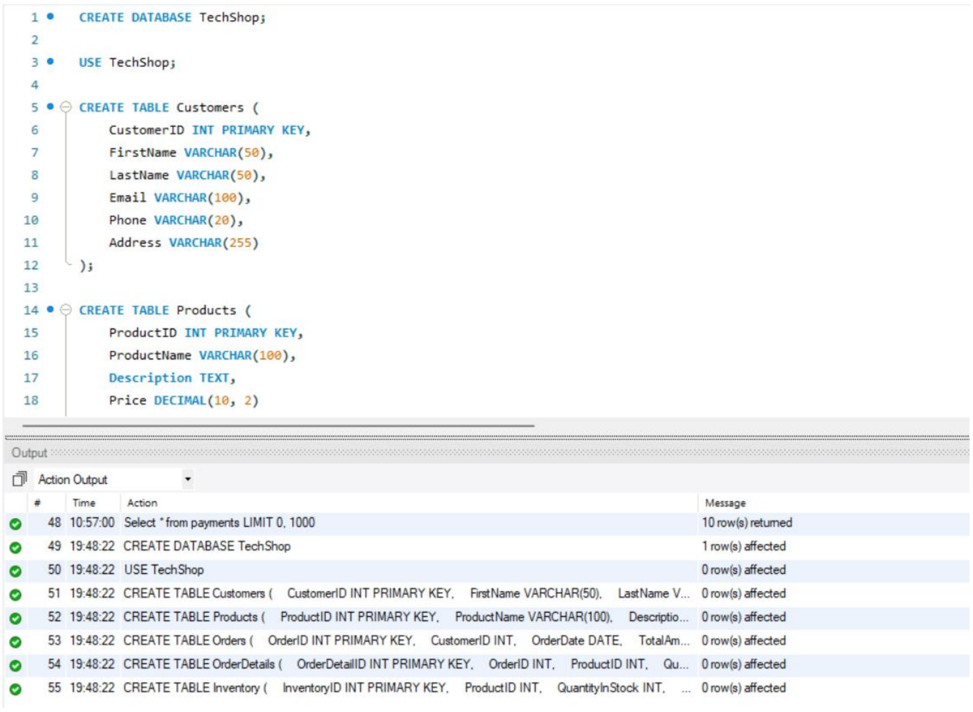
TASK 1

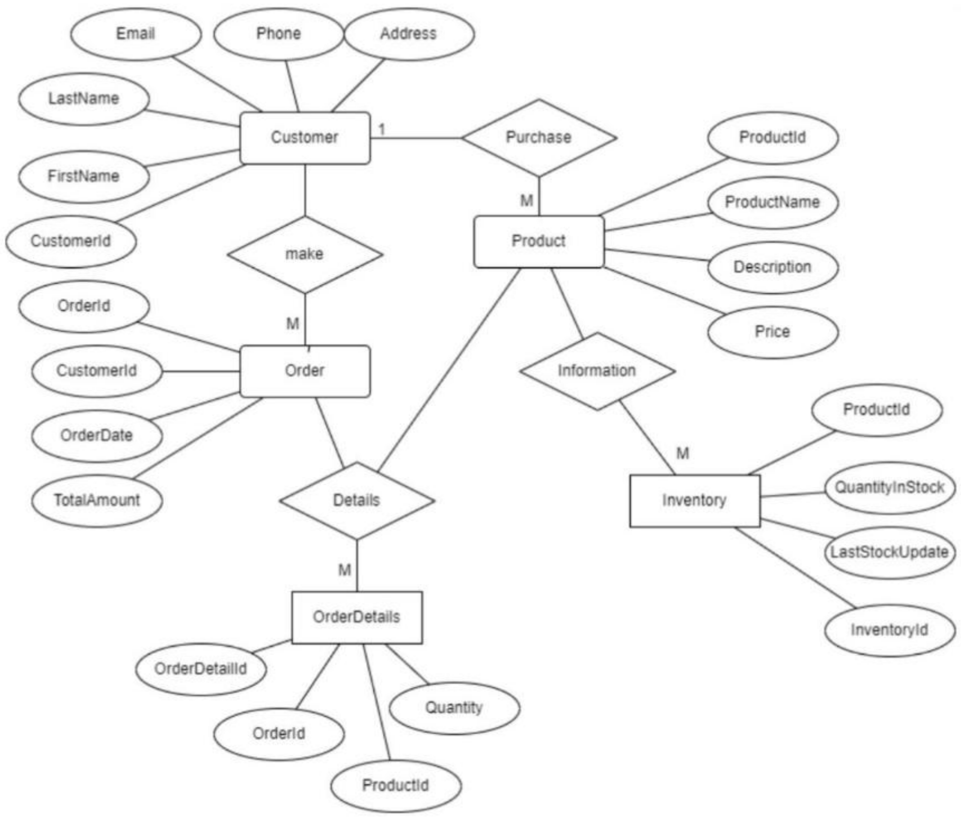
1. Create the database named "TechShop"



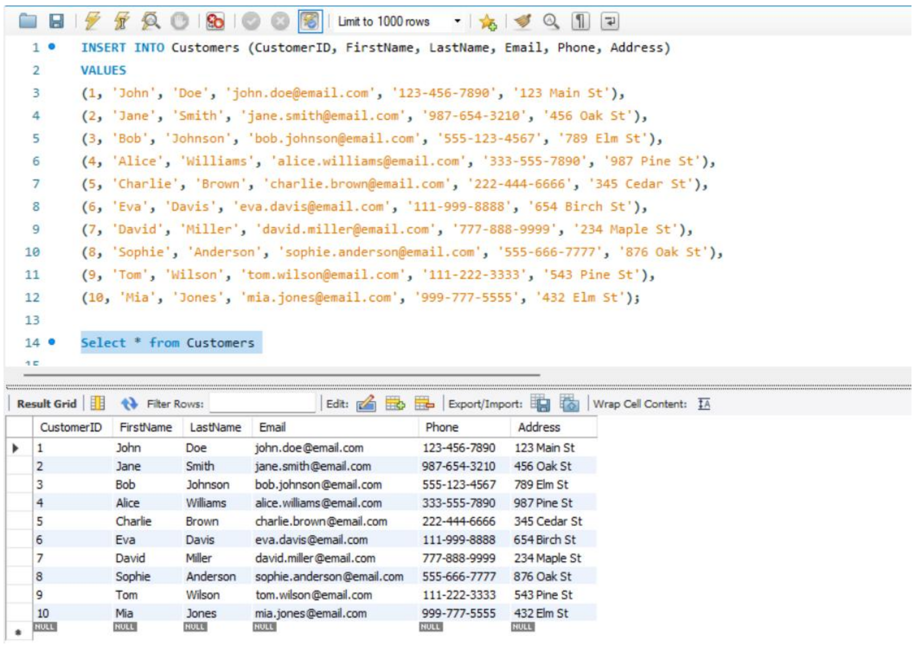
```sql
CREATE DATABASE TechShop;

USE TechShop;

CREATE TABLE Customers (
    CustomerID INT PRIMARY KEY,
    FirstName VARCHAR(50),
    LastName VARCHAR(50),
    Email VARCHAR(100),
    Phone VARCHAR(20),
    Address VARCHAR(255)
);

CREATE TABLE Products (
    ProductID INT PRIMARY KEY,
    ProductName VARCHAR(100),
    Description TEXT,
    Price DECIMAL(10, 2)
```

```
Output
Action Output
 #   Time      Action                                                                                              Message
 48  10:57:00  Select * from payments LIMIT 0, 1000                                                                10 row(s) returned
 49  19:48:22  CREATE DATABASE TechShop                                                                            1 row(s) affected
 50  19:48:22  USE TechShop                                                                                        0 row(s) affected
 51  19:48:22  CREATE TABLE Customers (   CustomerID INT PRIMARY KEY,   FirstName VARCHAR(50),   LastName V...     0 row(s) affected
 52  19:48:22  CREATE TABLE Products (   ProductID INT PRIMARY KEY,   ProductName VARCHAR(100),   Descriptio...    0 row(s) affected
 53  19:48:22  CREATE TABLE Orders (   OrderID INT PRIMARY KEY,   CustomerID INT,   OrderDate DATE,   TotalAm...   0 row(s) affected
 54  19:48:22  CREATE TABLE OrderDetails (   OrderDetailID INT PRIMARY KEY,   OrderID INT,   ProductID INT,   Qu...  0 row(s) affected
 55  19:48:22  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.

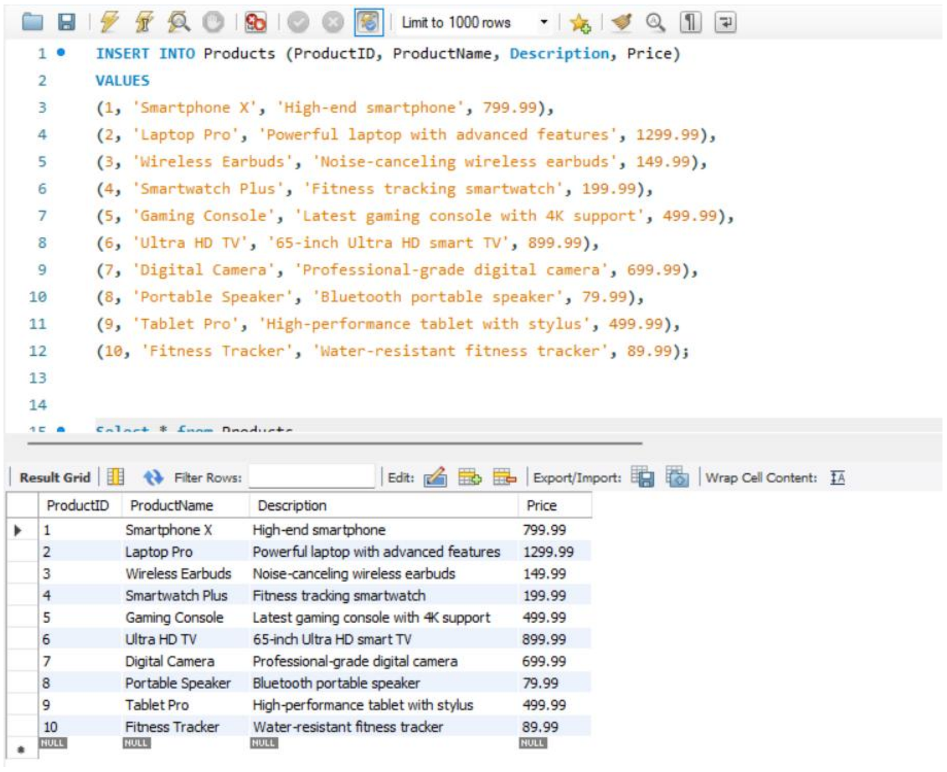


5. Insert at least 10 sample records into each of the following tables.
a. Customers
b. Products
c. Orders
d. OrderDetails
e. Inventory



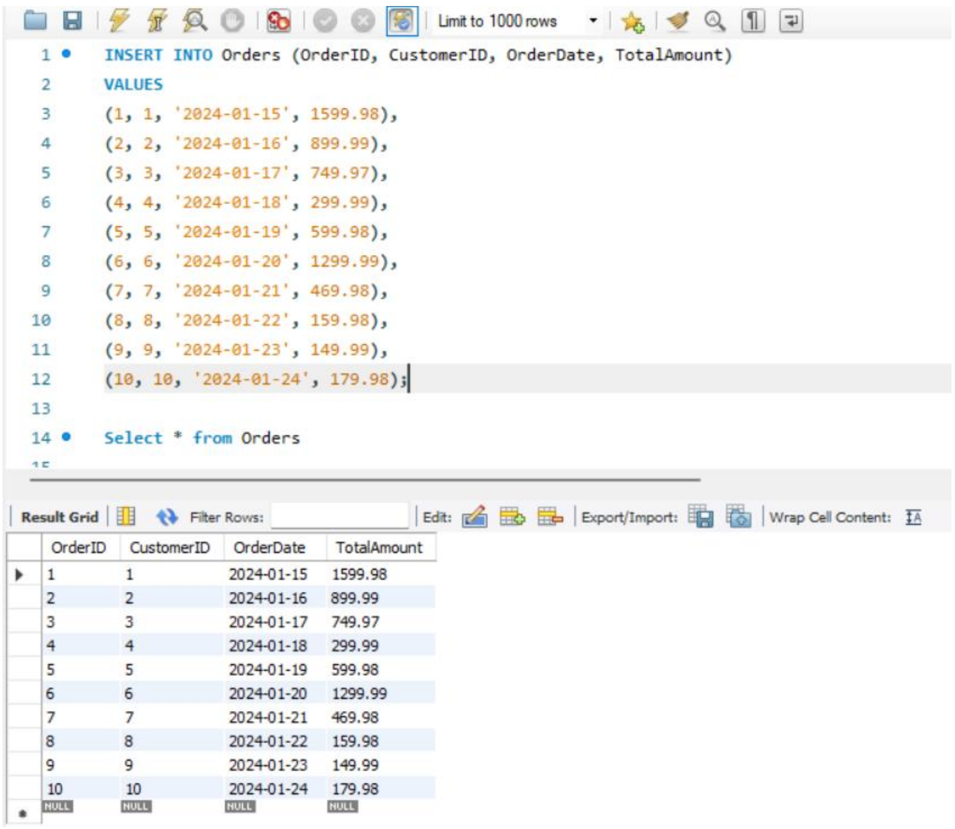
```sql
INSERT INTO Customers (CustomerID, FirstName, LastName, Email, Phone, Address)
VALUES
(1, 'John', 'Doe', 'john.doe@email.com', '123-456-7890', '123 Main St'),
(2, 'Jane', 'Smith', 'jane.smith@email.com', '987-654-3210', '456 Oak St'),
(3, 'Bob', 'Johnson', 'bob.johnson@email.com', '555-123-4567', '789 Elm St'),
(4, 'Alice', 'Williams', 'alice.williams@email.com', '333-555-7890', '987 Pine St'),
(5, 'Charlie', 'Brown', 'charlie.brown@email.com', '222-444-6666', '345 Cedar St'),
(6, 'Eva', 'Davis', 'eva.davis@email.com', '111-999-8888', '654 Birch St'),
(7, 'David', 'Miller', 'david.miller@email.com', '777-888-9999', '234 Maple St'),
(8, 'Sophie', 'Anderson', 'sophie.anderson@email.com', '555-666-7777', '876 Oak St'),
(9, 'Tom', 'Wilson', 'tom.wilson@email.com', '111-222-3333', '543 Pine St'),
(10, 'Mia', 'Jones', 'mia.jones@email.com', '999-777-5555', '432 Elm St');

