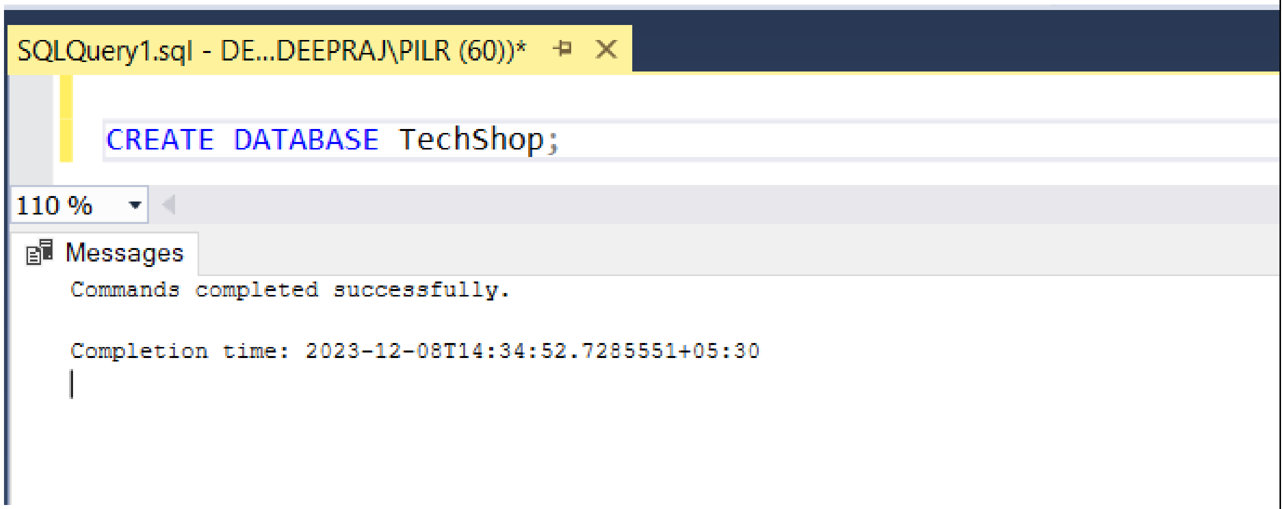
TechShop, an electronic gadgets shop

Task:1. Database Design:

1. Create the database named "TechShop"

Ans) `CREATE DATABASE TechShop;`



SQLQuery1.sql - DE...DEEPRAJ\PILR (60))*

```
CREATE DATABASE TechShop;
```

110 %

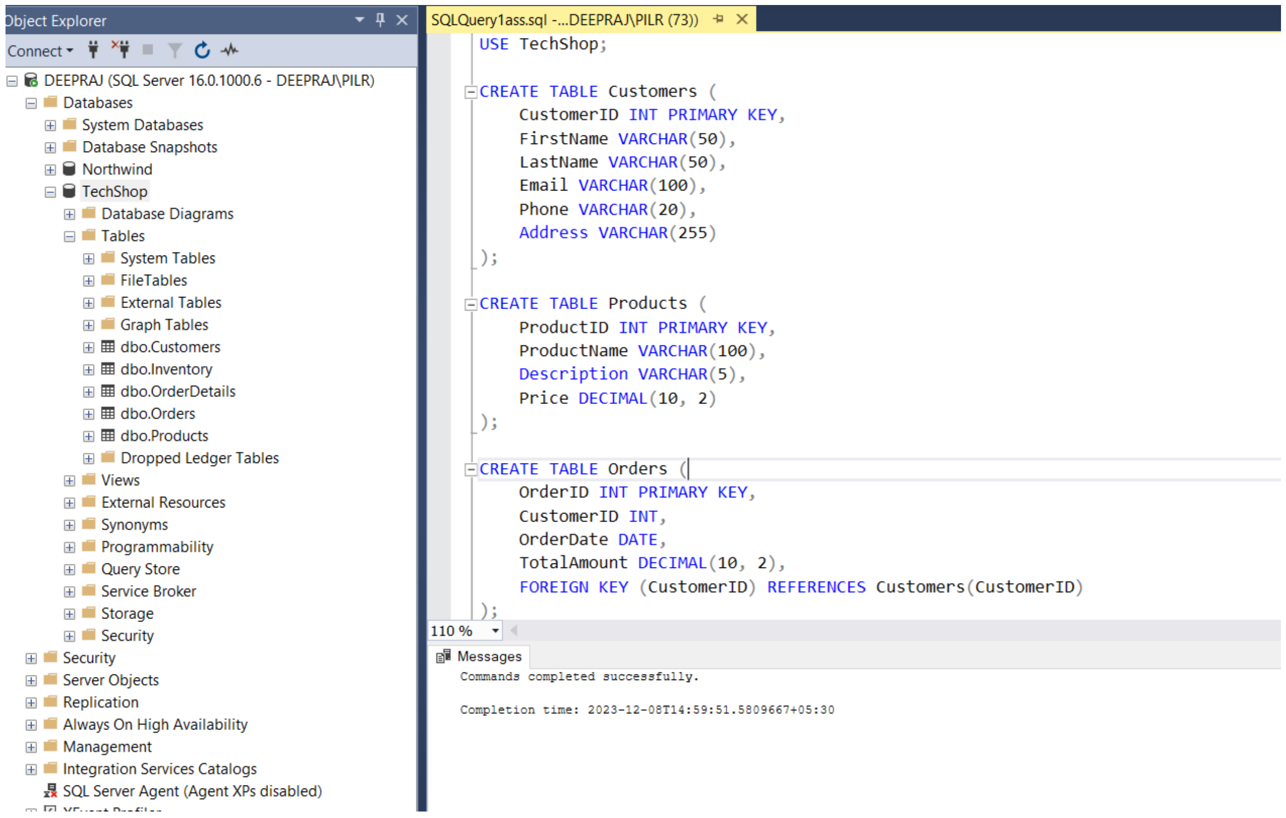
Messages

```
Commands completed successfully.

Completion time: 2023-12-08T14:34:52.7285551+05:30
```

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

Ans)

```
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 VARCHAR(5),
    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)
    )
```

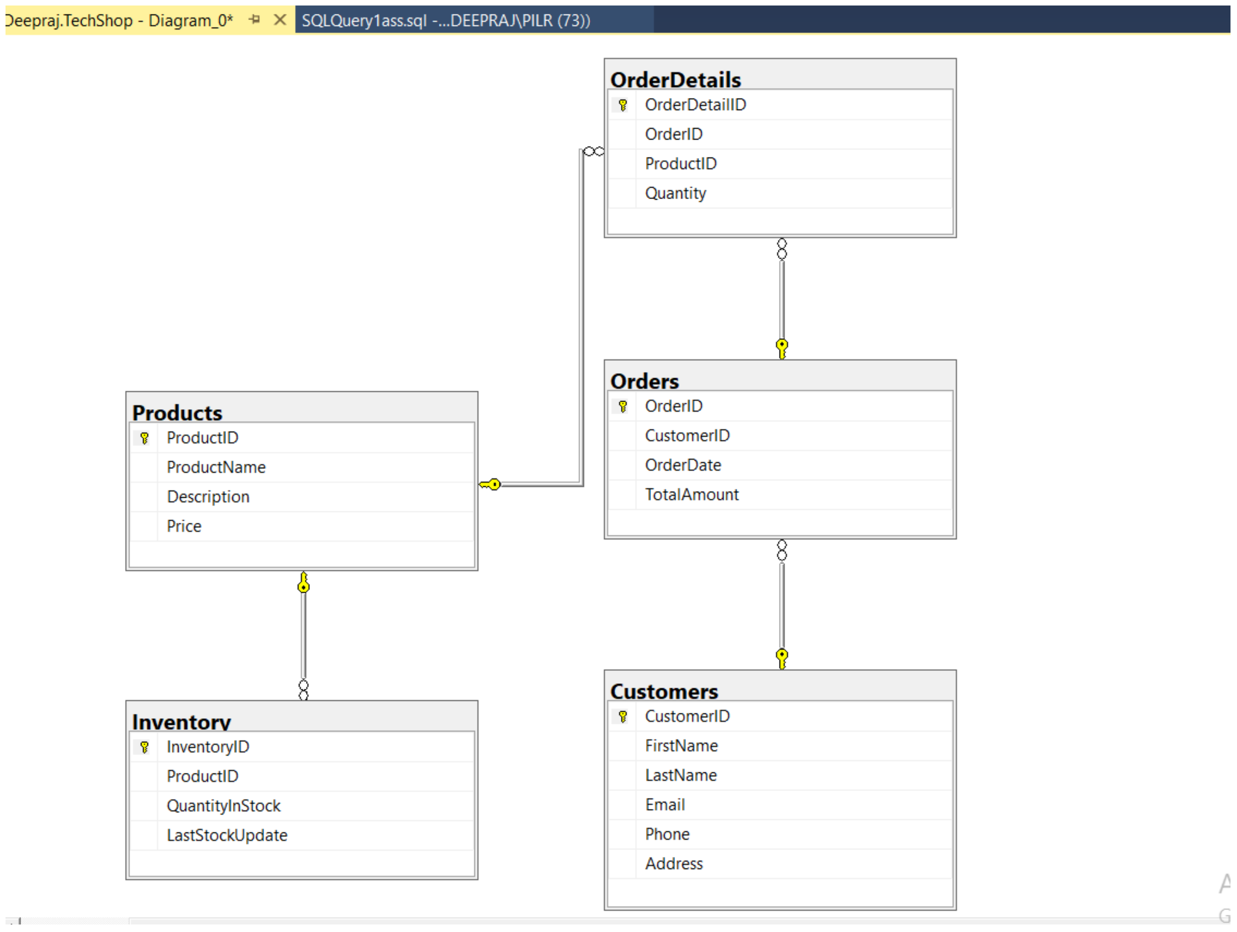


```
USE TechShop;

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 VARCHAR(5),
    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)
);
```

```
Commands completed successfully.

Completion time: 2023-12-08T14:59:51.5809667+05:30
```

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

Ans)



