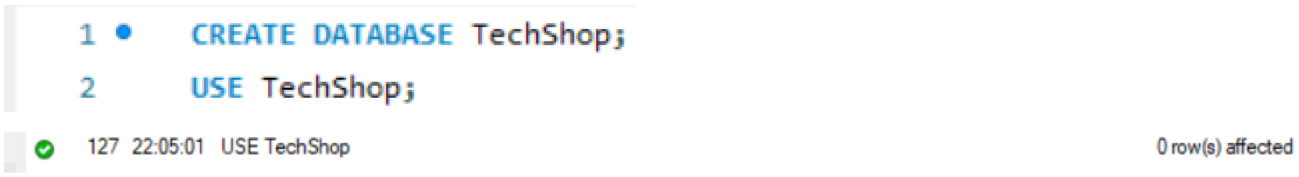
Tech Shop

Task 1

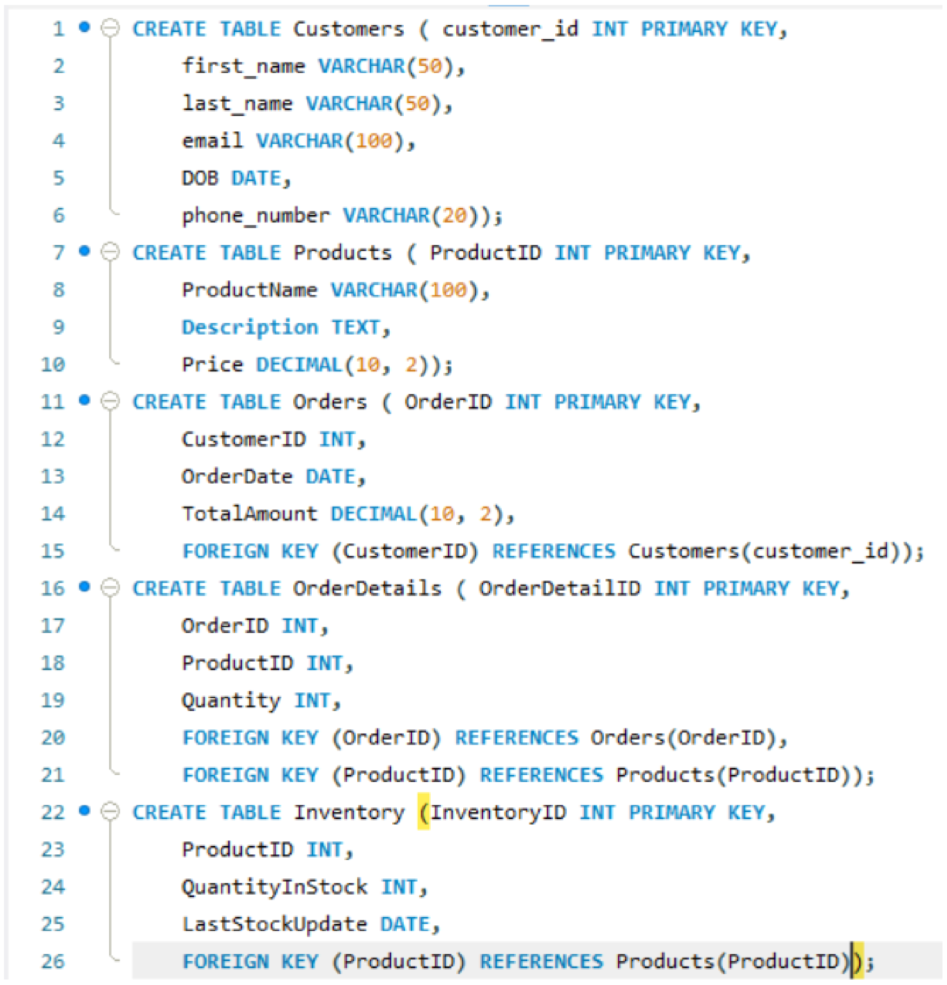
1. Create the database named "TechShop"



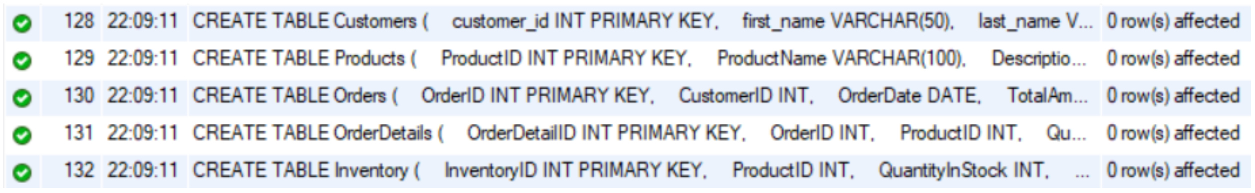
```
CREATE DATABASE TechShop;
USE TechShop;
```

```
127  22:05:01  USE TechShop                                                   0 row(s) affected
```

2. Define the schema for the Customers, Products, Orders, Order Details and Inventory tables based on the provided schema.

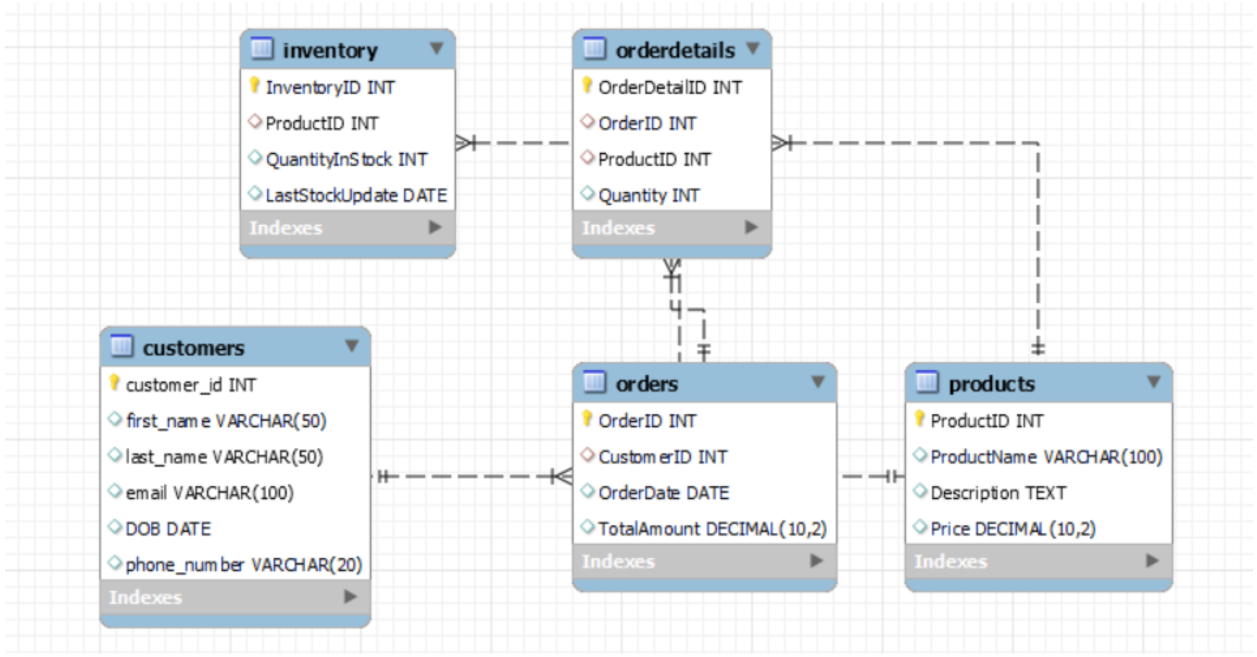


```
CREATE TABLE Customers ( customer_id INT PRIMARY KEY,
    first_name VARCHAR(50),
    last_name VARCHAR(50),
    email VARCHAR(100),
    DOB DATE,
    phone_number VARCHAR(20));
CREATE TABLE Products ( ProductID INT PRIMARY KEY,
    ProductName VARCHAR(100),
    Description TEXT,
    Price DECIMAL(10, 2));
CREATE TABLE Orders ( OrderID INT PRIMARY KEY,
    CustomerID INT,
    OrderDate DATE,
    TotalAmount DECIMAL(10, 2),
    FOREIGN KEY (CustomerID) REFERENCES Customers(customer_id));
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));
CREATE TABLE Inventory (InventoryID INT PRIMARY KEY,
    ProductID INT,
    QuantityInStock INT,
    LastStockUpdate DATE,
    FOREIGN KEY (ProductID) REFERENCES Products(ProductID));
```



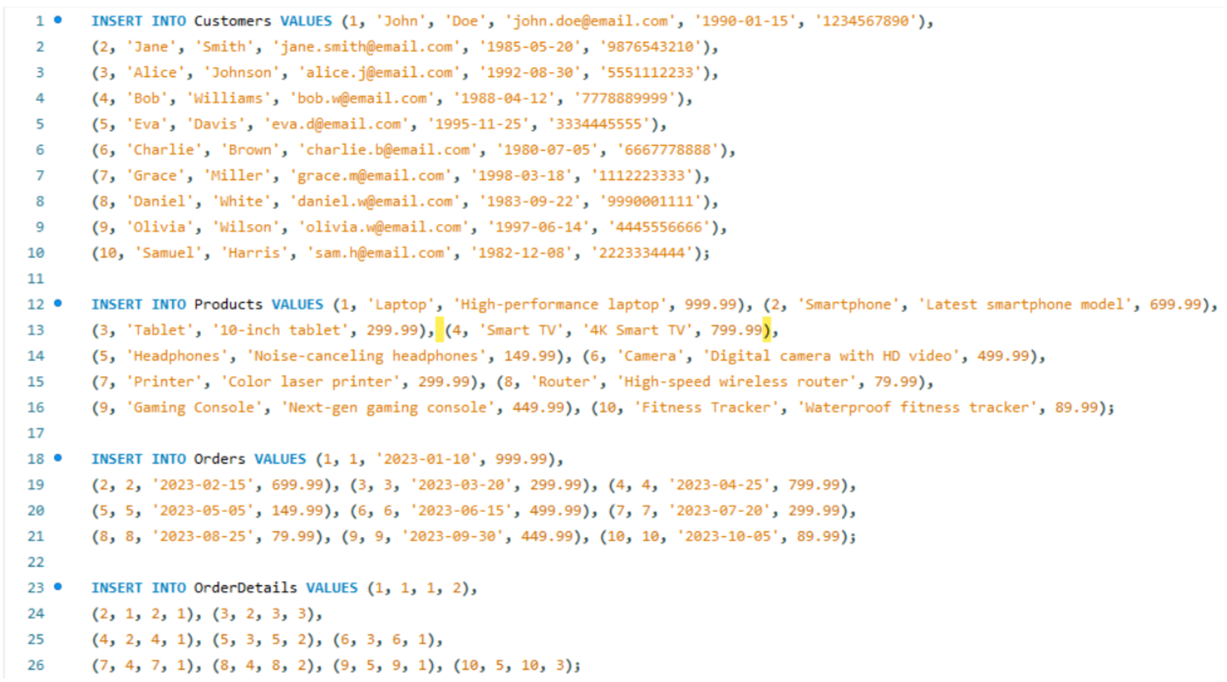
```
128 22:09:11 CREATE TABLE Customers (  customer_id INT PRIMARY KEY,  first_name VARCHAR(50),  last_name V...  0 row(s) affected
129 22:09:11 CREATE TABLE Products (  ProductID INT PRIMARY KEY,  ProductName VARCHAR(100),  Descriptio...  0 row(s) affected
130 22:09:11 CREATE TABLE Orders (  OrderID INT PRIMARY KEY,  CustomerID INT,  OrderDate DATE,  TotalAm...  0 row(s) affected
131 22:09:11 CREATE TABLE OrderDetails (  OrderDetailID INT PRIMARY KEY,  OrderID INT,  ProductID INT,  Qu...  0 row(s) affected
132 22:09:11 CREATE TABLE Inventory (  InventoryID INT PRIMARY KEY,  ProductID INT,  QuantityInStock INT,  ...  0 row(s) affected
```

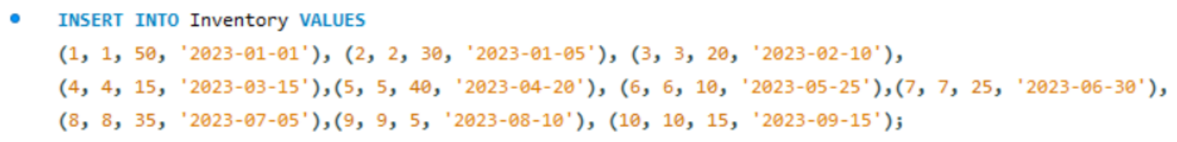
3. Create an ERD (Entity Relationship Diagram) for the database.