Select * from Customers
```

CustomerID	FirstName	LastName	Email	Phone	Address
1	John	Doe	john.doe@email.com	123-456-7890	123 Main St
2	Jane	Smith	jane.smith@email.com	987-654-3210	456 Oak St
3	Bob	Johnson	bob.johnson@email.com	555-123-4567	789 Elm St
4	Alice	Williams	alice.williams@email.com	333-555-7890	987 Pine St
5	Charlie	Brown	charlie.brown@email.com	222-444-6666	345 Cedar St
6	Eva	Davis	eva.davis@email.com	111-999-8888	654 Birch St
7	David	Miller	david.miller@email.com	777-888-9999	234 Maple St
8	Sophie	Anderson	sophie.anderson@email.com	555-666-7777	876 Oak St
9	Tom	Wilson	tom.wilson@email.com	111-222-3333	543 Pine St
10	Mia	Jones	mia.jones@email.com	999-777-5555	432 Elm St
NULL	NULL	NULL	NULL	NULL	NULL



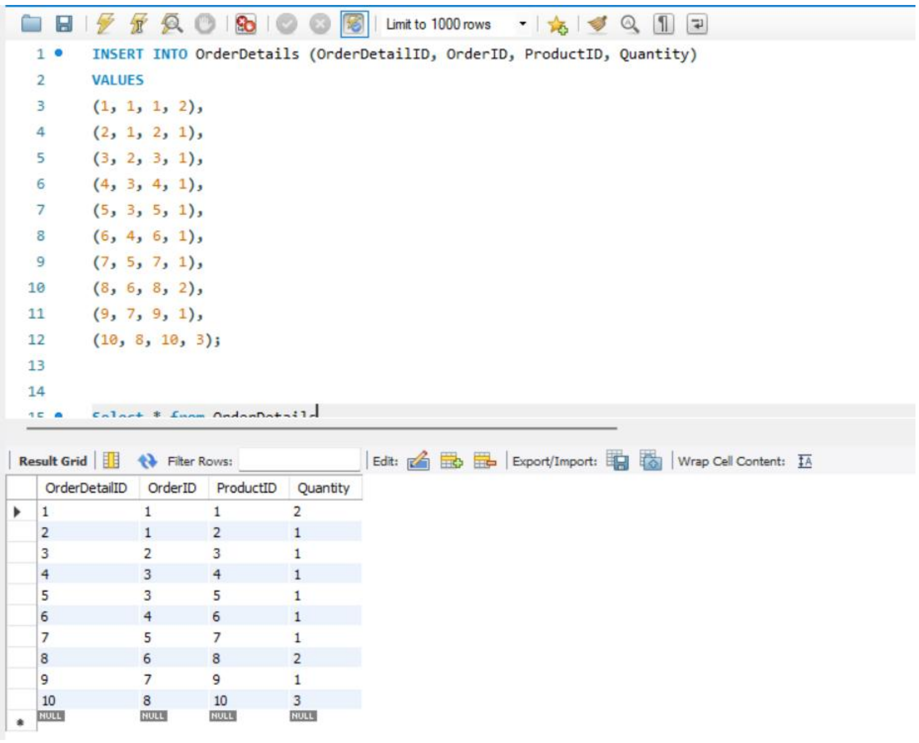
```sql
INSERT INTO Products (ProductID, ProductName, Description, Price)
VALUES
(1, 'Smartphone X', 'High-end smartphone', 799.99),
(2, 'Laptop Pro', 'Powerful laptop with advanced features', 1299.99),
(3, 'Wireless Earbuds', 'Noise-canceling wireless earbuds', 149.99),
(4, 'Smartwatch Plus', 'Fitness tracking smartwatch', 199.99),
(5, 'Gaming Console', 'Latest gaming console with 4K support', 499.99),
(6, 'Ultra HD TV', '65-inch Ultra HD smart TV', 899.99),
(7, 'Digital Camera', 'Professional-grade digital camera', 699.99),
(8, 'Portable Speaker', 'Bluetooth portable speaker', 79.99),
(9, 'Tablet Pro', 'High-performance tablet with stylus', 499.99),
(10, 'Fitness Tracker', 'Water-resistant fitness tracker', 89.99);


Select * from Products
```

ProductID	ProductName	Description	Price
1	Smartphone X	High-end smartphone	799.99
2	Laptop Pro	Powerful laptop with advanced features	1299.99
3	Wireless Earbuds	Noise-canceling wireless earbuds	149.99
4	Smartwatch Plus	Fitness tracking smartwatch	199.99
5	Gaming Console	Latest gaming console with 4K support	499.99
6	Ultra HD TV	65-inch Ultra HD smart TV	899.99
7	Digital Camera	Professional-grade digital camera	699.99
8	Portable Speaker	Bluetooth portable speaker	79.99
9	Tablet Pro	High-performance tablet with stylus	499.99
10	Fitness Tracker	Water-resistant fitness tracker	89.99
NULL	NULL	NULL	NULL



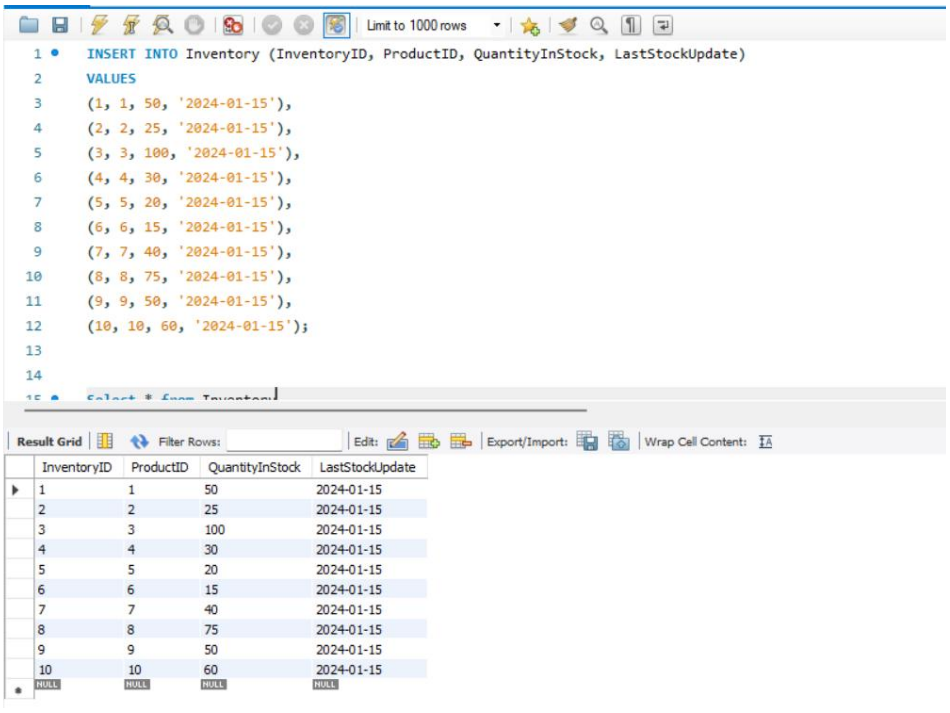
```sql
INSERT INTO Orders (OrderID, CustomerID, OrderDate, TotalAmount)
VALUES
(1, 1, '2024-01-15', 1599.98),
(2, 2, '2024-01-16', 899.99),
(3, 3, '2024-01-17', 749.97),
(4, 4, '2024-01-18', 299.99),
(5, 5, '2024-01-19', 599.98),
(6, 6, '2024-01-20', 1299.99),
(7, 7, '2024-01-21', 469.98),
(8, 8, '2024-01-22', 159.98),
(9, 9, '2024-01-23', 149.99),
(10, 10, '2024-01-24', 179.98);

Select * from Orders
```

OrderID	CustomerID	OrderDate	TotalAmount
1	1	2024-01-15	1599.98
2	2	2024-01-16	899.99
3	3	2024-01-17	749.97
4	4	2024-01-18	299.99
5	5	2024-01-19	599.98
6	6	2024-01-20	1299.99
7	7	2024-01-21	469.98
8	8	2024-01-22	159.98
9	9	2024-01-23	149.99
10	10	2024-01-24	179.98
NULL	NULL	NULL	NULL



```sql
INSERT INTO OrderDetails (OrderDetailID, OrderID, ProductID, Quantity)
VALUES
(1, 1, 1, 2),
(2, 1, 2, 1),
(3, 2, 3, 1),
(4, 3, 4, 1),
(5, 3, 5, 1),
(6, 4, 6, 1),
(7, 5, 7, 1),
(8, 6, 8, 2),
(9, 7, 9, 1),
(10, 8, 10, 3);


