Varshini E

Assignment- TechShop, an electronic gadgets shop

Task:1. Database Design:

1. Create the database named "TechShop"

create database TechShop;

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

- **Customers Table**

- CustomerID (Primary Key, INT)

- FirstName (VARCHAR)

- LastName (VARCHAR)

- Email (VARCHAR)

- Phone (VARCHAR)

- Address (VARCHAR)

- **Products Table**

- ProductID (Primary Key, INT)

- ProductName (VARCHAR)

- Description (VARCHAR)

- Price (DECIMAL)

- **Orders Table**

- OrderID (Primary Key, INT)

- CustomerID (Foreign Key referencing Customers, INT)

- OrderDate (DATE)

- TotalAmount (DECIMAL)

- **OrderDetails Table**

- OrderDetailID (Primary Key, INT)

- OrderID (Foreign Key referencing Orders, INT)

- ProductID (Foreign Key referencing Products, INT)

- Quantity (INT)

- **Inventory Table**

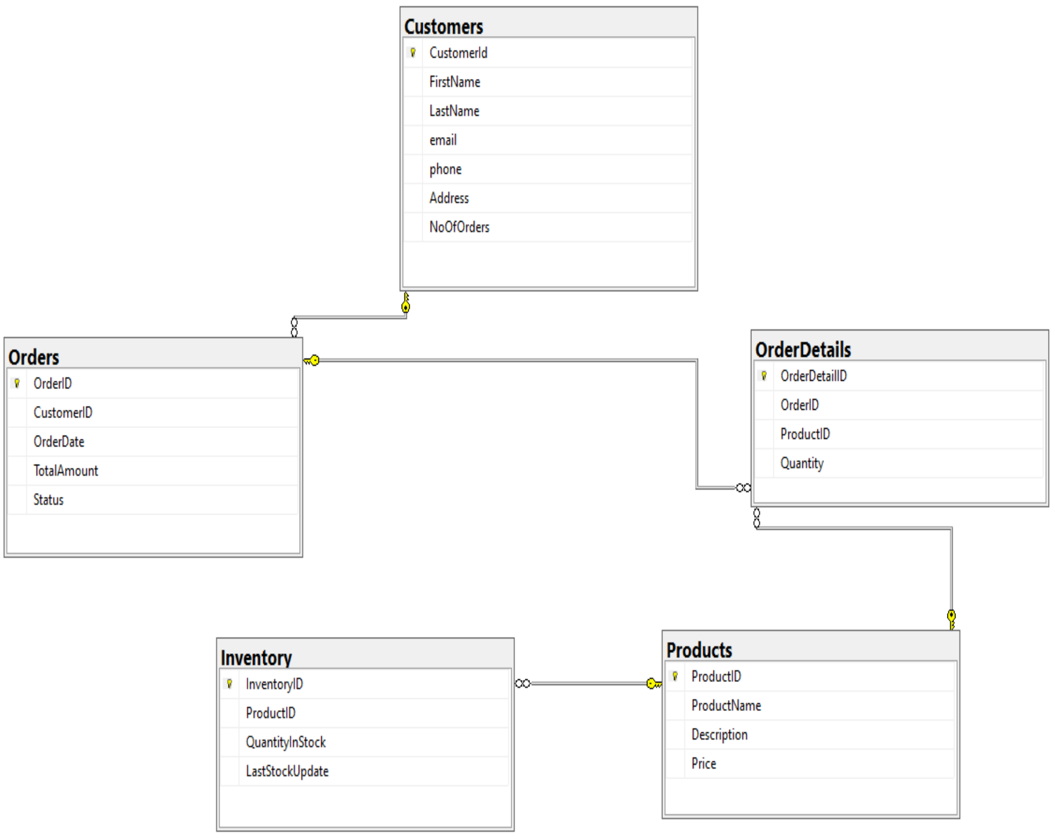
- InventoryID (Primary Key, INT)

- ProductID (Foreign Key referencing Products, INT)

- QuantityInStock (INT)

- LastStockUpdate (DATE)

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



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

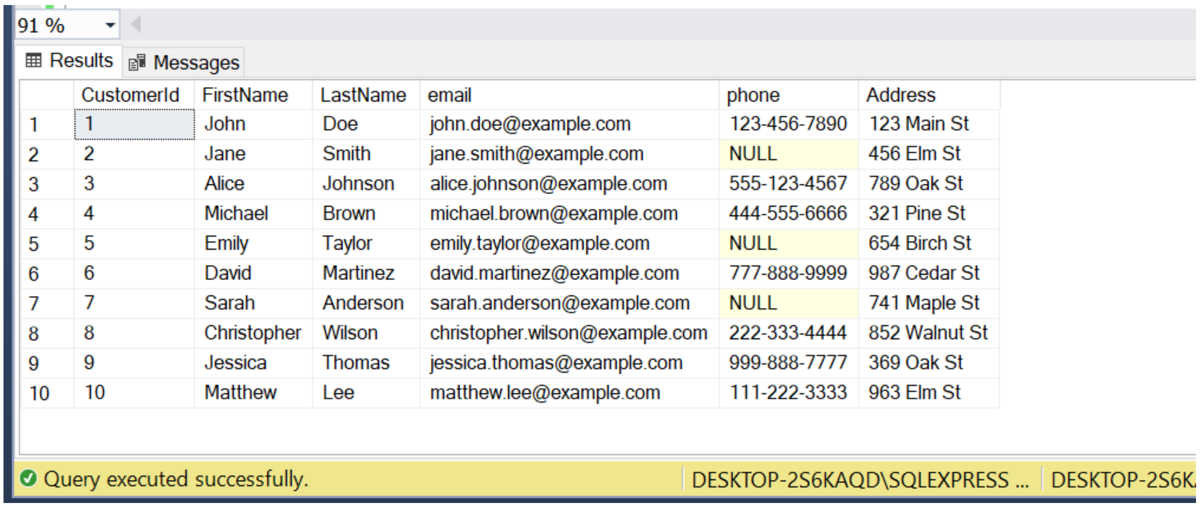
5. Insert at least 10 sample records into each of the following tables.

a. Customers

```
use TechShop;

create table Customers(

CustomerId int Identity constraint C_PK Primary Key,
FirstName varchar(45) not null,
LastName varchar(40),
email varchar(65) not null,
phone varchar(20),
Address varchar(80) not null);
INSERT INTO Customers (FirstName, LastName, Email, Phone, Address)
VALUES
('John', 'Doe', 'john.doe@example.com', '123-456-7890', '123 Main St'),
('Jane', 'Smith', 'jane.smith@example.com', NULL, '456 Elm St'),
('Alice', 'Johnson', 'alice.johnson@example.com', '555-123-4567', '789 Oak St'),
('Michael', 'Brown', 'michael.brown@example.com', '444-555-6666', '321 Pine St'),
('Emily', 'Taylor', 'emily.taylor@example.com', NULL, '654 Birch St'),
('David', 'Martinez', 'david.martinez@example.com', '777-888-9999', '987 Cedar St'),
('Sarah', 'Anderson', 'sarah.anderson@example.com', NULL, '741 Maple St'),
('Christopher', 'Wilson', 'christopher.wilson@example.com', '222-333-4444', '852 Walnut St'),
('Jessica', 'Thomas', 'jessica.thomas@example.com', '999-888-7777', '369 Oak St'),
('Matthew', 'Lee', 'matthew.lee@example.com', '111-222-3333', '963 Elm St');
select * from Customers;
```


91 %

Results | Messages

	CustomerId	FirstName	LastName	email	phone	Address
1	1	John	Doe	john.doe@example.com	123-456-7890	123 Main St
2	2	Jane	Smith	jane.smith@example.com	NULL	456 Elm St
3	3	Alice	Johnson	alice.johnson@example.com	555-123-4567	789 Oak St
4	4	Michael	Brown	michael.brown@example.com	444-555-6666	321 Pine St
5	5	Emily	Taylor	emily.taylor@example.com	NULL	654 Birch St
6	6	David	Martinez	david.martinez@example.com	777-888-9999	987 Cedar St
7	7	Sarah	Anderson	sarah.anderson@example.com	NULL	741 Maple St
8	8	Christopher	Wilson	christopher.wilson@example.com	222-333-4444	852 Walnut St
9	9	Jessica	Thomas	jessica.thomas@example.com	999-888-7777	369 Oak St
10	10	Matthew	Lee	matthew.lee@example.com	111-222-3333	963 Elm St