4. Insert at least 10 sample records into each of the following tables.

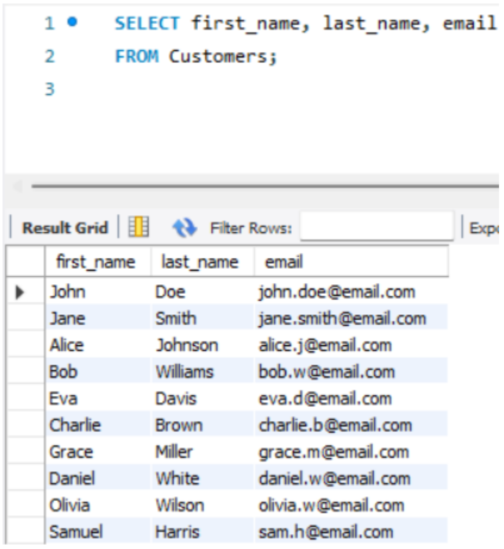
a. Customers
b. Products
c. Orders
d. Order Details


```sql
INSERT INTO Customers VALUES (1, 'John', 'Doe', 'john.doe@email.com', '1990-01-15', '1234567890'),
(2, 'Jane', 'Smith', 'jane.smith@email.com', '1985-05-20', '9876543210'),
(3, 'Alice', 'Johnson', 'alice.j@email.com', '1992-08-30', '5551112233'),
(4, 'Bob', 'Williams', 'bob.w@email.com', '1988-04-12', '7778889999'),
(5, 'Eva', 'Davis', 'eva.d@email.com', '1995-11-25', '3334445555'),
(6, 'Charlie', 'Brown', 'charlie.b@email.com', '1980-07-05', '6667778888'),
(7, 'Grace', 'Miller', 'grace.m@email.com', '1998-03-18', '1112223333'),
(8, 'Daniel', 'White', 'daniel.w@email.com', '1983-09-22', '9990001111'),
(9, 'Olivia', 'Wilson', 'olivia.w@email.com', '1997-06-14', '4445556666'),
(10, 'Samuel', 'Harris', 'sam.h@email.com', '1982-12-08', '2223334444');

INSERT INTO Products VALUES (1, 'Laptop', 'High-performance laptop', 999.99), (2, 'Smartphone', 'Latest smartphone model', 699.99),
(3, 'Tablet', '10-inch tablet', 299.99), (4, 'Smart TV', '4K Smart TV', 799.99),
(5, 'Headphones', 'Noise-canceling headphones', 149.99), (6, 'Camera', 'Digital camera with HD video', 499.99),
(7, 'Printer', 'Color laser printer', 299.99), (8, 'Router', 'High-speed wireless router', 79.99),
(9, 'Gaming Console', 'Next-gen gaming console', 449.99), (10, 'Fitness Tracker', 'Waterproof fitness tracker', 89.99);

INSERT INTO Orders VALUES (1, 1, '2023-01-10', 999.99),
(2, 2, '2023-02-15', 699.99), (3, 3, '2023-03-20', 299.99), (4, 4, '2023-04-25', 799.99),
(5, 5, '2023-05-05', 149.99), (6, 6, '2023-06-15', 499.99), (7, 7, '2023-07-20', 299.99),
(8, 8, '2023-08-25', 79.99), (9, 9, '2023-09-30', 449.99), (10, 10, '2023-10-05', 89.99);

INSERT INTO OrderDetails VALUES (1, 1, 1, 2),
(2, 1, 2, 1), (3, 2, 3, 3),
(4, 2, 4, 1), (5, 3, 5, 2), (6, 3, 6, 1),
(7, 4, 7, 1), (8, 4, 8, 2), (9, 5, 9, 1), (10, 5, 10, 3);
```


```sql
INSERT INTO Inventory VALUES
(1, 1, 50, '2023-01-01'), (2, 2, 30, '2023-01-05'), (3, 3, 20, '2023-02-10'),
(4, 4, 15, '2023-03-15'),(5, 5, 40, '2023-04-20'), (6, 6, 10, '2023-05-25'),(7, 7, 25, '2023-06-30'),
(8, 8, 35, '2023-07-05'),(9, 9, 5, '2023-08-10'), (10, 10, 15, '2023-09-15');
```

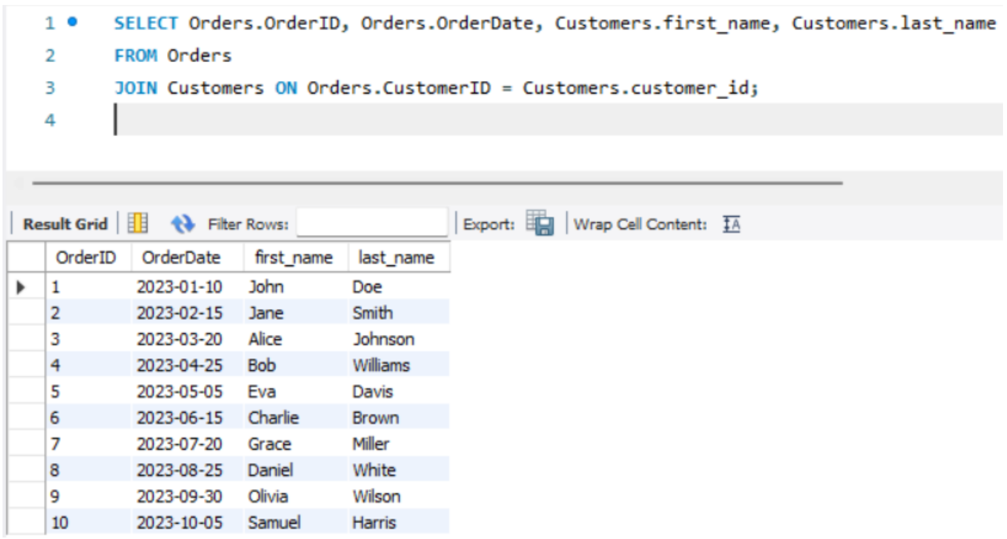
Task 2

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


```sql
SELECT first_name, last_name, email
FROM Customers;
```

first_name	last_name	email
John	Doe	john.doe@email.com
Jane	Smith	jane.smith@email.com
Alice	Johnson	alice.j@email.com
Bob	Williams	bob.w@email.com
Eva	Davis	eva.d@email.com
Charlie	Brown	charlie.b@email.com
Grace	Miller	grace.m@email.com
Daniel	White	daniel.w@email.com
Olivia	Wilson	olivia.w@email.com
Samuel	Harris	sam.h@email.com

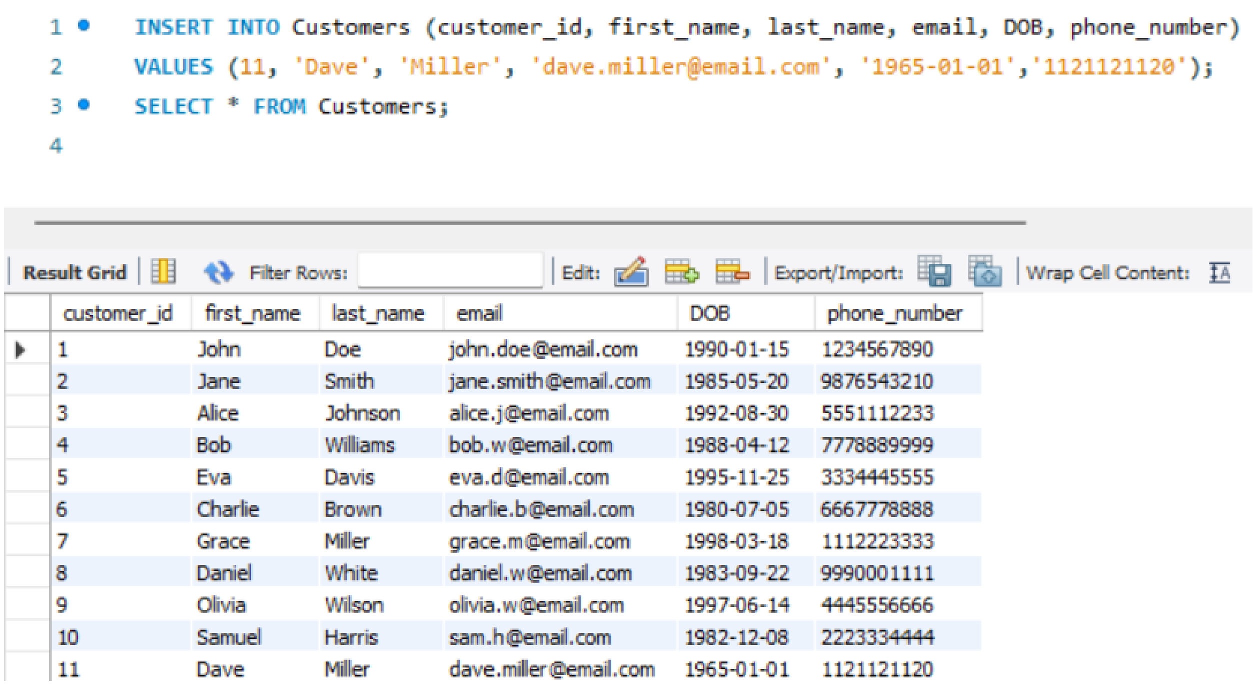
2. Write an SQL query to list all orders with their order dates and corresponding customer names.



```
SELECT Orders.OrderID, Orders.OrderDate, Customers.first_name, Customers.last_name
FROM Orders
JOIN Customers ON Orders.CustomerID = Customers.customer_id;
```

OrderID	OrderDate	first_name	last_name
1	2023-01-10	John	Doe
2	2023-02-15	Jane	Smith
3	2023-03-20	Alice	Johnson
4	2023-04-25	Bob	Williams
5	2023-05-05	Eva	Davis
6	2023-06-15	Charlie	Brown
7	2023-07-20	Grace	Miller
8	2023-08-25	Daniel	White
9	2023-09-30	Olivia	Wilson
10	2023-10-05	Samuel	Harris