Select * from OrderDetails
```

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

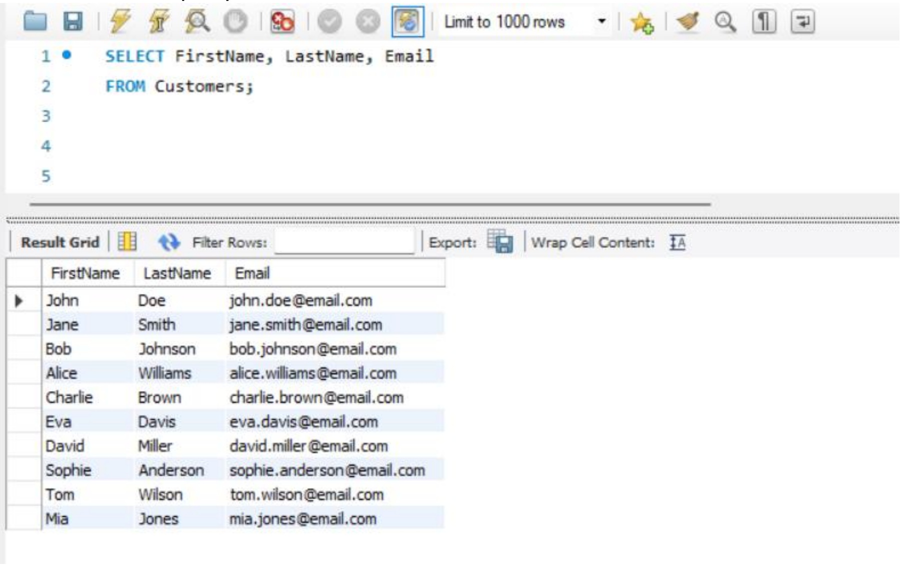


```sql
INSERT INTO Inventory (InventoryID, ProductID, QuantityInStock, LastStockUpdate)
VALUES
(1, 1, 50, '2024-01-15'),
(2, 2, 25, '2024-01-15'),
(3, 3, 100, '2024-01-15'),
(4, 4, 30, '2024-01-15'),
(5, 5, 20, '2024-01-15'),
(6, 6, 15, '2024-01-15'),
(7, 7, 40, '2024-01-15'),
(8, 8, 75, '2024-01-15'),
(9, 9, 50, '2024-01-15'),
(10, 10, 60, '2024-01-15');


Select * from Inventory;
```

InventoryID	ProductID	QuantityInStock	LastStockUpdate
1	1	50	2024-01-15
2	2	25	2024-01-15
3	3	100	2024-01-15
4	4	30	2024-01-15
5	5	20	2024-01-15
6	6	15	2024-01-15
7	7	40	2024-01-15
8	8	75	2024-01-15
9	9	50	2024-01-15
10	10	60	2024-01-15
NULL	NULL	NULL	NULL

Tasks 2: Select, Where, Between, AND, LIKE:

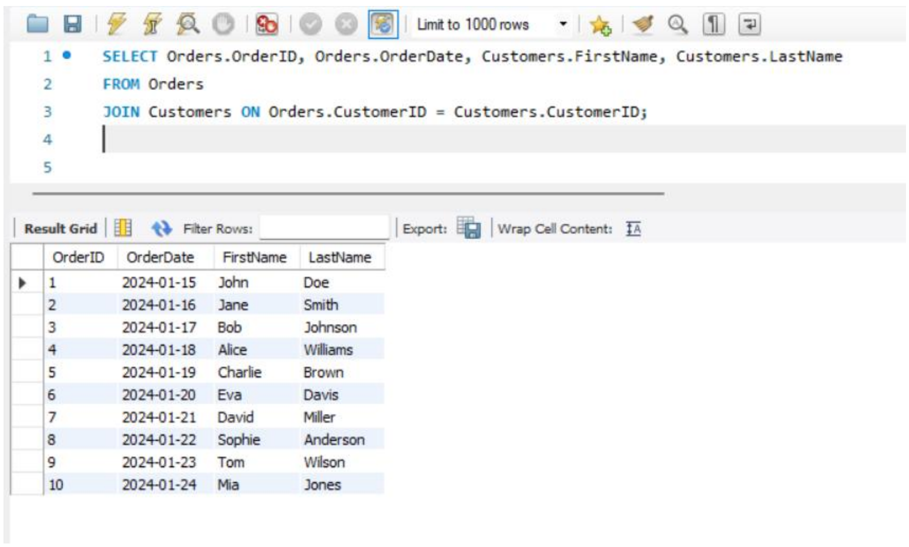
1. Write an SQL query to retrieve the names and emails of all customers.



```sql
SELECT FirstName, LastName, Email
FROM Customers;
```

FirstName	LastName	Email
John	Doe	john.doe@email.com
Jane	Smith	jane.smith@email.com
Bob	Johnson	bob.johnson@email.com
Alice	Williams	alice.williams@email.com
Charlie	Brown	charlie.brown@email.com
Eva	Davis	eva.davis@email.com
David	Miller	david.miller@email.com
Sophie	Anderson	sophie.anderson@email.com
Tom	Wilson	tom.wilson@email.com
Mia	Jones	mia.jones@email.com

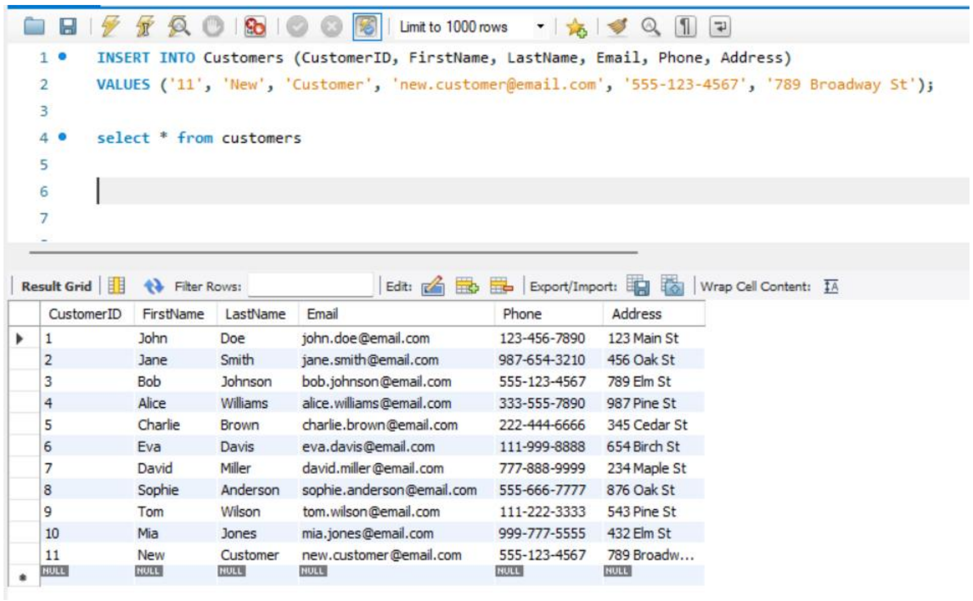
2. Write an SQL query to list all orders with their order dates and corresponding customer names.



```sql
SELECT Orders.OrderID, Orders.OrderDate, Customers.FirstName, Customers.LastName
FROM Orders
JOIN Customers ON Orders.CustomerID = Customers.CustomerID;
```

OrderID	OrderDate	FirstName	LastName
1	2024-01-15	John	Doe
2	2024-01-16	Jane	Smith
3	2024-01-17	Bob	Johnson
4	2024-01-18	Alice	Williams
5	2024-01-19	Charlie	Brown
6	2024-01-20	Eva	Davis
7	2024-01-21	David	Miller
8	2024-01-22	Sophie	Anderson
9	2024-01-23	Tom	Wilson
10	2024-01-24	Mia	Jones

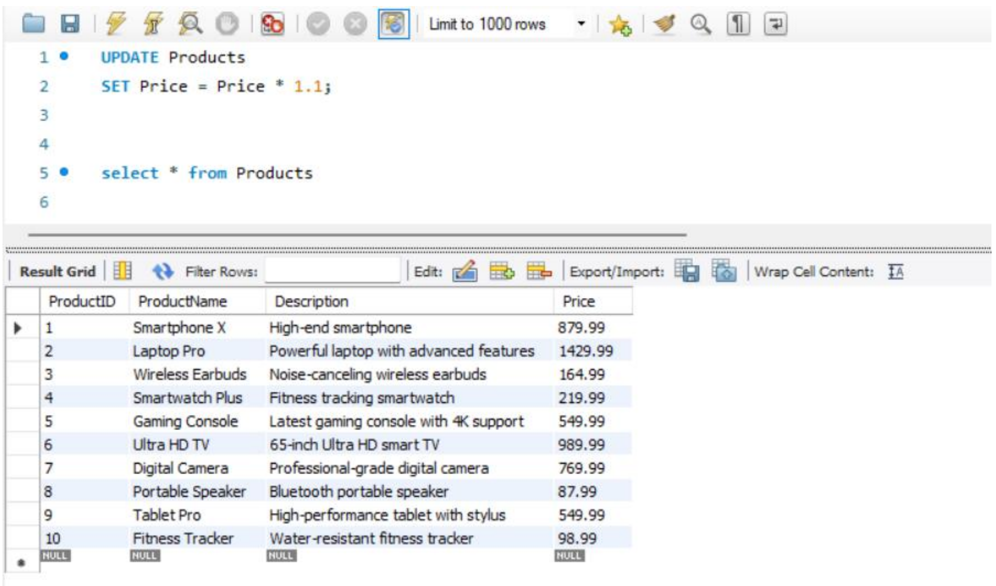
3. Write an SQL query to insert a new customer record into the "Customers" table. Include customer information such as name, email, and address.



