)
Rows matched: 11  Changed: 11  Warnings: 0
```

8. Write an SQL query to delete all orders and their associated order details for a specific customer from the "Orders" and "OrderDetails" tables. Allow users to input the customer ID as a parameter.



```
mysql> DELETE FROM OrderDetails WHERE OrderID IN (SELECT OrderID FROM Orders WHERE Cust
omerID = 3);
Query OK, 1 row affected (0.01 sec)

mysql> DELETE FROM Orders WHERE CustomerID = 3;
Query OK, 2 rows affected (0.00 sec)
```

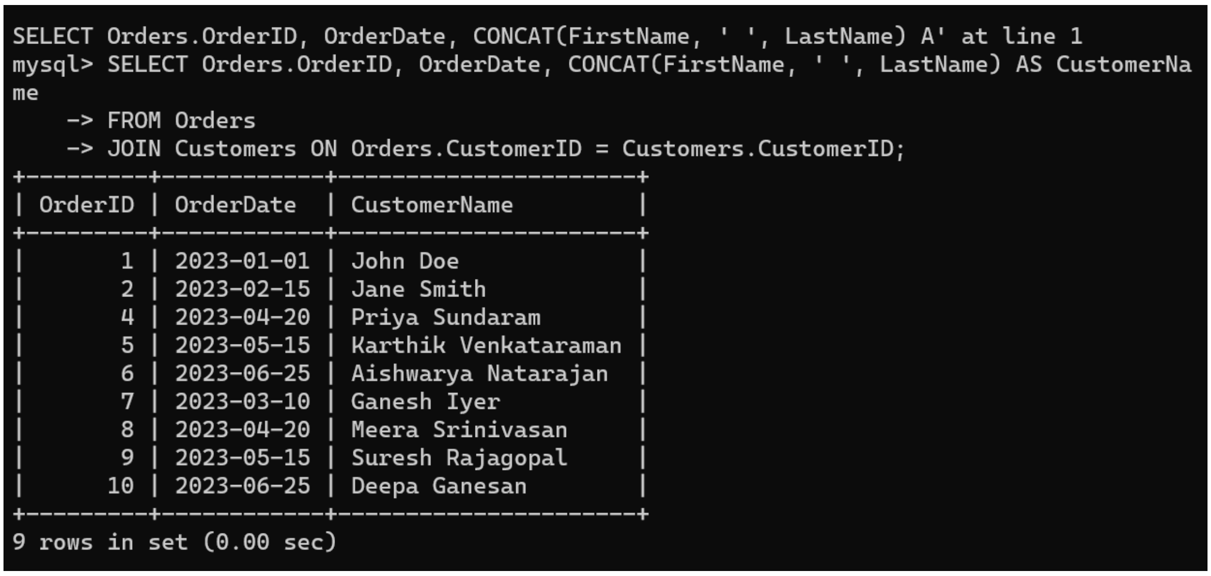
9. Write an SQL query to insert a new electronic gadget product into the "Products" table.



```
mysql> INSERT INTO Products (ProductID ,ProductName, Description, Price)
    -> VALUES (11,'Phone', 'Smart Phone', 499.99);
Query OK, 1 row affected (0.00 sec)
```

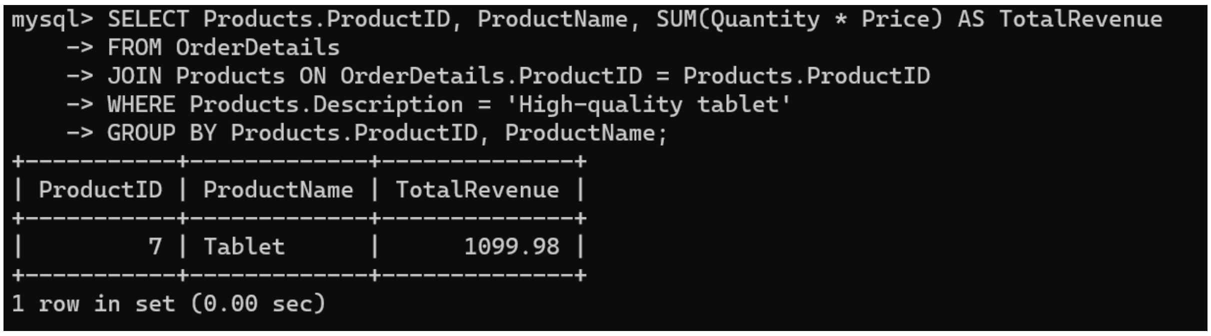
Task 3:

1. Write an SQL query to retrieve a list of all orders along with customer information (e.g.. customer name) for each order.