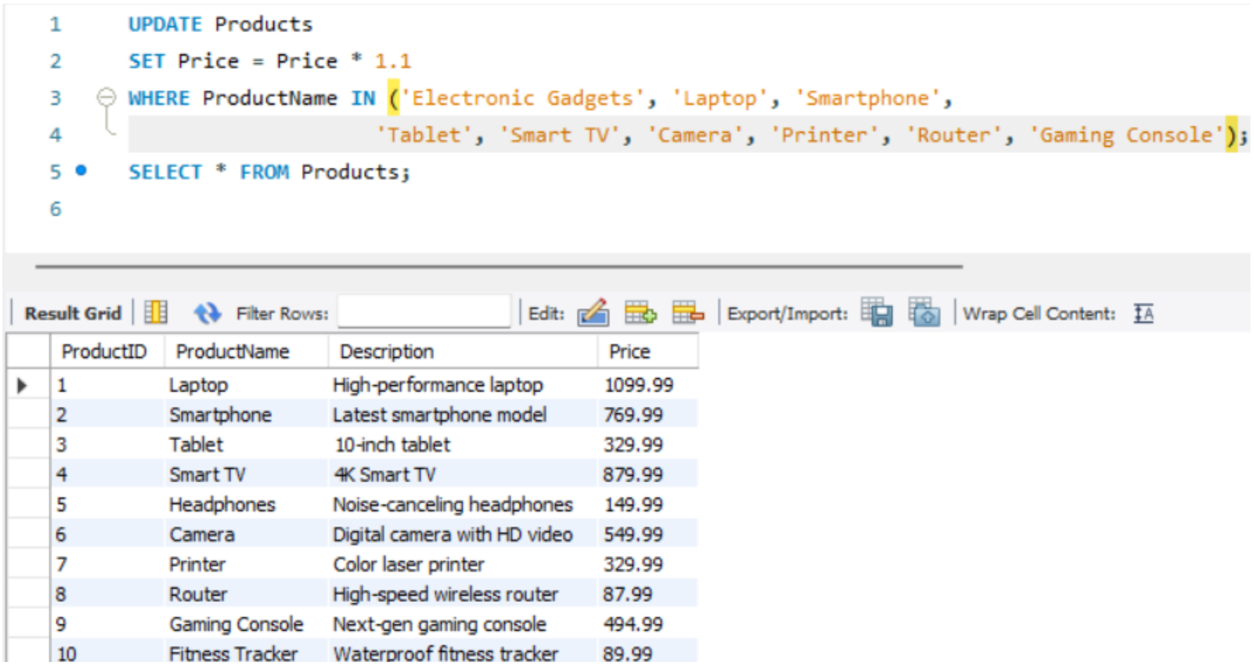
3. Write an SQL query to insert a new customer record into the "Customers" table. Include customer information such as name, email, and address.



```
INSERT INTO Customers (customer_id, first_name, last_name, email, DOB, phone_number)
VALUES (11, 'Dave', 'Miller', 'dave.miller@email.com', '1965-01-01','1121121120');
SELECT * FROM Customers;
```

customer_id	first_name	last_name	email	DOB	phone_number
1	John	Doe	john.doe@email.com	1990-01-15	1234567890
2	Jane	Smith	jane.smith@email.com	1985-05-20	9876543210
3	Alice	Johnson	alice.j@email.com	1992-08-30	5551112233
4	Bob	Williams	bob.w@email.com	1988-04-12	7778889999
5	Eva	Davis	eva.d@email.com	1995-11-25	3334445555
6	Charlie	Brown	charlie.b@email.com	1980-07-05	6667778888
7	Grace	Miller	grace.m@email.com	1998-03-18	1112223333
8	Daniel	White	daniel.w@email.com	1983-09-22	9990001111
9	Olivia	Wilson	olivia.w@email.com	1997-06-14	4445556666
10	Samuel	Harris	sam.h@email.com	1982-12-08	2223334444
11	Dave	Miller	dave.miller@email.com	1965-01-01	1121121120

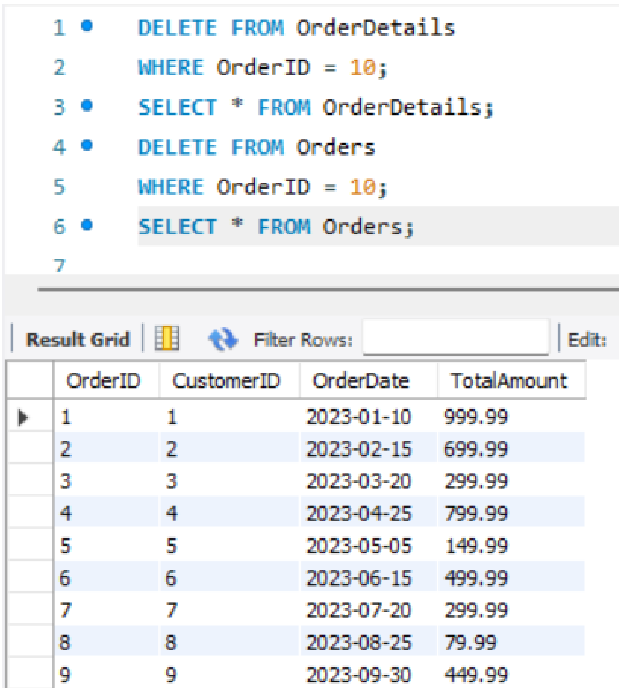
4. Write an SQL query to update the prices of all electronic gadgets in the "Products" table by increasing them by 10%.



```sql
UPDATE Products
SET Price = Price * 1.1
WHERE ProductName IN ('Electronic Gadgets', 'Laptop', 'Smartphone',
                      'Tablet', 'Smart TV', 'Camera', 'Printer', 'Router', 'Gaming Console');
SELECT * FROM Products;
```

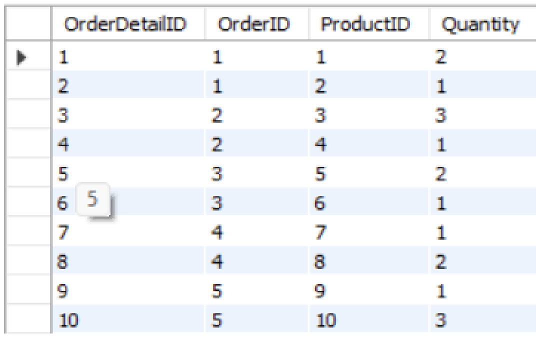
ProductID	ProductName	Description	Price
1	Laptop	High-performance laptop	1099.99
2	Smartphone	Latest smartphone model	769.99
3	Tablet	10-inch tablet	329.99
4	Smart TV	4K Smart TV	879.99
5	Headphones	Noise-canceling headphones	149.99
6	Camera	Digital camera with HD video	549.99
7	Printer	Color laser printer	329.99
8	Router	High-speed wireless router	87.99
9	Gaming Console	Next-gen gaming console	494.99
10	Fitness Tracker	Waterproof fitness tracker	89.99

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.



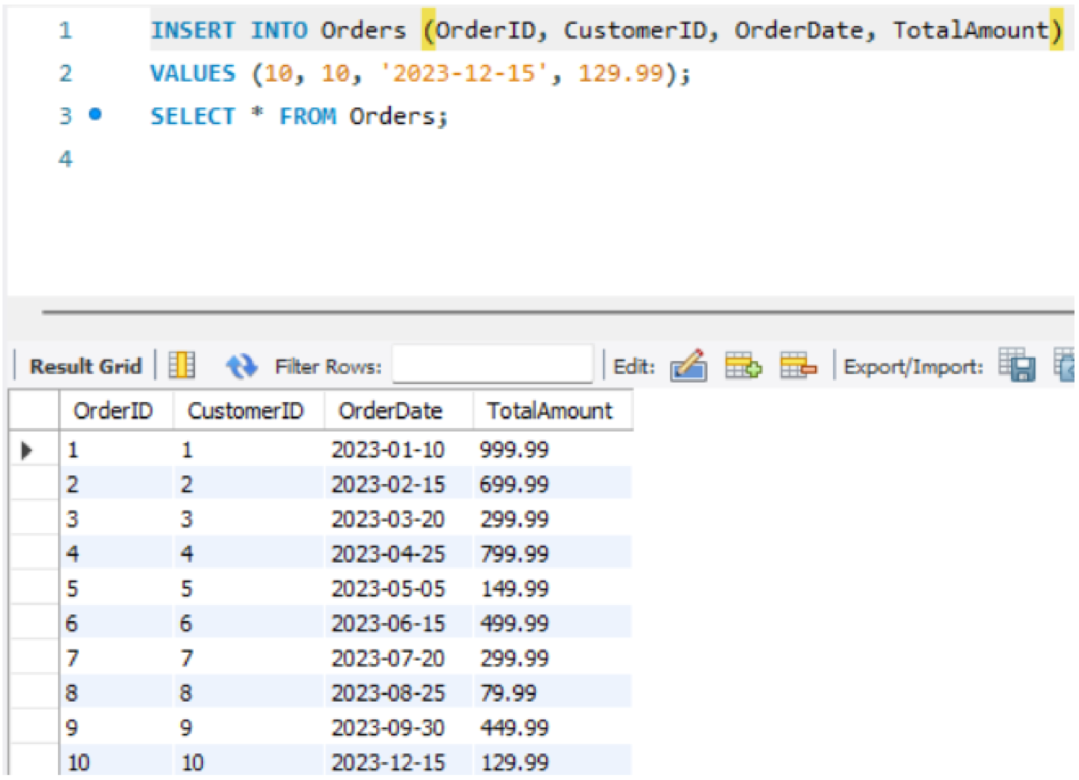
```sql
DELETE FROM OrderDetails
WHERE OrderID = 10;
SELECT * FROM OrderDetails;
DELETE FROM Orders
WHERE OrderID = 10;
SELECT * FROM Orders;
```

OrderID	CustomerID	OrderDate	TotalAmount
1	1	2023-01-10	999.99
2	2	2023-02-15	699.99
3	3	2023-03-20	299.99
4	4	2023-04-25	799.99
5	5	2023-05-05	149.99
6	6	2023-06-15	499.99
7	7	2023-07-20	299.99
8	8	2023-08-25	79.99
9	9	2023-09-30	449.99



OrderDetailID	OrderID	ProductID	Quantity
1	1	1	2
2	1	2	1
3	2	3	3
4	2	4	1
5	3	5	2
6	3	6	1
7	4	7	1
8	4	8	2
9	5	9	1
10	5	10	3

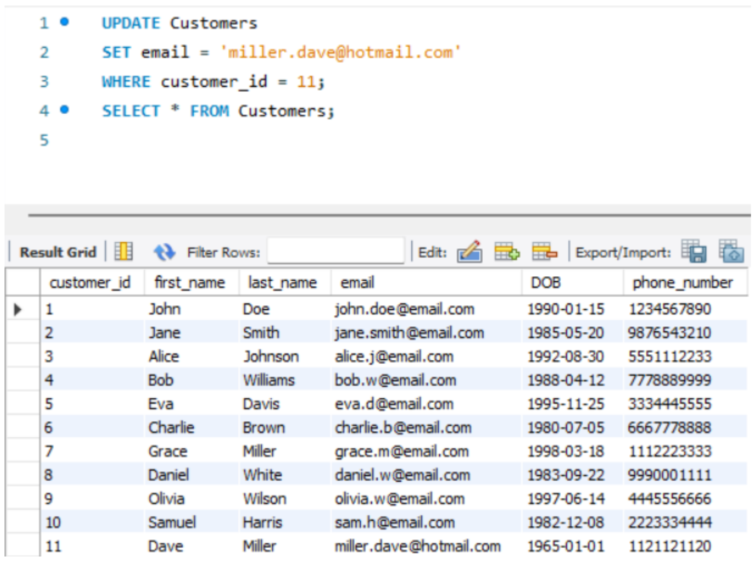
6. Write an SQL query to insert a new order into the "Orders" table. Include the customer ID, order date, and any other necessary information.



```sql
INSERT INTO Orders (OrderID, CustomerID, OrderDate, TotalAmount)
VALUES (10, 10, '2023-12-15', 129.99);
SELECT * FROM Orders;
```