```sql
INSERT INTO Customers (CustomerID, FirstName, LastName, Email, Phone, Address)
VALUES ('11', 'New', 'Customer', 'new.customer@email.com', '555-123-4567', '789 Broadway St');

select * from customers
```

CustomerID	FirstName	LastName	Email	Phone	Address
1	John	Doe	john.doe@email.com	123-456-7890	123 Main St
2	Jane	Smith	jane.smith@email.com	987-654-3210	456 Oak St
3	Bob	Johnson	bob.johnson@email.com	555-123-4567	789 Elm St
4	Alice	Williams	alice.williams@email.com	333-555-7890	987 Pine St
5	Charlie	Brown	charlie.brown@email.com	222-444-6666	345 Cedar St
6	Eva	Davis	eva.davis@email.com	111-999-8888	654 Birch St
7	David	Miller	david.miller@email.com	777-888-9999	234 Maple St
8	Sophie	Anderson	sophie.anderson@email.com	555-666-7777	876 Oak St
9	Tom	Wilson	tom.wilson@email.com	111-222-3333	543 Pine St
10	Mia	Jones	mia.jones@email.com	999-777-5555	432 Elm St
11	New	Customer	new.customer@email.com	555-123-4567	789 Broadw...
NULL	NULL	NULL	NULL	NULL	NULL

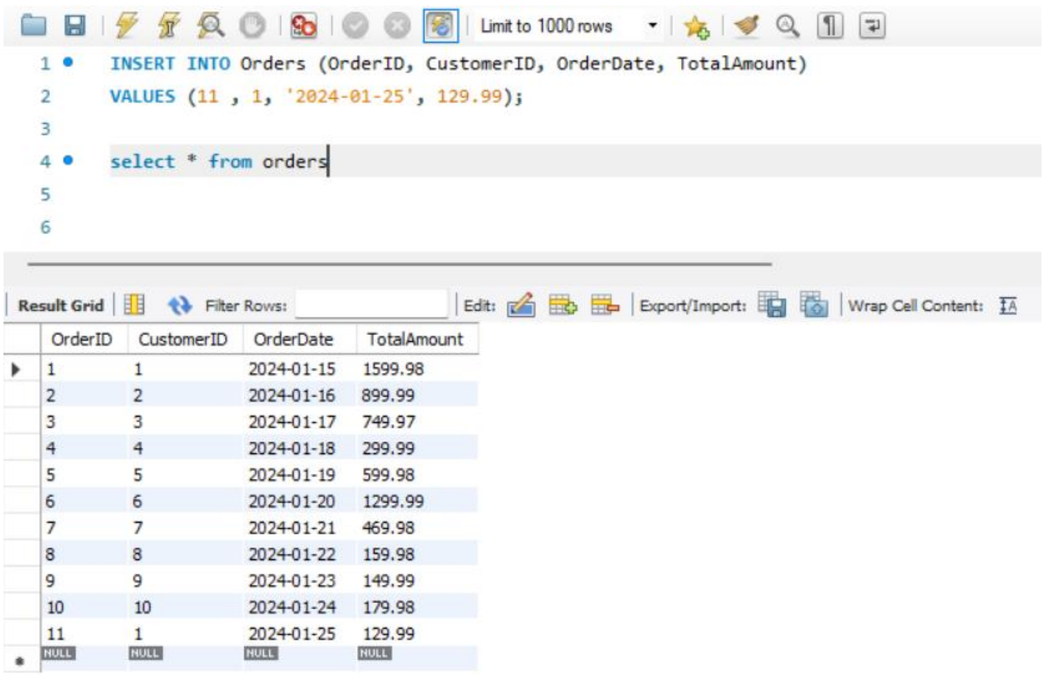
4. Write an SQL query to update the prices of all electronic gadgets in the "Products" table by increasing them by 10%.



```sql
UPDATE Products
SET Price = Price * 1.1;


select * from Products
```

ProductID	ProductName	Description	Price
1	Smartphone X	High-end smartphone	879.99
2	Laptop Pro	Powerful laptop with advanced features	1429.99
3	Wireless Earbuds	Noise-canceling wireless earbuds	164.99
4	Smartwatch Plus	Fitness tracking smartwatch	219.99
5	Gaming Console	Latest gaming console with 4K support	549.99
6	Ultra HD TV	65-inch Ultra HD smart TV	989.99
7	Digital Camera	Professional-grade digital camera	769.99
8	Portable Speaker	Bluetooth portable speaker	87.99
9	Tablet Pro	High-performance tablet with stylus	549.99
10	Fitness Tracker	Water-resistant fitness tracker	98.99
NULL	NULL	NULL	NULL

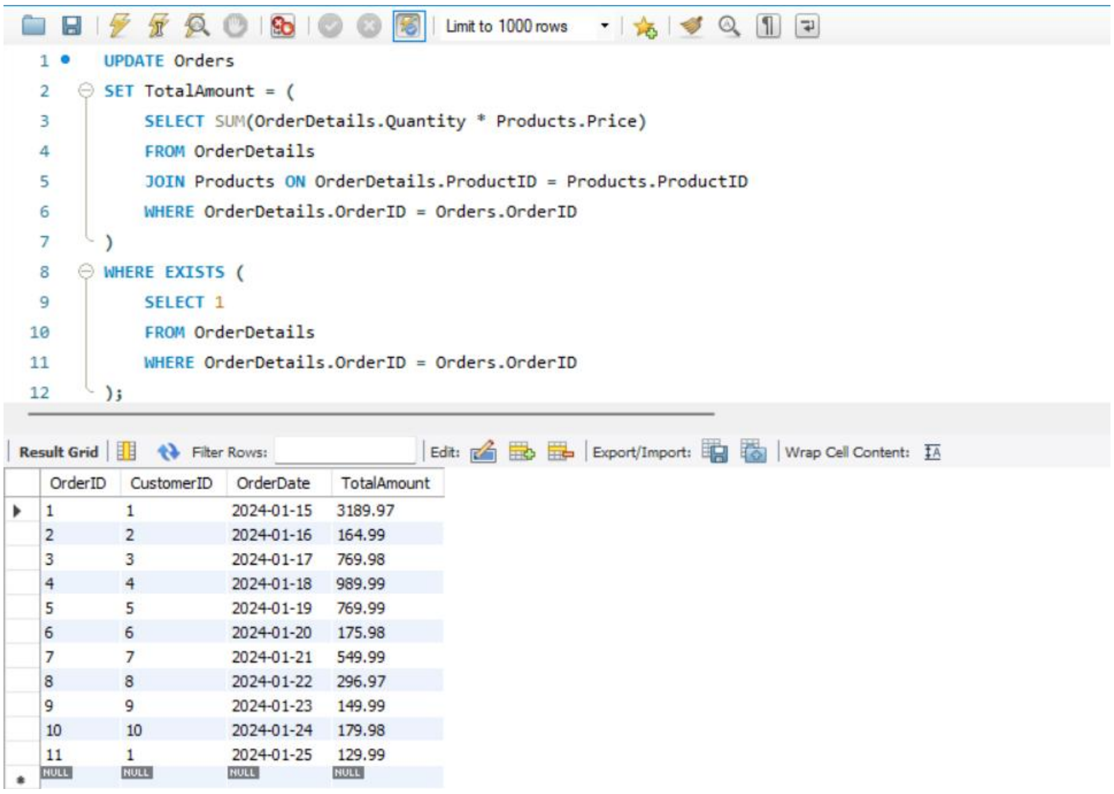
5. Write an SQL query to delete a specific order and its associated order details from the "Orders" and "OrderDetails" tables. Allow users to input the order ID as a parameter.
6. Write an SQL query to insert a new order into the "Orders" table. Include the customer ID, order date, and any other necessary information.



```sql
INSERT INTO Orders (OrderID, CustomerID, OrderDate, TotalAmount)
VALUES (11 , 1, '2024-01-25', 129.99);

select * from orders
```

OrderID	CustomerID	OrderDate	TotalAmount
1	1	2024-01-15	1599.98
2	2	2024-01-16	899.99
3	3	2024-01-17	749.97
4	4	2024-01-18	299.99
5	5	2024-01-19	599.98
6	6	2024-01-20	1299.99
7	7	2024-01-21	469.98
8	8	2024-01-22	159.98
9	9	2024-01-23	149.99
10	10	2024-01-24	179.98
11	1	2024-01-25	129.99
NULL	NULL	NULL	NULL

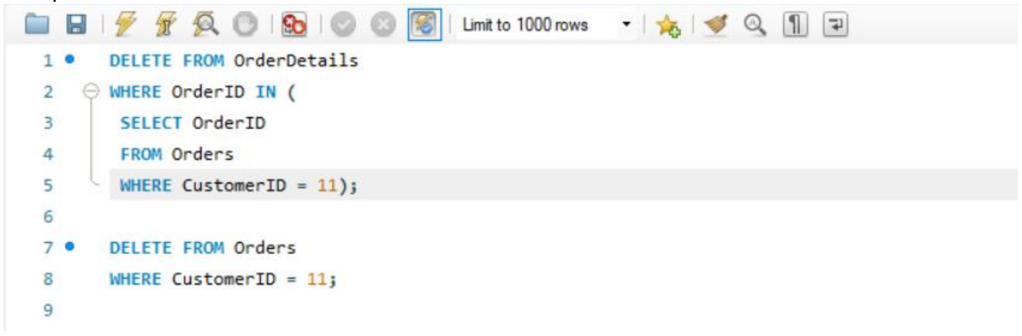
7. Write an SQL query to update the contact information (e.g., email and address) of a specific customer in the "Customers" table. Allow users to input the customer ID and new contact information.
8. Write an SQL query to recalculate and update the total cost of each order in the "Orders" table based on the prices and quantities in the "OrderDetails" table.



```sql
UPDATE Orders
SET TotalAmount = (
    SELECT SUM(OrderDetails.Quantity * Products.Price)
    FROM OrderDetails
    JOIN Products ON OrderDetails.ProductID = Products.ProductID
    WHERE OrderDetails.OrderID = Orders.OrderID
)
WHERE EXISTS (
    SELECT 1
    FROM OrderDetails
    WHERE OrderDetails.OrderID = Orders.OrderID
);
```

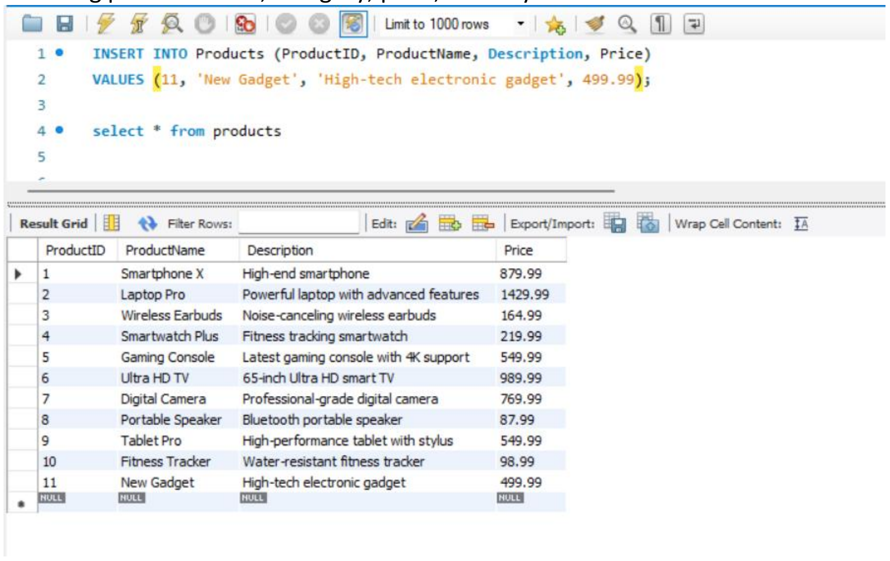
OrderID	CustomerID	OrderDate	TotalAmount
1	1	2024-01-15	3189.97
2	2	2024-01-16	164.99
3	3	2024-01-17	769.98
4	4	2024-01-18	989.99
5	5	2024-01-19	769.99
6	6	2024-01-20	175.98
7	7	2024-01-21	549.99
8	8	2024-01-22	296.97
9	9	2024-01-23	149.99
10	10	2024-01-24	179.98
11	1	2024-01-25	129.99
NULL	NULL	NULL	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.