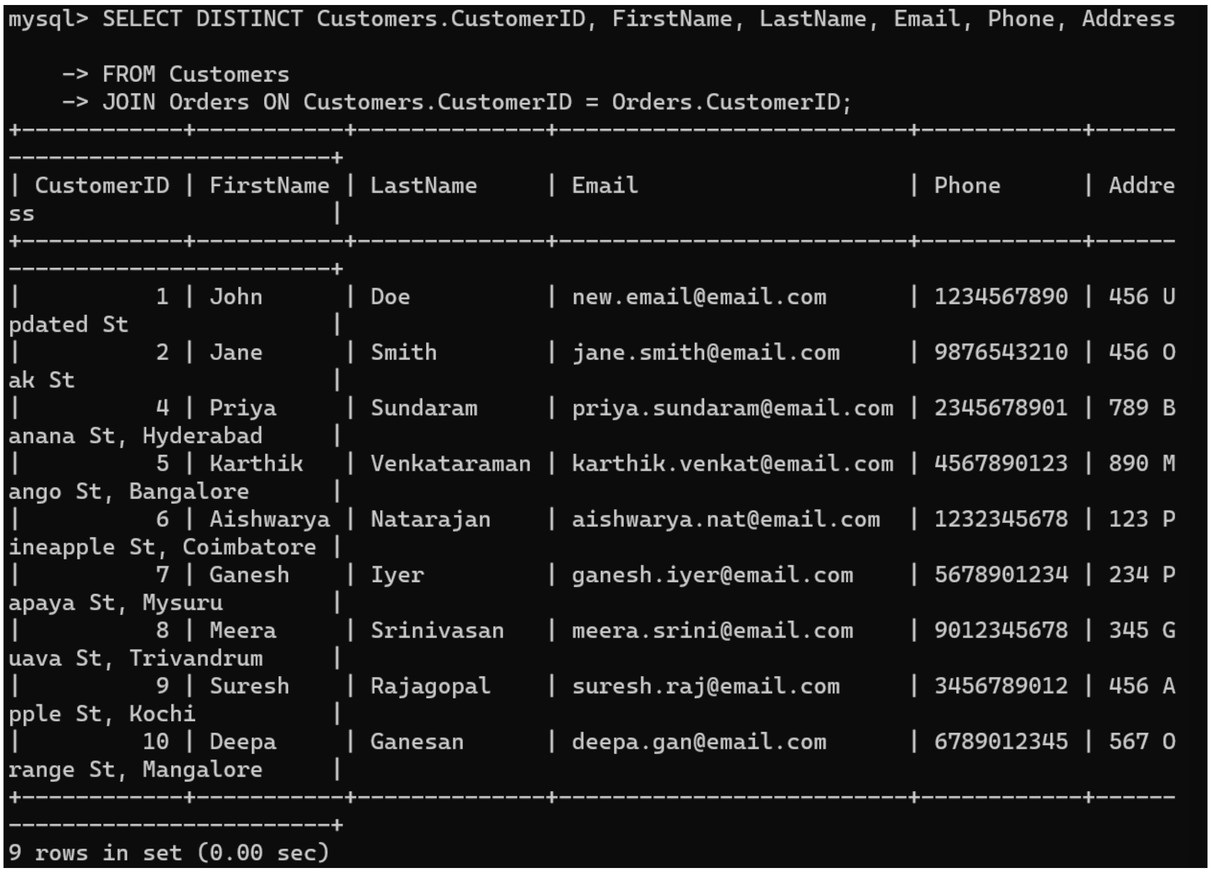
```
SELECT Orders.OrderID, OrderDate, CONCAT(FirstName, ' ', LastName) A' at line 1
mysql> SELECT Orders.OrderID, OrderDate, CONCAT(FirstName, ' ', LastName) AS CustomerNa
me
    -> FROM Orders
    -> JOIN Customers ON Orders.CustomerID = Customers.CustomerID;
+---------+------------+----------------------+
| OrderID | OrderDate  | CustomerName         |
+---------+------------+----------------------+
|       1 | 2023-01-01 | John Doe             |
|       2 | 2023-02-15 | Jane Smith           |
|       4 | 2023-04-20 | Priya Sundaram       |
|       5 | 2023-05-15 | Karthik Venkataraman |
|       6 | 2023-06-25 | Aishwarya Natarajan  |
|       7 | 2023-03-10 | Ganesh Iyer          |
|       8 | 2023-04-20 | Meera Srinivasan     |
|       9 | 2023-05-15 | Suresh Rajagopal     |
|      10 | 2023-06-25 | Deepa Ganesan        |
+---------+------------+----------------------+
9 rows in set (0.00 sec)
```

2. Write an SQL query to find the total revenue generated by each electronic gadget product. Include the product name and the total revenue.



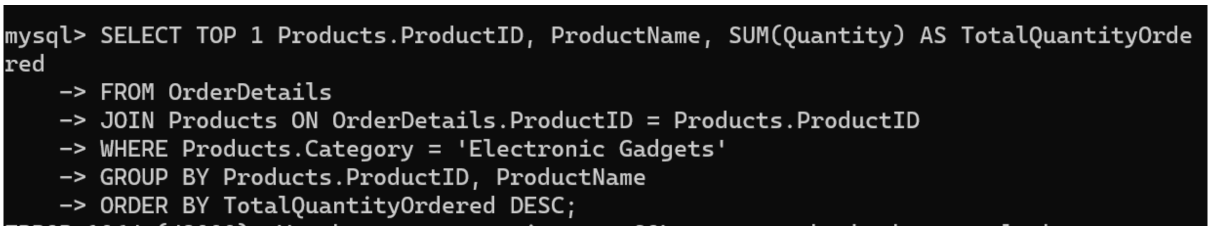
```
mysql> SELECT Products.ProductID, ProductName, SUM(Quantity * Price) AS TotalRevenue
    -> FROM OrderDetails
    -> JOIN Products ON OrderDetails.ProductID = Products.ProductID
    -> WHERE Products.Description = 'High-quality tablet'
    -> GROUP BY Products.ProductID, ProductName;
+-----------+-------------+--------------+
| ProductID | ProductName | TotalRevenue |
+-----------+-------------+--------------+
|         7 | Tablet      |      1099.98 |
+-----------+-------------+--------------+
1 row in set (0.00 sec)
```

3. Write an SQL query to list all customers who have made at least one purchase. Include their names and contact information.