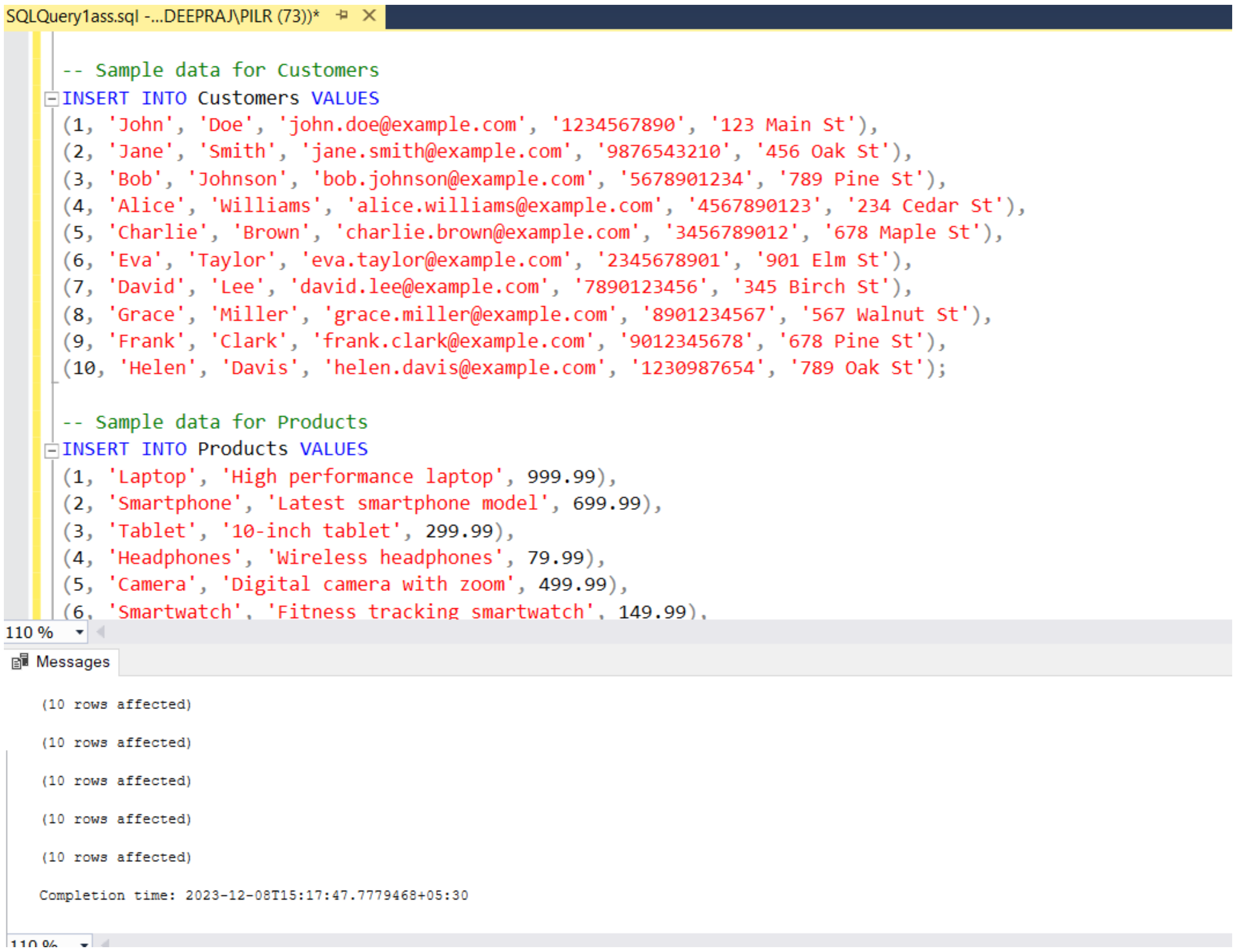
4. Create appropriate Primary Key and Foreign Key constraints for referential integrity.

   Ans) Already did in above question 1

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

ANS)

```sql
-- Sample data for Customers
INSERT INTO Customers VALUES
(1, 'John', 'Doe', 'john.doe@example.com', '1234567890', '123 Main St'),
(2, 'Jane', 'Smith', 'jane.smith@example.com', '9876543210', '456 Oak St'),
(3, 'Bob', 'Johnson', 'bob.johnson@example.com', '5678901234', '789 Pine St'),
(4, 'Alice', 'Williams', 'alice.williams@example.com', '4567890123', '234 Cedar St'),
(5, 'Charlie', 'Brown', 'charlie.brown@example.com', '3456789012', '678 Maple St'),
(6, 'Eva', 'Taylor', 'eva.taylor@example.com', '2345678901', '901 Elm St'),
(7, 'David', 'Lee', 'david.lee@example.com', '7890123456', '345 Birch St'),
(8, 'Grace', 'Miller', 'grace.miller@example.com', '8901234567', '567 Walnut St'),
(9, 'Frank', 'Clark', 'frank.clark@example.com', '9012345678', '678 Pine St'),
(10, 'Helen', 'Davis', 'helen.davis@example.com', '1230987654', '789 Oak St');

-- Sample data for Products
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, 'Headphones', 'Wireless headphones', 79.99),
(5, 'Camera', 'Digital camera with zoom', 499.99),
(6, 'Smartwatch', 'Fitness tracking smartwatch', 149.99),
(7, 'Printer', 'Color inkjet printer', 199.99),
(8, 'External Hard Drive', '1TB portable hard drive', 129.99),
(9, 'Gaming Console', 'Next-gen gaming console', 499.99),
(10, 'Bluetooth Speaker', 'Waterproof Bluetooth speaker', 59.99);

-- Sample data for Orders
INSERT INTO Orders VALUES
(1, 1, '2023-01-01', 1500.00),
(2, 2, '2023-02-01', 1200.00),
(3, 3, '2023-03-01', 800.00),
(4, 4, '2023-04-01', 500.00),
(5, 5, '2023-05-01', 2000.00),
(6, 6, '2023-06-01', 1000.00),
(7, 7, '2023-07-01', 700.00),
(8, 8, '2023-08-01', 900.00),
(9, 9, '2023-09-01', 300.00),
(10, 10, '2023-10-01', 1200.00);

-- Sample data for OrderDetails
INSERT INTO OrderDetails VALUES
(1, 1, 1, 2),
(2, 1, 2, 1),
(3, 2, 3, 3),
(4, 3, 4, 1),
(5, 4, 5, 2),
(6, 5, 6, 1),
(7, 6, 7, 2),
(8, 7, 8, 1),
(9, 8, 9, 3),
(10, 9, 10, 1);

-- Sample data for Inventory
INSERT INTO Inventory VALUES
(1, 1, 50, '2023-01-01'),
(2, 2, 100, '2023-02-01'),
(3, 3, 30, '2023-03-01'),
(4, 4, 20, '2023-04-01'),
(5, 5, 10, '2023-05-01'),
(6, 6, 40, '2023-06-01'),
(7, 7, 25, '2023-07-01'),
(8, 8, 15, '2023-08-01'),
(9, 9, 5, '2023-09-01'),
(10, 10, 35, '2023-10-01');
```



SQLQuery1ass.sql -...DEEPRAJ\PILR (73))*

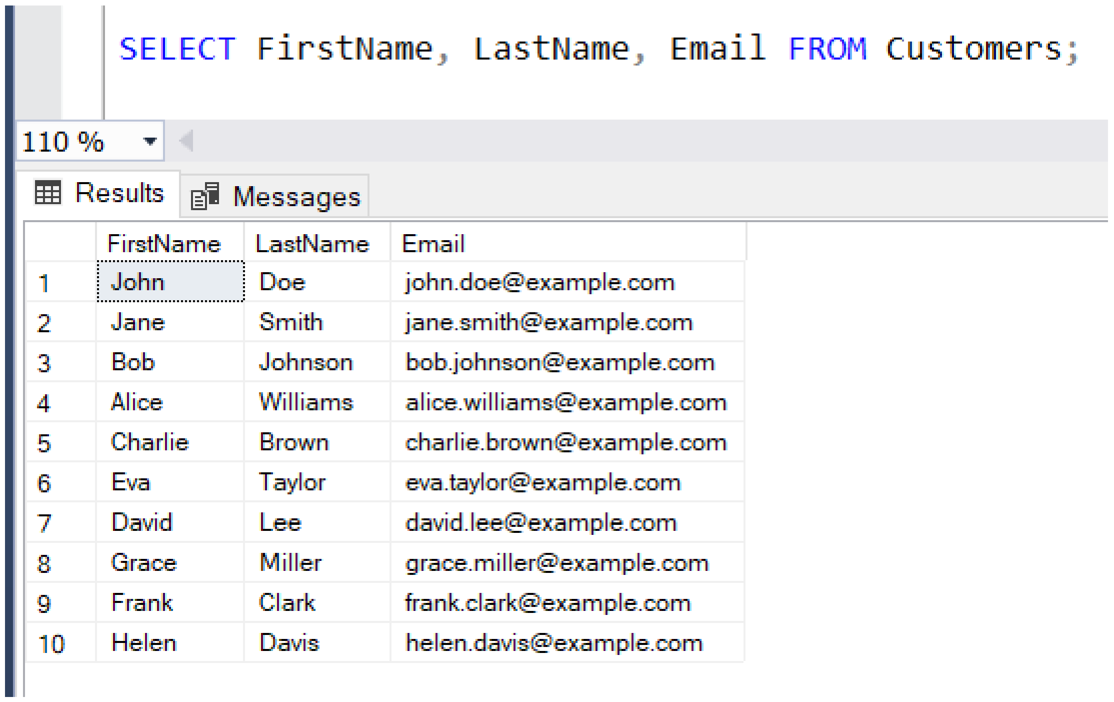
```
-- Sample data for Customers
INSERT INTO Customers VALUES
(1, 'John', 'Doe', 'john.doe@example.com', '1234567890', '123 Main St'),
(2, 'Jane', 'Smith', 'jane.smith@example.com', '9876543210', '456 Oak St'),
(3, 'Bob', 'Johnson', 'bob.johnson@example.com', '5678901234', '789 Pine St'),
(4, 'Alice', 'Williams', 'alice.williams@example.com', '4567890123', '234 Cedar St'),
(5, 'Charlie', 'Brown', 'charlie.brown@example.com', '3456789012', '678 Maple St'),
(6, 'Eva', 'Taylor', 'eva.taylor@example.com', '2345678901', '901 Elm St'),
(7, 'David', 'Lee', 'david.lee@example.com', '7890123456', '345 Birch St'),
(8, 'Grace', 'Miller', 'grace.miller@example.com', '8901234567', '567 Walnut St'),
(9, 'Frank', 'Clark', 'frank.clark@example.com', '9012345678', '678 Pine St'),
(10, 'Helen', 'Davis', 'helen.davis@example.com', '1230987654', '789 Oak St');

-- Sample data for Products
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, 'Headphones', 'Wireless headphones', 79.99),
(5, 'Camera', 'Digital camera with zoom', 499.99),
(6, 'Smartwatch', 'Fitness tracking smartwatch', 149.99),
```

```
(10 rows affected)

(10 rows affected)

(10 rows affected)

(10 rows affected)

(10 rows affected)

Completion time: 2023-12-08T15:17:47.7779468+05:30
```