```sql
DELETE FROM OrderDetails
WHERE OrderID IN (
 SELECT OrderID
 FROM Orders
 WHERE CustomerID = 11);

DELETE FROM Orders
WHERE CustomerID = 11;

```

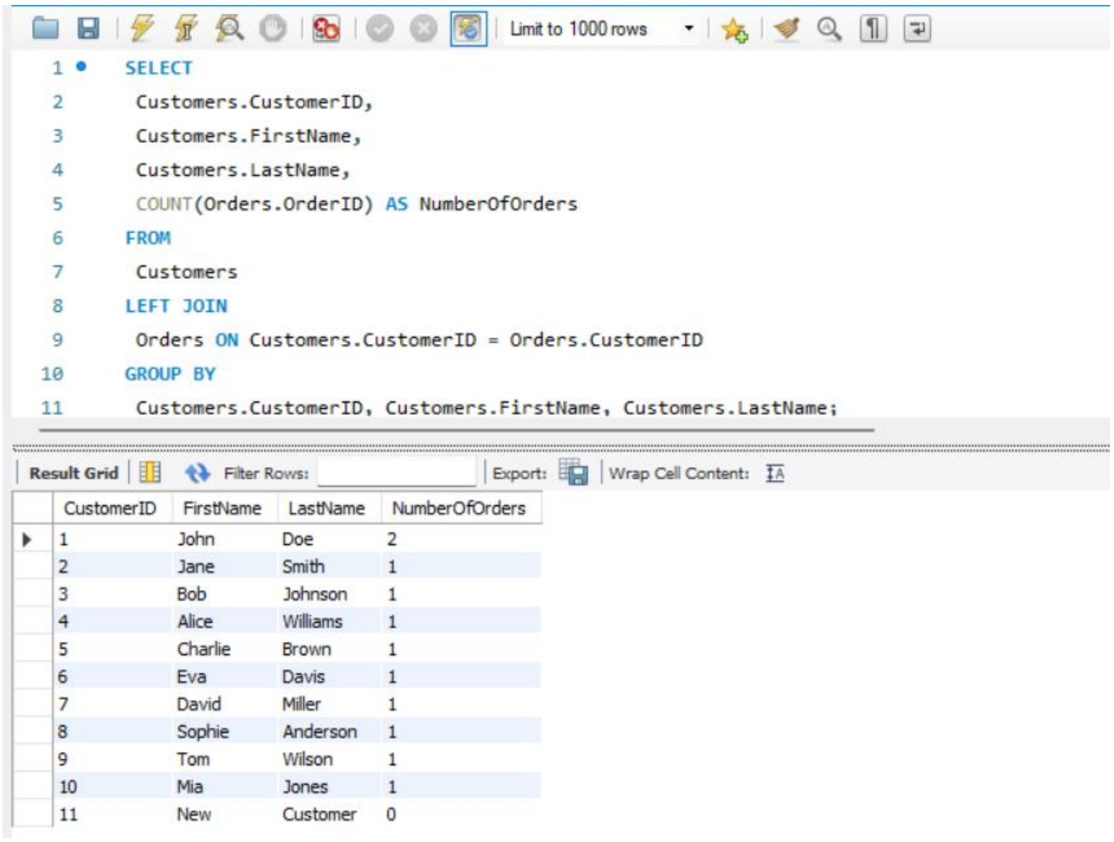
10. Write an SQL query to insert a new electronic gadget product into the "Products" table, including product name, category, price, and any other relevant details.



```sql
INSERT INTO Products (ProductID, ProductName, Description, Price)
VALUES (11, 'New Gadget', 'High-tech electronic gadget', 499.99);

select * from products

```

ProductID	ProductName	Description	Price
1	Smartphone X	High-end smartphone	879.99
2	Laptop Pro	Powerful laptop with advanced features	1429.99
3	Wireless Earbuds	Noise-canceling wireless earbuds	164.99
4	Smartwatch Plus	Fitness tracking smartwatch	219.99
5	Gaming Console	Latest gaming console with 4K support	549.99
6	Ultra HD TV	65-inch Ultra HD smart TV	989.99
7	Digital Camera	Professional-grade digital camera	769.99
8	Portable Speaker	Bluetooth portable speaker	87.99
9	Tablet Pro	High-performance tablet with stylus	549.99
10	Fitness Tracker	Water-resistant fitness tracker	98.99
11	New Gadget	High-tech electronic gadget	499.99
NULL	NULL	NULL	NULL

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

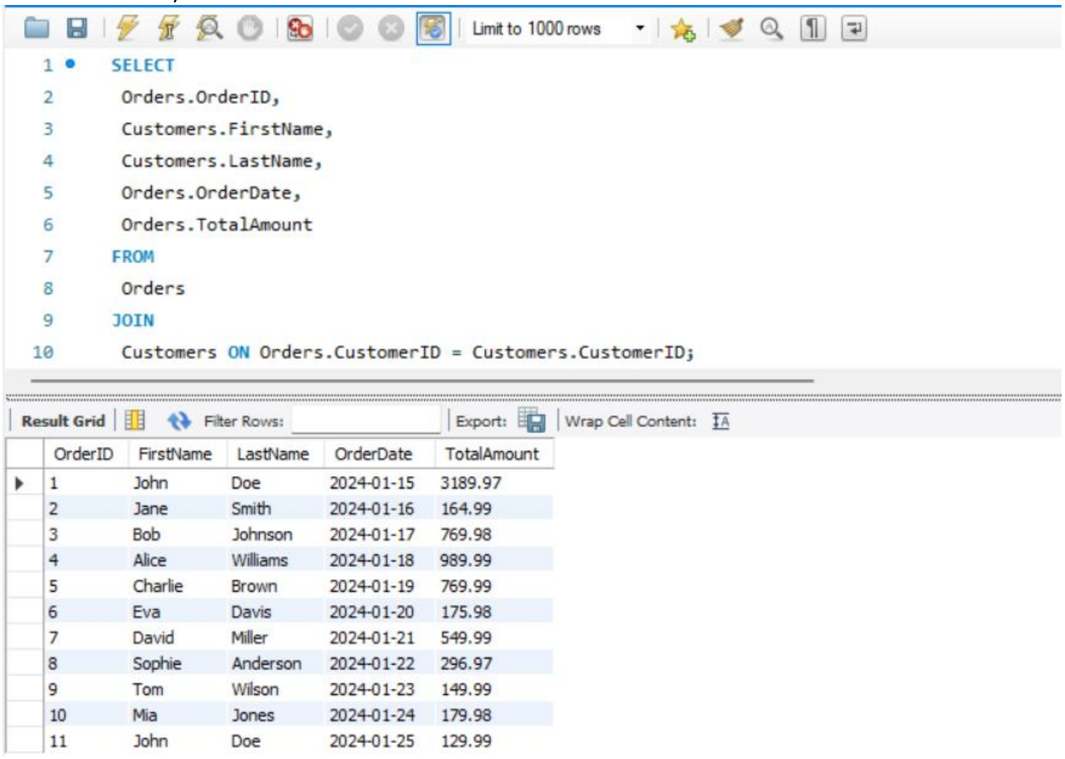


```
SELECT
 Customers.CustomerID,
 Customers.FirstName,
 Customers.LastName,
 COUNT(Orders.OrderID) AS NumberOfOrders
FROM
 Customers
LEFT JOIN
 Orders ON Customers.CustomerID = Orders.CustomerID
GROUP BY
 Customers.CustomerID, Customers.FirstName, Customers.LastName;
```

CustomerID	FirstName	LastName	NumberOfOrders
1	John	Doe	2
2	Jane	Smith	1
3	Bob	Johnson	1
4	Alice	Williams	1
5	Charlie	Brown	1
6	Eva	Davis	1
7	David	Miller	1
8	Sophie	Anderson	1
9	Tom	Wilson	1
10	Mia	Jones	1
11	New	Customer	0

Task 3. Aggregate functions, Having, Order By, GroupBy and Joins:

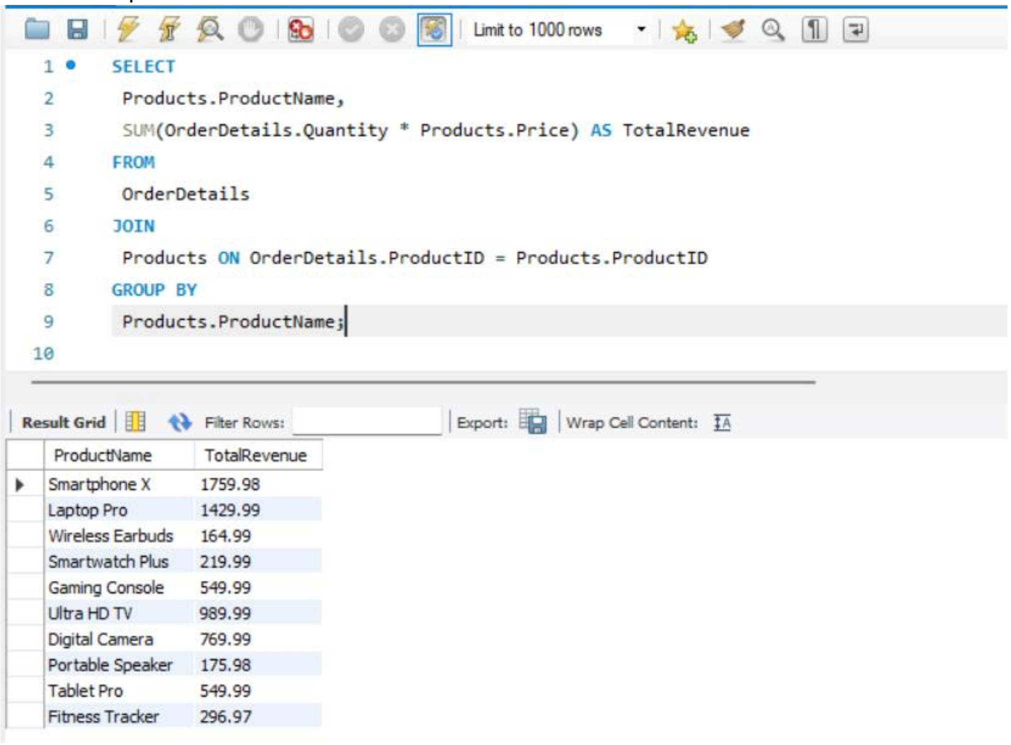
1. Write an SQL query to retrieve a list of all orders along with customer information (e.g., customer name) for each order.



```
SELECT
 Orders.OrderID,
 Customers.FirstName,
 Customers.LastName,
 Orders.OrderDate,
 Orders.TotalAmount
FROM
 Orders
JOIN
 Customers ON Orders.CustomerID = Customers.CustomerID;
```

OrderID	FirstName	LastName	OrderDate	TotalAmount
1	John	Doe	2024-01-15	3189.97
2	Jane	Smith	2024-01-16	164.99
3	Bob	Johnson	2024-01-17	769.98
4	Alice	Williams	2024-01-18	989.99
5	Charlie	Brown	2024-01-19	769.99
6	Eva	Davis	2024-01-20	175.98
7	David	Miller	2024-01-21	549.99
8	Sophie	Anderson	2024-01-22	296.97
9	Tom	Wilson	2024-01-23	149.99
10	Mia	Jones	2024-01-24	179.98
11	John	Doe	2024-01-25	129.99

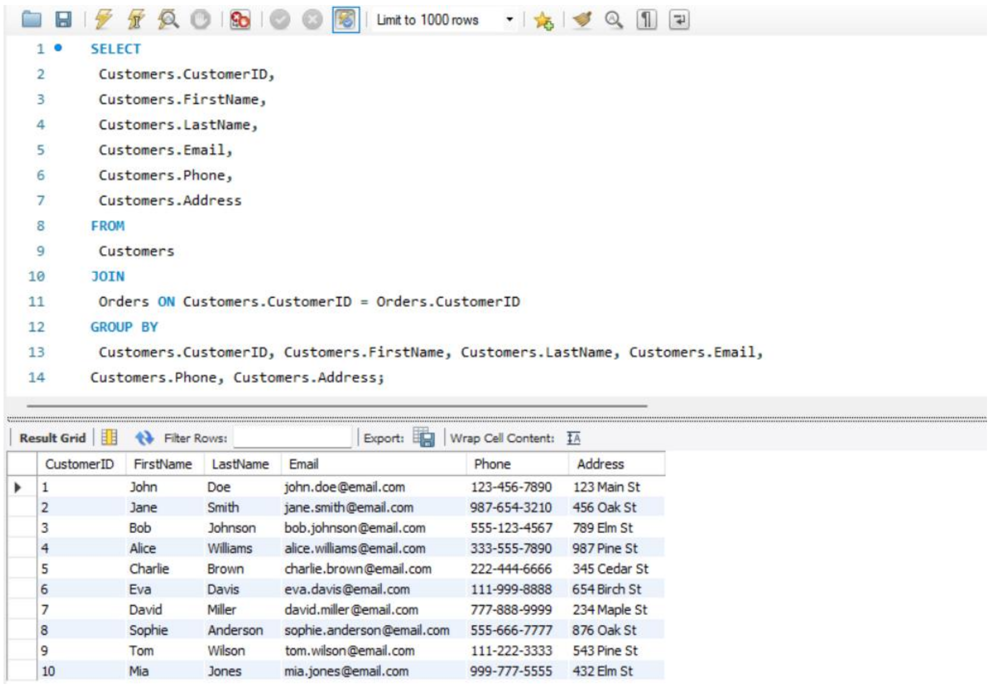
2. Write an SQL query to find the total revenue generated by each electronic gadget product. Include the product name and the total revenue.