OrderID	CustomerID	OrderDate	TotalAmount
1	1	2023-01-10	999.99
2	2	2023-02-15	699.99
3	3	2023-03-20	299.99
4	4	2023-04-25	799.99
5	5	2023-05-05	149.99
6	6	2023-06-15	499.99
7	7	2023-07-20	299.99
8	8	2023-08-25	79.99
9	9	2023-09-30	449.99
10	10	2023-12-15	129.99

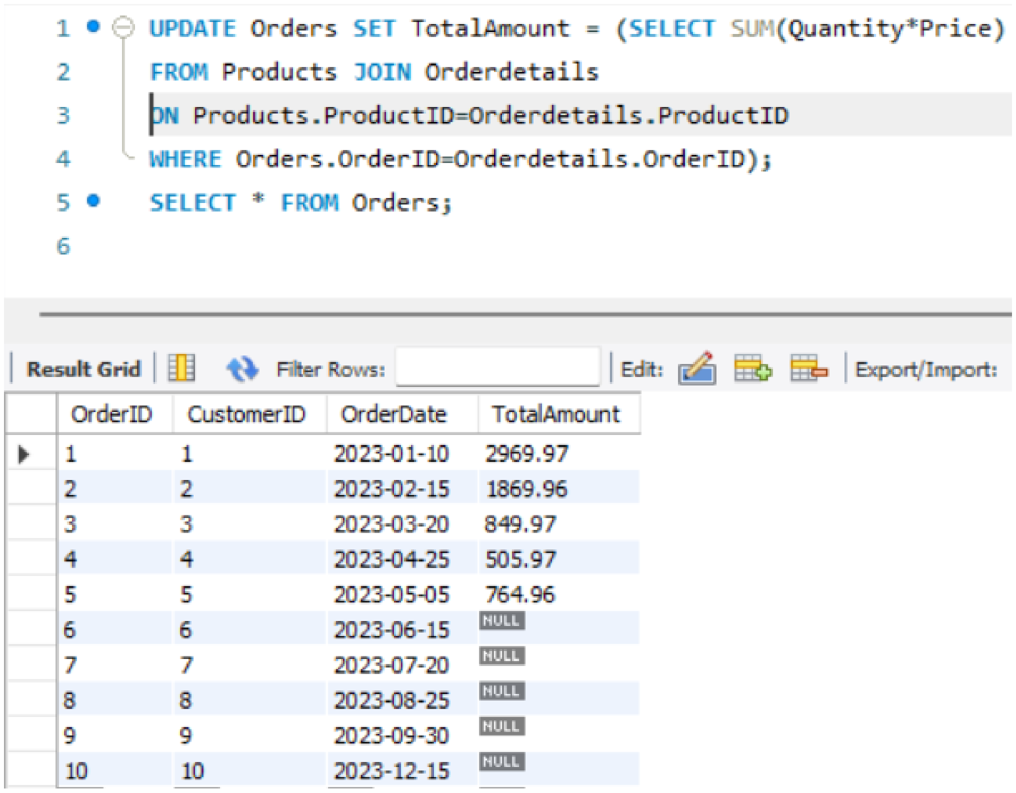
7. Write an SQL query to update the contact information (e.g., email and address) of a specific customer in the "Customers" table. Allow users to input the customer ID and new contact information.



```sql
UPDATE Customers
SET email = 'miller.dave@hotmail.com'
WHERE customer_id = 11;
SELECT * FROM Customers;
```

customer_id	first_name	last_name	email	DOB	phone_number
1	John	Doe	john.doe@email.com	1990-01-15	1234567890
2	Jane	Smith	jane.smith@email.com	1985-05-20	9876543210
3	Alice	Johnson	alice.j@email.com	1992-08-30	5551112233
4	Bob	Williams	bob.w@email.com	1988-04-12	7778889999
5	Eva	Davis	eva.d@email.com	1995-11-25	3334445555
6	Charlie	Brown	charlie.b@email.com	1980-07-05	6667778888
7	Grace	Miller	grace.m@email.com	1998-03-18	1112223333
8	Daniel	White	daniel.w@email.com	1983-09-22	9990001111
9	Olivia	Wilson	olivia.w@email.com	1997-06-14	4445556666
10	Samuel	Harris	sam.h@email.com	1982-12-08	2223334444
11	Dave	Miller	miller.dave@hotmail.com	1965-01-01	1121121120

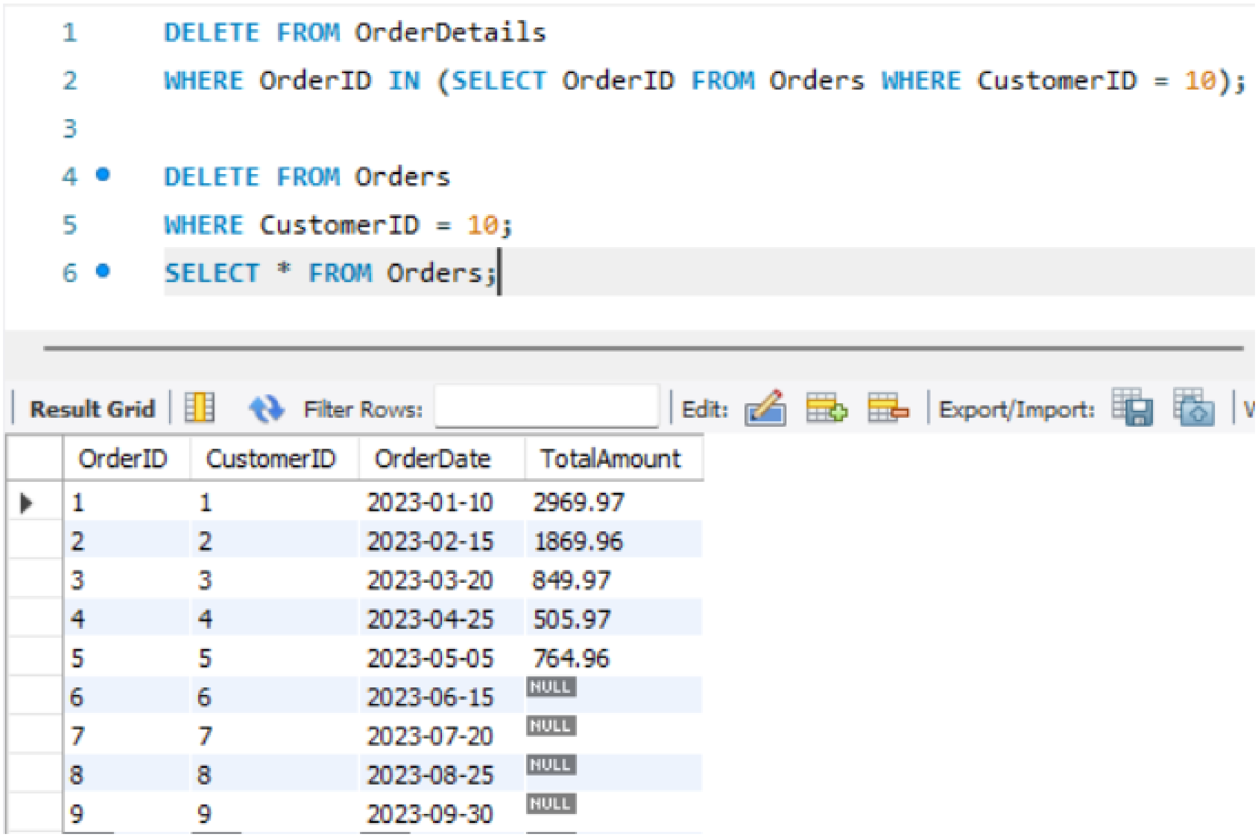
8. Write an SQL query to recalculate and update the total cost of each order in the "Orders" table based on the prices and quantities in the "OrderDetails" table.



```
UPDATE Orders SET TotalAmount = (SELECT SUM(Quantity*Price)
FROM Products JOIN Orderdetails
ON Products.ProductID=Orderdetails.ProductID
WHERE Orders.OrderID=Orderdetails.OrderID);
SELECT * FROM Orders;
```

OrderID	CustomerID	OrderDate	TotalAmount
1	1	2023-01-10	2969.97
2	2	2023-02-15	1869.96
3	3	2023-03-20	849.97
4	4	2023-04-25	505.97
5	5	2023-05-05	764.96
6	6	2023-06-15	NULL
7	7	2023-07-20	NULL
8	8	2023-08-25	NULL
9	9	2023-09-30	NULL
10	10	2023-12-15	NULL

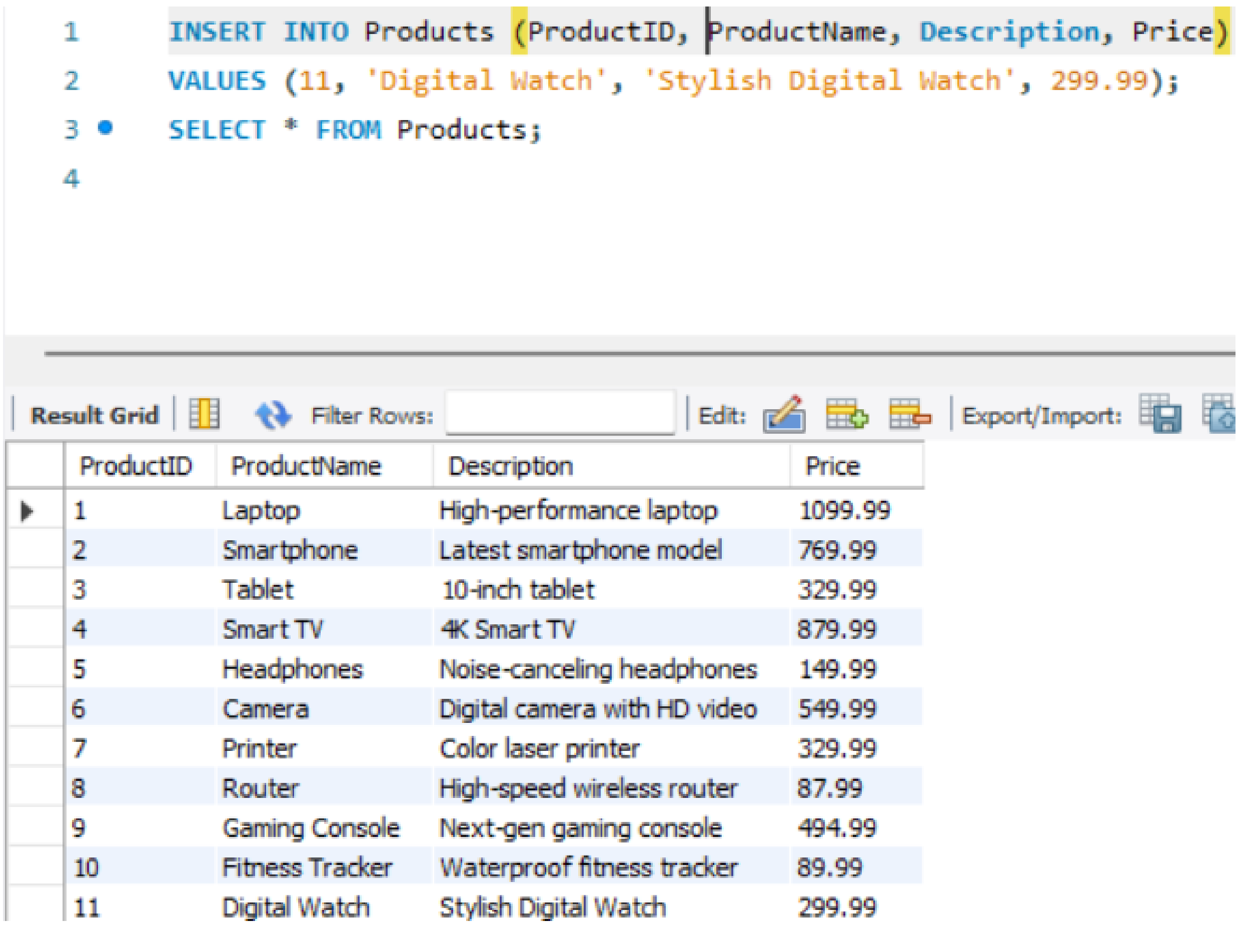
9. Write an SQL query to delete all orders and their associated order details for a specific customer from the "Orders" and "OrderDetails" tables. Allow users to input the customer ID as a parameter.



```
DELETE FROM OrderDetails
WHERE OrderID IN (SELECT OrderID FROM Orders WHERE CustomerID = 10);

DELETE FROM Orders
WHERE CustomerID = 10;
SELECT * FROM Orders;
```