Query executed successfully. | DESKTOP-2S6KAQD\SQLEXPRESS ... | DESKTOP-2S6K

b. Products

CREATE TABLE Products (

ProductID INT PRIMARY KEY,

ProductName VARCHAR(100),

Description VARCHAR(MAX),

Price DECIMAL(10, 2)

);

INSERT INTO Products (ProductID, ProductName, Description, Price)

VALUES

(1, 'Laptop', '15.6-inch Full HD display, 8GB RAM, 256GB SSD', 799.99),

(2, 'Smartphone', '6.5-inch AMOLED display, 128GB storage, 4000mAh battery', 699.00),

(3, 'Tablet', '10.2-inch Retina display, 64GB storage, Wi-Fi + Cellular', 449.99),

(4, 'Headphones', 'Noise-cancelling, Bluetooth, 30-hour battery life', 149.95),

(5, 'Smartwatch', 'Water-resistant, heart rate monitor, GPS', 199.99),

(6, 'Camera', '24.2MP sensor, 4K video recording, Wi-Fi connectivity', 899.00),

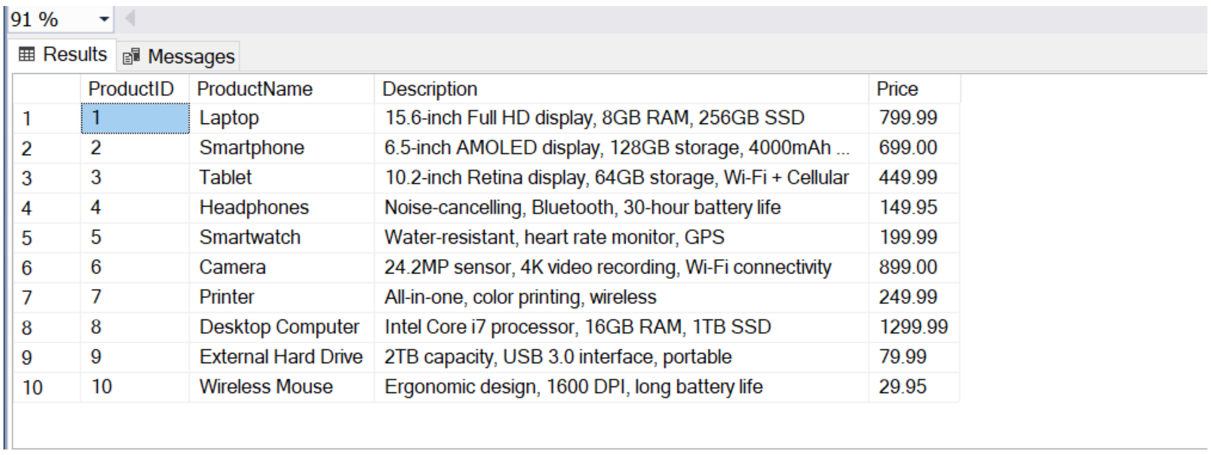
(7, 'Printer', 'All-in-one, color printing, wireless', 249.99),

(8, 'Desktop Computer', 'Intel Core i7 processor, 16GB RAM, 1TB SSD', 1299.99),

(9, 'External Hard Drive', '2TB capacity, USB 3.0 interface, portable', 79.99),

(10, 'Wireless Mouse', 'Ergonomic design, 1600 DPI, long battery life', 29.95);

select * from Products;


91 %

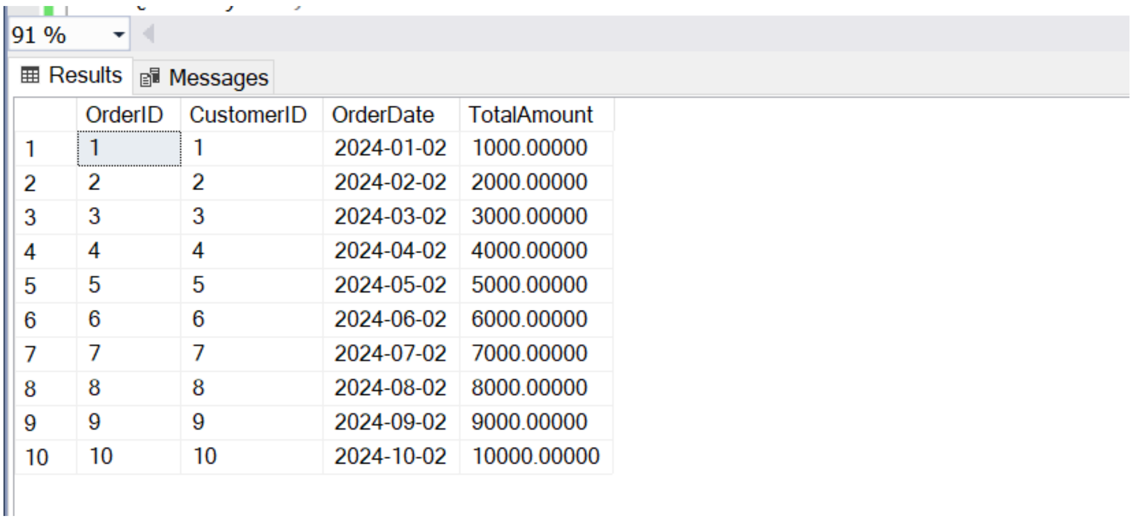
Results | Messages

	ProductID	ProductName	Description	Price
1	1	Laptop	15.6-inch Full HD display, 8GB RAM, 256GB SSD	799.99
2	2	Smartphone	6.5-inch AMOLED display, 128GB storage, 4000mAh ...	699.00
3	3	Tablet	10.2-inch Retina display, 64GB storage, Wi-Fi + Cellular	449.99
4	4	Headphones	Noise-cancelling, Bluetooth, 30-hour battery life	149.95
5	5	Smartwatch	Water-resistant, heart rate monitor, GPS	199.99
6	6	Camera	24.2MP sensor, 4K video recording, Wi-Fi connectivity	899.00
7	7	Printer	All-in-one, color printing, wireless	249.99
8	8	Desktop Computer	Intel Core i7 processor, 16GB RAM, 1TB SSD	1299.99
9	9	External Hard Drive	2TB capacity, USB 3.0 interface, portable	79.99
10	10	Wireless Mouse	Ergonomic design, 1600 DPI, long battery life	29.95

c. Orders

```
CREATE TABLE Orders (
OrderID INT PRIMARY KEY,
CustomerID INT FOREIGN KEY (CustomerID) REFERENCES Customers(CustomerID),
OrderDate DATE,
TotalAmount DECIMAL(10, 5),
);

INSERT INTO Orders (OrderID, CustomerID, OrderDate, TotalAmount)
VALUES
(1, 1, '01-02-2024', 1000),
(2, 2, '02-02-2024', 2000),
(3, 3, '03-02-2024', 3000),
(4, 4, '04-02-2024', 4000),
(5, 5, '05-02-2024', 5000),
(6, 6, '06-02-2024', 6000),
(7, 7, '07-02-2024', 7000),
(8, 8, '08-02-2024', 8000),
(9, 9, '09-02-2024', 9000),
(10, 10, '10-02-2024', 10000);
select * from Orders;
```

