Assignment 1

You are working as a database administrator for a fictional company named "TechShop," which sells electronic gadgets. TechShop maintains data related to their products, customers, and orders. Your task is to design and implement a database for TechShop based on the following requirements.

Database Tables:

1. Customers:

- **CustomerID (Primary Key)**
- **FirstName**
- **LastName**
- **Email**
- **Phone**
- **Address**

2. Products:

- **ProductID (Primary Key)**
- **ProductName**
- **Description**
- **Price**

3. Orders:

- **OrderID (Primary Key)**
- **CustomerID (Foreign Key referencing Customers)**
- **OrderDate**
- **TotalAmount**

4. OrderDetails:

- **OrderDetailID (Primary Key)**
- **OrderID (Foreign Key referencing Orders)**
- **ProductID (Foreign Key referencing Products)**
- **Quantity**

5. Inventory

- **InventoryID (Primary Key)**
- **ProductID (Foreign Key referencing Products)**
- **QuantityInStock**
- **LastStockUpdate**

Task:1. Database Design:

1. Create the database named "TechShop" CREATE DATABASE TechShop;

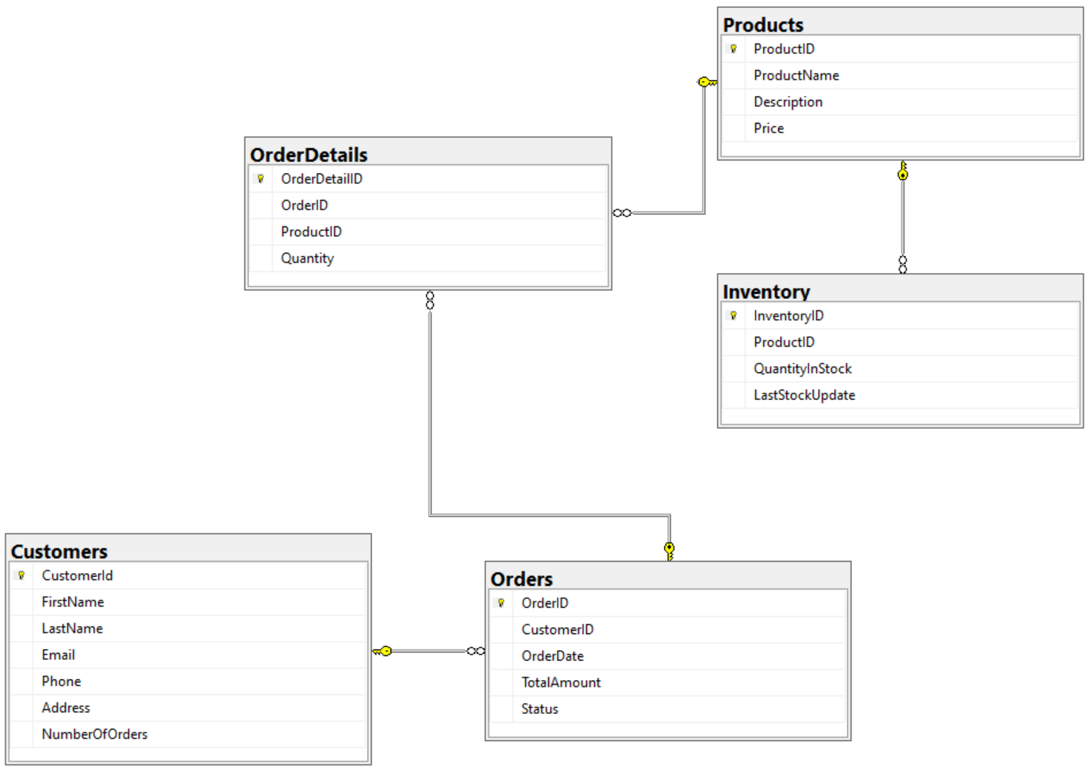
```
USE TechShop;
```

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

```
CREATE TABLE Customers(
```

```sql
CustomerId INT PRIMARY KEY,
FirstName VARCHAR(20),
LastName VARCHAR(20),
Email VARCHAR(40),
Phone VARCHAR(10),
Address VARCHAR(30),
);

SELECT * FROM Customers;

CREATE TABLE Products(
ProductID INT PRIMARY KEY,
ProductName VARCHAR(30),
Description VARCHAR(50),
Price DECIMAL(10,2)
);

CREATE TABLE Orders(
OrderID INT PRIMARY KEY,
CustomerID INT,
OrderDate DATE,
TotalAmount DECIMAL(10,2),
FOREIGN KEY (CustomerID) REFERENCES Customers(CustomerId)
);

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)
);
```

2. Define the schema for the Customers, Products, Orders, OrderDetails and Inventory tables based on the provided schema.
3. Create an ERD (Entity Relationship Diagram) for the database.
4. Create appropriate Primary Key and Foreign Key constraints for referential integrity.



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

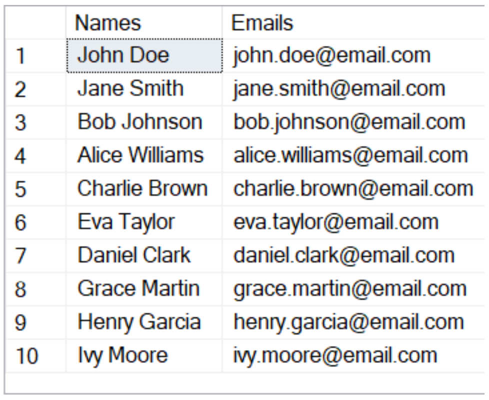
```
INSERT INTO Customers VALUES

(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'),

-- ... (Insert 8 more records)


-- Step 8: Insert sample records into Products table

INSERT INTO Products VALUES

(1, 'Laptop', 'High-performance laptop', 999.99),

(2, 'Smartphone', 'Flagship smartphone', 699.99),

-- ... (Insert 8 more records)


-- Step 9: Insert sample records into Orders table

INSERT INTO Orders VALUES
```

```
(1, 1, '2023-01-01', 999.99),
(2, 2, '2023-02-01', 699.99),
-- ... (Insert 8 more records)

-- Step 10: Insert sample records into OrderDetails table
INSERT INTO OrderDetails VALUES
(1, 1, 1, 2),
(2, 1, 2, 1),
-- ... (Insert 8 more records)

-- Step 11: Insert sample records into Inventory table
INSERT INTO Inventory VALUES
(1, 1, 50, '2023-01-01'),
(2, 2, 100, '2023-02-01'),
-- ... (Insert 8 more records)
```