OrderID	CustomerID	OrderDate	TotalAmount
1	1	2023-01-10	2969.97
2	2	2023-02-15	1869.96
3	3	2023-03-20	849.97
4	4	2023-04-25	505.97
5	5	2023-05-05	764.96
6	6	2023-06-15	NULL
7	7	2023-07-20	NULL
8	8	2023-08-25	NULL
9	9	2023-09-30	NULL

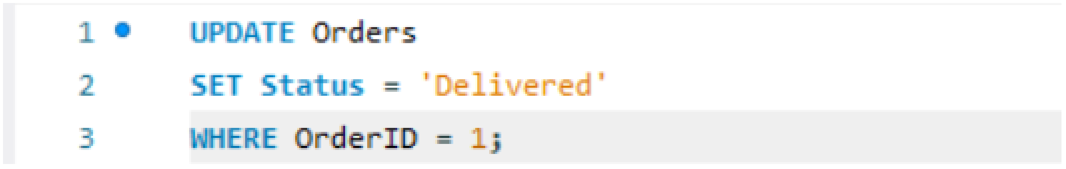
10. Write an SQL query to insert a new electronic gadget product into the "Products" table, including product name, category, price, and any other relevant details.


```sql
INSERT INTO Products (ProductID, ProductName, Description, Price)
VALUES (11, 'Digital Watch', 'Stylish Digital Watch', 299.99);
SELECT * FROM Products;
```

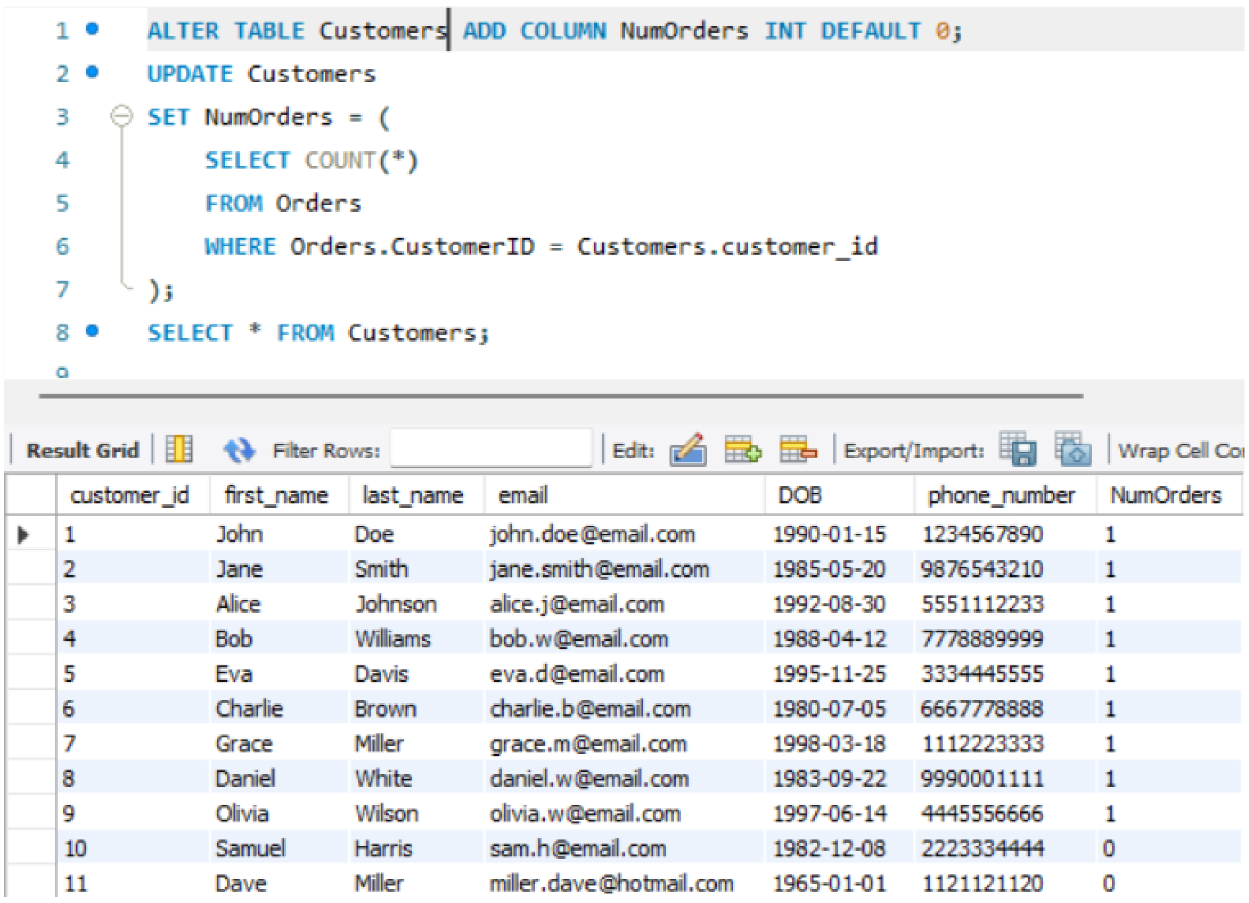
Result Grid | Filter Rows: | Edit: | Export/Import:

ProductID	ProductName	Description	Price
1	Laptop	High-performance laptop	1099.99
2	Smartphone	Latest smartphone model	769.99
3	Tablet	10-inch tablet	329.99
4	Smart TV	4K Smart TV	879.99
5	Headphones	Noise-canceling headphones	149.99
6	Camera	Digital camera with HD video	549.99
7	Printer	Color laser printer	329.99
8	Router	High-speed wireless router	87.99
9	Gaming Console	Next-gen gaming console	494.99
10	Fitness Tracker	Waterproof fitness tracker	89.99
11	Digital Watch	Stylish Digital Watch	299.99

11. Write an SQL query to update the status of a specific order in the "Orders" table (e.g., from "Pending" to "Shipped"). Allow users to input the order ID and the new status.


```sql
UPDATE Orders
SET Status = 'Delivered'
WHERE OrderID = 1;
```

12. Write an SQL query to calculate and update the number of orders placed by each customer in the "Customers" table based on the data in the "Orders" table.

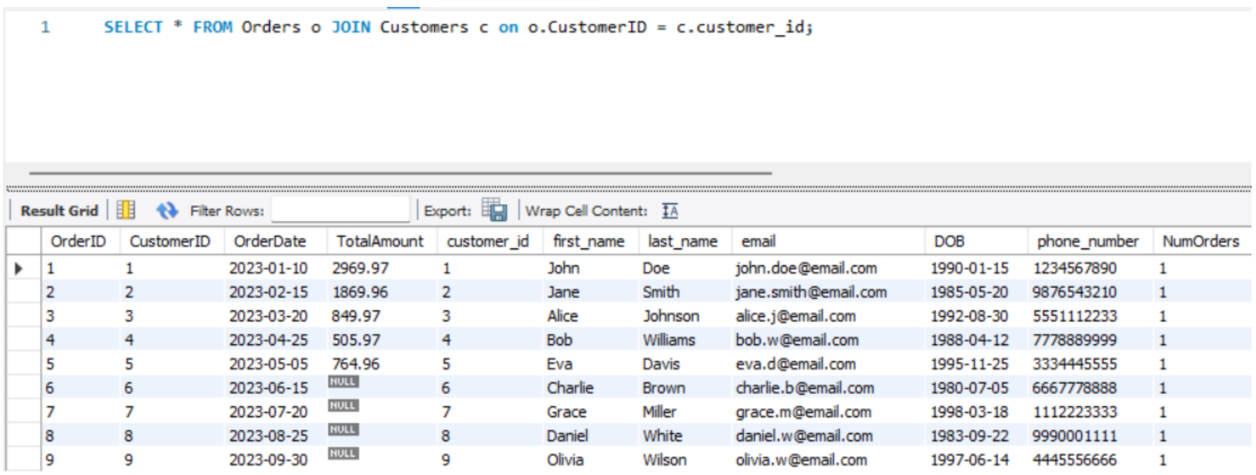


```sql
ALTER TABLE Customers ADD COLUMN NumOrders INT DEFAULT 0;
UPDATE Customers
SET NumOrders = (
    SELECT COUNT(*)
    FROM Orders
    WHERE Orders.CustomerID = Customers.customer_id
);
SELECT * FROM Customers;
```

customer_id	first_name	last_name	email	DOB	phone_number	NumOrders
1	John	Doe	john.doe@email.com	1990-01-15	1234567890	1
2	Jane	Smith	jane.smith@email.com	1985-05-20	9876543210	1
3	Alice	Johnson	alice.j@email.com	1992-08-30	5551112233	1
4	Bob	Williams	bob.w@email.com	1988-04-12	7778889999	1
5	Eva	Davis	eva.d@email.com	1995-11-25	3334445555	1
6	Charlie	Brown	charlie.b@email.com	1980-07-05	6667778888	1
7	Grace	Miller	grace.m@email.com	1998-03-18	1112223333	1
8	Daniel	White	daniel.w@email.com	1983-09-22	9990001111	1
9	Olivia	Wilson	olivia.w@email.com	1997-06-14	4445556666	1
10	Samuel	Harris	sam.h@email.com	1982-12-08	2223334444	0
11	Dave	Miller	miller.dave@hotmail.com	1965-01-01	1121121120	0

Task 3

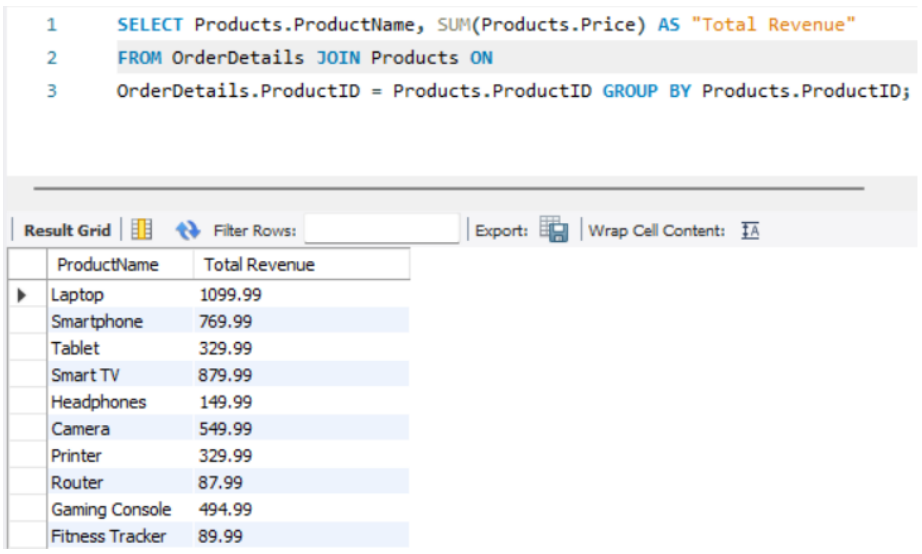
1. Write an SQL query to retrieve a list of all orders along with customer information (e.g., customer name) for each order.



```sql
SELECT * FROM Orders o JOIN Customers c on o.CustomerID = c.customer_id;
```