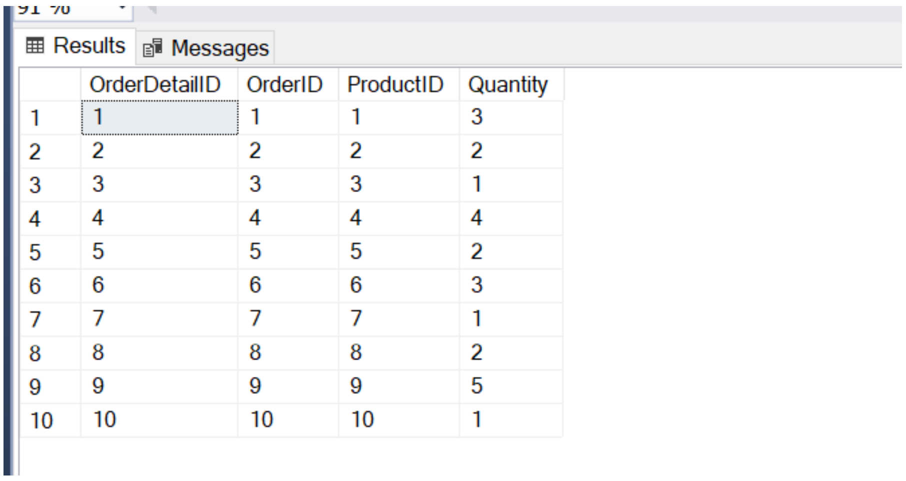

91 %

Results | Messages

	OrderID	CustomerID	OrderDate	TotalAmount
1	1	1	2024-01-02	1000.00000
2	2	2	2024-02-02	2000.00000
3	3	3	2024-03-02	3000.00000
4	4	4	2024-04-02	4000.00000
5	5	5	2024-05-02	5000.00000
6	6	6	2024-06-02	6000.00000
7	7	7	2024-07-02	7000.00000
8	8	8	2024-08-02	8000.00000
9	9	9	2024-09-02	9000.00000
10	10	10	2024-10-02	10000.00000

d. OrderDetails

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

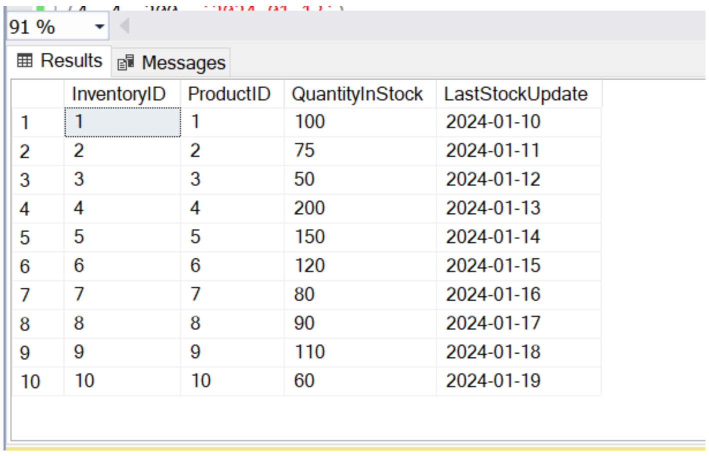

Results | Messages

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

e. Inventory

```
CREATE TABLE Inventory (
InventoryID INT PRIMARY KEY,
ProductID INT,
QuantityInStock INT,
LastStockUpdate DATE,
FOREIGN KEY (ProductID) REFERENCES Products(ProductID)
);

INSERT INTO Inventory (InventoryID, ProductID, QuantityInStock, LastStockUpdate)
VALUES
(1, 1, 100, '2024-01-10'),
(2, 2, 75, '2024-01-11'),
(3, 3, 50, '2024-01-12'),
(4, 4, 200, '2024-01-13'),
(5, 5, 150, '2024-01-14'),
(6, 6, 120, '2024-01-15'),
(7, 7, 80, '2024-01-16'),
(8, 8, 90, '2024-01-17'),
(9, 9, 110, '2024-01-18'),
(10, 10, 60, '2024-01-19');
select * from Inventory;
```



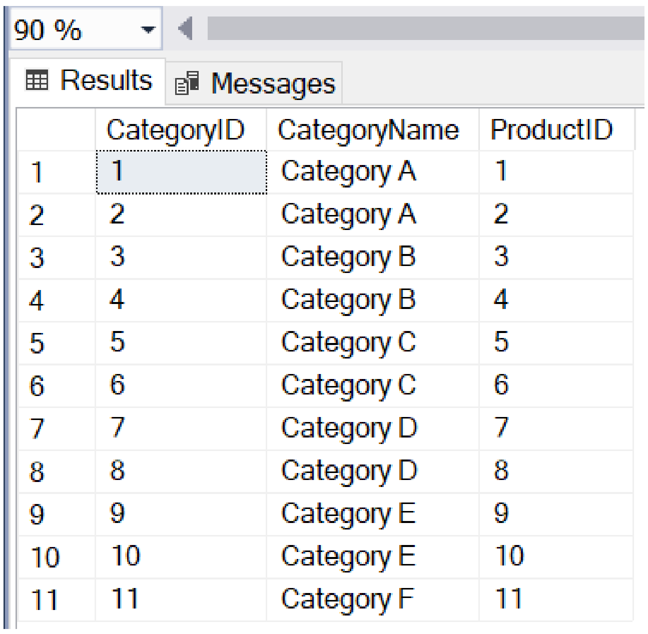
91 %

Results | Messages

	InventoryID	ProductID	QuantityInStock	LastStockUpdate
1	1	1	100	2024-01-10
2	2	2	75	2024-01-11
3	3	3	50	2024-01-12
4	4	4	200	2024-01-13
5	5	5	150	2024-01-14
6	6	6	120	2024-01-15
7	7	7	80	2024-01-16
8	8	8	90	2024-01-17
9	9	9	110	2024-01-18
10	10	10	60	2024-01-19

Categories Table

```
CREATE TABLE Categories (
CategoryID INT PRIMARY KEY,
CategoryName VARCHAR(255),
ProductID INT,
FOREIGN KEY (ProductID) REFERENCES Products(ProductID)
);
INSERT INTO Categories (CategoryID, CategoryName, ProductID)
VALUES
(1, 'Category A', 1),
(2, 'Category A', 2),
(3, 'Category B', 3),
(4, 'Category B', 4),
(5, 'Category C', 5),
(6, 'Category C', 6),
(7, 'Category D', 7),
(8, 'Category D', 8),
(9, 'Category E', 9),
(10, 'Category E', 10),
(11, 'Category F', 11);
select * from Categories;
```

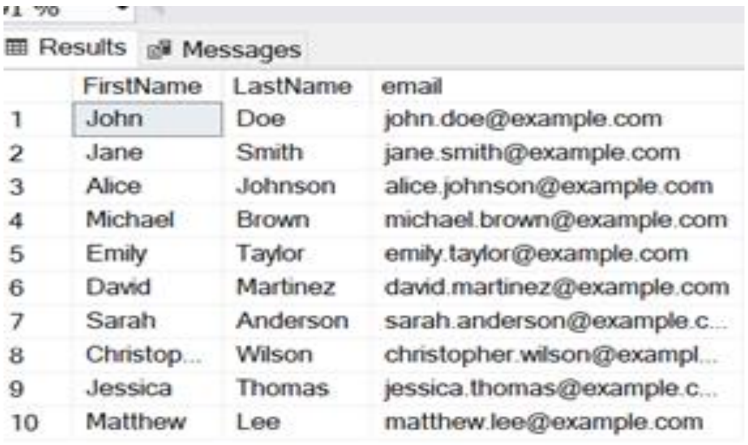


	CategoryID	CategoryName	ProductID
1	1	Category A	1
2	2	Category A	2
3	3	Category B	3
4	4	Category B	4
5	5	Category C	5
6	6	Category C	6
7	7	Category D	7
8	8	Category D	8
9	9	Category E	9
10	10	Category E	10
11	11	Category F	11

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

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

select FirstName,LastName,email from customers;