Tasks 2: Select, Where, Between, AND LIKE

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

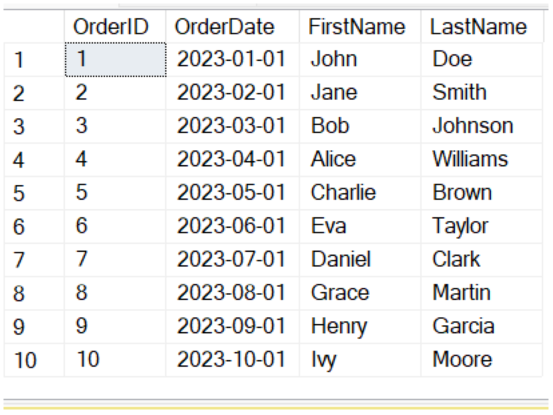
```sql
SELECT FirstName + ' ' + LastName AS Names, Email as Emails from Customers;
```



	Names	Emails
1	John Doe	john.doe@email.com
2	Jane Smith	jane.smith@email.com
3	Bob Johnson	bob.johnson@email.com
4	Alice Williams	alice.williams@email.com
5	Charlie Brown	charlie.brown@email.com
6	Eva Taylor	eva.taylor@email.com
7	Daniel Clark	daniel.clark@email.com
8	Grace Martin	grace.martin@email.com
9	Henry Garcia	henry.garcia@email.com
10	Ivy Moore	ivy.moore@email.com

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

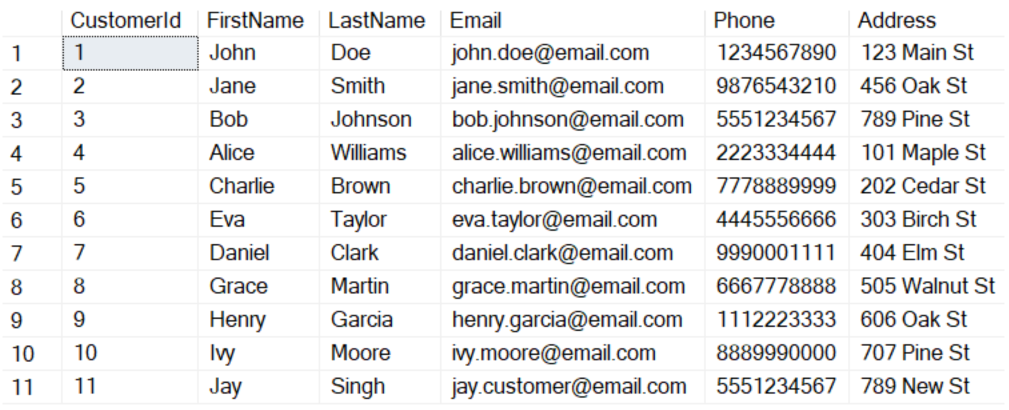
```sql
SELECT O.OrderID, O.OrderDate, C.FirstName, C.LastName from Orders O INNER
JOIN Customers C ON O.OrderID= C.CustomerId;
```



	OrderID	OrderDate	FirstName	LastName
1	1	2023-01-01	John	Doe
2	2	2023-02-01	Jane	Smith
3	3	2023-03-01	Bob	Johnson
4	4	2023-04-01	Alice	Williams
5	5	2023-05-01	Charlie	Brown
6	6	2023-06-01	Eva	Taylor
7	7	2023-07-01	Daniel	Clark
8	8	2023-08-01	Grace	Martin
9	9	2023-09-01	Henry	Garcia
10	10	2023-10-01	Ivy	Moore

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

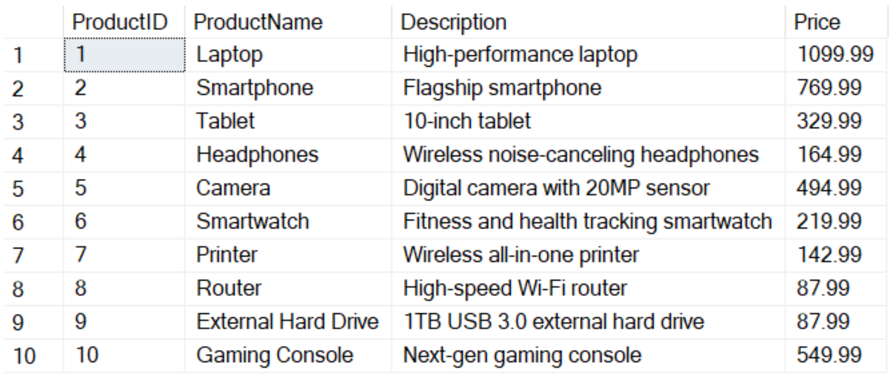
```sql
INSERT INTO Customers VALUES (11,'Jay', 'Singh', 'jay.customer@email.com',
'5551234567', '789 New St');
```



	CustomerId	FirstName	LastName	Email	Phone	Address
1	1	John	Doe	john.doe@email.com	1234567890	123 Main St
2	2	Jane	Smith	jane.smith@email.com	9876543210	456 Oak St
3	3	Bob	Johnson	bob.johnson@email.com	5551234567	789 Pine St
4	4	Alice	Williams	alice.williams@email.com	2223334444	101 Maple St
5	5	Charlie	Brown	charlie.brown@email.com	7778889999	202 Cedar St
6	6	Eva	Taylor	eva.taylor@email.com	4445556666	303 Birch St
7	7	Daniel	Clark	daniel.clark@email.com	9990001111	404 Elm St
8	8	Grace	Martin	grace.martin@email.com	6667778888	505 Walnut St
9	9	Henry	Garcia	henry.garcia@email.com	1112223333	606 Oak St
10	10	Ivy	Moore	ivy.moore@email.com	8889990000	707 Pine St
11	11	Jay	Singh	jay.customer@email.com	5551234567	789 New St