OrderID	CustomerID	OrderDate	TotalAmount	customer_id	first_name	last_name	email	DOB	phone_number	NumOrders
1	1	2023-01-10	2969.97	1	John	Doe	john.doe@email.com	1990-01-15	1234567890	1
2	2	2023-02-15	1869.96	2	Jane	Smith	jane.smith@email.com	1985-05-20	9876543210	1
3	3	2023-03-20	849.97	3	Alice	Johnson	alice.j@email.com	1992-08-30	5551112233	1
4	4	2023-04-25	505.97	4	Bob	Williams	bob.w@email.com	1988-04-12	7778889999	1
5	5	2023-05-05	764.96	5	Eva	Davis	eva.d@email.com	1995-11-25	3334445555	1
6	6	2023-06-15	NULL	6	Charlie	Brown	charlie.b@email.com	1980-07-05	6667778888	1
7	7	2023-07-20	NULL	7	Grace	Miller	grace.m@email.com	1998-03-18	1112223333	1
8	8	2023-08-25	NULL	8	Daniel	White	daniel.w@email.com	1983-09-22	9990001111	1
9	9	2023-09-30	NULL	9	Olivia	Wilson	olivia.w@email.com	1997-06-14	4445556666	1

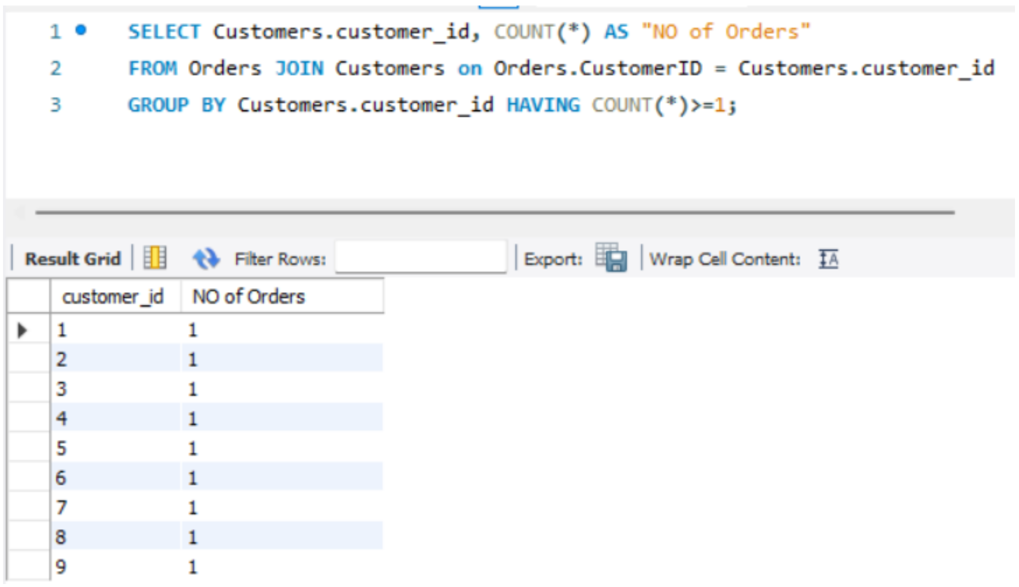
2. Write an SQL query to find the total revenue generated by each electronic gadget product. Include the product name and the total revenue.



```
SELECT Products.ProductName, SUM(Products.Price) AS "Total Revenue"
FROM OrderDetails JOIN Products ON
OrderDetails.ProductID = Products.ProductID GROUP BY Products.ProductID;
```

ProductName	Total Revenue
Laptop	1099.99
Smartphone	769.99
Tablet	329.99
Smart TV	879.99
Headphones	149.99
Camera	549.99
Printer	329.99
Router	87.99
Gaming Console	494.99
Fitness Tracker	89.99

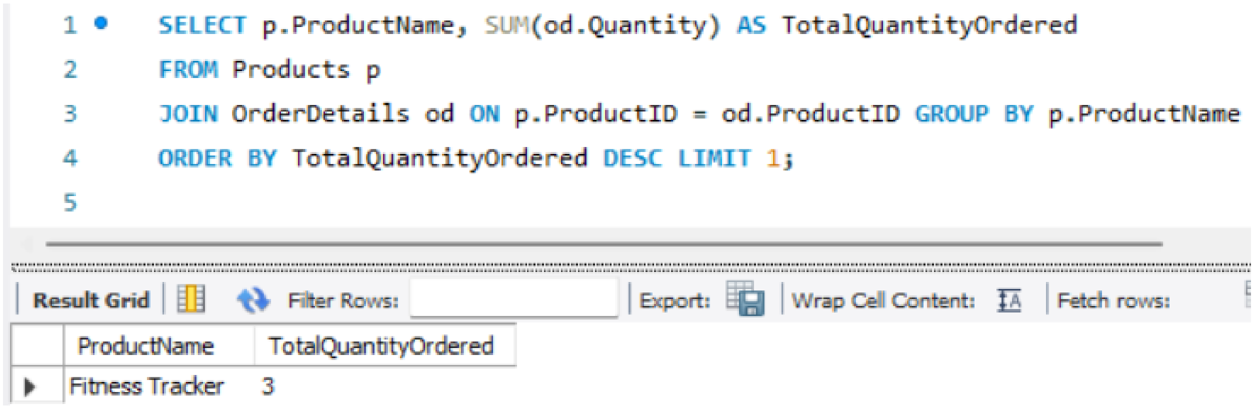
3. Write an SQL query to list all customers who have made at least one purchase. Include their names and contact information.



```
SELECT Customers.customer_id, COUNT(*) AS "NO of Orders"
FROM Orders JOIN Customers on Orders.CustomerID = Customers.customer_id
GROUP BY Customers.customer_id HAVING COUNT(*)>=1;
```

customer_id	NO of Orders
1	1
2	1
3	1
4	1
5	1
6	1
7	1
8	1
9	1

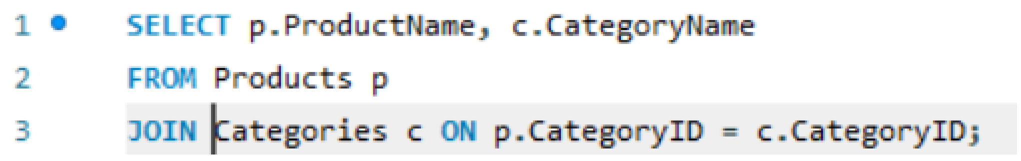
4. Write an SQL query to find the most popular electronic gadget, which is the one with the highest total quantity ordered. Include the product name and the total quantity ordered.



```
SELECT p.ProductName, SUM(od.Quantity) AS TotalQuantityOrdered
FROM Products p
JOIN OrderDetails od ON p.ProductID = od.ProductID GROUP BY p.ProductName
ORDER BY TotalQuantityOrdered DESC LIMIT 1;
```

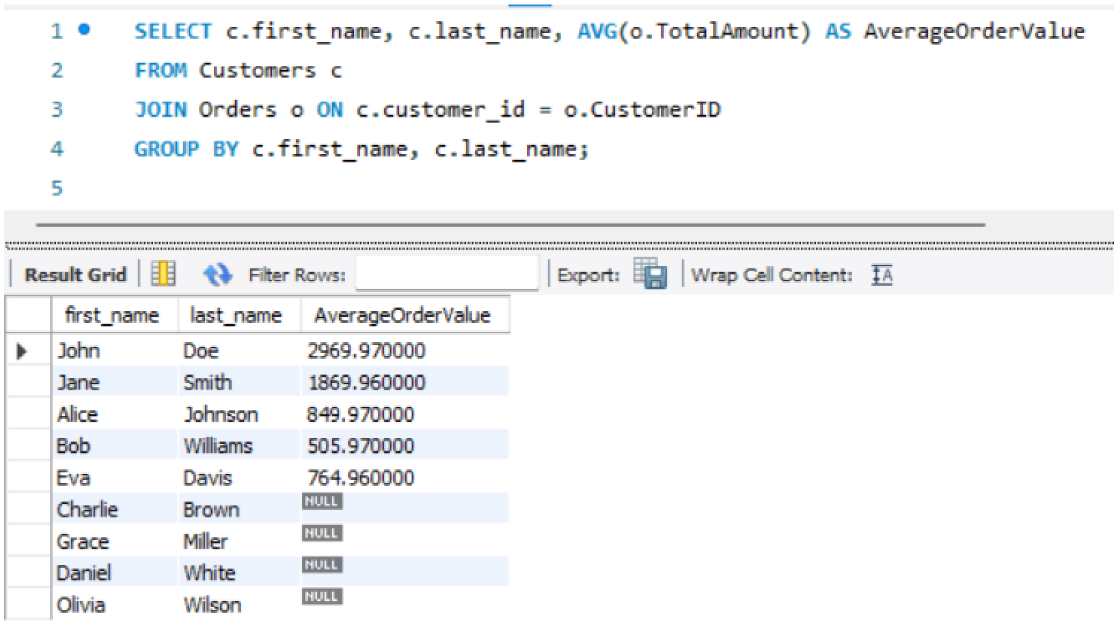
ProductName	TotalQuantityOrdered
Fitness Tracker	3

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



```
SELECT p.ProductName, c.CategoryName
FROM Products p
JOIN Categories c ON p.CategoryID = c.CategoryID;
```

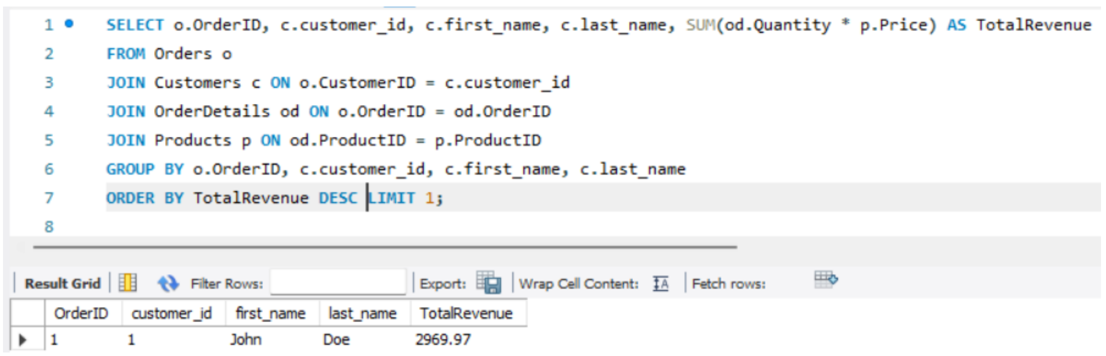
6. Write an SQL query to calculate the average order value for each customer. Include the customer's name and their average order value.



```
SELECT c.first_name, c.last_name, AVG(o.TotalAmount) AS AverageOrderValue
FROM Customers c
JOIN Orders o ON c.customer_id = o.CustomerID
GROUP BY c.first_name, c.last_name;
```

first_name	last_name	AverageOrderValue
John	Doe	2969.970000
Jane	Smith	1869.960000
Alice	Johnson	849.970000
Bob	Williams	505.970000
Eva	Davis	764.960000
Charlie	Brown	NULL
Grace	Miller	NULL
Daniel	White	NULL
Olivia	Wilson	NULL