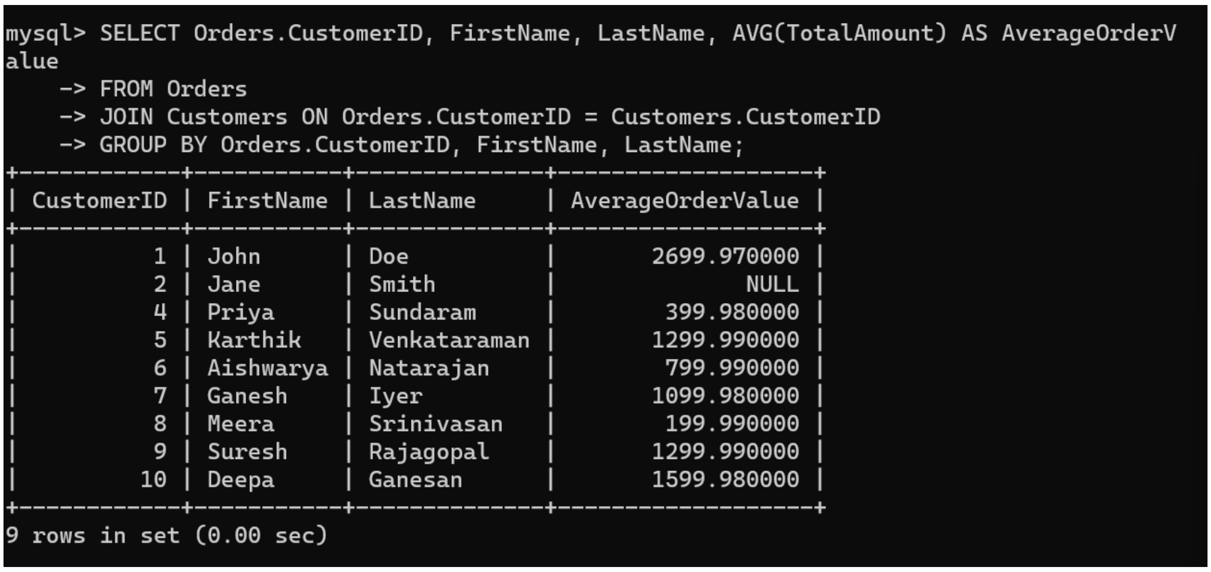
```
mysql> SELECT DISTINCT Customers.CustomerID, FirstName, LastName, Email, Phone, Address

    -> FROM Customers
    -> JOIN Orders ON Customers.CustomerID = Orders.CustomerID;
+------------+-----------+--------------+--------------------------+------------+------
------------------------+
| CustomerID | FirstName | LastName     | Email                    | Phone      | Addre
ss                      |
+------------+-----------+--------------+--------------------------+------------+------
------------------------+
|          1 | John      | Doe          | new.email@email.com      | 1234567890 | 456 U
pdated St               |
|          2 | Jane      | Smith        | jane.smith@email.com     | 9876543210 | 456 O
ak St                   |
|          4 | Priya     | Sundaram     | priya.sundaram@email.com | 2345678901 | 789 B
anana St, Hyderabad     |
|          5 | Karthik   | Venkataraman | karthik.venkat@email.com | 4567890123 | 890 M
ango St, Bangalore      |
|          6 | Aishwarya | Natarajan    | aishwarya.nat@email.com  | 1232345678 | 123 P
ineapple St, Coimbatore |
|          7 | Ganesh    | Iyer         | ganesh.iyer@email.com    | 5678901234 | 234 P
apaya St, Mysuru        |
|          8 | Meera     | Srinivasan   | meera.srini@email.com    | 9012345678 | 345 G
uava St, Trivandrum     |
|          9 | Suresh    | Rajagopal    | suresh.raj@email.com     | 3456789012 | 456 A
pple St, Kochi          |
|         10 | Deepa     | Ganesan      | deepa.gan@email.com      | 6789012345 | 567 O
range St, Mangalore     |
+------------+-----------+--------------+--------------------------+------------+------
------------------------+
9 rows in set (0.00 sec)
```

4. Write an SQL query to find the most popular electronic gadget, which is the one with the highest total quantity ordered. Include the product name and the total quantity ordered.



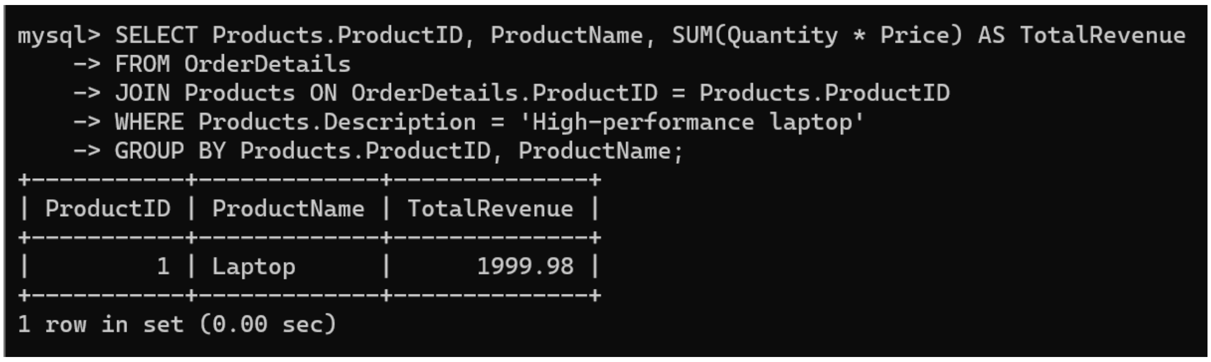
```
mysql> SELECT TOP 1 Products.ProductID, ProductName, SUM(Quantity) AS TotalQuantityOrde
red
    -> FROM OrderDetails
    -> JOIN Products ON OrderDetails.ProductID = Products.ProductID
    -> WHERE Products.Category = 'Electronic Gadgets'
    -> GROUP BY Products.ProductID, ProductName
    -> ORDER BY TotalQuantityOrdered DESC;
```