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

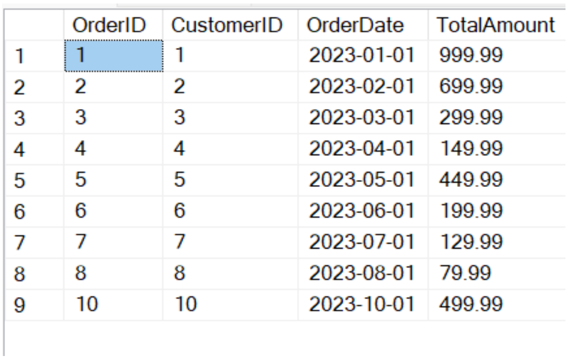
```sql
UPDATE Products
SET Price = Price * 1.1;
```



	ProductID	ProductName	Description	Price
1	1	Laptop	High-performance laptop	1099.99
2	2	Smartphone	Flagship smartphone	769.99
3	3	Tablet	10-inch tablet	329.99
4	4	Headphones	Wireless noise-canceling headphones	164.99
5	5	Camera	Digital camera with 20MP sensor	494.99
6	6	Smartwatch	Fitness and health tracking smartwatch	219.99
7	7	Printer	Wireless all-in-one printer	142.99
8	8	Router	High-speed Wi-Fi router	87.99
9	9	External Hard Drive	1TB USB 3.0 external hard drive	87.99
10	10	Gaming Console	Next-gen gaming console	549.99

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.

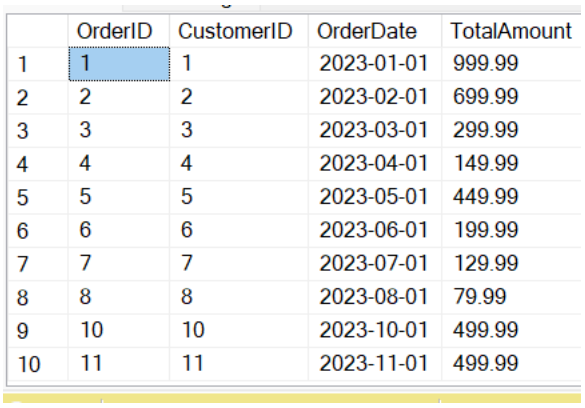
```sql
DECLARE @OrderID INT; -- Declare the parameter
SET @OrderID = 9;
DELETE FROM OrderDetails
WHERE OrderID = @OrderID;
DELETE FROM Orders
WHERE OrderID = @OrderID;
```



	OrderID	CustomerID	OrderDate	TotalAmount
1	1	1	2023-01-01	999.99
2	2	2	2023-02-01	699.99
3	3	3	2023-03-01	299.99
4	4	4	2023-04-01	149.99
5	5	5	2023-05-01	449.99
6	6	6	2023-06-01	199.99
7	7	7	2023-07-01	129.99
8	8	8	2023-08-01	79.99
9	10	10	2023-10-01	499.99

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.

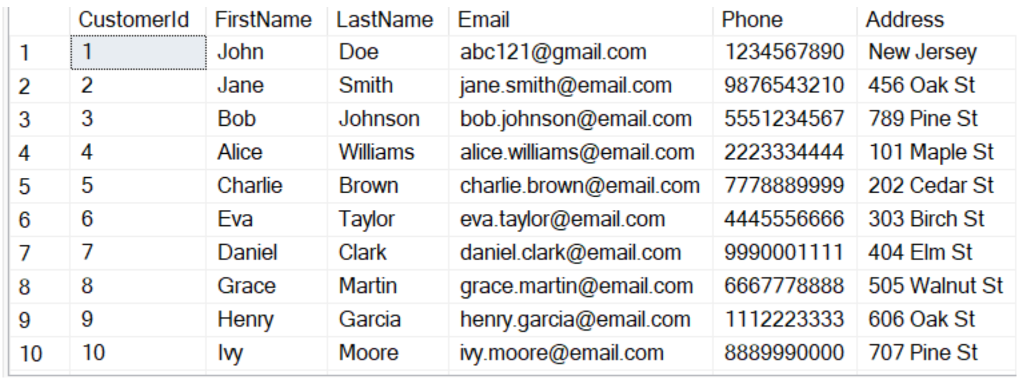
```sql
INSERT INTO Orders VALUES (11, 11, '2023-11-01', 499.99);
```



	OrderID	CustomerID	OrderDate	TotalAmount
1	1	1	2023-01-01	999.99
2	2	2	2023-02-01	699.99
3	3	3	2023-03-01	299.99
4	4	4	2023-04-01	149.99
5	5	5	2023-05-01	449.99
6	6	6	2023-06-01	199.99
7	7	7	2023-07-01	129.99
8	8	8	2023-08-01	79.99
9	10	10	2023-10-01	499.99
10	11	11	2023-11-01	499.99

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.