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

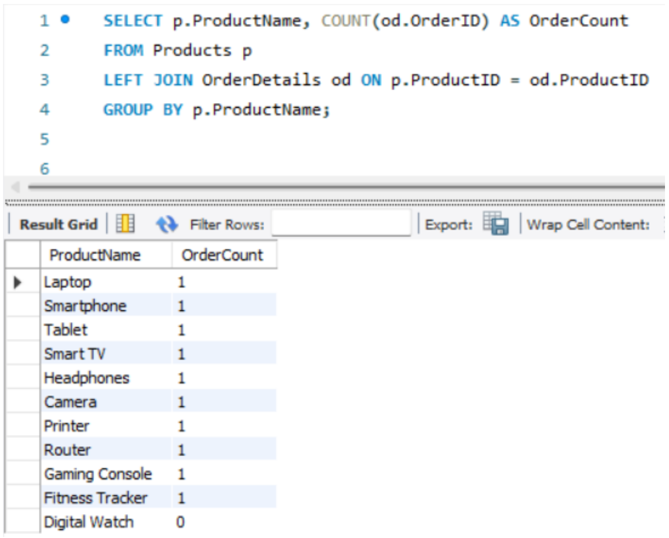


```sql
SELECT o.OrderID, c.customer_id, c.first_name, c.last_name, SUM(od.Quantity * p.Price) AS TotalRevenue
FROM Orders o
JOIN Customers c ON o.CustomerID = c.customer_id
JOIN OrderDetails od ON o.OrderID = od.OrderID
JOIN Products p ON od.ProductID = p.ProductID
GROUP BY o.OrderID, c.customer_id, c.first_name, c.last_name
ORDER BY TotalRevenue DESC LIMIT 1;
```

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

OrderID	customer_id	first_name	last_name	TotalRevenue
1	1	John	Doe	2969.97

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

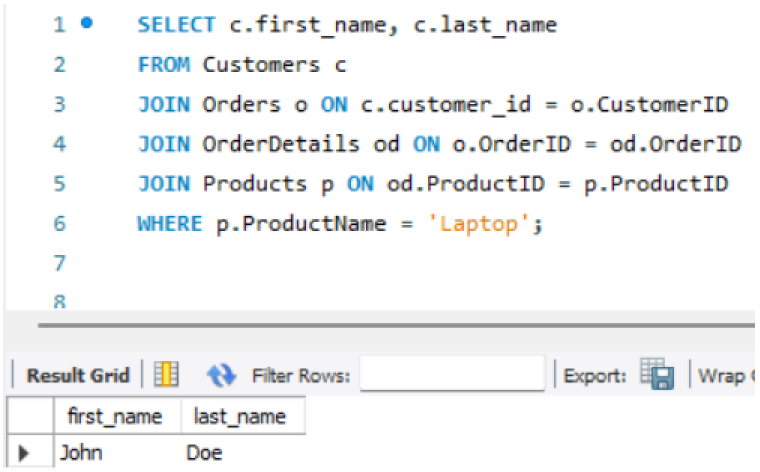


```sql
SELECT p.ProductName, COUNT(od.OrderID) AS OrderCount
FROM Products p
LEFT JOIN OrderDetails od ON p.ProductID = od.ProductID
GROUP BY p.ProductName;
```

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

ProductName	OrderCount
Laptop	1
Smartphone	1
Tablet	1
Smart TV	1
Headphones	1
Camera	1
Printer	1
Router	1
Gaming Console	1
Fitness Tracker	1
Digital Watch	0

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.

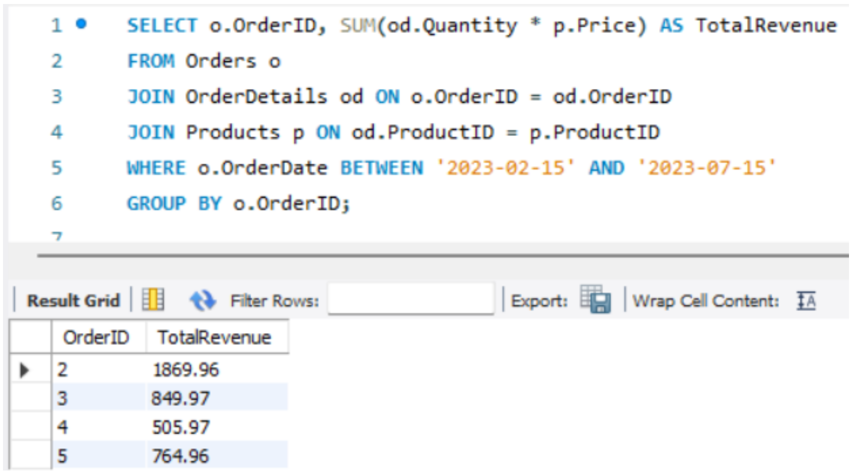


```sql
SELECT c.first_name, c.last_name
FROM Customers c
JOIN Orders o ON c.customer_id = o.CustomerID
JOIN OrderDetails od ON o.OrderID = od.OrderID
JOIN Products p ON od.ProductID = p.ProductID
WHERE p.ProductName = 'Laptop';
```

Result Grid | Filter Rows: | Export: | Wrap

first_name	last_name
John	Doe

10. Write an SQL query to calculate the total revenue generated by all orders placed within a specific time period. Allow users to input the start and end dates as parameters.

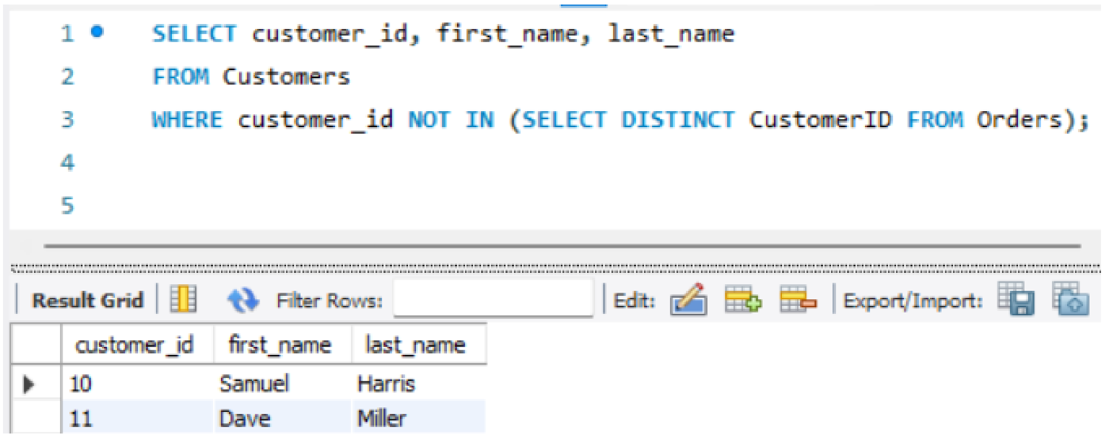


```sql
SELECT o.OrderID, SUM(od.Quantity * p.Price) AS TotalRevenue
FROM Orders o
JOIN OrderDetails od ON o.OrderID = od.OrderID
JOIN Products p ON od.ProductID = p.ProductID
WHERE o.OrderDate BETWEEN '2023-02-15' AND '2023-07-15'
GROUP BY o.OrderID;
```

OrderID	TotalRevenue
2	1869.96
3	849.97
4	505.97
5	764.96

Task 4 : SubQuery and its Types

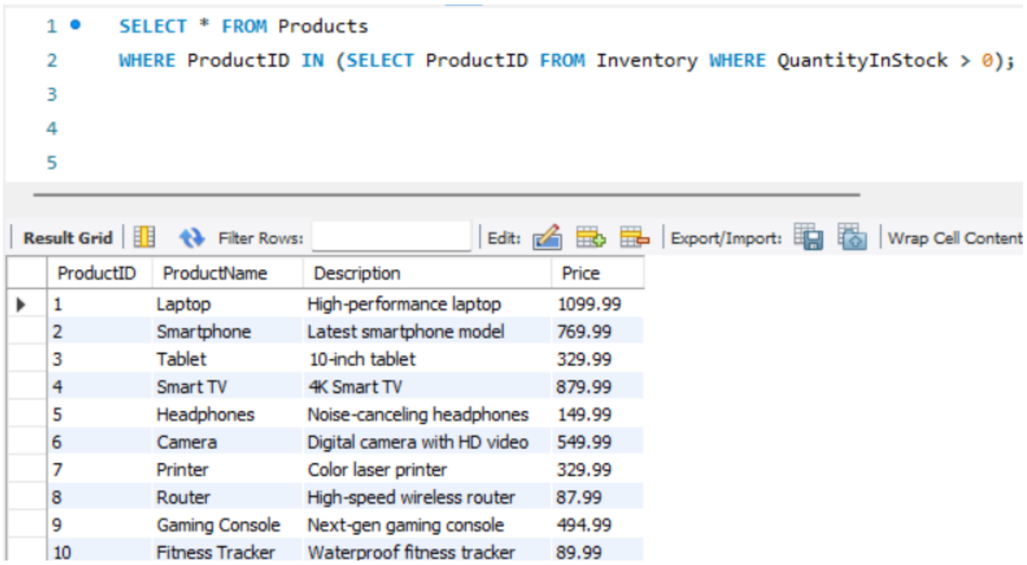
1. Write an SQL query to find out which customers have not placed any orders.



```sql
SELECT customer_id, first_name, last_name
FROM Customers
WHERE customer_id NOT IN (SELECT DISTINCT CustomerID FROM Orders);
```

customer_id	first_name	last_name
10	Samuel	Harris
11	Dave	Miller

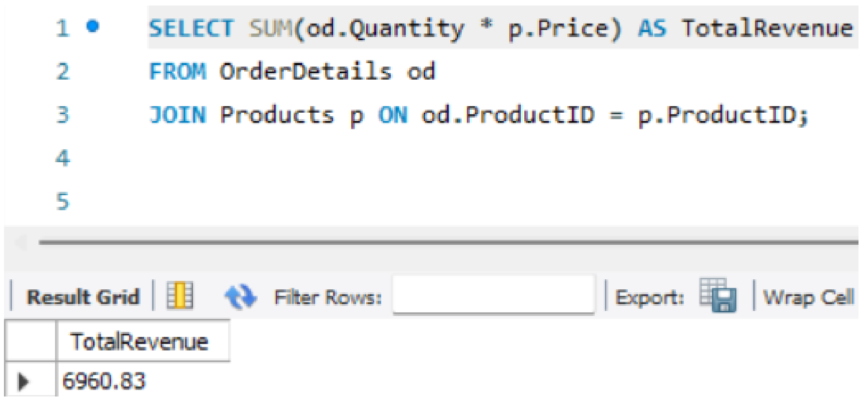
2. Write an SQL query to find the total number of products available for sale.



```sql
SELECT * FROM Products
WHERE ProductID IN (SELECT ProductID FROM Inventory WHERE QuantityInStock > 0);
```

ProductID	ProductName	Description	Price
1	Laptop	High-performance laptop	1099.99
2	Smartphone	Latest smartphone model	769.99
3	Tablet	10-inch tablet	329.99
4	Smart TV	4K Smart TV	879.99
5	Headphones	Noise-canceling headphones	149.99
6	Camera	Digital camera with HD video	549.99
7	Printer	Color laser printer	329.99
8	Router	High-speed wireless router	87.99
9	Gaming Console	Next-gen gaming console	494.99
10	Fitness Tracker	Waterproof fitness tracker	89.99