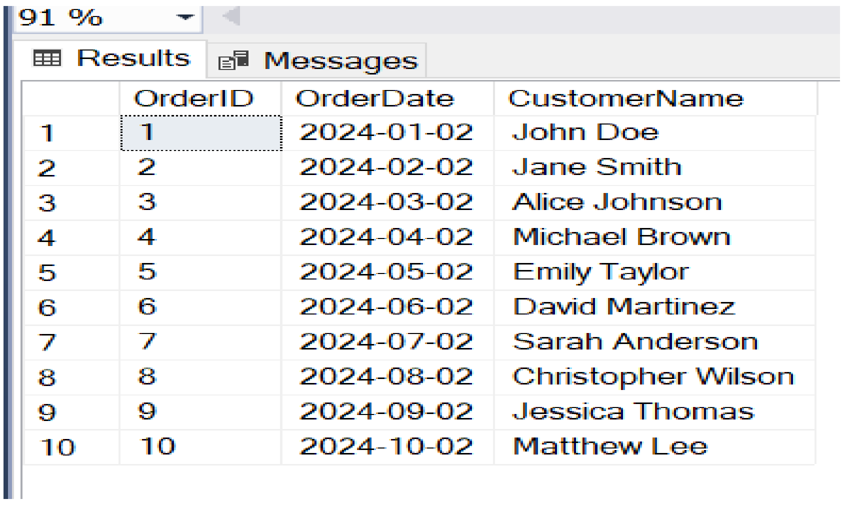
	FirstName	LastName	email
1	John	Doe	john.doe@example.com
2	Jane	Smith	jane.smith@example.com
3	Alice	Johnson	alice.johnson@example.com
4	Michael	Brown	michael.brown@example.com
5	Emily	Taylor	emily.taylor@example.com
6	David	Martinez	david.martinez@example.com
7	Sarah	Anderson	sarah.anderson@example.c...
8	Christop...	Wilson	christopher.wilson@exampl...
9	Jessica	Thomas	jessica.thomas@example.c...
10	Matthew	Lee	matthew.lee@example.com

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

SELECT o.OrderID, o.OrderDate, CONCAT(c.FirstName, ' ', c.LastName) AS CustomerName

FROM Orders o

JOIN Customers c ON o.CustomerID = c.CustomerID;

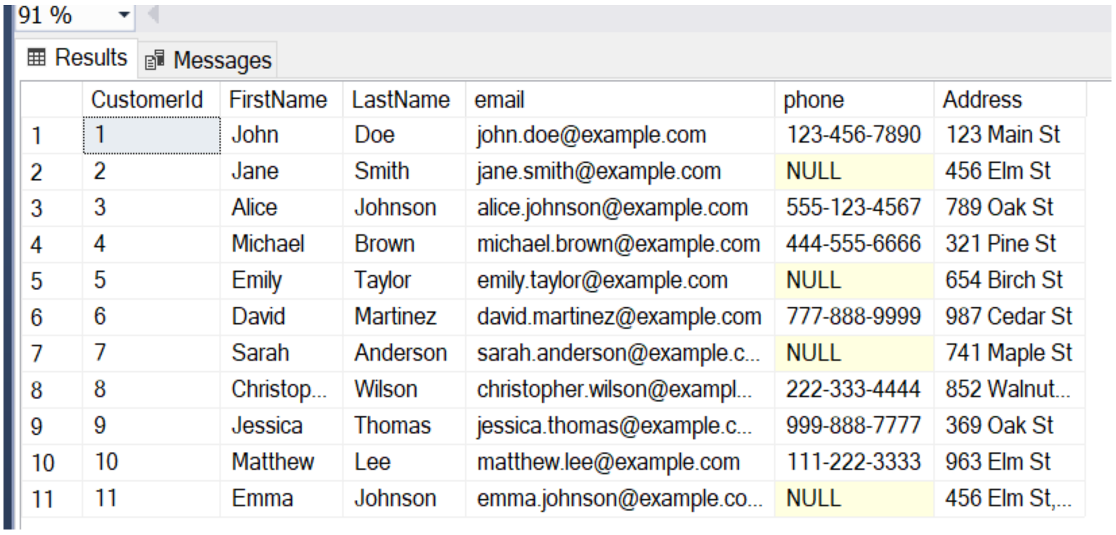

91 %

Results | Messages

	OrderID	OrderDate	CustomerName
1	1	2024-01-02	John Doe
2	2	2024-02-02	Jane Smith
3	3	2024-03-02	Alice Johnson
4	4	2024-04-02	Michael Brown
5	5	2024-05-02	Emily Taylor
6	6	2024-06-02	David Martinez
7	7	2024-07-02	Sarah Anderson
8	8	2024-08-02	Christopher Wilson
9	9	2024-09-02	Jessica Thomas
10	10	2024-10-02	Matthew Lee

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

INSERT INTO Customers(FirstName, LastName, email, Address)
VALUES ('Emma', 'Johnson', 'emma.johnson@example.com', '456 Elm St, Springfield, USA');

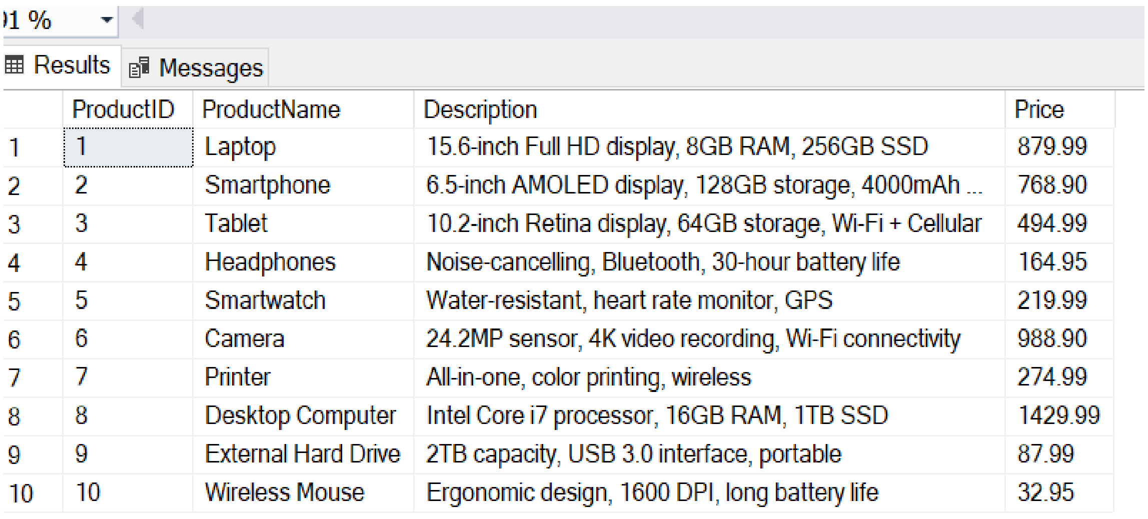

91 %

Results | Messages

	CustomerId	FirstName	LastName	email	phone	Address
1	1	John	Doe	john.doe@example.com	123-456-7890	123 Main St
2	2	Jane	Smith	jane.smith@example.com	NULL	456 Elm St
3	3	Alice	Johnson	alice.johnson@example.com	555-123-4567	789 Oak St
4	4	Michael	Brown	michael.brown@example.com	444-555-6666	321 Pine St
5	5	Emily	Taylor	emily.taylor@example.com	NULL	654 Birch St
6	6	David	Martinez	david.martinez@example.com	777-888-9999	987 Cedar St
7	7	Sarah	Anderson	sarah.anderson@example.c...	NULL	741 Maple St
8	8	Christop...	Wilson	christopher.wilson@exampl...	222-333-4444	852 Walnut...
9	9	Jessica	Thomas	jessica.thomas@example.c...	999-888-7777	369 Oak St
10	10	Matthew	Lee	matthew.lee@example.com	111-222-3333	963 Elm St
11	11	Emma	Johnson	emma.johnson@example.co...	NULL	456 Elm St,...