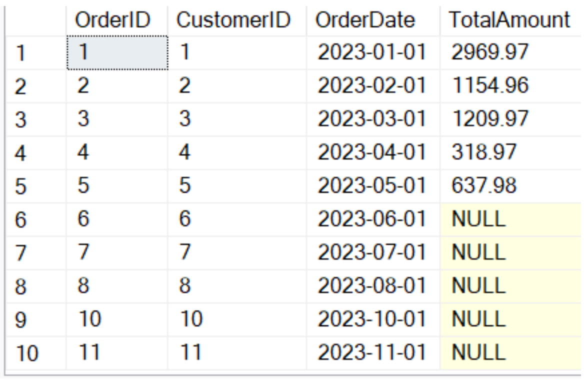
```sql
DECLARE @CustomerID INT = 1;
UPDATE Customers
SET Email='abc121@gmail.com', Address = 'New Jersey'
WHERE CustomerId = @CustomerID;
```



	CustomerId	FirstName	LastName	Email	Phone	Address
1	1	John	Doe	abc121@gmail.com	1234567890	New Jersey
2	2	Jane	Smith	jane.smith@email.com	9876543210	456 Oak St
3	3	Bob	Johnson	bob.johnson@email.com	5551234567	789 Pine St
4	4	Alice	Williams	alice.williams@email.com	2223334444	101 Maple St
5	5	Charlie	Brown	charlie.brown@email.com	7778889999	202 Cedar St
6	6	Eva	Taylor	eva.taylor@email.com	4445556666	303 Birch St
7	7	Daniel	Clark	daniel.clark@email.com	9990001111	404 Elm St
8	8	Grace	Martin	grace.martin@email.com	6667778888	505 Walnut St
9	9	Henry	Garcia	henry.garcia@email.com	1112223333	606 Oak St
10	10	Ivy	Moore	ivy.moore@email.com	8889990000	707 Pine St

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.

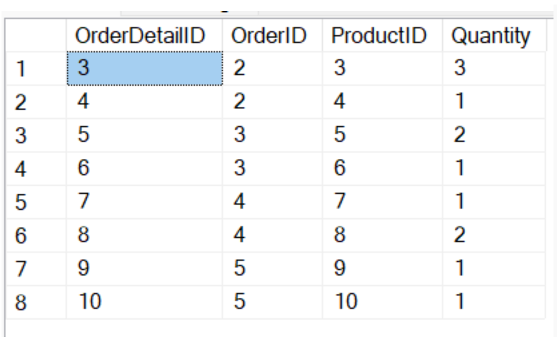
```sql
UPDATE Orders
SET TotalAmount =(
SELECT  SUM(P.Price*O.Quantity)From OrderDetails O JOIN Products P
ON O.ProductID = P.ProductID WHERE O.OrderID = Orders.OrderID);
```



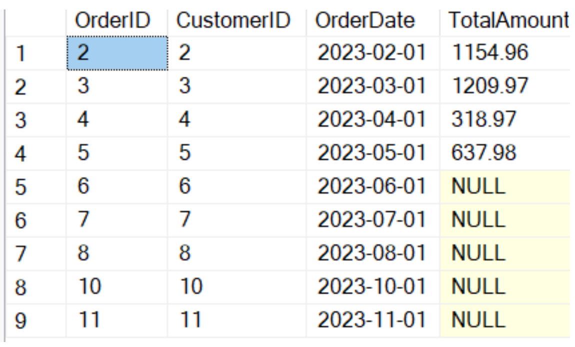
	OrderID	CustomerID	OrderDate	TotalAmount
1	1	1	2023-01-01	2969.97
2	2	2	2023-02-01	1154.96
3	3	3	2023-03-01	1209.97
4	4	4	2023-04-01	318.97
5	5	5	2023-05-01	637.98
6	6	6	2023-06-01	NULL
7	7	7	2023-07-01	NULL
8	8	8	2023-08-01	NULL
9	10	10	2023-10-01	NULL
10	11	11	2023-11-01	NULL