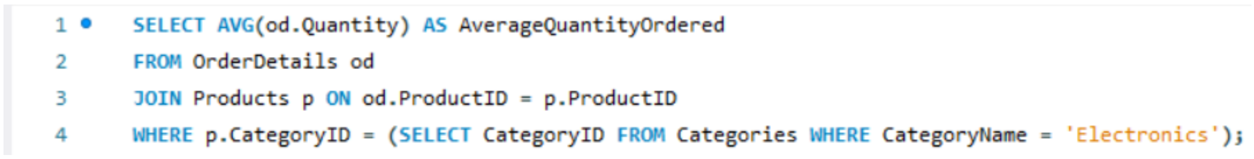
3. Write an SQL query to calculate the total revenue generated by TechShop.



```sql
SELECT SUM(od.Quantity * p.Price) AS TotalRevenue
FROM OrderDetails od
JOIN Products p ON od.ProductID = p.ProductID;
```

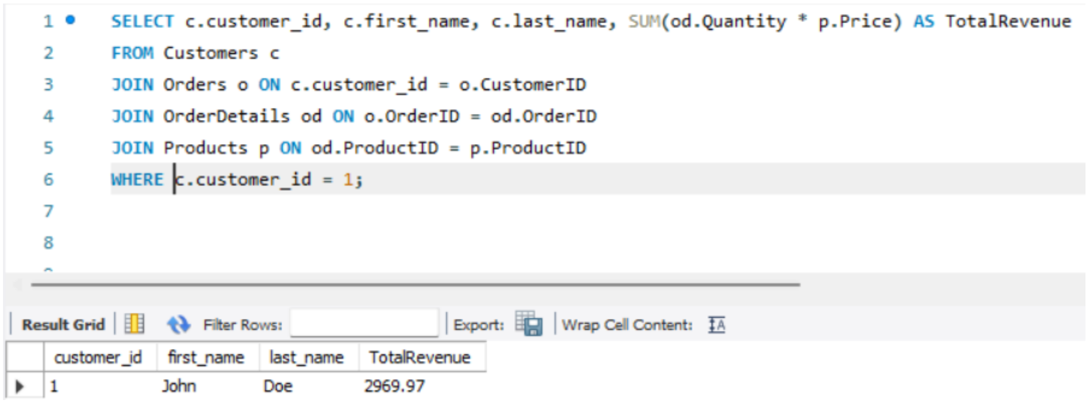
TotalRevenue
6960.83

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.



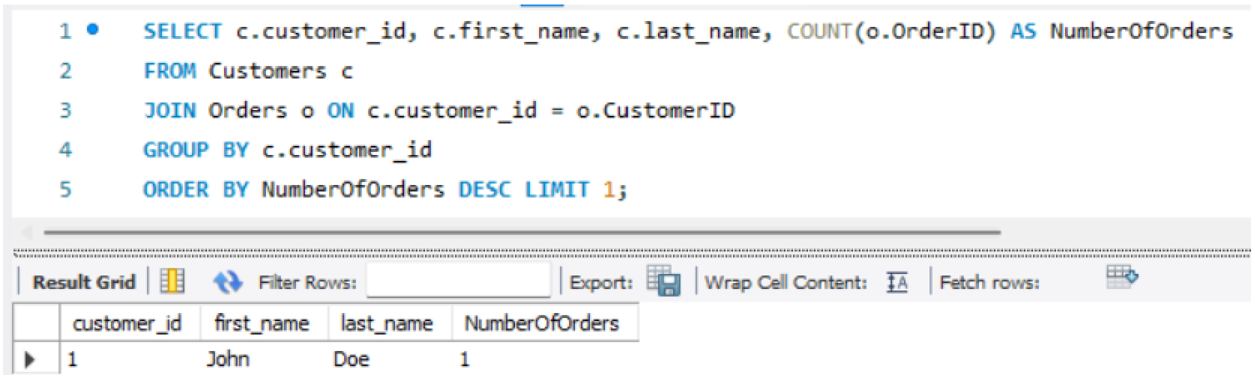
```sql
SELECT AVG(od.Quantity) AS AverageQuantityOrdered
FROM OrderDetails od
JOIN Products p ON od.ProductID = p.ProductID
WHERE p.CategoryID = (SELECT CategoryID FROM Categories WHERE CategoryName = 'Electronics');
```

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.


```sql
SELECT c.customer_id, c.first_name, c.last_name, SUM(od.Quantity * p.Price) AS TotalRevenue
FROM Customers c
JOIN Orders o ON c.customer_id = o.CustomerID
JOIN OrderDetails od ON o.OrderID = od.OrderID
JOIN Products p ON od.ProductID = p.ProductID
WHERE c.customer_id = 1;
```

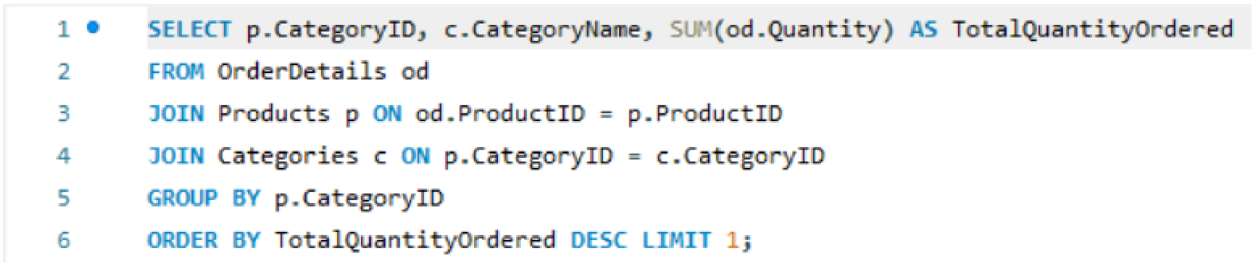
customer_id	first_name	last_name	TotalRevenue
1	John	Doe	2969.97

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.

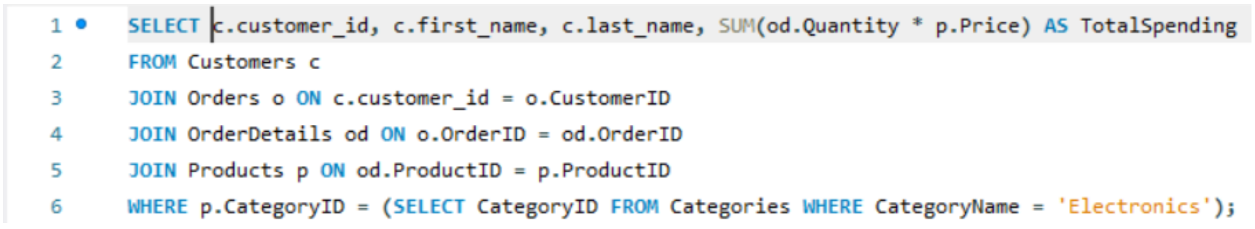

```sql
SELECT c.customer_id, c.first_name, c.last_name, COUNT(o.OrderID) AS NumberOfOrders
FROM Customers c
JOIN Orders o ON c.customer_id = o.CustomerID
GROUP BY c.customer_id
ORDER BY NumberOfOrders DESC LIMIT 1;
```

customer_id	first_name	last_name	NumberOfOrders
1	John	Doe	1

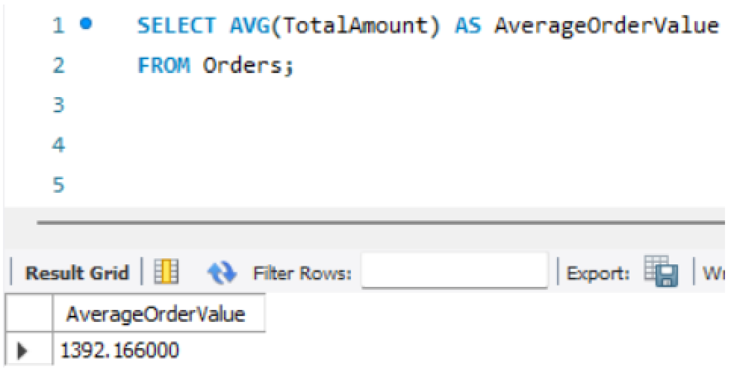
7. Write an SQL query to find the most popular product category, which is the one with the highest total quantity ordered across all orders.


```sql
SELECT p.CategoryID, c.CategoryName, SUM(od.Quantity) AS TotalQuantityOrdered
FROM OrderDetails od
JOIN Products p ON od.ProductID = p.ProductID
JOIN Categories c ON p.CategoryID = c.CategoryID
GROUP BY p.CategoryID
ORDER BY TotalQuantityOrdered DESC LIMIT 1;
```

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.


```sql
SELECT c.customer_id, c.first_name, c.last_name, SUM(od.Quantity * p.Price) AS TotalSpending
FROM Customers c
JOIN Orders o ON c.customer_id = o.CustomerID
JOIN OrderDetails od ON o.OrderID = od.OrderID
JOIN Products p ON od.ProductID = p.ProductID
WHERE p.CategoryID = (SELECT CategoryID FROM Categories WHERE CategoryName = 'Electronics');
```

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

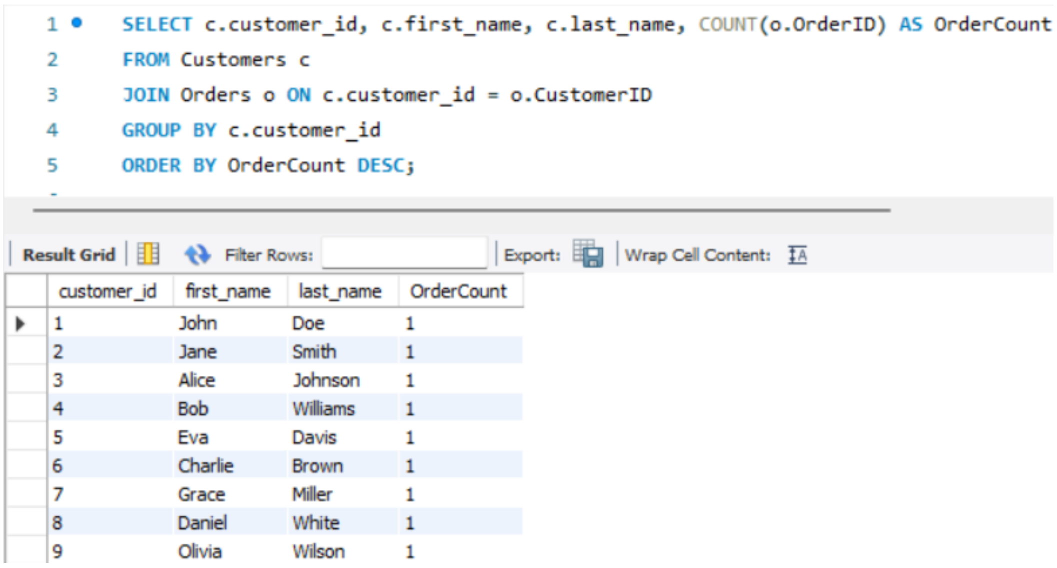


```sql
SELECT AVG(TotalAmount) AS AverageOrderValue
FROM Orders;
```

Result Grid | Filter Rows: | Export: | Wr

AverageOrderValue
1392.166000

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.



```sql
SELECT c.customer_id, c.first_name, c.last_name, COUNT(o.OrderID) AS OrderCount
FROM Customers c
JOIN Orders o ON c.customer_id = o.CustomerID
GROUP BY c.customer_id
ORDER BY OrderCount DESC;
```

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

customer_id	first_name	last_name	OrderCount
1	John	Doe	1
2	Jane	Smith	1
3	Alice	Johnson	1
4	Bob	Williams	1
5	Eva	Davis	1
6	Charlie	Brown	1
7	Grace	Miller	1
8	Daniel	White	1
9	Olivia	Wilson	1