```sql
SELECT
 Products.ProductName,
 SUM(OrderDetails.Quantity * Products.Price) AS TotalRevenue
FROM
 OrderDetails
JOIN
 Products ON OrderDetails.ProductID = Products.ProductID
GROUP BY
 Products.ProductName;
```

ProductName	TotalRevenue
Smartphone X	1759.98
Laptop Pro	1429.99
Wireless Earbuds	164.99
Smartwatch Plus	219.99
Gaming Console	549.99
Ultra HD TV	989.99
Digital Camera	769.99
Portable Speaker	175.98
Tablet Pro	549.99
Fitness Tracker	296.97

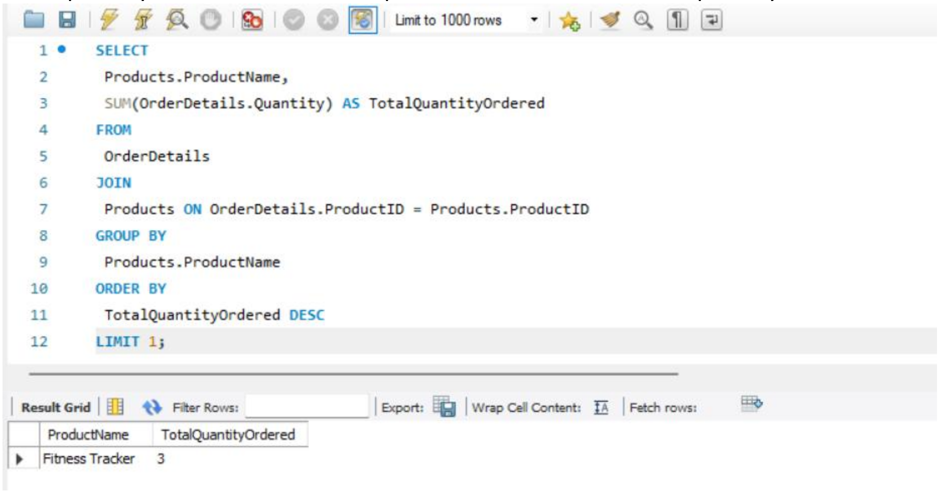
3. Write an SQL query to list all customers who have made at least one purchase. Include their names and contact information.



```sql
SELECT
 Customers.CustomerID,
 Customers.FirstName,
 Customers.LastName,
 Customers.Email,
 Customers.Phone,
 Customers.Address
FROM
 Customers
JOIN
 Orders ON Customers.CustomerID = Orders.CustomerID
GROUP BY
 Customers.CustomerID, Customers.FirstName, Customers.LastName, Customers.Email,
Customers.Phone, Customers.Address;
```

CustomerID	FirstName	LastName	Email	Phone	Address
1	John	Doe	john.doe@email.com	123-456-7890	123 Main St
2	Jane	Smith	jane.smith@email.com	987-654-3210	456 Oak St
3	Bob	Johnson	bob.johnson@email.com	555-123-4567	789 Elm St
4	Alice	Williams	alice.williams@email.com	333-555-7890	987 Pine St
5	Charlie	Brown	charlie.brown@email.com	222-444-6666	345 Cedar St
6	Eva	Davis	eva.davis@email.com	111-999-8888	654 Birch St
7	David	Miller	david.miller@email.com	777-888-9999	234 Maple St
8	Sophie	Anderson	sophie.anderson@email.com	555-666-7777	876 Oak St
9	Tom	Wilson	tom.wilson@email.com	111-222-3333	543 Pine St
10	Mia	Jones	mia.jones@email.com	999-777-5555	432 Elm St

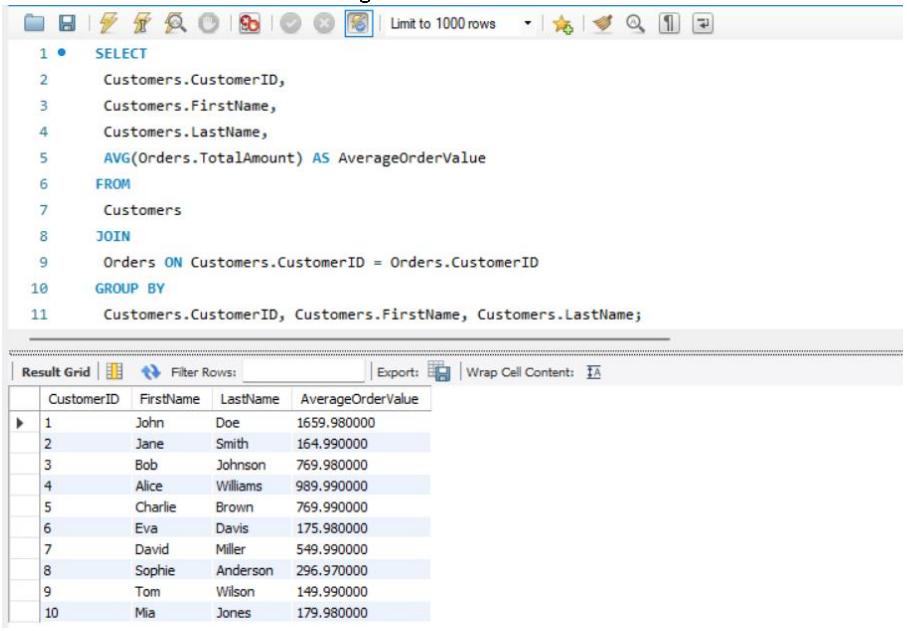
4. Write an SQL query to find the most popular electronic gadget, which is the one with the highest total quantity ordered. Include the product name and the total quantity ordered.



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

ProductName	TotalQuantityOrdered
Fitness Tracker	3

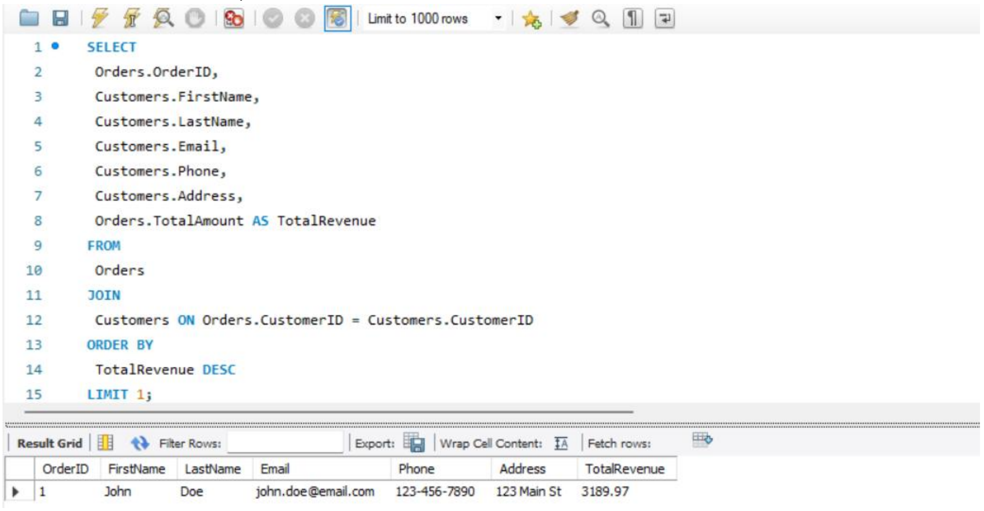
6. Write an SQL query to calculate the average order value for each customer. Include the customer's name and their average order value.



```sql
SELECT
 Customers.CustomerID,
 Customers.FirstName,
 Customers.LastName,
 AVG(Orders.TotalAmount) AS AverageOrderValue
FROM
 Customers
JOIN
 Orders ON Customers.CustomerID = Orders.CustomerID
GROUP BY
 Customers.CustomerID, Customers.FirstName, Customers.LastName;
```

CustomerID	FirstName	LastName	AverageOrderValue
1	John	Doe	1659.980000
2	Jane	Smith	164.990000
3	Bob	Johnson	769.980000
4	Alice	Williams	989.990000
5	Charlie	Brown	769.990000
6	Eva	Davis	175.980000
7	David	Miller	549.990000
8	Sophie	Anderson	296.970000
9	Tom	Wilson	149.990000
10	Mia	Jones	179.980000

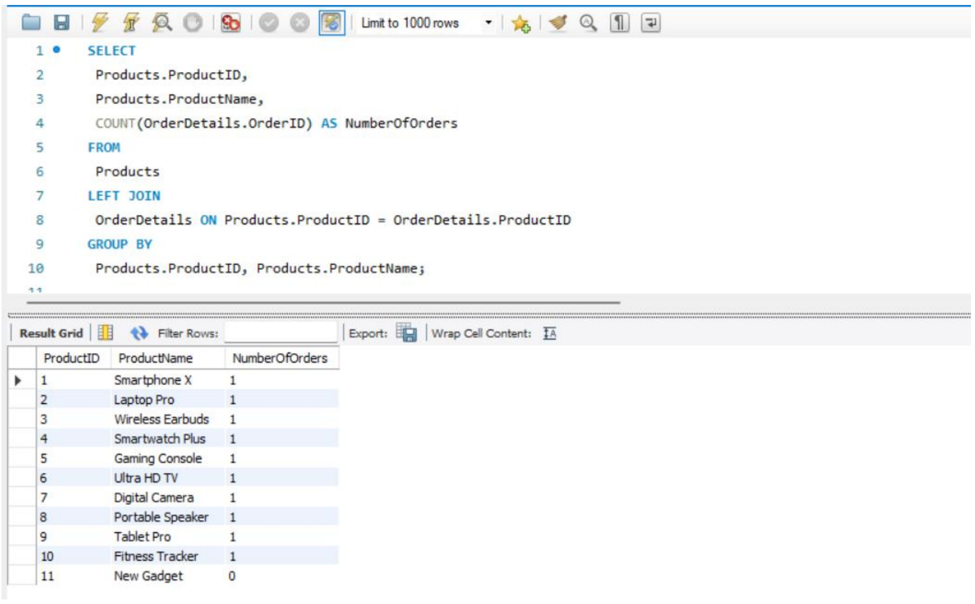
7. Write an SQL query to find the order with the highest total revenue. Include the order ID, customer information, and the total revenue.