5. Write an SQL query to retrieve a list of electronic gadgets along with their corresponding categories.



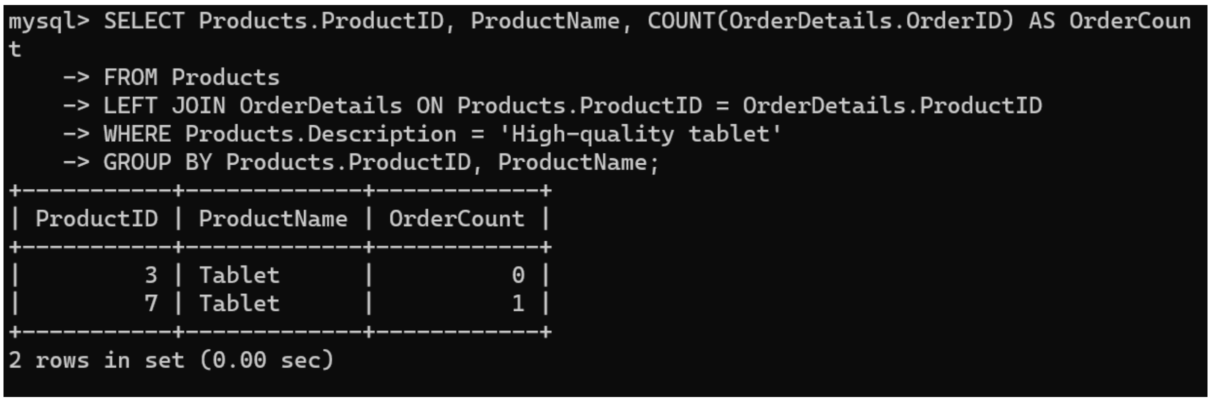
```
mysql> SELECT Orders.CustomerID, FirstName, LastName, AVG(TotalAmount) AS AverageOrderV
alue
    -> FROM Orders
    -> JOIN Customers ON Orders.CustomerID = Customers.CustomerID
    -> GROUP BY Orders.CustomerID, FirstName, LastName;
+------------+-----------+--------------+-------------------+
| CustomerID | FirstName | LastName     | AverageOrderValue |
+------------+-----------+--------------+-------------------+
|          1 | John      | Doe          |       2699.970000 |
|          2 | Jane      | Smith        |              NULL |
|          4 | Priya     | Sundaram     |        399.980000 |
|          5 | Karthik   | Venkataraman |       1299.990000 |
|          6 | Aishwarya | Natarajan    |        799.990000 |
|          7 | Ganesh    | Iyer         |       1099.980000 |
|          8 | Meera     | Srinivasan   |        199.990000 |
|          9 | Suresh    | Rajagopal    |       1299.990000 |
|         10 | Deepa     | Ganesan      |       1599.980000 |
+------------+-----------+--------------+-------------------+
9 rows in set (0.00 sec)
```

6. Write an SQL query to calculate the average order value for each customer. Include the customer's name and their average order value.



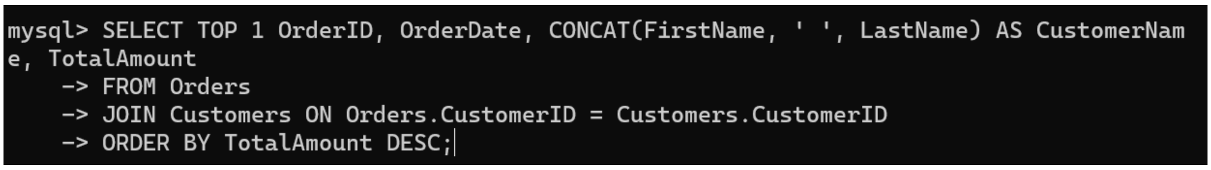
```
mysql> SELECT Products.ProductID, ProductName, SUM(Quantity * Price) AS TotalRevenue
    -> FROM OrderDetails
    -> JOIN Products ON OrderDetails.ProductID = Products.ProductID
    -> WHERE Products.Description = 'High-performance laptop'
    -> GROUP BY Products.ProductID, ProductName;
+-----------+-------------+--------------+
| ProductID | ProductName | TotalRevenue |
+-----------+-------------+--------------+
|         1 | Laptop      |      1999.98 |
+-----------+-------------+--------------+
1 row in set (0.00 sec)
```

7. Write an SQL query to find the order with the highest total revenue. Include the order ID, customer information, and the total revenue.