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

UPDATE Products
SET Price = Price * 1.10;

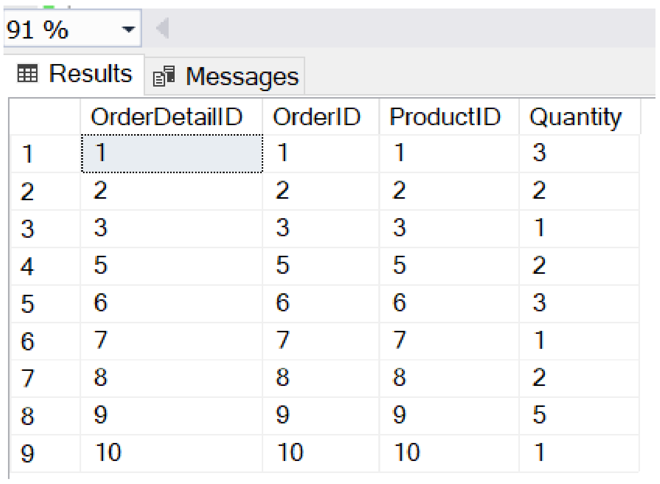

)1 %

Results | Messages

	ProductID	ProductName	Description	Price
1	1	Laptop	15.6-inch Full HD display, 8GB RAM, 256GB SSD	879.99
2	2	Smartphone	6.5-inch AMOLED display, 128GB storage, 4000mAh ...	768.90
3	3	Tablet	10.2-inch Retina display, 64GB storage, Wi-Fi + Cellular	494.99
4	4	Headphones	Noise-cancelling, Bluetooth, 30-hour battery life	164.95
5	5	Smartwatch	Water-resistant, heart rate monitor, GPS	219.99
6	6	Camera	24.2MP sensor, 4K video recording, Wi-Fi connectivity	988.90
7	7	Printer	All-in-one, color printing, wireless	274.99
8	8	Desktop Computer	Intel Core i7 processor, 16GB RAM, 1TB SSD	1429.99
9	9	External Hard Drive	2TB capacity, USB 3.0 interface, portable	87.99
10	10	Wireless Mouse	Ergonomic design, 1600 DPI, long battery life	32.95

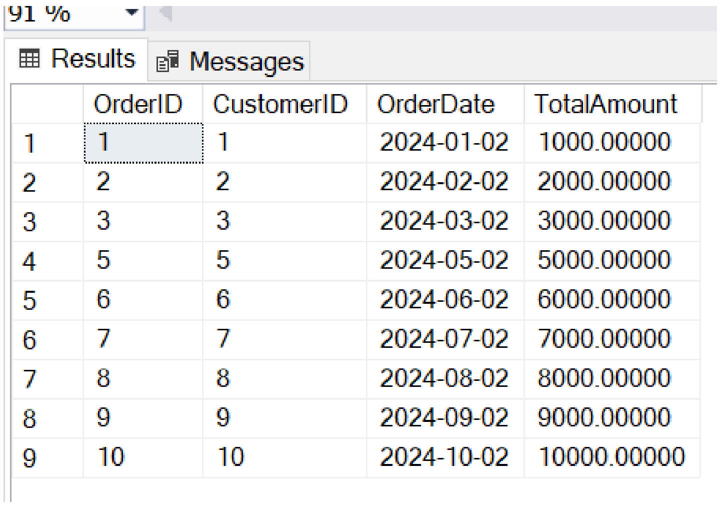
5. Write an SQL query to delete a specific order and its associated order details from the "Orders" and "OrderDetails" tables. Allow users to input the order ID as a parameter.

```
DECLARE @OrderID INT= 4;
DELETE FROM OrderDetails
WHERE OrderID =@OrderID ;
DELETE FROM Orders
WHERE OrderID = @OrderID ;
```


91 %

Results | Messages

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

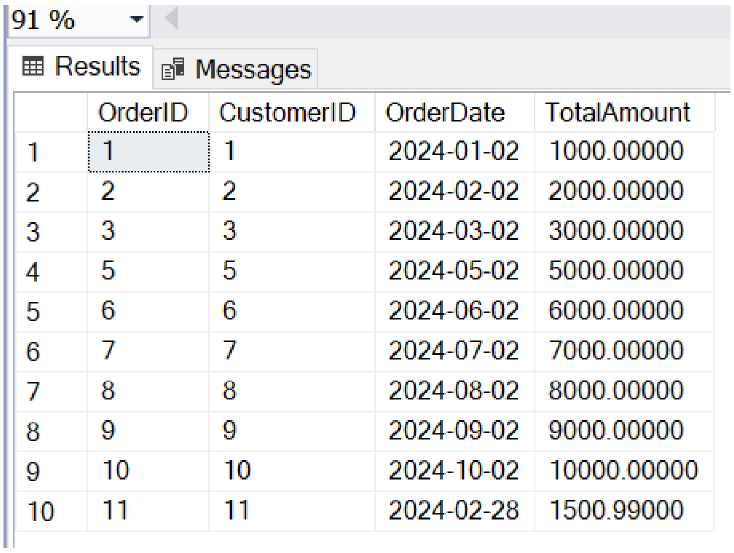

91 %

Results | Messages

	OrderID	CustomerID	OrderDate	TotalAmount
1	1	1	2024-01-02	1000.00000
2	2	2	2024-02-02	2000.00000
3	3	3	2024-03-02	3000.00000
4	5	5	2024-05-02	5000.00000
5	6	6	2024-06-02	6000.00000
6	7	7	2024-07-02	7000.00000
7	8	8	2024-08-02	8000.00000
8	9	9	2024-09-02	9000.00000
9	10	10	2024-10-02	10000.00000

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

INSERT INTO Orders (CustomerID, OrderID, OrderDate, TotalAmount)
VALUES (11, 11, '2024-02-28', 1500.99);


91 %

Results | Messages

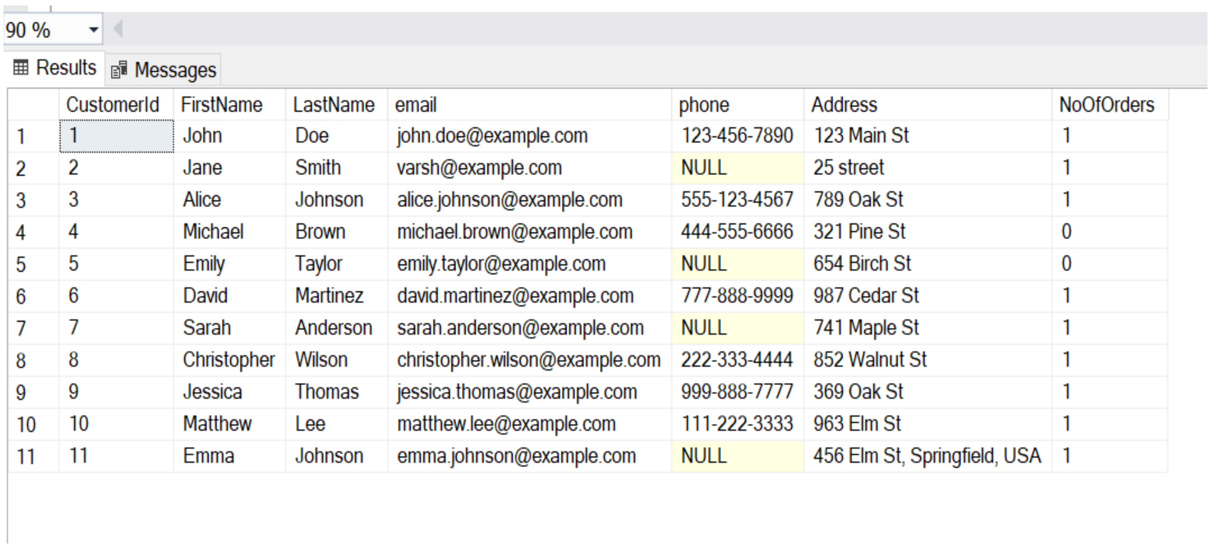
	OrderID	CustomerID	OrderDate	TotalAmount
1	1	1	2024-01-02	1000.00000
2	2	2	2024-02-02	2000.00000
3	3	3	2024-03-02	3000.00000
4	5	5	2024-05-02	5000.00000
5	6	6	2024-06-02	6000.00000
6	7	7	2024-07-02	7000.00000
7	8	8	2024-08-02	8000.00000
8	9	9	2024-09-02	9000.00000
9	10	10	2024-10-02	10000.00000
10	11	11	2024-02-28	1500.99000

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.