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
DECLARE @CustomerID INT = 1;
DELETE FROM OrderDetails WHERE OrderID = (
SELECT OrderID from Orders WHERE CustomerID = @CustomerID)
DELETE FROM Orders WHERE Orders.CustomerID = @CustomerID;
```



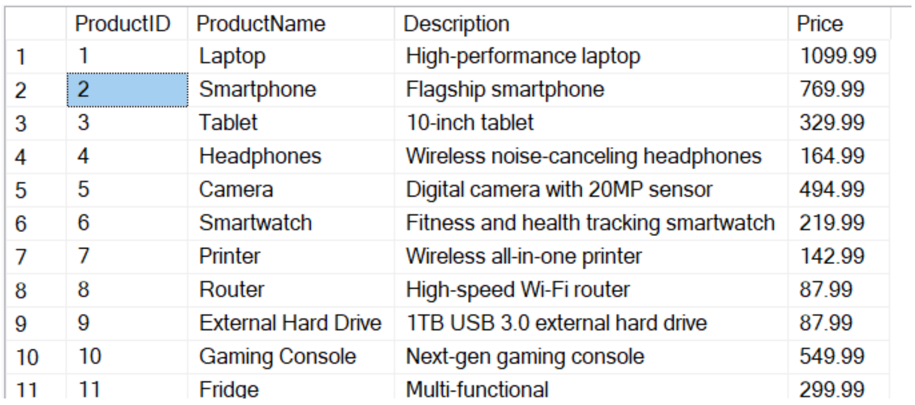
	OrderDetailID	OrderID	ProductID	Quantity
1	3	2	3	3
2	4	2	4	1
3	5	3	5	2
4	6	3	6	1
5	7	4	7	1
6	8	4	8	2
7	9	5	9	1
8	10	5	10	1



	OrderID	CustomerID	OrderDate	TotalAmount
1	2	2	2023-02-01	1154.96
2	3	3	2023-03-01	1209.97
3	4	4	2023-04-01	318.97
4	5	5	2023-05-01	637.98
5	6	6	2023-06-01	NULL
6	7	7	2023-07-01	NULL
7	8	8	2023-08-01	NULL
8	10	10	2023-10-01	NULL
9	11	11	2023-11-01	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.

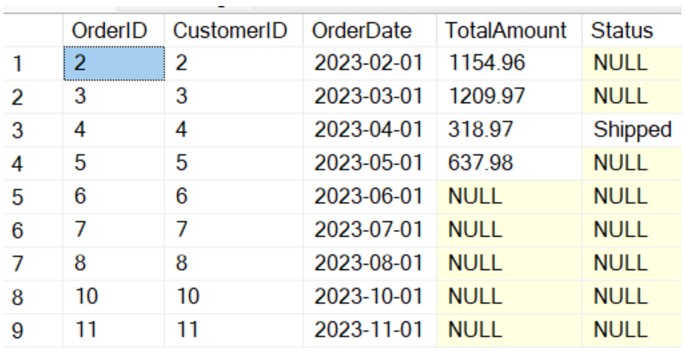
```sql
INSERT INTO Products
VALUES (11,'Fridge', 'Multi-functional', 299.99);
```



	ProductID	ProductName	Description	Price
1	1	Laptop	High-performance laptop	1099.99
2	2	Smartphone	Flagship smartphone	769.99
3	3	Tablet	10-inch tablet	329.99
4	4	Headphones	Wireless noise-canceling headphones	164.99
5	5	Camera	Digital camera with 20MP sensor	494.99
6	6	Smartwatch	Fitness and health tracking smartwatch	219.99
7	7	Printer	Wireless all-in-one printer	142.99
8	8	Router	High-speed Wi-Fi router	87.99
9	9	External Hard Drive	1TB USB 3.0 external hard drive	87.99
10	10	Gaming Console	Next-gen gaming console	549.99
11	11	Fridge	Multi-functional	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.

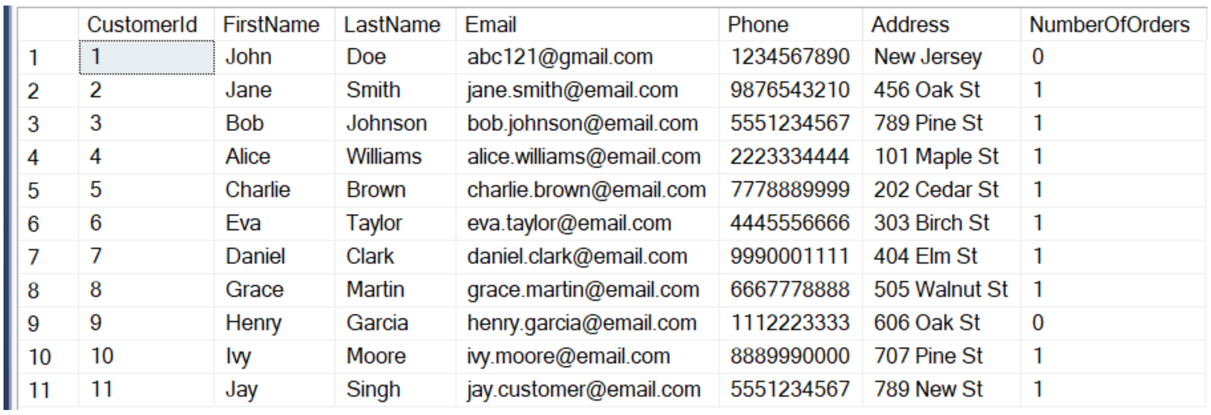
```
DECLARE @OrderID INT = 4; -- Replace with the actual order ID
UPDATE Orders
SET Status = 'Shipped'
WHERE OrderID = @OrderID;
```



	OrderID	CustomerID	OrderDate	TotalAmount	Status
1	2	2	2023-02-01	1154.96	NULL
2	3	3	2023-03-01	1209.97	NULL
3	4	4	2023-04-01	318.97	Shipped
4	5	5	2023-05-01	637.98	NULL
5	6	6	2023-06-01	NULL	NULL
6	7	7	2023-07-01	NULL	NULL
7	8	8	2023-08-01	NULL	NULL
8	10	10	2023-10-01	NULL	NULL
9	11	11	2023-11-01	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.