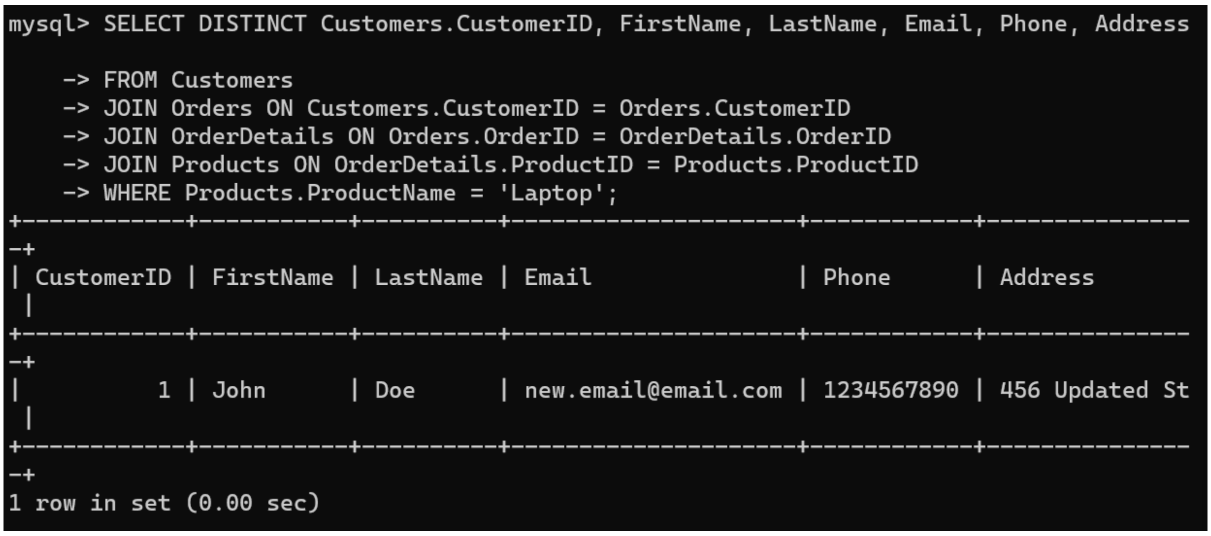
```
mysql> SELECT Products.ProductID, ProductName, COUNT(OrderDetails.OrderID) AS OrderCoun
t
    -> FROM Products
    -> LEFT JOIN OrderDetails ON Products.ProductID = OrderDetails.ProductID
    -> WHERE Products.Description = 'High-quality tablet'
    -> GROUP BY Products.ProductID, ProductName;
+-----------+-------------+------------+
| ProductID | ProductName | OrderCount |
+-----------+-------------+------------+
|         3 | Tablet      |          0 |
|         7 | Tablet      |          1 |
+-----------+-------------+------------+
2 rows in set (0.00 sec)
```

8. Write an SQL query to list electronic gadgets and the number of times each product has been ordered.



```
mysql> SELECT TOP 1 OrderID, OrderDate, CONCAT(FirstName, ' ', LastName) AS CustomerNam
e, TotalAmount
    -> FROM Orders
    -> JOIN Customers ON Orders.CustomerID = Customers.CustomerID
    -> ORDER BY TotalAmount DESC;
```

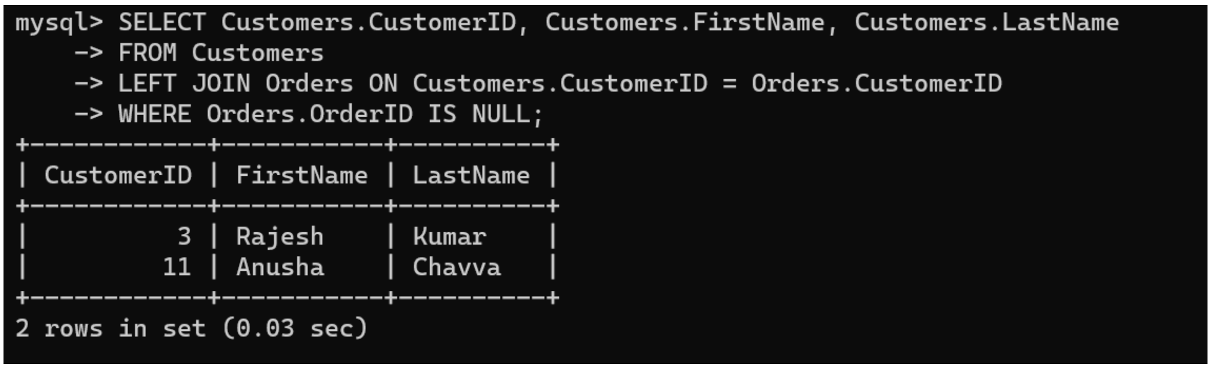
9. Write an SQL query to find customers who have purchased a specific electronic gadget product. Allow users to input the product name as a parameter.



```
mysql> SELECT DISTINCT Customers.CustomerID, FirstName, LastName, Email, Phone, Address

    -> FROM Customers
    -> JOIN Orders ON Customers.CustomerID = Orders.CustomerID
    -> JOIN OrderDetails ON Orders.OrderID = OrderDetails.OrderID
    -> JOIN Products ON OrderDetails.ProductID = Products.ProductID
    -> WHERE Products.ProductName = 'Laptop';
+------------+-----------+----------+---------------------+------------+---------------
-+
| CustomerID | FirstName | LastName | Email               | Phone      | Address
 |
+------------+-----------+----------+---------------------+------------+---------------
-+
|          1 | John      | Doe      | new.email@email.com | 1234567890 | 456 Updated St
 |
+------------+-----------+----------+---------------------+------------+---------------
-+
1 row in set (0.00 sec)
```

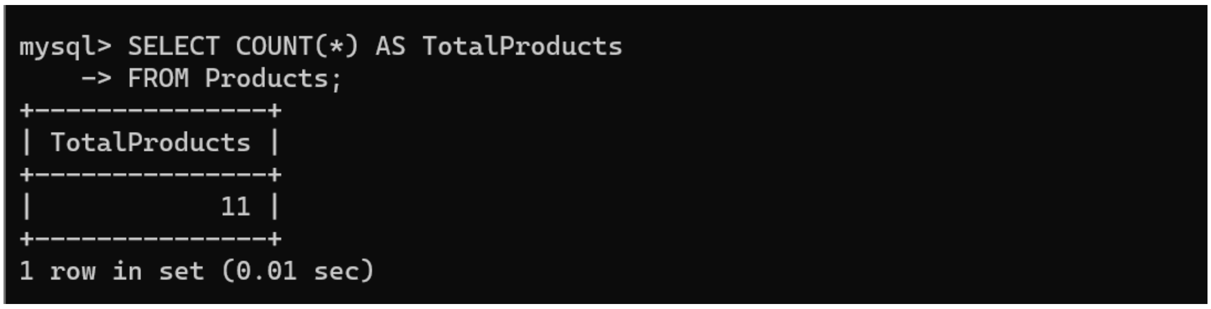
Task 4:

1. Write an SQL query to find out which customers have not placed any orders.



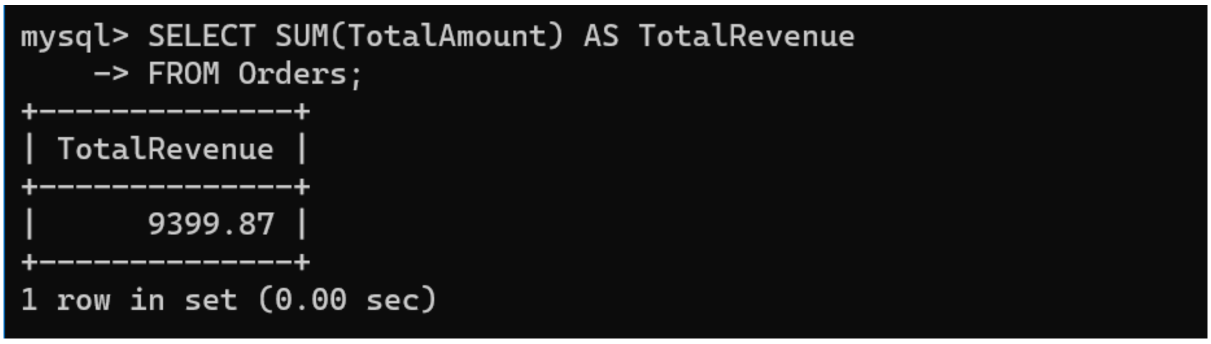
```
mysql> SELECT Customers.CustomerID, Customers.FirstName, Customers.LastName
    -> FROM Customers
    -> LEFT JOIN Orders ON Customers.CustomerID = Orders.CustomerID
    -> WHERE Orders.OrderID IS NULL;
+------------+-----------+----------+
| CustomerID | FirstName | LastName |
+------------+-----------+----------+
|          3 | Rajesh    | Kumar    |
|         11 | Anusha    | Chavva   |
+------------+-----------+----------+
2 rows in set (0.03 sec)
```

2. Write an SQL query to find the total number of products available for sale.



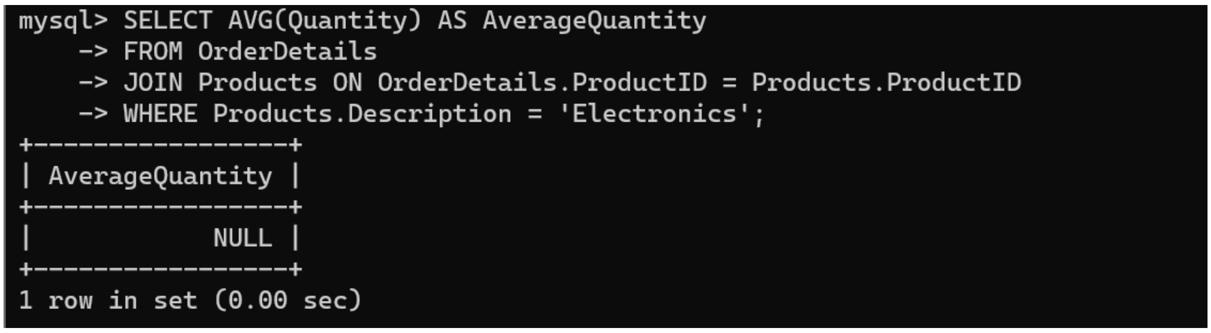
```
mysql> SELECT COUNT(*) AS TotalProducts
    -> FROM Products;
+---------------+
| TotalProducts |
+---------------+
|            11 |
+---------------+
1 row in set (0.01 sec)
```

3. Write an SQL query to calculate the total revenue generated by TechShop.



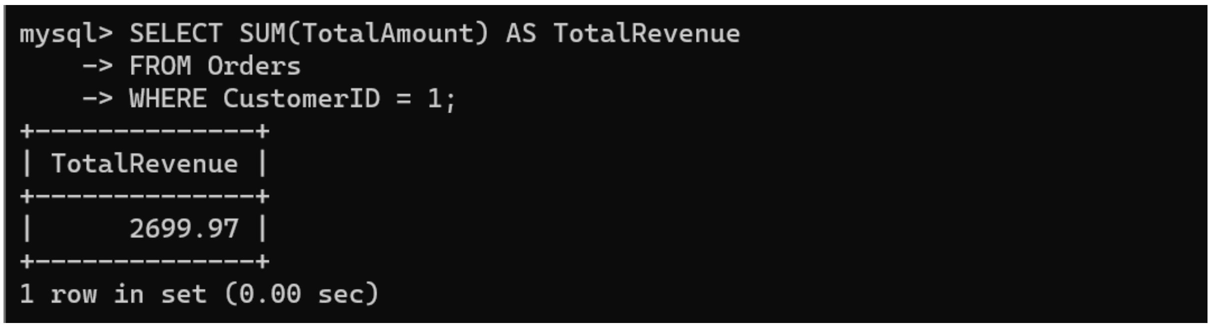
```
mysql> SELECT SUM(TotalAmount) AS TotalRevenue
    -> FROM Orders;
+--------------+
| TotalRevenue |
+--------------+
|      9399.87 |
+--------------+
1 row in set (0.00 sec)
```

4.Write an SQL query to calculate the average quantity ordered for products in a specific category. Allow users to input the category name as a parameter.