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

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

   Ans) `SELECT FirstName, LastName, Email FROM Customers;`



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

110 %

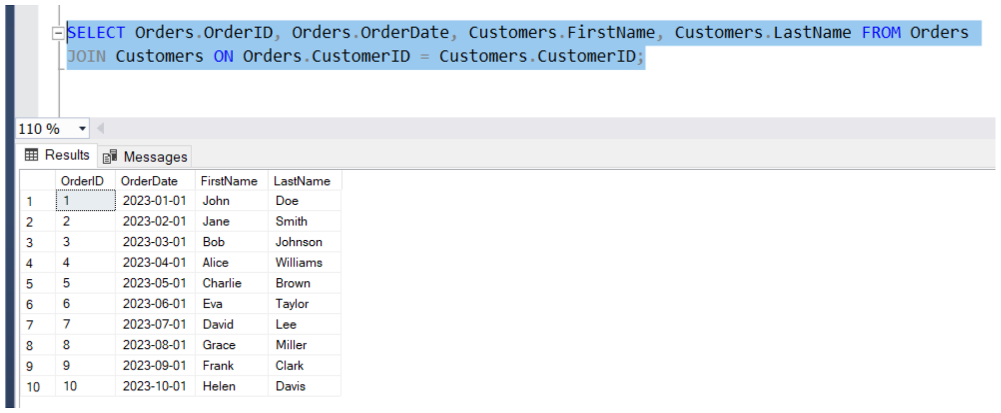
Results | Messages

	FirstName	LastName	Email
1	John	Doe	john.doe@example.com
2	Jane	Smith	jane.smith@example.com
3	Bob	Johnson	bob.johnson@example.com
4	Alice	Williams	alice.williams@example.com
5	Charlie	Brown	charlie.brown@example.com
6	Eva	Taylor	eva.taylor@example.com
7	David	Lee	david.lee@example.com
8	Grace	Miller	grace.miller@example.com
9	Frank	Clark	frank.clark@example.com
10	Helen	Davis	helen.davis@example.com

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

   Ans) `SELECT Orders.OrderID, Orders.OrderDate, Customers.FirstName, Customers.LastName FROM Orders`

   `JOIN Customers ON Orders.CustomerID = Customers.CustomerID;`



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

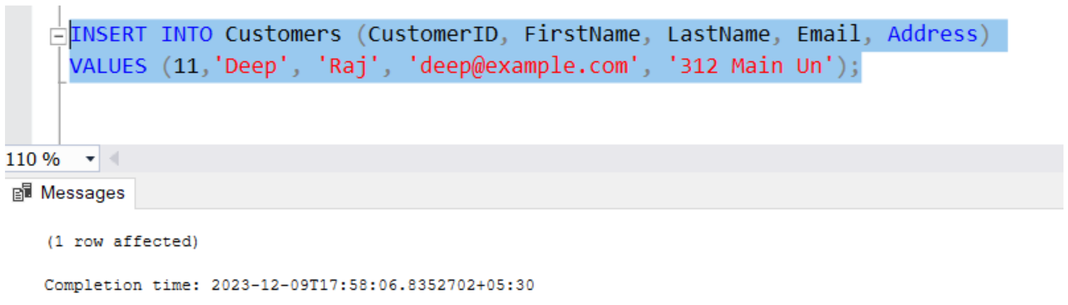
110 %

Results | Messages

	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	David	Lee
8	8	2023-08-01	Grace	Miller
9	9	2023-09-01	Frank	Clark
10	10	2023-10-01	Helen	Davis

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

Ans) `INSERT INTO Customers (CustomerID, FirstName, LastName, Email, Address)`
`VALUES (11,'Deep', 'Raj', 'deep@example.com', '312 Main Un');`

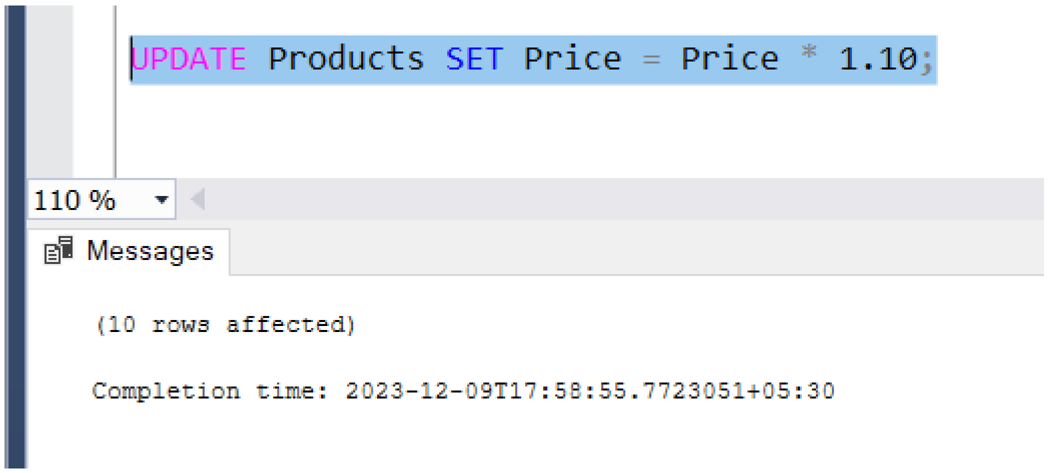


```
INSERT INTO Customers (CustomerID, FirstName, LastName, Email, Address)
VALUES (11,'Deep', 'Raj', 'deep@example.com', '312 Main Un');
```

```
110 %
Messages

(1 row affected)

Completion time: 2023-12-09T17:58:06.8352702+05:30
```

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

   Ans) `UPDATE Products SET Price = Price * 1.10;`



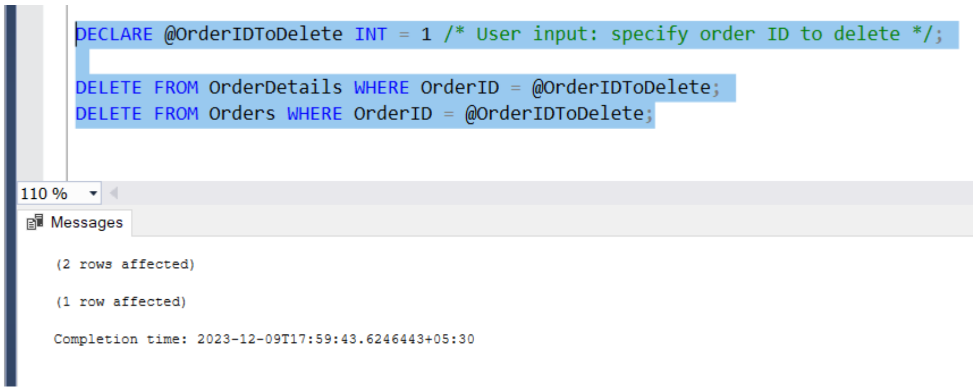
```
UPDATE Products SET Price = Price * 1.10;
```

```
110 %
Messages

(10 rows affected)

Completion time: 2023-12-09T17:58:55.7723051+05:30
```

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.

   Ans) `DECLARE @OrderIDToDelete INT = 1 /* User input: order ID to delete */;`

```
DELETE FROM OrderDetails WHERE OrderID = @OrderIDToDelete;
DELETE FROM Orders WHERE OrderID = @OrderIDToDelete;
```

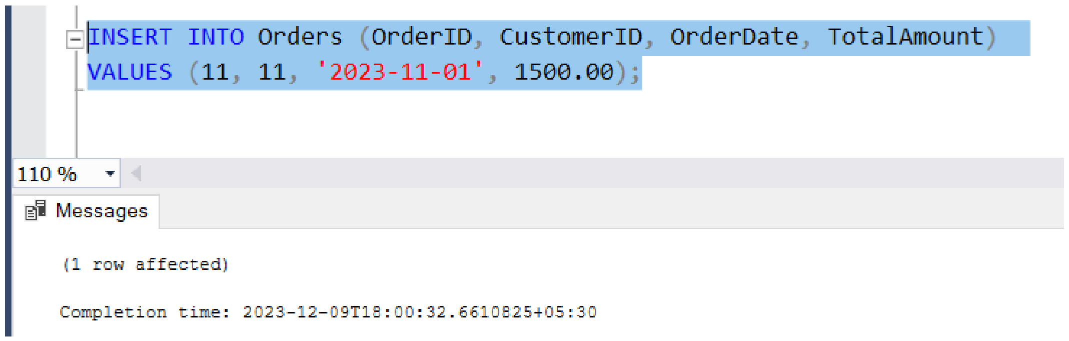


```
DECLARE @OrderIDToDelete INT = 1 /* User input: specify order ID to delete */;

DELETE FROM OrderDetails WHERE OrderID = @OrderIDToDelete;
DELETE FROM Orders WHERE OrderID = @OrderIDToDelete;
```

```
110 %
Messages

(2 rows affected)

(1 row affected)

Completion time: 2023-12-09T17:59:43.6246443+05:30
```

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.

Ans)
```
INSERT INTO Orders (OrderID, CustomerID, OrderDate, TotalAmount)
VALUES (11, 11, '2023-11-01', 1500.00);
```



```
INSERT INTO Orders (OrderID, CustomerID, OrderDate, TotalAmount)
VALUES (11, 11, '2023-11-01', 1500.00);
```

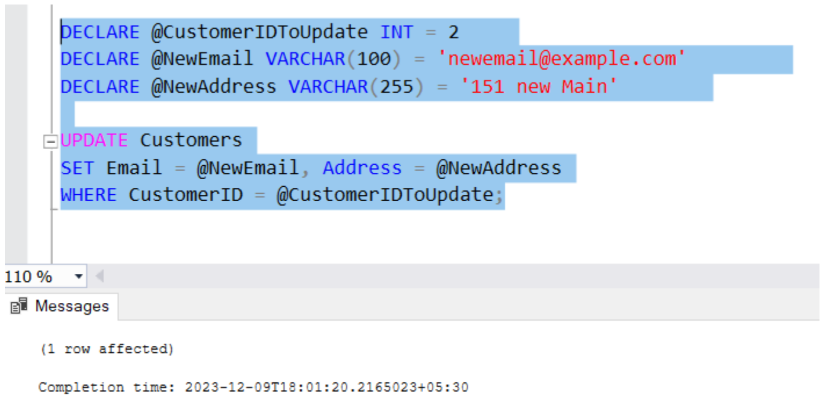
110 %

Messages

```
(1 row affected)

Completion time: 2023-12-09T18:00:32.6610825+05:30
```

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.