DECLARE @CustomerIDToUpdate int=2;
DECLARE @NewEmail VARCHAR(65)='varsh@example.com';
DECLARE @NewAddress VARCHAR(80)='25 street';
UPDATE Customers
SET email = @NewEmail,

Address = @NewAddress

WHERE Customerid = @CustomerIDToUpdate;


90 %

Results | Messages

	CustomerId	FirstName	LastName	email	phone	Address	NoOfOrders
1	1	John	Doe	john.doe@example.com	123-456-7890	123 Main St	1
2	2	Jane	Smith	varsh@example.com	NULL	25 street	1
3	3	Alice	Johnson	alice.johnson@example.com	555-123-4567	789 Oak St	1
4	4	Michael	Brown	michael.brown@example.com	444-555-6666	321 Pine St	0
5	5	Emily	Taylor	emily.taylor@example.com	NULL	654 Birch St	0
6	6	David	Martinez	david.martinez@example.com	777-888-9999	987 Cedar St	1
7	7	Sarah	Anderson	sarah.anderson@example.com	NULL	741 Maple St	1
8	8	Christopher	Wilson	christopher.wilson@example.com	222-333-4444	852 Walnut St	1
9	9	Jessica	Thomas	jessica.thomas@example.com	999-888-7777	369 Oak St	1
10	10	Matthew	Lee	matthew.lee@example.com	111-222-3333	963 Elm St	1
11	11	Emma	Johnson	emma.johnson@example.com	NULL	456 Elm St, Springfield, USA	1

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

UPDATE Orders

SET TotalAmount = (

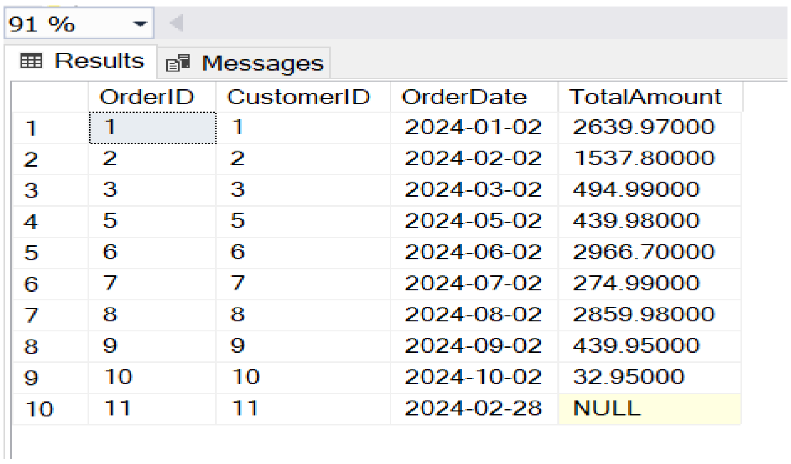
SELECT sum(OD.Quantity * P.Price)

FROM OrderDetails OD

JOIN Products P ON OD.ProductID = P.ProductID

WHERE OD.OrderID = Orders.OrderID

)


91 %

Results | Messages

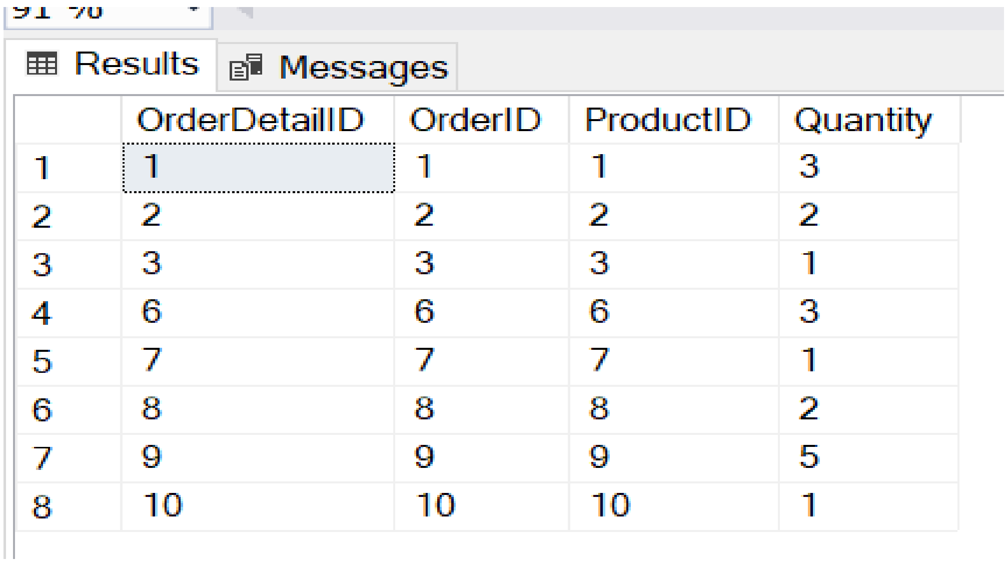
	OrderID	CustomerID	OrderDate	TotalAmount
1	1	1	2024-01-02	2639.97000
2	2	2	2024-02-02	1537.80000
3	3	3	2024-03-02	494.99000
4	5	5	2024-05-02	439.98000
5	6	6	2024-06-02	2966.70000
6	7	7	2024-07-02	274.99000
7	8	8	2024-08-02	2859.98000
8	9	9	2024-09-02	439.95000
9	10	10	2024-10-02	32.95000
10	11	11	2024-02-28	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.

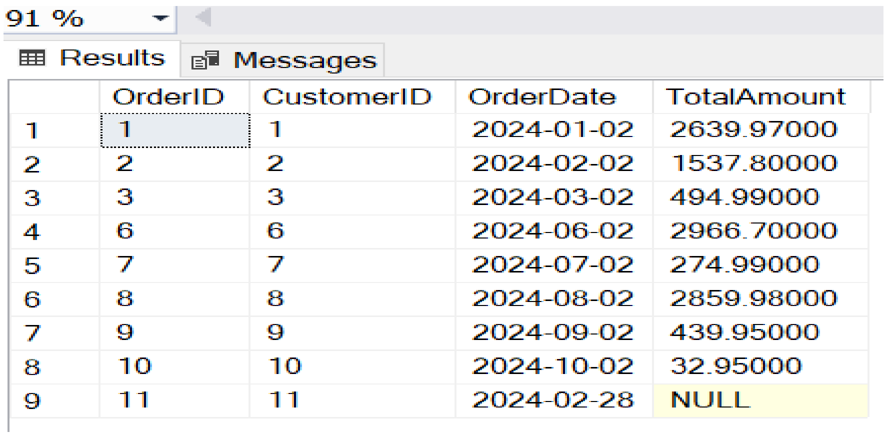
```
DECLARE @CustomerIDToDelete INT;
SET @CustomerIDToDelete = 5;

BEGIN TRANSACTION;
DELETE FROM OrderDetails
WHERE OrderID IN (
SELECT OrderID
FROM Orders
WHERE CustomerID = @CustomerIDToDelete
);
DECLARE @CustomerIDToDelete INT;
SET @CustomerIDToDelete = 5;
DELETE FROM Orders
WHERE CustomerID = @CustomerIDToDelete;

COMMIT TRANSACTION;
```