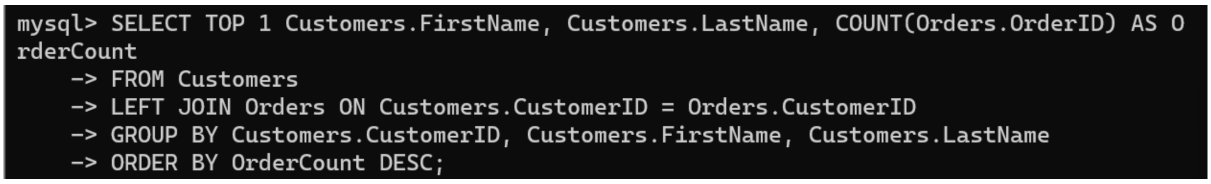
```
mysql> SELECT AVG(Quantity) AS AverageQuantity
    -> FROM OrderDetails
    -> JOIN Products ON OrderDetails.ProductID = Products.ProductID
    -> WHERE Products.Description = 'Electronics';
+-----------------+
| AverageQuantity |
+-----------------+
|            NULL |
+-----------------+
1 row in set (0.00 sec)
```

5.Write an SQL query to calculate the total revenue generated by a specific customer. Allow users to input the customer ID as a parameter.



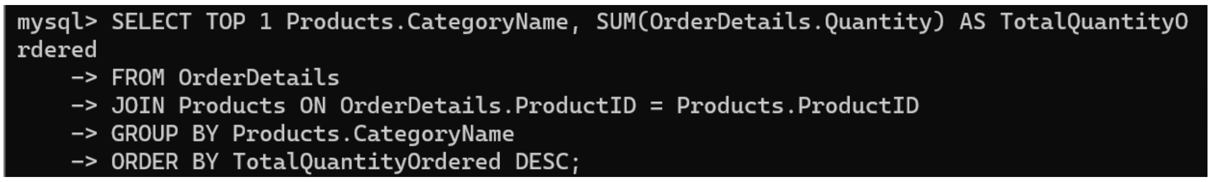
```
mysql> SELECT SUM(TotalAmount) AS TotalRevenue
    -> FROM Orders
    -> WHERE CustomerID = 1;
+--------------+
| TotalRevenue |
+--------------+
|      2699.97 |
+--------------+
1 row in set (0.00 sec)
```

6. Write an SQL query to find the customers who have placed the most orders. List their names and the number of orders they've placed.



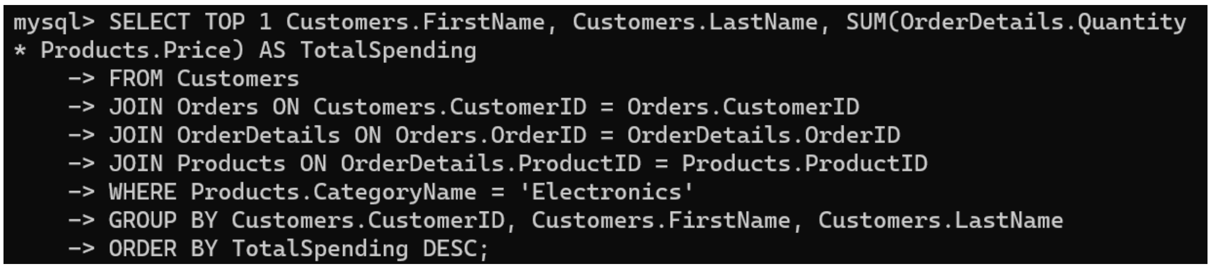
```
mysql> SELECT TOP 1 Customers.FirstName, Customers.LastName, COUNT(Orders.OrderID) AS O
rderCount
    -> FROM Customers
    -> LEFT JOIN Orders ON Customers.CustomerID = Orders.CustomerID
    -> GROUP BY Customers.CustomerID, Customers.FirstName, Customers.LastName
    -> ORDER BY OrderCount DESC;
```

7. Write an SQL query to find the most popular product category, which is the one with the highest total quantity ordered across all orders.