Ans)
```
DECLARE @CustomerIDToUpdate INT = 2
DECLARE @NewEmail VARCHAR(100) = 'newemail@example.com'
DECLARE @NewAddress VARCHAR(255) = '151 new Main'

UPDATE Customers
SET Email = @NewEmail, Address = @NewAddress
WHERE CustomerID = @CustomerIDToUpdate;
```



```
DECLARE @CustomerIDToUpdate INT = 2
DECLARE @NewEmail VARCHAR(100) = 'newemail@example.com'
DECLARE @NewAddress VARCHAR(255) = '151 new Main'

UPDATE Customers
SET Email = @NewEmail, Address = @NewAddress
WHERE CustomerID = @CustomerIDToUpdate;
```

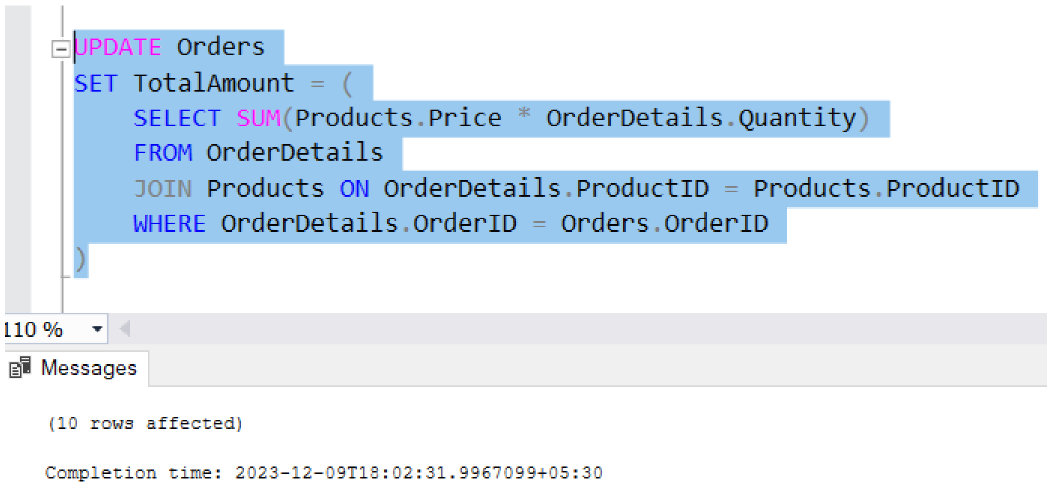
110 %

Messages

```
(1 row affected)

Completion time: 2023-12-09T18:01:20.2165023+05:30
```

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.

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



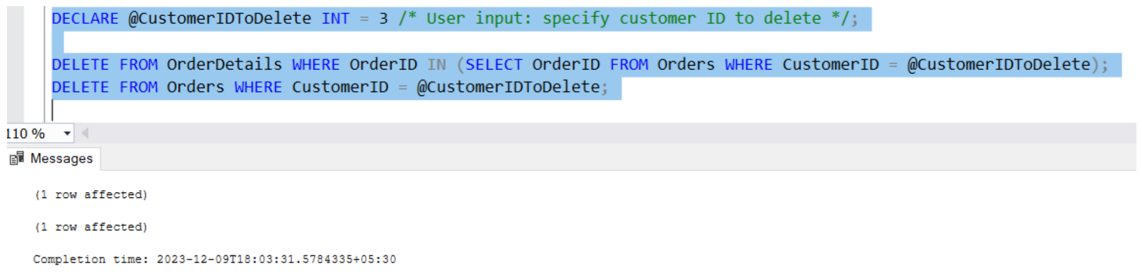
```sql
UPDATE Orders
SET TotalAmount = (
    SELECT SUM(Products.Price * OrderDetails.Quantity)
    FROM OrderDetails
    JOIN Products ON OrderDetails.ProductID = Products.ProductID
    WHERE OrderDetails.OrderID = Orders.OrderID
)
```

```
(10 rows affected)

Completion time: 2023-12-09T18:02:31.9967099+05:30
```

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.

Ans) `DECLARE @CustomerIDToDelete INT = 3 /* User input:customer ID to delete */;`

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

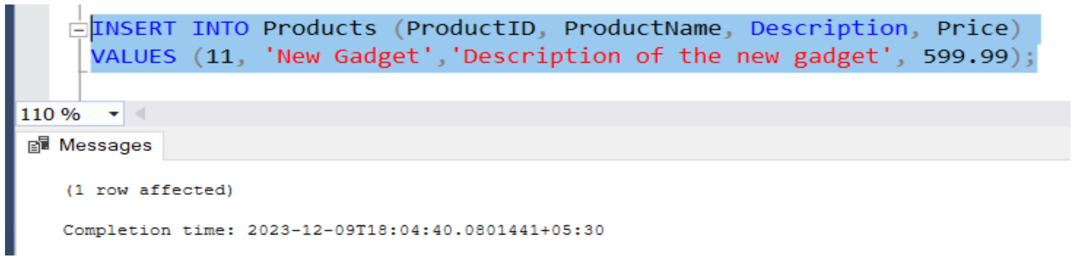


```sql
DECLARE @CustomerIDToDelete INT = 3 /* User input: specify customer ID to delete */;

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

```
(1 row affected)

(1 row affected)

Completion time: 2023-12-09T18:03:31.5784335+05:30
```

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.

Ans)
```sql
INSERT INTO Products (ProductID, ProductName, Description, Price)
VALUES (11, 'New Gadget','Description of the new gadget', 599.99);
```



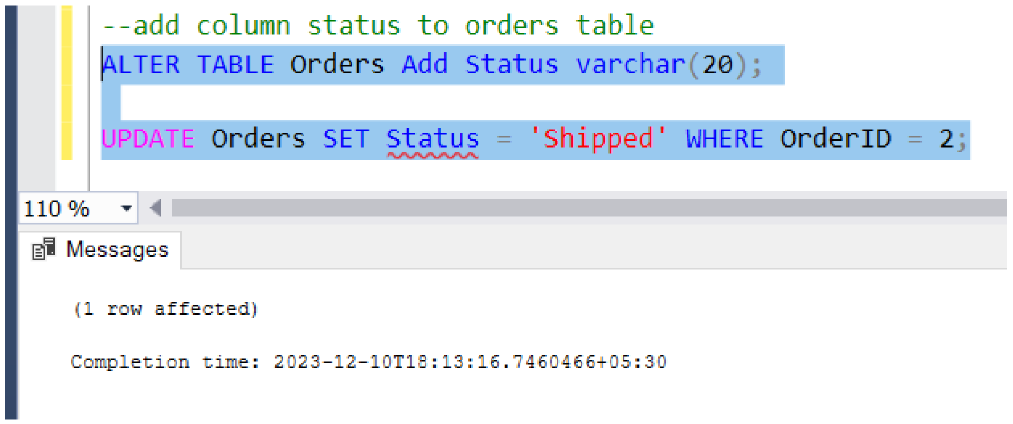
```sql
INSERT INTO Products (ProductID, ProductName, Description, Price)
VALUES (11, 'New Gadget','Description of the new gadget', 599.99);
```

```
(1 row affected)

Completion time: 2023-12-09T18:04:40.0801441+05:30
```

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.

Ans) `--add column status to orders table`

```
ALTER TABLE Orders Add Status varchar(20);

UPDATE Orders SET Status = 'Shipped' WHERE OrderID = 2;
```

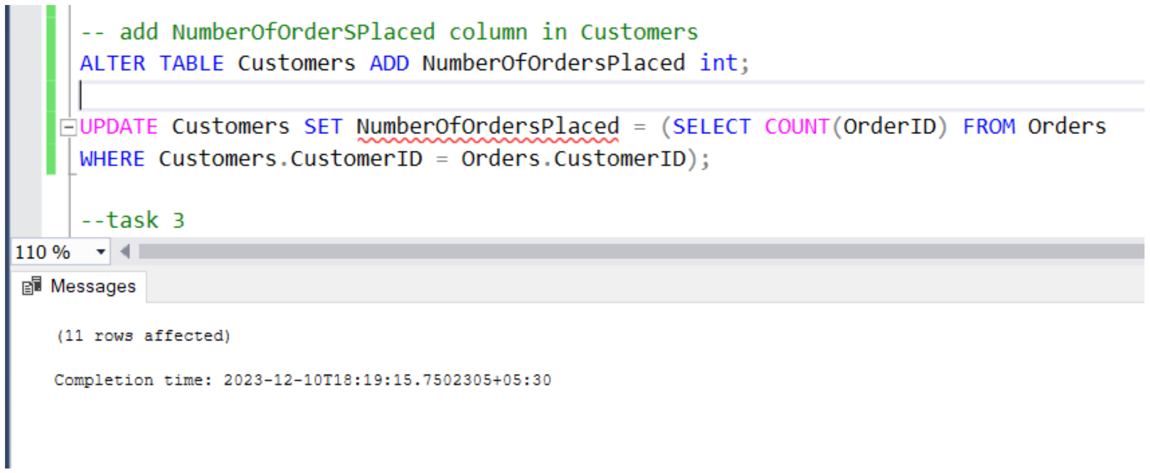


```
--add column status to orders table
ALTER TABLE Orders Add Status varchar(20);

UPDATE Orders SET Status = 'Shipped' WHERE OrderID = 2;
```

```
(1 row affected)

Completion time: 2023-12-10T18:13:16.7460466+05:30
```

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.

Ans)
```
-- add NumberOfOrderSPlaced column in Customers
ALTER TABLE Customers ADD NumberOfOrdersPlaced int;

UPDATE Customers SET NumberOfOrdersPlaced = (SELECT COUNT(OrderID) FROM Orders
WHERE Customers.CustomerID = Orders.CustomerID);
```



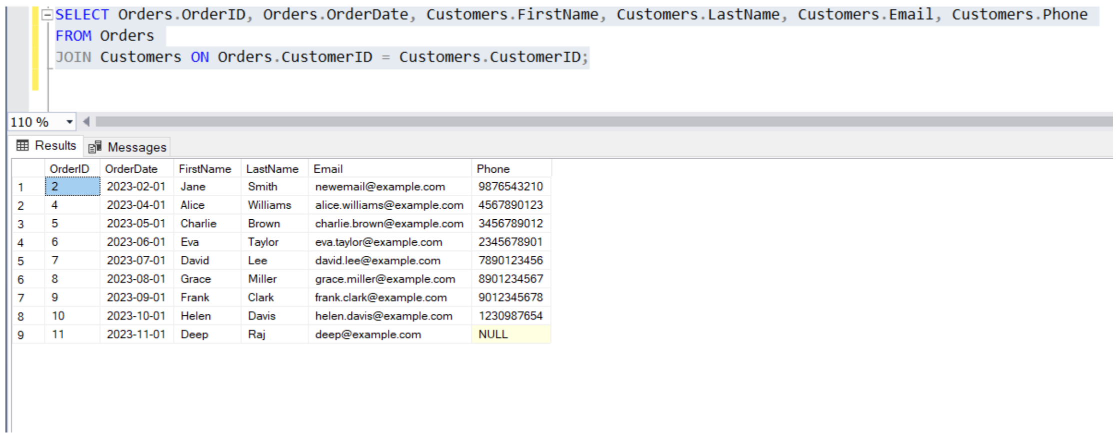
```
-- add NumberOfOrderSPlaced column in Customers
ALTER TABLE Customers ADD NumberOfOrdersPlaced int;