```
UPDATE Customers
SET NumberOfOrders = (
    SELECT COUNT(*)
    FROM Orders
    WHERE Orders.CustomerID = Customers.CustomerId
);
```

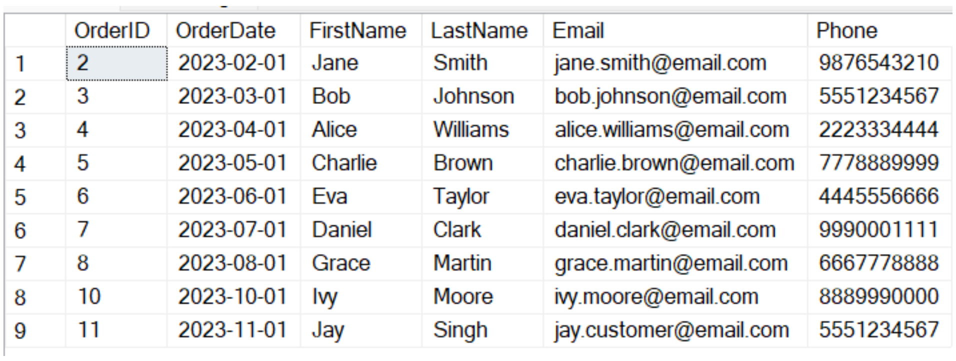


	CustomerId	FirstName	LastName	Email	Phone	Address	NumberOfOrders
1	1	John	Doe	abc121@gmail.com	1234567890	New Jersey	0
2	2	Jane	Smith	jane.smith@email.com	9876543210	456 Oak St	1
3	3	Bob	Johnson	bob.johnson@email.com	5551234567	789 Pine St	1
4	4	Alice	Williams	alice.williams@email.com	2223334444	101 Maple St	1
5	5	Charlie	Brown	charlie.brown@email.com	7778889999	202 Cedar St	1
6	6	Eva	Taylor	eva.taylor@email.com	4445556666	303 Birch St	1
7	7	Daniel	Clark	daniel.clark@email.com	9990001111	404 Elm St	1
8	8	Grace	Martin	grace.martin@email.com	6667778888	505 Walnut St	1
9	9	Henry	Garcia	henry.garcia@email.com	1112223333	606 Oak St	0
10	10	Ivy	Moore	ivy.moore@email.com	8889990000	707 Pine St	1
11	11	Jay	Singh	jay.customer@email.com	5551234567	789 New St	1

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.

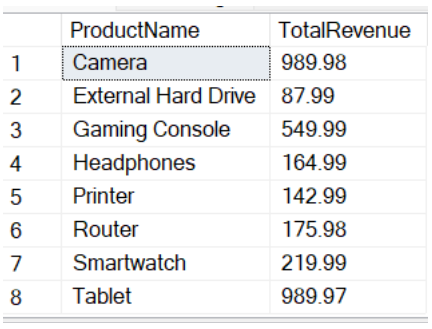
```
SELECT Orders.OrderID, Orders.OrderDate, Customers.FirstName,
Customers.LastName, Customers.Email, Customers.Phone
FROM Orders
JOIN Customers ON Orders.CustomerID = Customers.CustomerId;
```



	OrderID	OrderDate	FirstName	LastName	Email	Phone
1	2	2023-02-01	Jane	Smith	jane.smith@email.com	9876543210
2	3	2023-03-01	Bob	Johnson	bob.johnson@email.com	5551234567
3	4	2023-04-01	Alice	Williams	alice.williams@email.com	2223334444
4	5	2023-05-01	Charlie	Brown	charlie.brown@email.com	7778889999
5	6	2023-06-01	Eva	Taylor	eva.taylor@email.com	4445556666
6	7	2023-07-01	Daniel	Clark	daniel.clark@email.com	9990001111
7	8	2023-08-01	Grace	Martin	grace.martin@email.com	6667778888
8	10	2023-10-01	Ivy	Moore	ivy.moore@email.com	8889990000
9	11	2023-11-01	Jay	Singh	jay.customer@email.com	5551234567

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

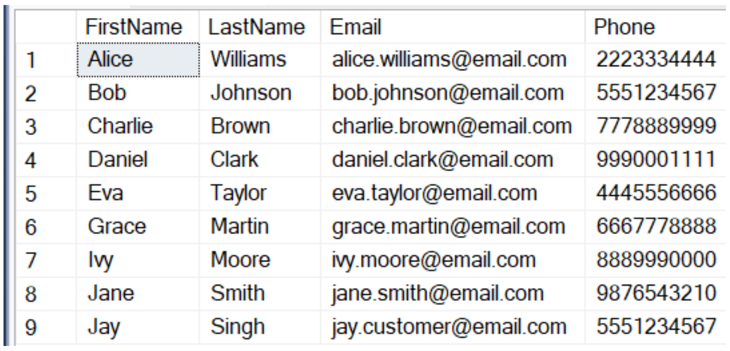
```
SELECT p.ProductName, SUM(od.Quantity * p.Price) AS TotalRevenue
FROM OrderDetails od
JOIN Products p ON od.ProductID = p.ProductID
GROUP BY p.ProductName;
```



	ProductName	TotalRevenue
1	Camera	989.98
2	External Hard Drive	87.99
3	Gaming Console	549.99
4	Headphones	164.99
5	Printer	142.99
6	Router	175.98
7	Smartwatch	219.99
8	Tablet	989.97

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

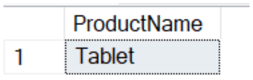
```
SELECT DISTINCT
    c.FirstName,
    c.LastName,
    c.Email,
    c.Phone
FROM Customers c
JOIN Orders o ON c.CustomerID = o.CustomerID;
```



	FirstName	LastName	Email	Phone
1	Alice	Williams	alice.williams@email.com	2223334444
2	Bob	Johnson	bob.johnson@email.com	5551234567
3	Charlie	Brown	charlie.brown@email.com	7778889999
4	Daniel	Clark	daniel.clark@email.com	9990001111
5	Eva	Taylor	eva.taylor@email.com	4445556666
6	Grace	Martin	grace.martin@email.com	6667778888
7	Ivy	Moore	ivy.moore@email.com	8889990000
8	Jane	Smith	jane.smith@email.com	9876543210
9	Jay	Singh	jay.customer@email.com	5551234567

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.

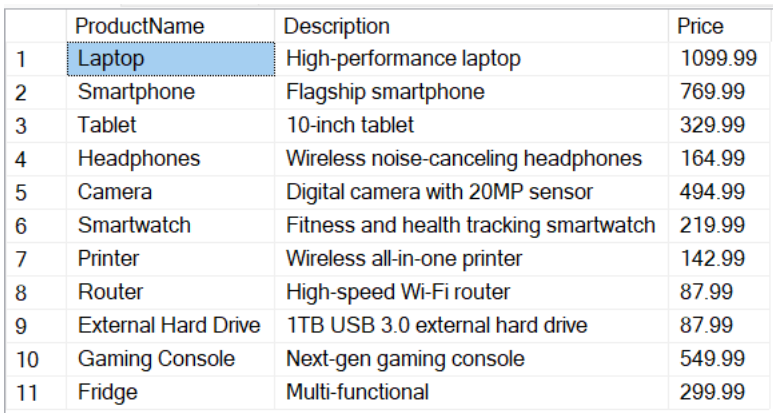
```
SELECT ProductName FROM Products WHERE ProductID = (
SELECT ProductID FROM OrderDetails O WHERE Quantity =
(SELECT MAX(Quantity) FROM OrderDetails) GROUP BY ProductID );
```



	ProductName
1	Tablet

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

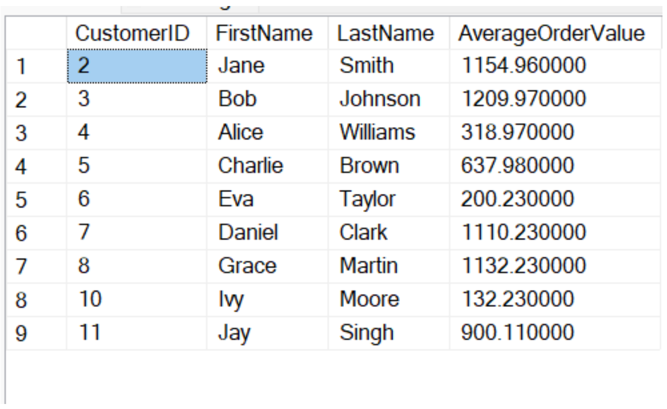
```
SELECT p.ProductName, p.Description, p.Price
FROM Products p;
```



	ProductName	Description	Price
1	Laptop	High-performance laptop	1099.99
2	Smartphone	Flagship smartphone	769.99
3	Tablet	10-inch tablet	329.99
4	Headphones	Wireless noise-canceling headphones	164.99
5	Camera	Digital camera with 20MP sensor	494.99
6	Smartwatch	Fitness and health tracking smartwatch	219.99
7	Printer	Wireless all-in-one printer	142.99
8	Router	High-speed Wi-Fi router	87.99
9	External Hard Drive	1TB USB 3.0 external hard drive	87.99
10	Gaming Console	Next-gen gaming console	549.99
11	Fridge	Multi-functional	299.99

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

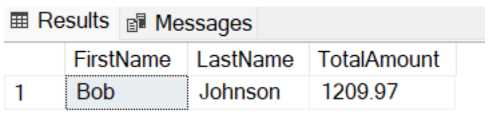
```
SELECT c.CustomerID, c.FirstName, c.LastName, AVG(o.TotalAmount) AS
AverageOrderValue
FROM Customers c
JOIN Orders o ON c.CustomerID = o.CustomerID
GROUP BY c.CustomerID, c.FirstName, c.LastName;
```



	CustomerID	FirstName	LastName	AverageOrderValue
1	2	Jane	Smith	1154.960000
2	3	Bob	Johnson	1209.970000
3	4	Alice	Williams	318.970000
4	5	Charlie	Brown	637.980000
5	6	Eva	Taylor	200.230000
6	7	Daniel	Clark	1110.230000
7	8	Grace	Martin	1132.230000
8	10	Ivy	Moore	132.230000
9	11	Jay	Singh	900.110000

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

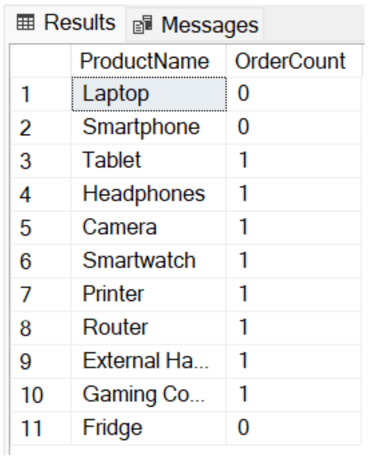
```
SELECT TOP 1 C.FirstName, C.LastName, O.TotalAmount From Orders O
JOIN Customers C ON O.CustomerID = C.CustomerId
ORDER BY TotalAmount DESC;
```



	FirstName	LastName	TotalAmount
1	Bob	Johnson	1209.97

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

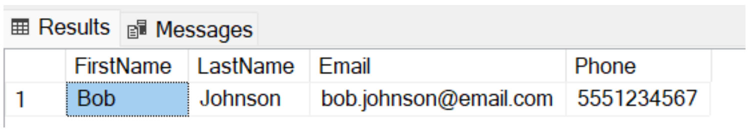
```
SELECT p.ProductName, COUNT(od.OrderDetailID) AS OrderCount
FROM Products p
LEFT JOIN OrderDetails od ON p.ProductID = od.ProductID
GROUP BY p.ProductID, p.ProductName;
```



	ProductName	OrderCount
1	Laptop	0
2	Smartphone	0
3	Tablet	1
4	Headphones	1
5	Camera	1
6	Smartwatch	1
7	Printer	1
8	Router	1
9	External Ha...	1
10	Gaming Co...	1
11	Fridge	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 c.FirstName, c.LastName, c.Email, c.Phone
FROM Customers c
JOIN Orders o ON c.CustomerID = o.CustomerID
JOIN OrderDetails od ON o.OrderID = od.OrderID
JOIN Products p ON od.ProductID = p.ProductID
WHERE p.ProductName = @ProductName;
```