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

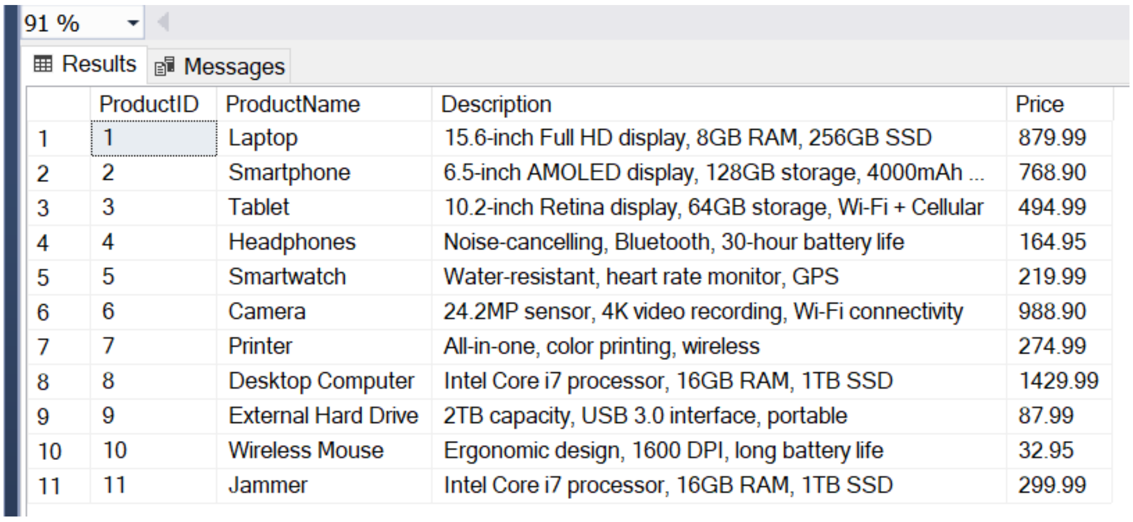


	OrderID	CustomerID	OrderDate	TotalAmount
1	1	1	2024-01-02	2639.97000
2	2	2	2024-02-02	1537.80000
3	3	3	2024-03-02	494.99000
4	6	6	2024-06-02	2966.70000
5	7	7	2024-07-02	274.99000
6	8	8	2024-08-02	2859.98000
7	9	9	2024-09-02	439.95000
8	10	10	2024-10-02	32.95000
9	11	11	2024-02-28	NULL

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.

INSERT INTO Products (ProductID, ProductName, Description, Price)

VALUES ('11','Jammer', 'Intel Core i7 processor, 16GB RAM, 1TB SSD', 299.99);



	ProductID	ProductName	Description	Price
1	1	Laptop	15.6-inch Full HD display, 8GB RAM, 256GB SSD	879.99
2	2	Smartphone	6.5-inch AMOLED display, 128GB storage, 4000mAh ...	768.90
3	3	Tablet	10.2-inch Retina display, 64GB storage, Wi-Fi + Cellular	494.99
4	4	Headphones	Noise-cancelling, Bluetooth, 30-hour battery life	164.95
5	5	Smartwatch	Water-resistant, heart rate monitor, GPS	219.99
6	6	Camera	24.2MP sensor, 4K video recording, Wi-Fi connectivity	988.90
7	7	Printer	All-in-one, color printing, wireless	274.99
8	8	Desktop Computer	Intel Core i7 processor, 16GB RAM, 1TB SSD	1429.99
9	9	External Hard Drive	2TB capacity, USB 3.0 interface, portable	87.99
10	10	Wireless Mouse	Ergonomic design, 1600 DPI, long battery life	32.95
11	11	Jammer	Intel Core i7 processor, 16GB RAM, 1TB SSD	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

ALTER TABLE Orders

ADD Status VARCHAR(50);

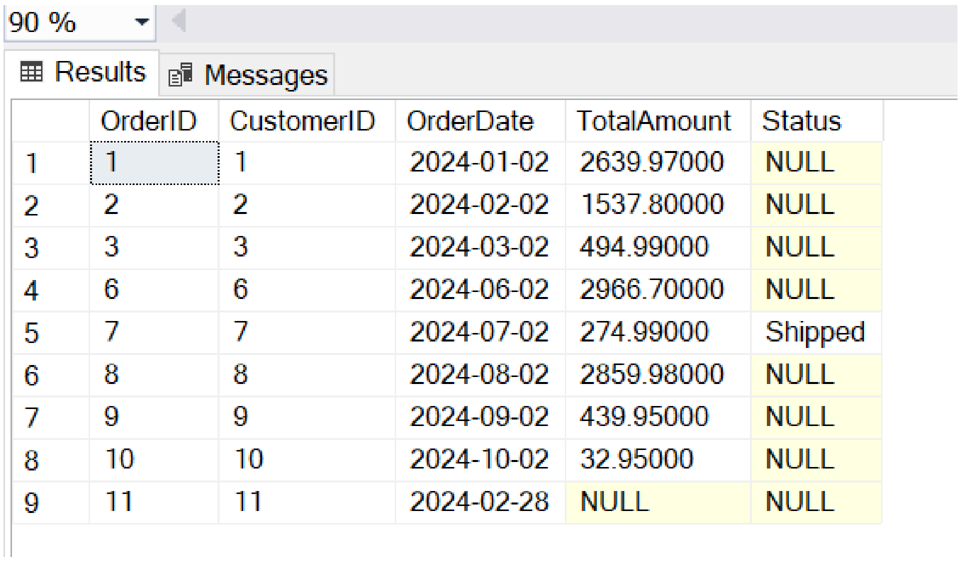
DECLARE @OrderID INT = 7;

DECLARE @NewStatus VARCHAR(50)='Shipped';

UPDATE Orders

SET Status = @NewStatus

WHERE OrderID = @OrderID;

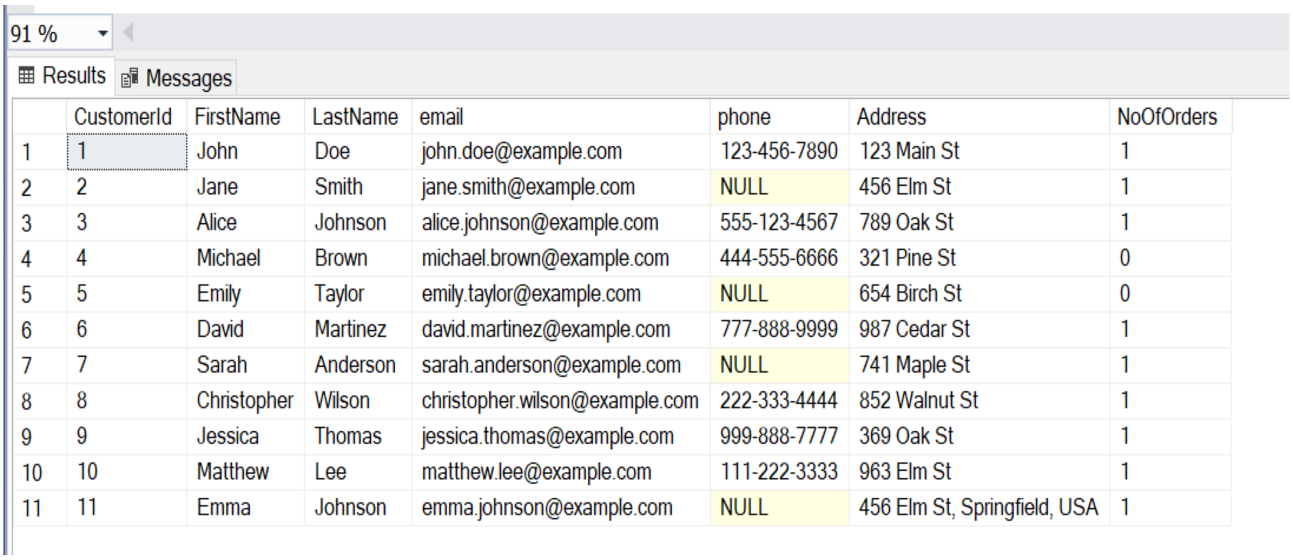

90 %

Results | Messages

	OrderID	CustomerID	OrderDate	TotalAmount	Status
1	1	1	2024-01-02	2639.97000	NULL
2	2	2	2024-02-02	1537.80000	NULL
3	3	3	2024-03-02	494.99000	NULL
4	6	6	2024-06-02	2966.70000	NULL
5	7	7	2024-07-02	274.99000	Shipped
6	8	8	2024-08-02	2859.98000	NULL
7	9	9	2024-09-02	439.95000	NULL
8	10	10	2024-10-02	32.95000	NULL
9	11	11	2024-02-28	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

```
alter table Customers
add NoOfOrders int;
update Customers
set NoOfOrders=(
select count(ORderID)
from Orders
where ORders.CustomerID=Customers.CustomerId);
```