UPDATE Customers SET NumberOfOrdersPlaced = (SELECT COUNT(OrderID) FROM Orders
WHERE Customers.CustomerID = Orders.CustomerID);

--task 3
```

```
(11 rows affected)

Completion time: 2023-12-10T18:19:15.7502305+05:30
```

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.

Ans)
```
SELECT Orders.OrderID, Orders.OrderDate, Customers.FirstName,
Customers.LastName, Customers.Email, Customers.Phone FROM Orders

 JOIN Customers ON Orders.CustomerID = Customers.CustomerID;
```



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

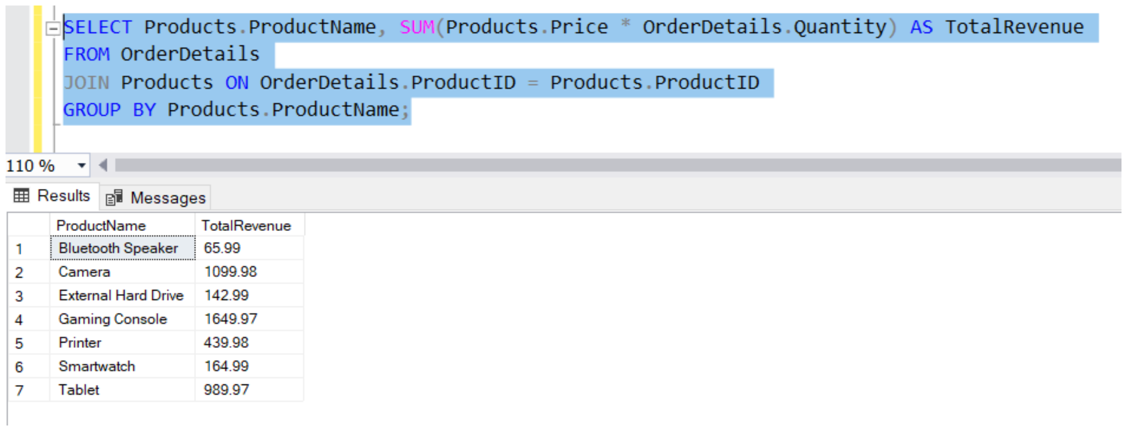
110 %

Results | Messages

	OrderID	OrderDate	FirstName	LastName	Email	Phone
1	2	2023-02-01	Jane	Smith	newemail@example.com	9876543210
2	4	2023-04-01	Alice	Williams	alice.williams@example.com	4567890123
3	5	2023-05-01	Charlie	Brown	charlie.brown@example.com	3456789012
4	6	2023-06-01	Eva	Taylor	eva.taylor@example.com	2345678901
5	7	2023-07-01	David	Lee	david.lee@example.com	7890123456
6	8	2023-08-01	Grace	Miller	grace.miller@example.com	8901234567
7	9	2023-09-01	Frank	Clark	frank.clark@example.com	9012345678
8	10	2023-10-01	Helen	Davis	helen.davis@example.com	1230987654
9	11	2023-11-01	Deep	Raj	deep@example.com	NULL

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

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



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

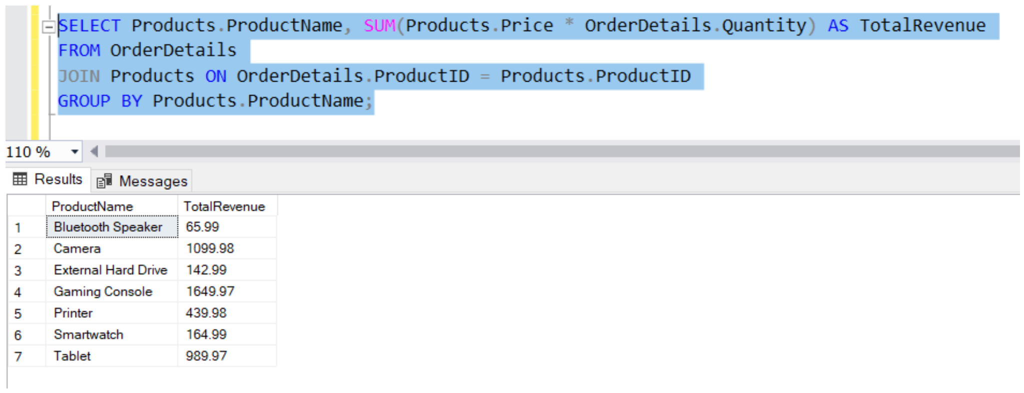
110 %

Results | Messages

	ProductName	TotalRevenue
1	Bluetooth Speaker	65.99
2	Camera	1099.98
3	External Hard Drive	142.99
4	Gaming Console	1649.97
5	Printer	439.98
6	Smartwatch	164.99
7	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.

Ans)
```
SELECT Customers.FirstName, Customers.LastName, Customers.Email,
Customers.Phone FROM Customers
 JOIN Orders ON Customers.CustomerID = Orders.CustomerID;
```



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

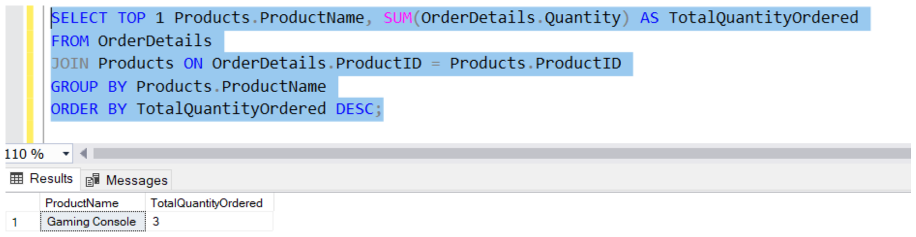
110 %

Results Messages

	ProductName	TotalRevenue
1	Bluetooth Speaker	65.99
2	Camera	1099.98
3	External Hard Drive	142.99
4	Gaming Console	1649.97
5	Printer	439.98
6	Smartwatch	164.99
7	Tablet	989.97

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.

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



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

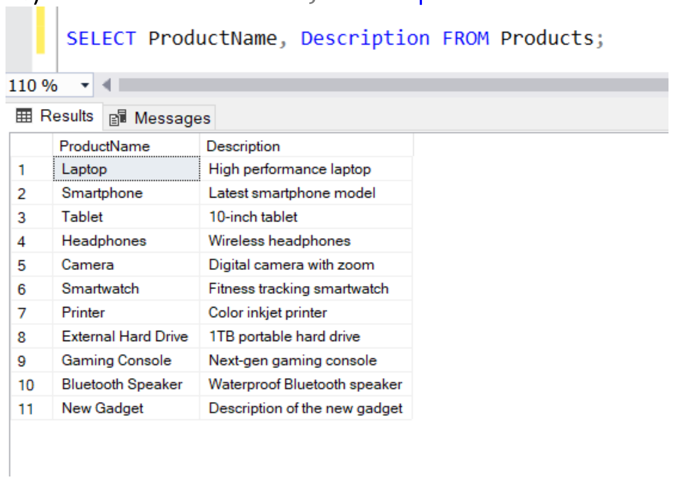
110 %

Results Messages

	ProductName	TotalQuantityOrdered
1	Gaming Console	3

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

Ans)
```
SELECT ProductName, Description FROM Products;
```



```
SELECT ProductName, Description FROM Products;
```

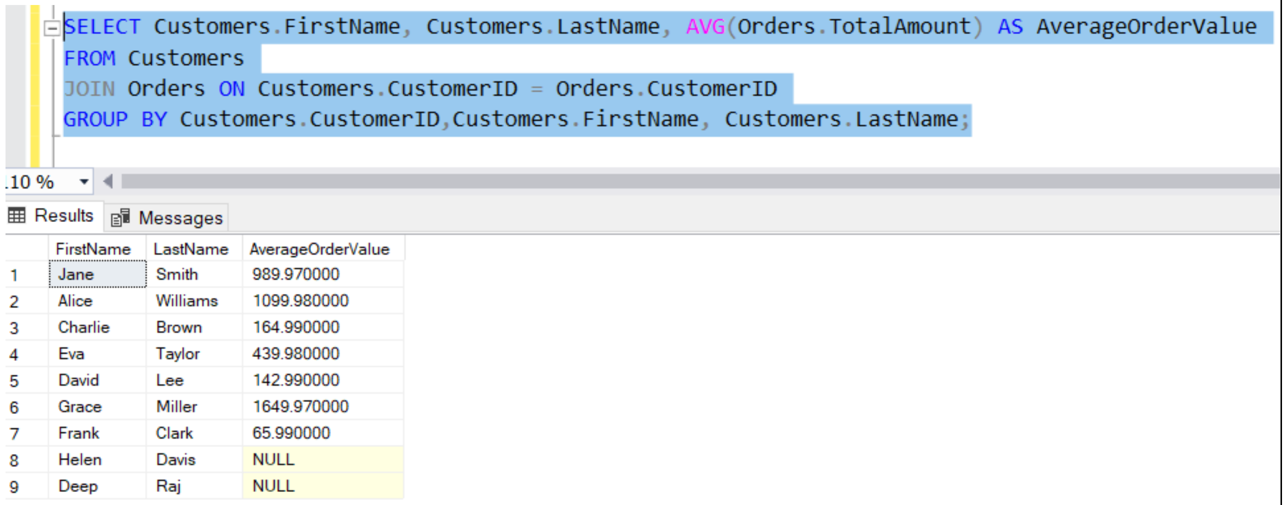
110 %

Results Messages

	ProductName	Description
1	Laptop	High performance laptop
2	Smartphone	Latest smartphone model
3	Tablet	10-inch tablet
4	Headphones	Wireless headphones
5	Camera	Digital camera with zoom
6	Smartwatch	Fitness tracking smartwatch
7	Printer	Color inkjet printer
8	External Hard Drive	1TB portable hard drive
9	Gaming Console	Next-gen gaming console
10	Bluetooth Speaker	Waterproof Bluetooth speaker
11	New Gadget	Description of the new gadget

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

Ans)
```
SELECT 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;
```