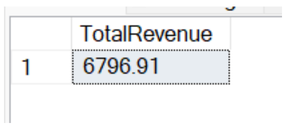


Results | Messages

	FirstName	LastName	Email	Phone
1	Bob	Johnson	bob.johnson@email.com	5551234567

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.

```
DECLARE @StartDate DATE = '2023-01-01';
DECLARE @EndDate DATE = '2023-12-31';
SELECT SUM(TotalAmount) AS TotalRevenue FROM Orders
WHERE Orders.OrderDate BETWEEN @StartDate AND @EndDate;
```

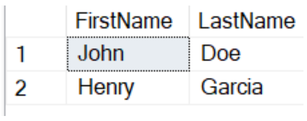


	TotalRevenue
1	6796.91

Task 4: Subquery and its type:

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

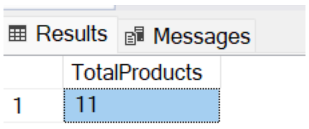
```
SELECT C.FirstName, C.LastName FROM Customers C
WHERE C.CustomerId NOT IN(
SELECT O.CustomerID FROM Orders O
);
```



	FirstName	LastName
1	John	Doe
2	Henry	Garcia

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

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



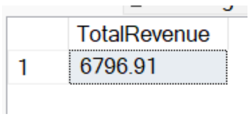
Results | Messages

	TotalProducts
1	11

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

```
SELECT SUM(TotalAmount) AS TotalRevenue
```

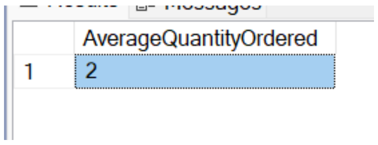
```
FROM Orders;
```



```
    TotalRevenue
1   6796.91
```

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.