```sql
SELECT
 Orders.OrderID,
 Customers.FirstName,
 Customers.LastName,
 Customers.Email,
 Customers.Phone,
 Customers.Address,
 Orders.TotalAmount AS TotalRevenue
FROM
 Orders
JOIN
 Customers ON Orders.CustomerID = Customers.CustomerID
ORDER BY
 TotalRevenue DESC
LIMIT 1;
```

OrderID	FirstName	LastName	Email	Phone	Address	TotalRevenue
1	John	Doe	john.doe@email.com	123-456-7890	123 Main St	3189.97

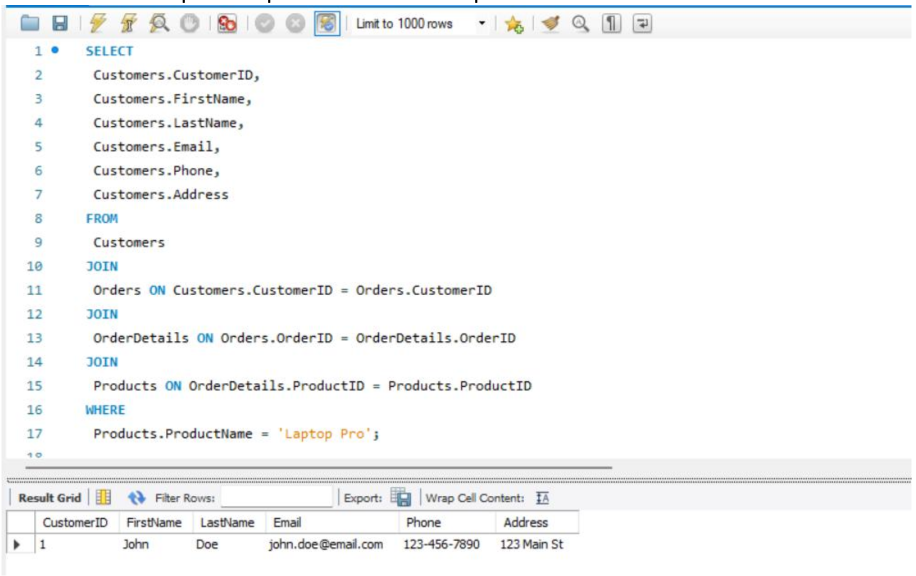
8. Write an SQL query to list electronic gadgets and the number of times each product has been ordered.



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

ProductID	ProductName	NumberOfOrders
1	Smartphone X	1
2	Laptop Pro	1
3	Wireless Earbuds	1
4	Smartwatch Plus	1
5	Gaming Console	1
6	Ultra HD TV	1
7	Digital Camera	1
8	Portable Speaker	1
9	Tablet Pro	1
10	Fitness Tracker	1
11	New Gadget	0

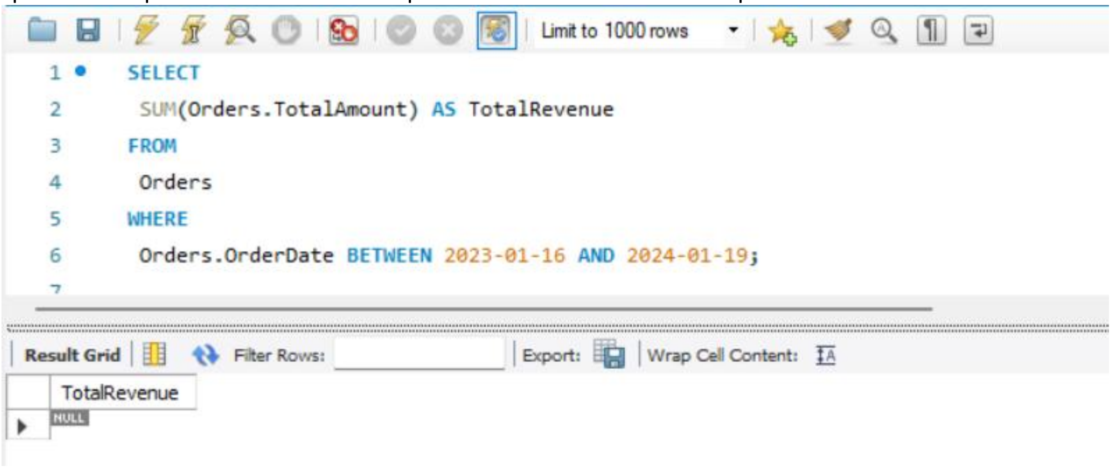
9. Write an SQL query to find customers who have purchased a specific electronic gadget product. Allow users to input the product name as a parameter.



```
SELECT
 Customers.CustomerID,
 Customers.FirstName,
 Customers.LastName,
 Customers.Email,
 Customers.Phone,
 Customers.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 Pro';
```

CustomerID	FirstName	LastName	Email	Phone	Address
1	John	Doe	john.doe@email.com	123-456-7890	123 Main St

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.

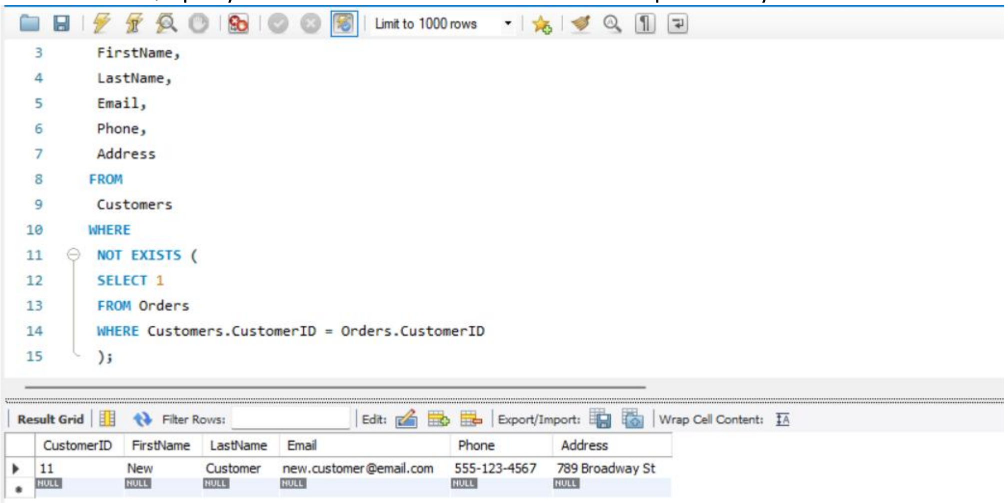


```
SELECT
 SUM(Orders.TotalAmount) AS TotalRevenue
FROM
 Orders
WHERE
 Orders.OrderDate BETWEEN 2023-01-16 AND 2024-01-19;
```

TotalRevenue
NULL

Task 4. Subquery and its type:

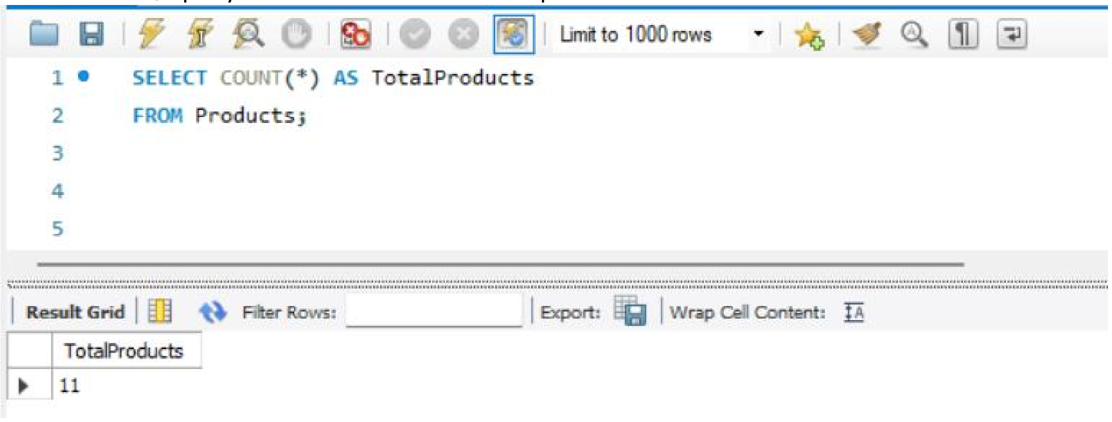
1. Write an SQL query to find out which customers have not placed any orders.



```sql
    FirstName,
    LastName,
    Email,
    Phone,
    Address
   FROM
    Customers
   WHERE
    NOT EXISTS (
    SELECT 1
    FROM Orders
    WHERE Customers.CustomerID = Orders.CustomerID
    );
```

```
CustomerID  FirstName  LastName  Email                    Phone         Address
11          New        Customer  new.customer@email.com   555-123-4567  789 Broadway St
NULL        NULL       NULL      NULL                     NULL          NULL
```

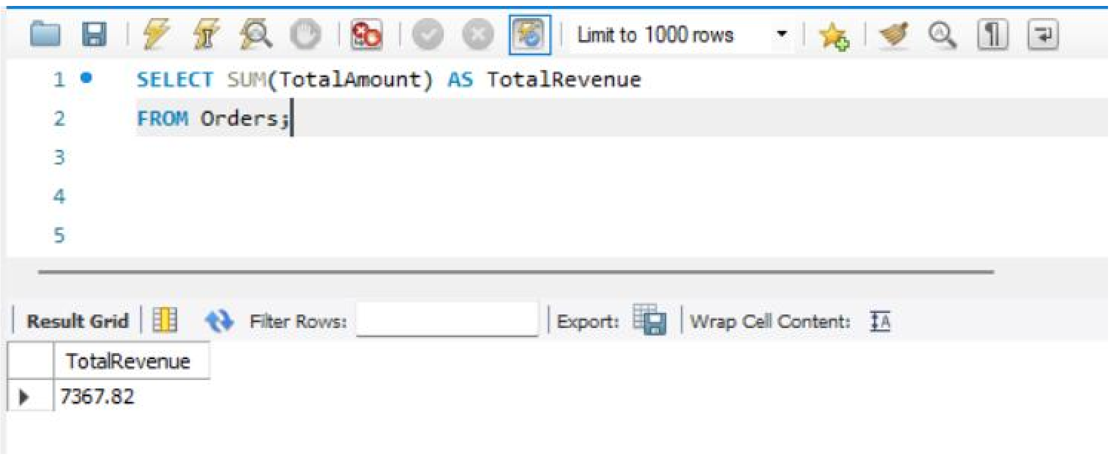
2. Write an SQL query to find the total number of products available for sale.