```
SELECT 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;
```

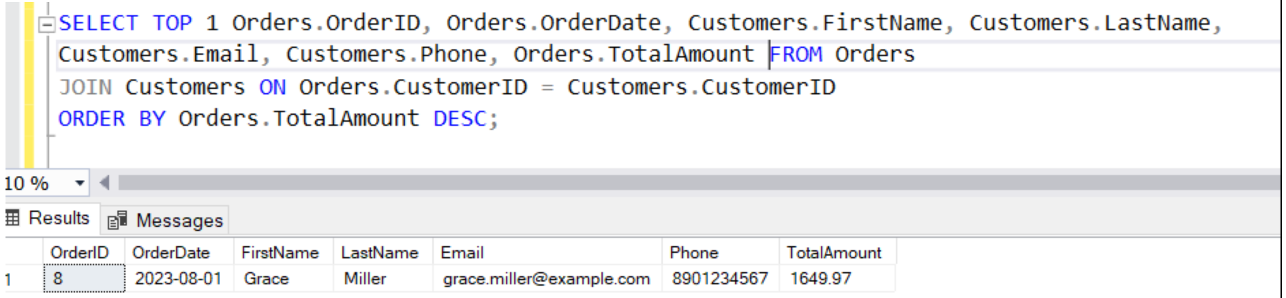
110 %

Results | Messages

	FirstName	LastName	AverageOrderValue
1	Jane	Smith	989.970000
2	Alice	Williams	1099.980000
3	Charlie	Brown	164.990000
4	Eva	Taylor	439.980000
5	David	Lee	142.990000
6	Grace	Miller	1649.970000
7	Frank	Clark	65.990000
8	Helen	Davis	NULL
9	Deep	Raj	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.

Ans)
```
SELECT TOP 1 Orders.OrderID, Orders.OrderDate, Customers.FirstName,
Customers.LastName, Customers.Email, Customers.Phone, Orders.TotalAmount FROM Orders
JOIN Customers ON Orders.CustomerID = Customers.CustomerID
 ORDER BY Orders.TotalAmount DESC;
```



```
SELECT TOP 1 Orders.OrderID, Orders.OrderDate, Customers.FirstName, Customers.LastName,
Customers.Email, Customers.Phone, Orders.TotalAmount FROM Orders
JOIN Customers ON Orders.CustomerID = Customers.CustomerID
ORDER BY Orders.TotalAmount DESC;
```

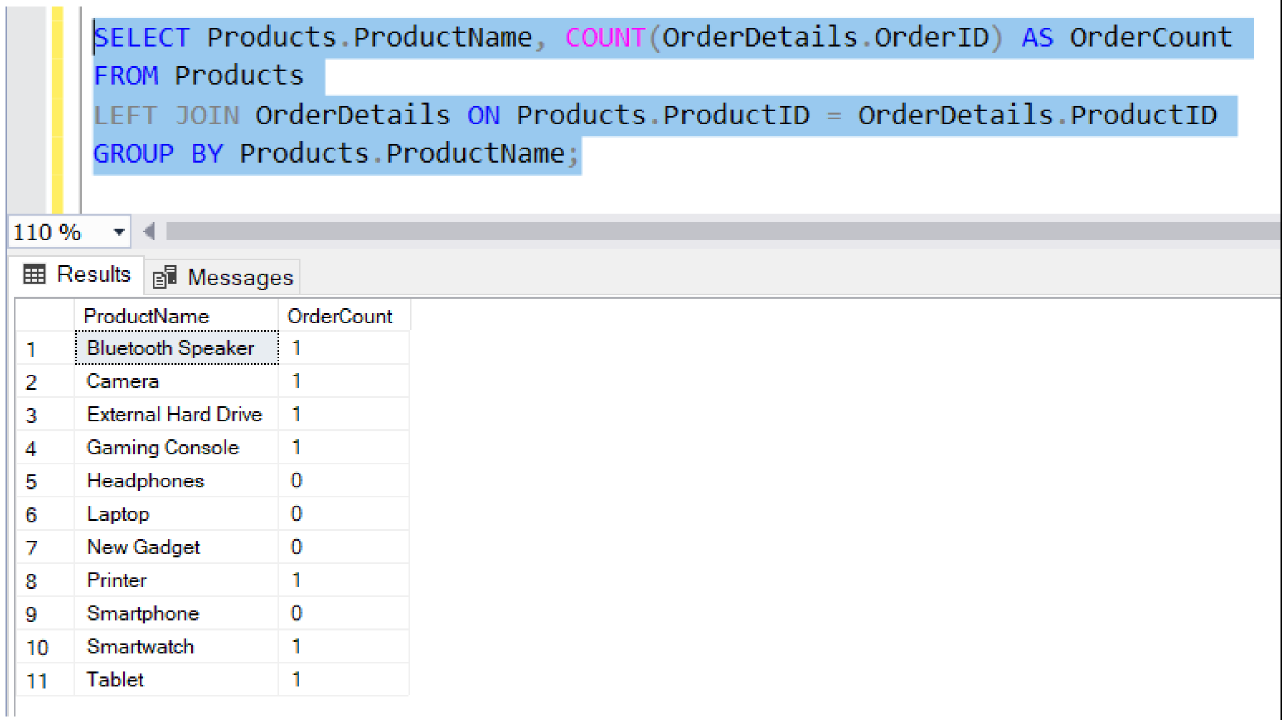
110 %

Results | Messages

	OrderID	OrderDate	FirstName	LastName	Email	Phone	TotalAmount
1	8	2023-08-01	Grace	Miller	grace.miller@example.com	8901234567	1649.97

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

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



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

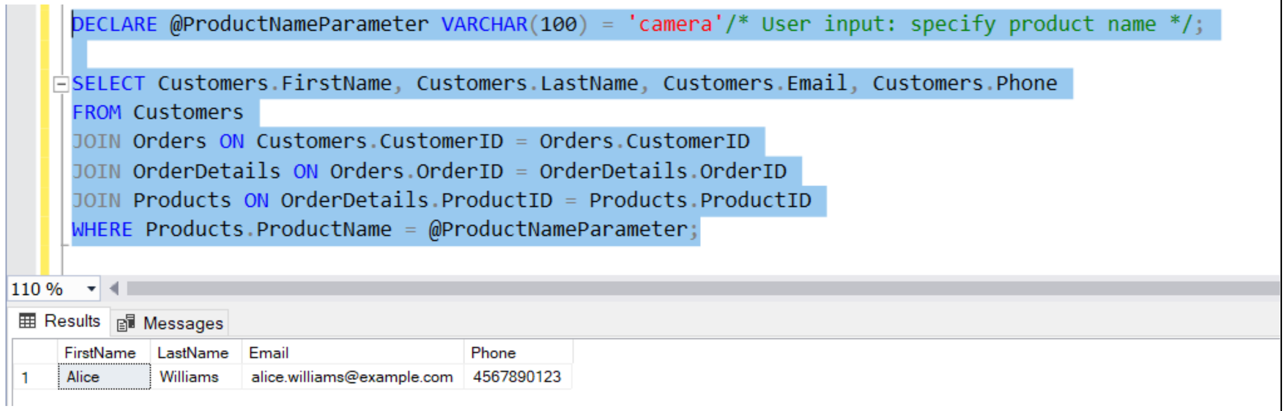
110 %

Results | Messages

	ProductName	OrderCount
1	Bluetooth Speaker	1
2	Camera	1
3	External Hard Drive	1
4	Gaming Console	1
5	Headphones	0
6	Laptop	0
7	New Gadget	0
8	Printer	1
9	Smartphone	0
10	Smartwatch	1
11	Tablet	1

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.

Ans)
```
DECLARE @ProductNameParameter VARCHAR(100) = 'camera'/* User input: */;

SELECT Customers.FirstName, Customers.LastName, Customers.Email, Customers.Phone
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 = @ProductNameParameter;
```



```
DECLARE @ProductNameParameter VARCHAR(100) = 'camera'/* User input: specify product name */;

SELECT Customers.FirstName, Customers.LastName, Customers.Email, Customers.Phone
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 = @ProductNameParameter;
```

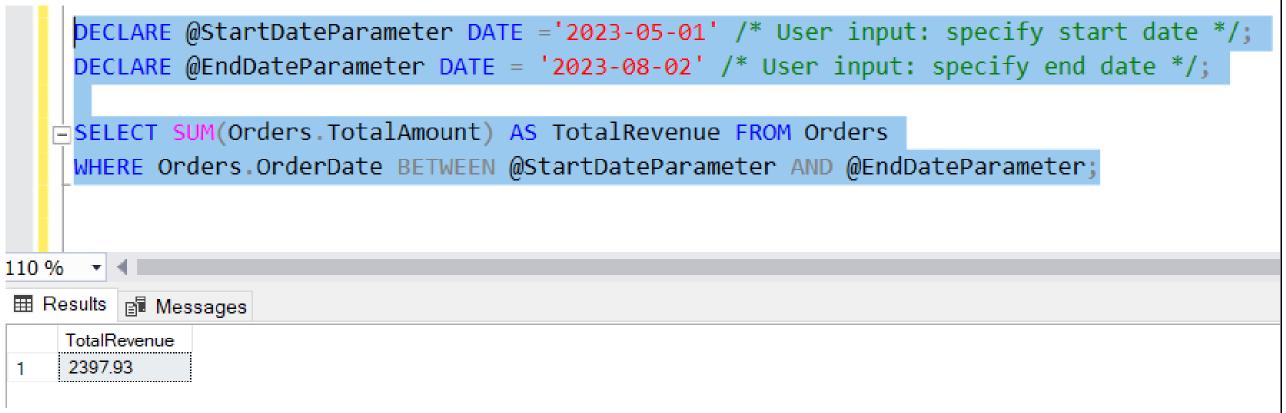
110 %

Results | Messages

	FirstName	LastName	Email	Phone
1	Alice	Williams	alice.williams@example.com	4567890123

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.

Ans)
```
DECLARE @StartDateParameter DATE ='2023-05-01'/* User input: start date */;
DECLARE @EndDateParameter DATE = '2023-08-02' /* User input: specify end date */;