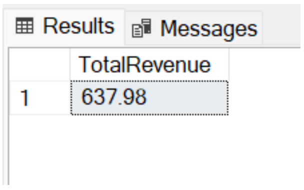
```
DECLARE @PRODUCTNAME VARCHAR(20) = 'Camera';

SELECT AVG(O.Quantity) AS AverageQunantity FROM OrderDetails O WHERE
O.ProductID IN
(SELECT P.ProductID FROM Products P WHERE P.ProductName = @PRODUCTNAME)
```



```
    AverageQuantityOrdered
1   2
```

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.

```
DECLARE @CUSTID INT = 5;
SELECT SUM(O.TotalAmount) AS TotalRevenue FROM Orders O
WHERE O.CustomerID = @CUSTID;
```

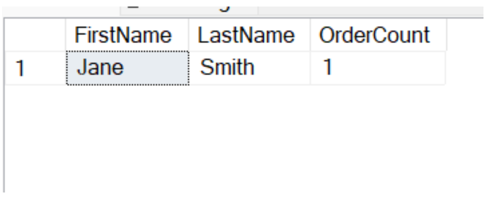


Results   Messages

```
    TotalRevenue
1   637.98
```

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.

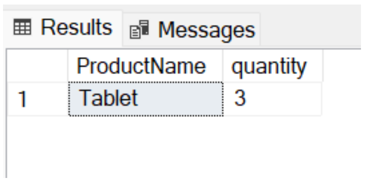
```
SELECT TOP 1 FirstName, LastName, OrderCount
FROM (
    SELECT c.FirstName, c.LastName, COUNT(o.OrderID) AS OrderCount,
           RANK() OVER (ORDER BY COUNT(o.OrderID) DESC) AS CustomerRank
    FROM Customers c
    LEFT JOIN Orders o ON c.CustomerID = o.CustomerID
    GROUP BY c.CustomerID, c.FirstName, c.LastName
) AS RankedCustomers WHERE CustomerRank = 1;
```



```
    FirstName   LastName   OrderCount
1   Jane        Smith      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.

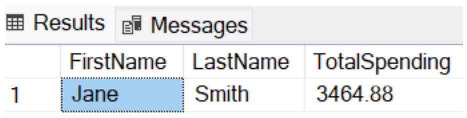
```
SELECT p.ProductName, od.quantity FROM Products p
JOIN OrderDetails od ON p.ProductID = od.ProductID
WHERE Quantity = (
SELECT TOP 1 Quantity FROM OrderDetails
ORDER BY Quantity DESC
)
```


Results | Messages

	ProductName	quantity
1	Tablet	3

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.

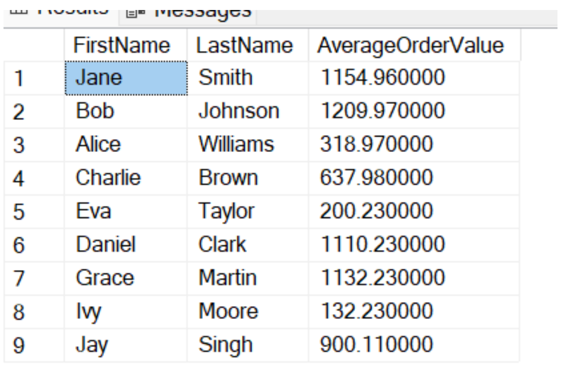
```
SELECT C.FirstName, C.LastName, TotalSpending
FROM Customers C
JOIN (
    SELECT TOP 1 O.CustomerID, (O.TotalAmount * Od.Quantity) AS
TotalSpending
    FROM Orders O
    JOIN OrderDetails Od ON Od.OrderID = O.OrderID
    ORDER BY (O.TotalAmount * Od.Quantity) DESC
) Orders ON C.CustomerID = Orders.CustomerID;
```


Results | Messages

	FirstName	LastName	TotalSpending
1	Jane	Smith	3464.88

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

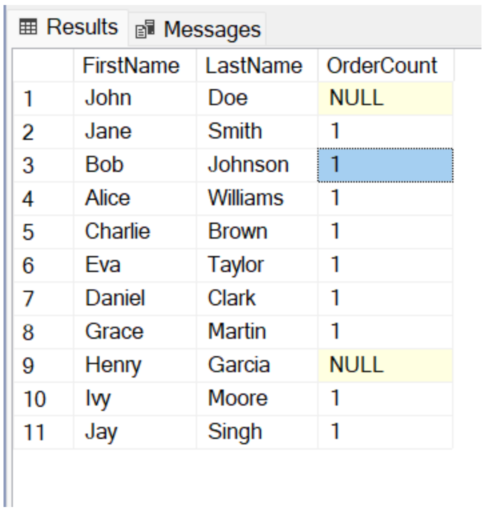
```
SELECT c.FirstName, c.LastName, AVG(OrderValue) AS AverageOrderValue
FROM Customers c
JOIN (
    SELECT o.CustomerID, SUM(o.TotalAmount) AS OrderValue
    FROM Orders o
    GROUP BY o.CustomerID
) AS CustomerOrderValues ON c.CustomerID = CustomerOrderValues.CustomerID
GROUP BY c.CustomerID, c.FirstName, c.LastName;
```


Results | Messages

	FirstName	LastName	AverageOrderValue
1	Jane	Smith	1154.960000
2	Bob	Johnson	1209.970000
3	Alice	Williams	318.970000
4	Charlie	Brown	637.980000
5	Eva	Taylor	200.230000
6	Daniel	Clark	1110.230000
7	Grace	Martin	1132.230000
8	Ivy	Moore	132.230000
9	Jay	Singh	900.110000

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

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



	FirstName	LastName	OrderCount
1	John	Doe	NULL
2	Jane	Smith	1
3	Bob	Johnson	1
4	Alice	Williams	1
5	Charlie	Brown	1
6	Eva	Taylor	1
7	Daniel	Clark	1
8	Grace	Martin	1
9	Henry	Garcia	NULL
10	Ivy	Moore	1
11	Jay	Singh	1