```sql
SELECT COUNT(*) AS TotalProducts
FROM Products;
```

```
TotalProducts
11
```

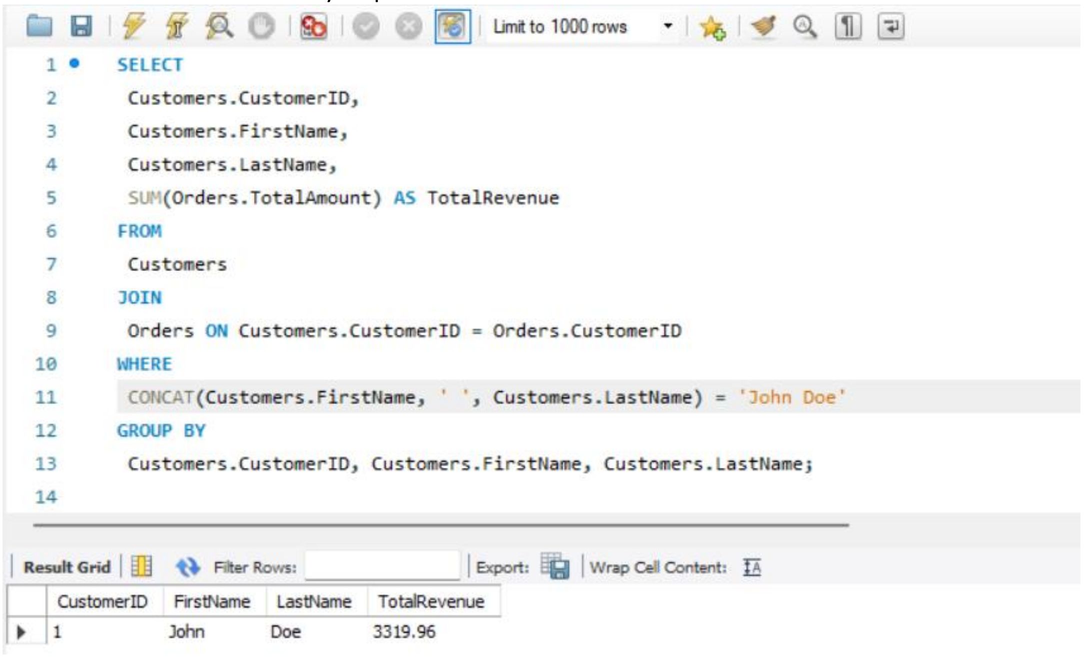
3. Write an SQL query to calculate the total revenue generated by TechShop.



```sql
SELECT SUM(TotalAmount) AS TotalRevenue
FROM Orders;
```

```
TotalRevenue
7367.82
```

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.



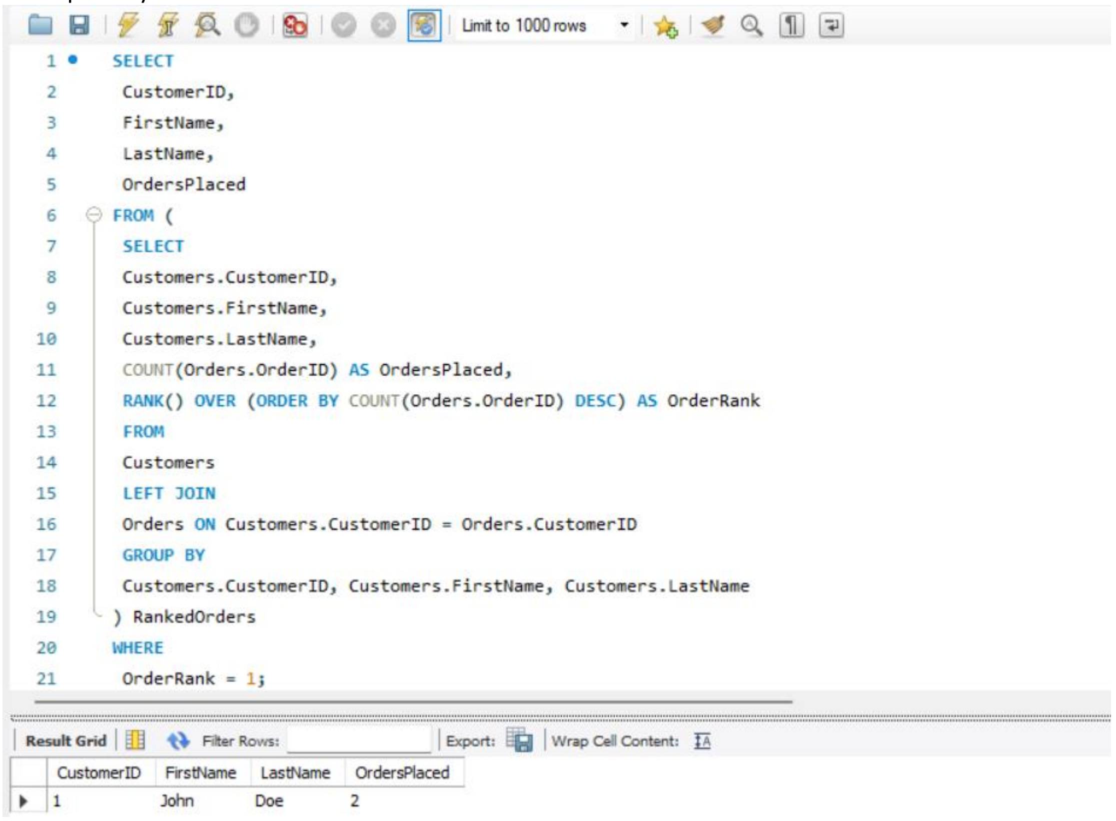
Limit to 1000 rows

```sql
SELECT
 Customers.CustomerID,
 Customers.FirstName,
 Customers.LastName,
 SUM(Orders.TotalAmount) AS TotalRevenue
FROM
 Customers
JOIN
 Orders ON Customers.CustomerID = Orders.CustomerID
WHERE
 CONCAT(Customers.FirstName, ' ', Customers.LastName) = 'John Doe'
GROUP BY
 Customers.CustomerID, Customers.FirstName, Customers.LastName;
```

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

CustomerID	FirstName	LastName	TotalRevenue
1	John	Doe	3319.96

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.



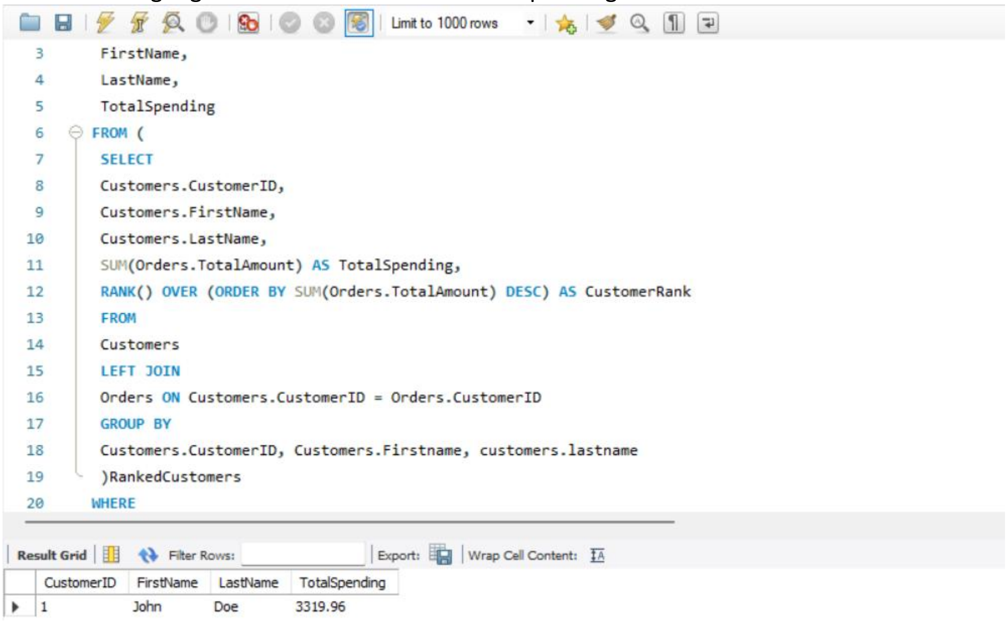
Limit to 1000 rows

```sql
SELECT
 CustomerID,
 FirstName,
 LastName,
 OrdersPlaced
FROM (
 SELECT
 Customers.CustomerID,
 Customers.FirstName,
 Customers.LastName,
 COUNT(Orders.OrderID) AS OrdersPlaced,
 RANK() OVER (ORDER BY COUNT(Orders.OrderID) DESC) AS OrderRank
 FROM
 Customers
 LEFT JOIN
 Orders ON Customers.CustomerID = Orders.CustomerID
 GROUP BY
 Customers.CustomerID, Customers.FirstName, Customers.LastName
) RankedOrders
WHERE
 OrderRank = 1;
```

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

CustomerID	FirstName	LastName	OrdersPlaced
1	John	Doe	2

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.



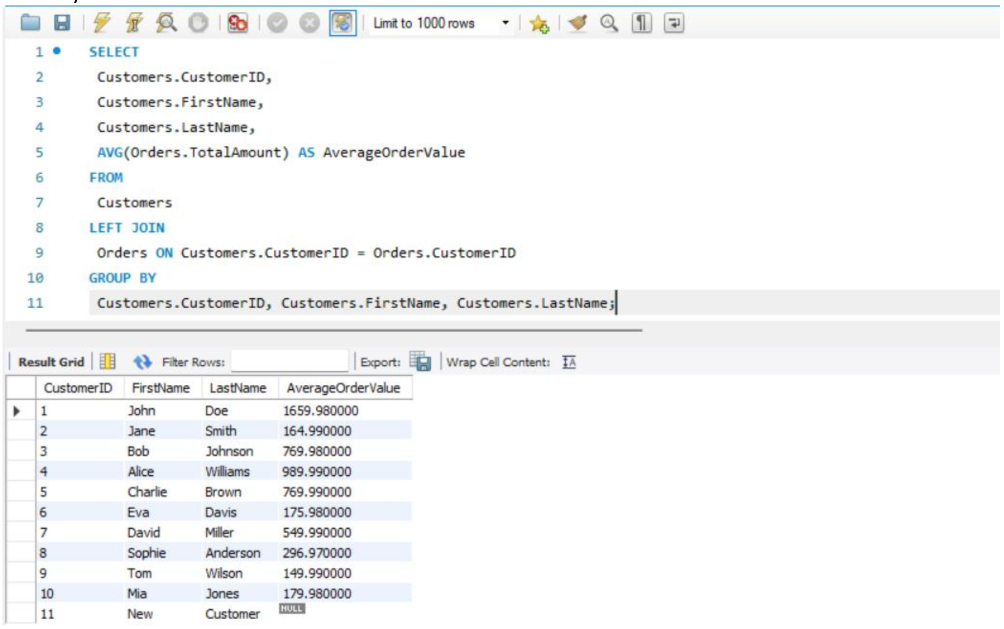
Limit to 1000 rows

```sql
 FirstName,
 LastName,
 TotalSpending
FROM (
 SELECT
 Customers.CustomerID,
 Customers.FirstName,
 Customers.LastName,
 SUM(Orders.TotalAmount) AS TotalSpending,
 RANK() OVER (ORDER BY SUM(Orders.TotalAmount) DESC) AS CustomerRank
 FROM
 Customers
 LEFT JOIN
 Orders ON Customers.CustomerID = Orders.CustomerID
 GROUP BY
 Customers.CustomerID, Customers.Firstname, customers.lastname
 )RankedCustomers
WHERE
```

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

CustomerID	FirstName	LastName	TotalSpending
1	John	Doe	3319.96

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



Limit to 1000 rows

```sql
SELECT
 Customers.CustomerID,
 Customers.FirstName,
 Customers.LastName,
 AVG(Orders.TotalAmount) AS AverageOrderValue
FROM
 Customers
LEFT JOIN
 Orders ON Customers.CustomerID = Orders.CustomerID
GROUP BY
 Customers.CustomerID, Customers.FirstName, Customers.LastName;
```

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

CustomerID	FirstName	LastName	AverageOrderValue
1	John	Doe	1659.980000
2	Jane	Smith	164.990000
3	Bob	Johnson	769.980000
4	Alice	Williams	989.990000
5	Charlie	Brown	769.990000
6	Eva	Davis	175.980000
7	David	Miller	549.990000
8	Sophie	Anderson	296.970000
9	Tom	Wilson	149.990000
10	Mia	Jones	179.980000
11	New	Customer	NULL