SELECT SUM(Orders.TotalAmount) AS TotalRevenue FROM Orders
 WHERE Orders.OrderDate BETWEEN @StartDateParameter AND @EndDateParameter;
```



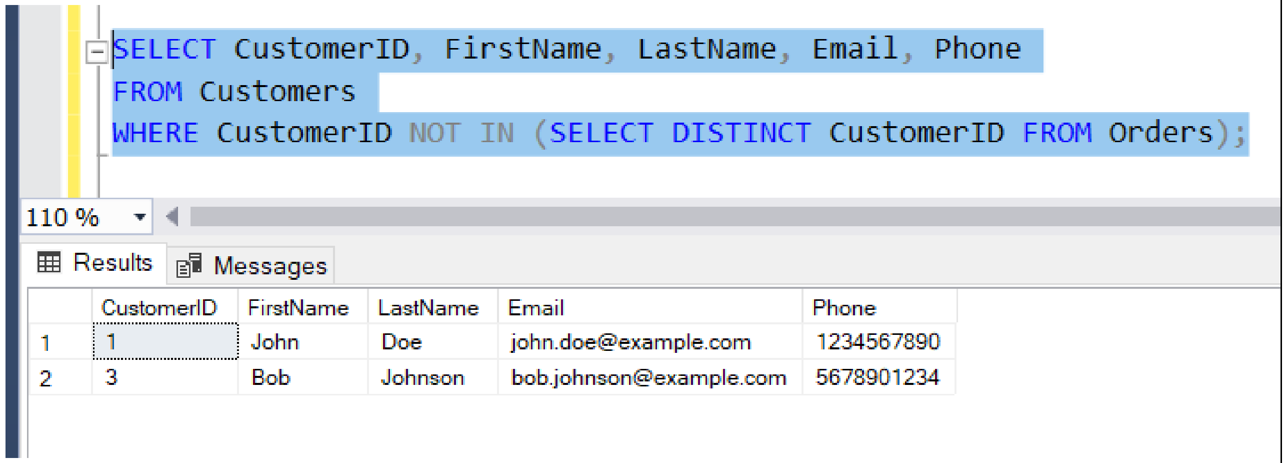
```
DECLARE @StartDateParameter DATE ='2023-05-01' /* User input: specify start date */;
DECLARE @EndDateParameter DATE = '2023-08-02' /* User input: specify end date */;

SELECT SUM(Orders.TotalAmount) AS TotalRevenue FROM Orders
WHERE Orders.OrderDate BETWEEN @StartDateParameter AND @EndDateParameter;
```

```
   TotalRevenue
1  2397.93
```

Task 4. Subquery and its type:

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

Ans)
```
SELECT CustomerID, FirstName, LastName, Email, Phone FROM Customers
WHERE CustomerID NOT IN (SELECT DISTINCT CustomerID FROM Orders);
```

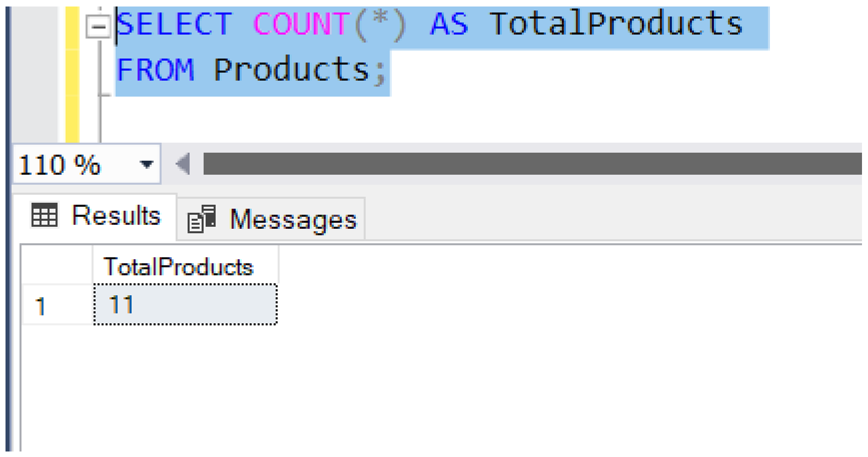


```
SELECT CustomerID, FirstName, LastName, Email, Phone
FROM Customers
WHERE CustomerID NOT IN (SELECT DISTINCT CustomerID FROM Orders);
```

```
   CustomerID  FirstName  LastName  Email                    Phone
1  1           John       Doe       john.doe@example.com     1234567890
2  3           Bob        Johnson   bob.johnson@example.com  5678901234
```

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

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

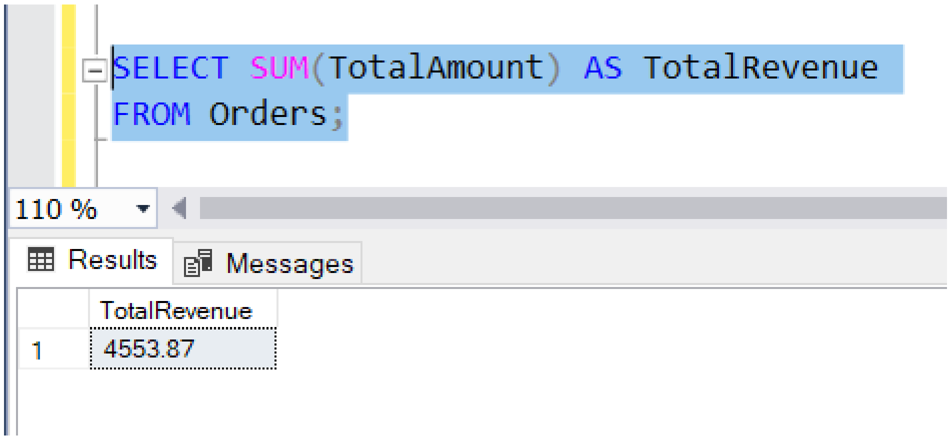


```
SELECT COUNT(*) AS TotalProducts
FROM Products;
```

```
   TotalProducts
1  11
```

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

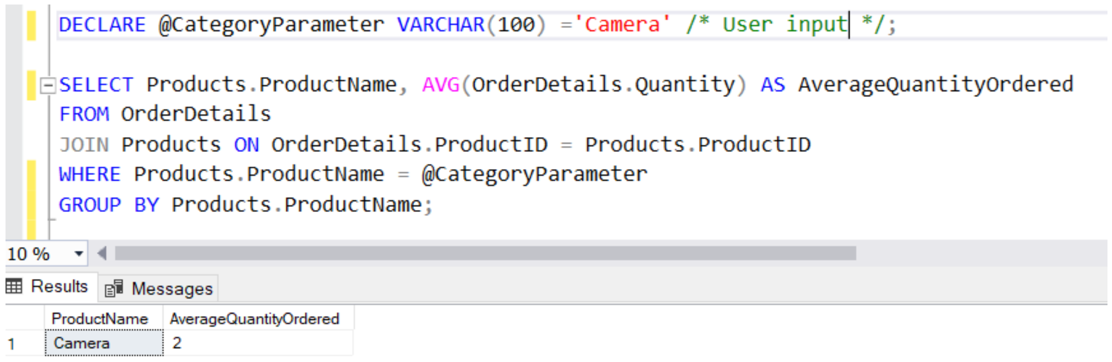
Ans)
```
SELECT SUM(TotalAmount) AS TotalRevenue
FROM Orders;
```


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

	TotalRevenue
1	4553.87

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.

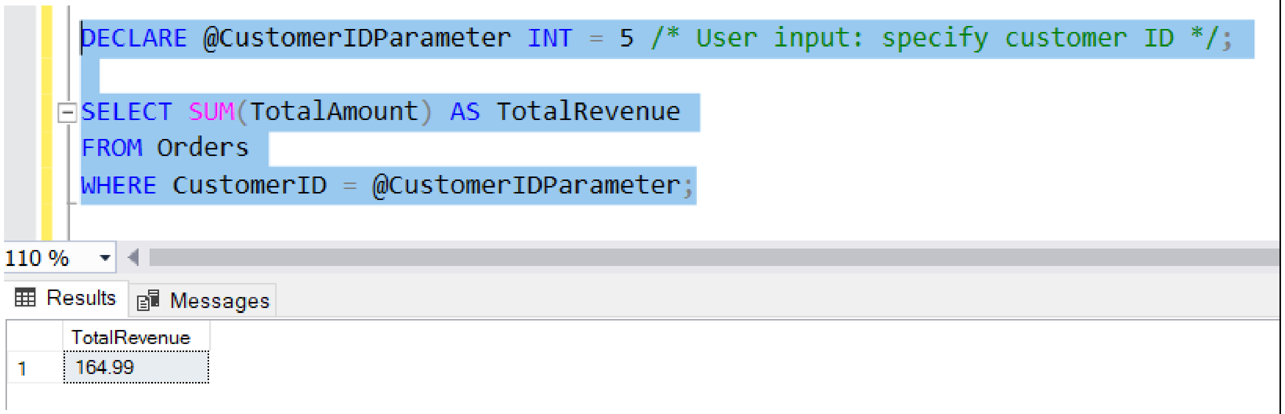
Ans)
```
DECLARE @CategoryParameter VARCHAR(100) ='Camera' /* User input */;

SELECT Products.ProductName, AVG(OrderDetails.Quantity) AS AverageQuantityOrdered
FROM OrderDetails
JOIN Products ON OrderDetails.ProductID = Products.ProductID
WHERE Products.ProductName = @CategoryParameter
 GROUP BY Products.ProductName;
```


```
DECLARE @CategoryParameter VARCHAR(100) ='Camera' /* User input */;

SELECT Products.ProductName, AVG(OrderDetails.Quantity) AS AverageQuantityOrdered
FROM OrderDetails
JOIN Products ON OrderDetails.ProductID = Products.ProductID
WHERE Products.ProductName = @CategoryParameter
GROUP BY Products.ProductName;
```

	ProductName	AverageQuantityOrdered
1	Camera	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.

Ans)
```
DECLARE @CustomerIDParameter INT = 5 /* User input: specify customer ID */;

SELECT SUM(TotalAmount) AS TotalRevenue
FROM Orders
 WHERE CustomerID = @CustomerIDParameter;
```

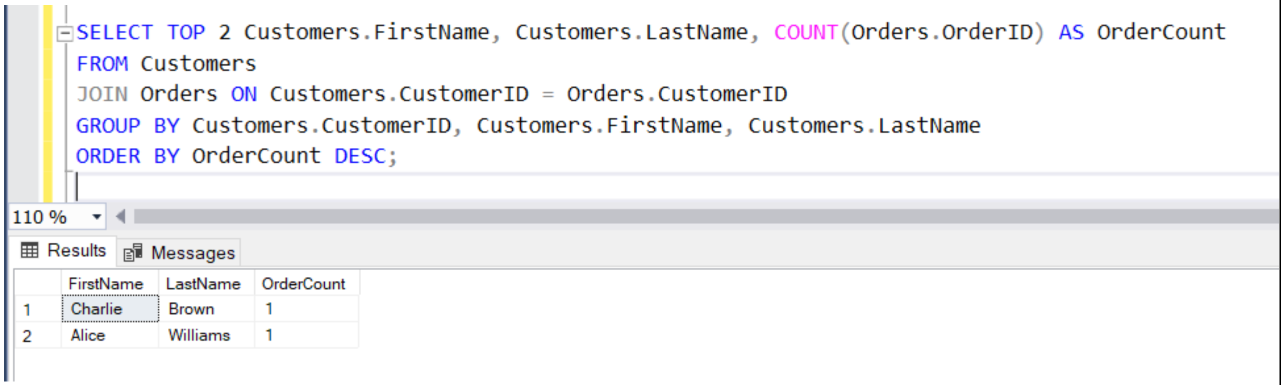