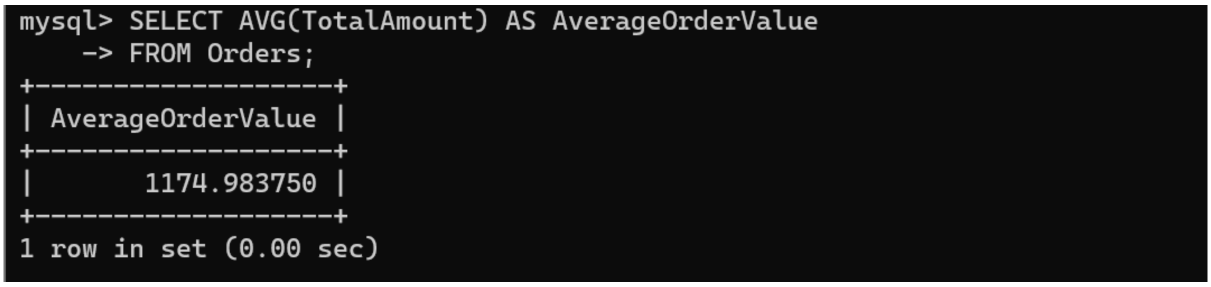
```
mysql> SELECT TOP 1 Products.CategoryName, SUM(OrderDetails.Quantity) AS TotalQuantityO
rdered
    -> FROM OrderDetails
    -> JOIN Products ON OrderDetails.ProductID = Products.ProductID
    -> GROUP BY Products.CategoryName
    -> ORDER BY TotalQuantityOrdered DESC;
```

8. Write an SQL query to find the customer who has spent the most money (highest total revenue) on electronic gadgets. List their name and total spending.



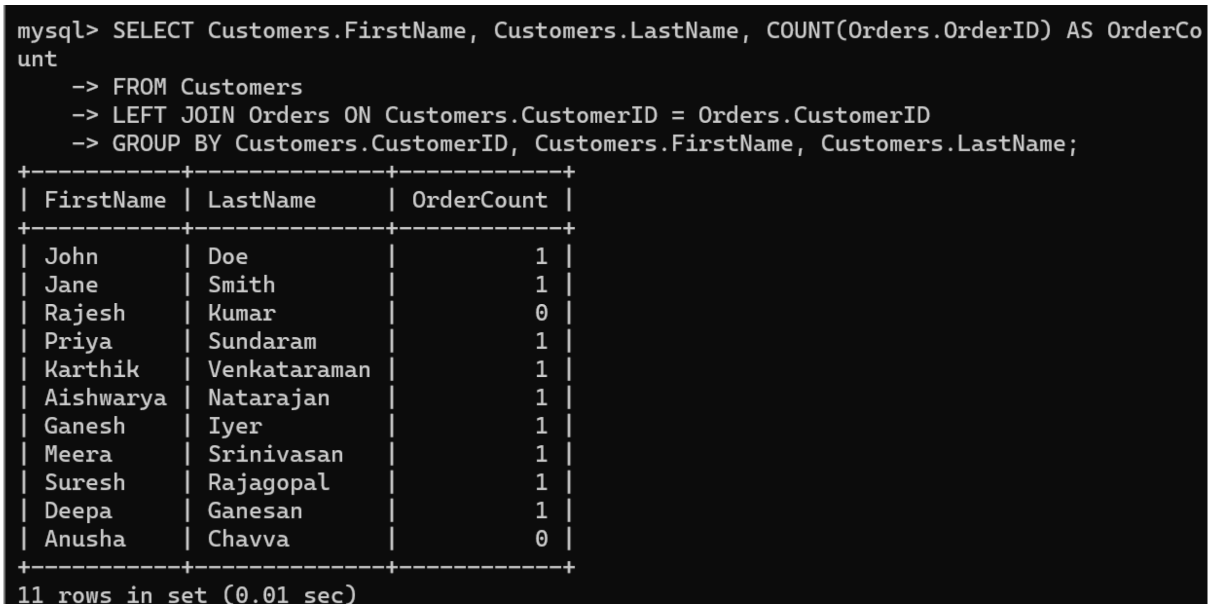
```
mysql> SELECT TOP 1 Customers.FirstName, Customers.LastName, SUM(OrderDetails.Quantity
* Products.Price) AS TotalSpending
    -> FROM Customers
    -> JOIN Orders ON Customers.CustomerID = Orders.CustomerID
    -> JOIN OrderDetails ON Orders.OrderID = OrderDetails.OrderID
    -> JOIN Products ON OrderDetails.ProductID = Products.ProductID
    -> WHERE Products.CategoryName = 'Electronics'
    -> GROUP BY Customers.CustomerID, Customers.FirstName, Customers.LastName
    -> ORDER BY TotalSpending DESC;
```

9. Write an SQL query to calculate the average order value (total revenue divided by the number of orders) for all customers.



```
mysql> SELECT AVG(TotalAmount) AS AverageOrderValue
    -> FROM Orders;
+-------------------+
| AverageOrderValue |
+-------------------+
|       1174.983750 |
+-------------------+
1 row in set (0.00 sec)
```

10. Write an SQL query to find the total number of orders placed by each customer and list their names along with the order count.



```
mysql> SELECT Customers.FirstName, Customers.LastName, COUNT(Orders.OrderID) AS OrderCo
unt
    -> FROM Customers
    -> LEFT JOIN Orders ON Customers.CustomerID = Orders.CustomerID
    -> GROUP BY Customers.CustomerID, Customers.FirstName, Customers.LastName;
+-----------+--------------+------------+
| FirstName | LastName     | OrderCount |
+-----------+--------------+------------+
| John      | Doe          |          1 |
| Jane      | Smith        |          1 |
| Rajesh    | Kumar        |          0 |
| Priya     | Sundaram     |          1 |
| Karthik   | Venkataraman |          1 |
| Aishwarya | Natarajan    |          1 |
| Ganesh    | Iyer         |          1 |
| Meera     | Srinivasan   |          1 |
| Suresh    | Rajagopal    |          1 |
| Deepa     | Ganesan      |          1 |
| Anusha    | Chavva       |          0 |
+-----------+--------------+------------+
11 rows in set (0.01 sec)
```