```
DECLARE @CustomerIDParameter INT = 5 /* User input: specify customer ID */;

SELECT SUM(TotalAmount) AS TotalRevenue
FROM Orders
WHERE CustomerID = @CustomerIDParameter;
```

```
110 %
Results  Messages
   TotalRevenue
1  164.99
```

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.

Ans)
```
SELECT TOP 2 Customers.FirstName, Customers.LastName, COUNT(Orders.OrderID) AS
OrderCount FROM Customers
JOIN Orders ON Customers.CustomerID = Orders.CustomerID
GROUP BY Customers.CustomerID, Customers.FirstName, Customers.LastName
ORDER BY OrderCount DESC;
```

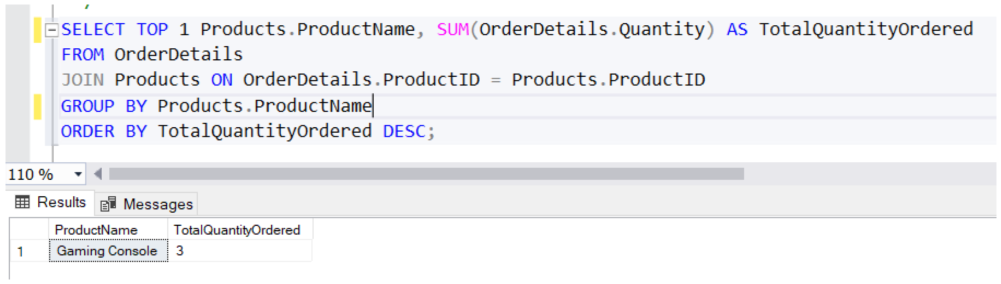


```
SELECT TOP 2 Customers.FirstName, Customers.LastName, COUNT(Orders.OrderID) AS OrderCount
FROM Customers
JOIN Orders ON Customers.CustomerID = Orders.CustomerID
GROUP BY Customers.CustomerID, Customers.FirstName, Customers.LastName
ORDER BY OrderCount DESC;
```

```
110 %
Results  Messages
   FirstName  LastName  OrderCount
1  Charlie    Brown     1
2  Alice      Williams  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.

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

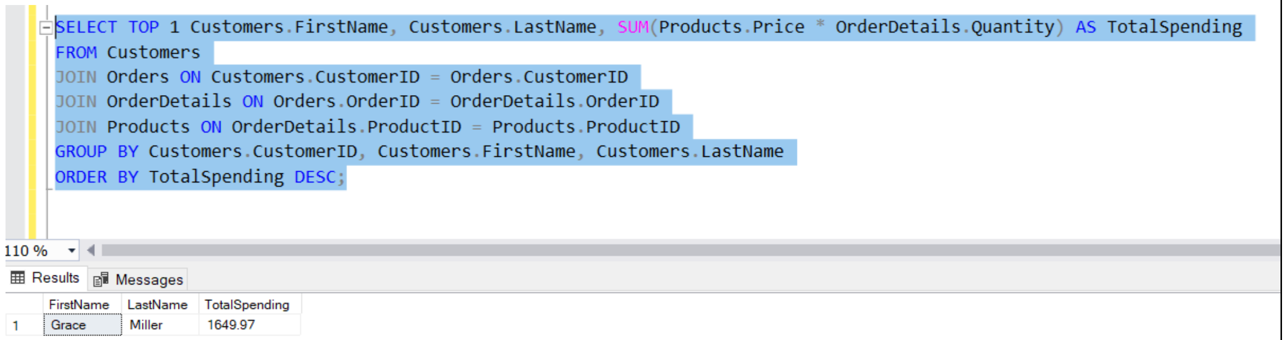


```
SELECT TOP 1 Products.ProductName, SUM(OrderDetails.Quantity) AS TotalQuantityOrdered
FROM OrderDetails
JOIN Products ON OrderDetails.ProductID = Products.ProductID
GROUP BY Products.ProductName
ORDER BY TotalQuantityOrdered DESC;
```

```
110 %
Results  Messages
   ProductName     TotalQuantityOrdered
1  Gaming Console  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.

Ans)
```
SELECT TOP 1 Customers.FirstName, Customers.LastName, SUM(Products.Price *
OrderDetails.Quantity) 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
GROUP BY Customers.CustomerID, Customers.FirstName, Customers.LastName
 ORDER BY TotalSpending DESC;
```

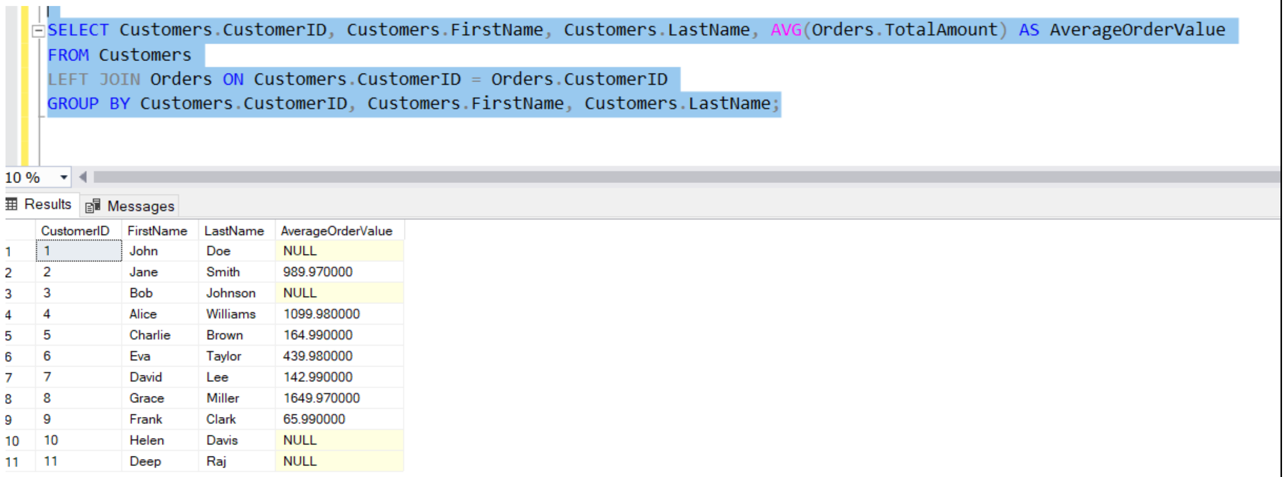


```
SELECT TOP 1 Customers.FirstName, Customers.LastName, SUM(Products.Price * OrderDetails.Quantity) AS TotalSpending
FROM Customers
JOIN Orders ON Customers.CustomerID = Orders.CustomerID
JOIN OrderDetails ON Orders.OrderID = OrderDetails.OrderID
JOIN Products ON OrderDetails.ProductID = Products.ProductID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

	FirstName	LastName	TotalSpending
1	Grace	Miller	1649.97

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

Ans)
```
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;
```

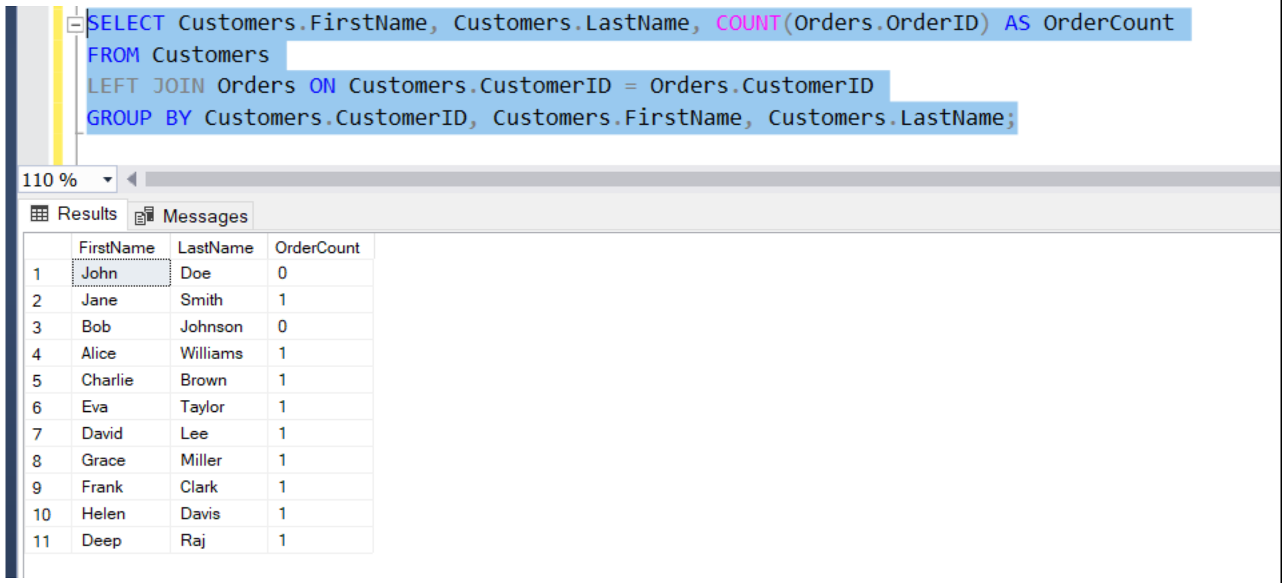


```
SELECT Customers.CustomerID, Customers.FirstName, Customers.LastName, AVG(Orders.TotalAmount) AS AverageOrderValue
FROM Customers
LEFT JOIN Orders ON Customers.CustomerID = Orders.CustomerID
GROUP BY Customers.CustomerID, Customers.FirstName, Customers.LastName;
```

	CustomerID	FirstName	LastName	AverageOrderValue
1	1	John	Doe	NULL
2	2	Jane	Smith	989.970000
3	3	Bob	Johnson	NULL
4	4	Alice	Williams	1099.980000
5	5	Charlie	Brown	164.990000
6	6	Eva	Taylor	439.980000
7	7	David	Lee	142.990000
8	8	Grace	Miller	1649.970000
9	9	Frank	Clark	65.990000
10	10	Helen	Davis	NULL
11	11	Deep	Raj	NULL

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.

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



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

110 %

Results | Messages

	FirstName	LastName	OrderCount
1	John	Doe	0
2	Jane	Smith	1
3	Bob	Johnson	0
4	Alice	Williams	1
5	Charlie	Brown	1
6	Eva	Taylor	1
7	David	Lee	1
8	Grace	Miller	1
9	Frank	Clark	1
10	Helen	Davis	1
11	Deep	Raj	1