91 %

Results | Messages

	CustomerId	FirstName	LastName	email	phone	Address	NoOfOrders
1	1	John	Doe	john.doe@example.com	123-456-7890	123 Main St	1
2	2	Jane	Smith	jane.smith@example.com	NULL	456 Elm St	1
3	3	Alice	Johnson	alice.johnson@example.com	555-123-4567	789 Oak St	1
4	4	Michael	Brown	michael.brown@example.com	444-555-6666	321 Pine St	0
5	5	Emily	Taylor	emily.taylor@example.com	NULL	654 Birch St	0
6	6	David	Martinez	david.martinez@example.com	777-888-9999	987 Cedar St	1
7	7	Sarah	Anderson	sarah.anderson@example.com	NULL	741 Maple St	1
8	8	Christopher	Wilson	christopher.wilson@example.com	222-333-4444	852 Walnut St	1
9	9	Jessica	Thomas	jessica.thomas@example.com	999-888-7777	369 Oak St	1
10	10	Matthew	Lee	matthew.lee@example.com	111-222-3333	963 Elm St	1
11	11	Emma	Johnson	emma.johnson@example.com	NULL	456 Elm St, Springfield, USA	1

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

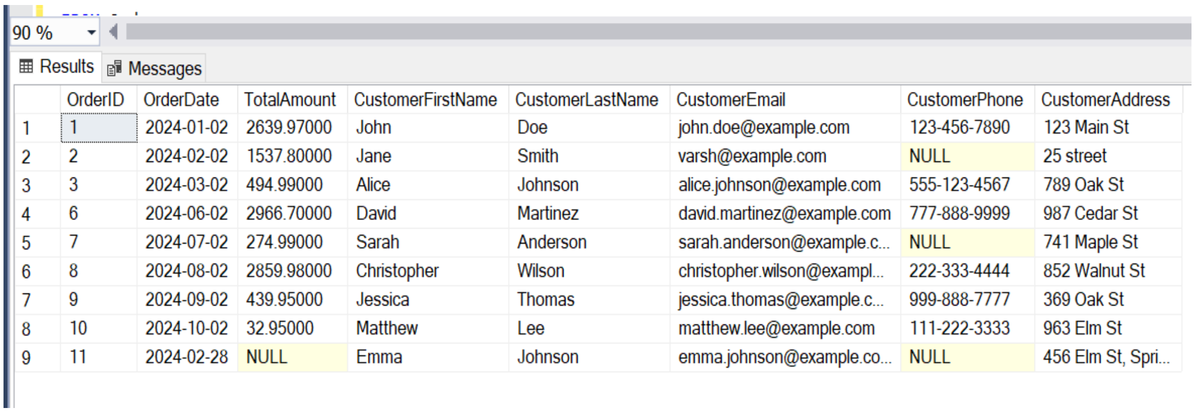
1.Write an SQL query to retrieve a list of all orders along with customer information (e.g., customer name) for each order.

SELECT o.OrderID, o.OrderDate, o.TotalAmount,

c.FirstName AS CustomerFirstName, c.LastName AS CustomerLastName, c.Email AS CustomerEmail, c.Phone AS CustomerPhone, c.Address AS CustomerAddress

FROM Orders o

JOIN Customers c ON o.CustomerID = c.CustomerID;



	OrderID	OrderDate	TotalAmount	CustomerFirstName	CustomerLastName	CustomerEmail	CustomerPhone	CustomerAddress
1	1	2024-01-02	2639.97000	John	Doe	john.doe@example.com	123-456-7890	123 Main St
2	2	2024-02-02	1537.80000	Jane	Smith	varsh@example.com	NULL	25 street
3	3	2024-03-02	494.99000	Alice	Johnson	alice.johnson@example.com	555-123-4567	789 Oak St
4	6	2024-06-02	2966.70000	David	Martinez	david.martinez@example.com	777-888-9999	987 Cedar St
5	7	2024-07-02	274.99000	Sarah	Anderson	sarah.anderson@example.c...	NULL	741 Maple St
6	8	2024-08-02	2859.98000	Christopher	Wilson	christopher.wilson@exampl...	222-333-4444	852 Walnut St
7	9	2024-09-02	439.95000	Jessica	Thomas	jessica.thomas@example.c...	999-888-7777	369 Oak St
8	10	2024-10-02	32.95000	Matthew	Lee	matthew.lee@example.com	111-222-3333	963 Elm St
9	11	2024-02-28	NULL	Emma	Johnson	emma.johnson@example.co...	NULL	456 Elm St, Spri...

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

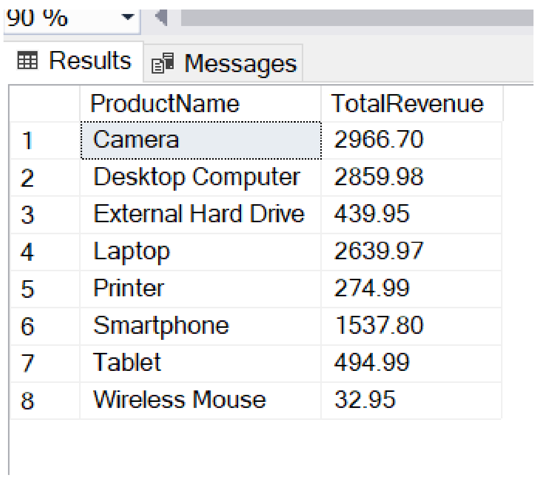
SELECT p.ProductName, SUM(od.Quantity * p.Price) AS TotalRevenue

FROM Products p

JOIN OrderDetails od ON p.ProductID = od.ProductID

JOIN Orders o ON od.OrderID = o.OrderID

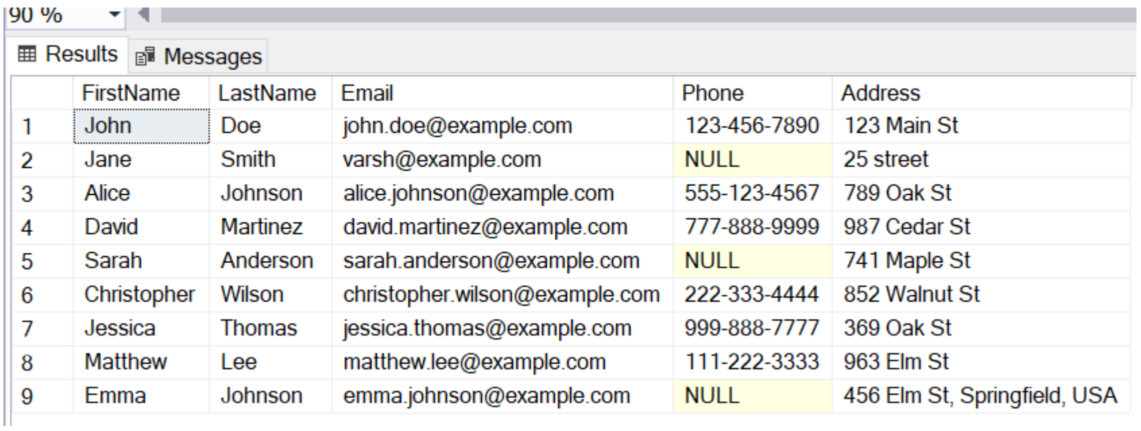
GROUP BY p.ProductName;



	ProductName	TotalRevenue
1	Camera	2966.70
2	Desktop Computer	2859.98
3	External Hard Drive	439.95
4	Laptop	2639.97
5	Printer	274.99
6	Smartphone	1537.80
7	Tablet	494.99
8	Wireless Mouse	32.95

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

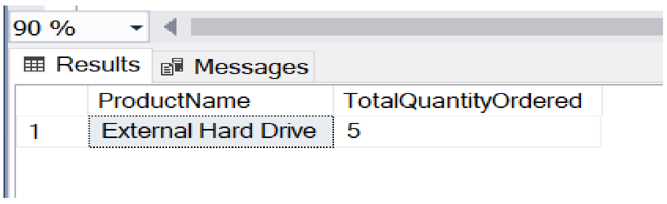
```
SELECT c.FirstName, c.LastName, c.Email, c.Phone, c.Address
FROM Customers c
WHERE EXISTS (
SELECT 1
FROM Orders o
WHERE o.CustomerID = c.CustomerID
);
```



	FirstName	LastName	Email	Phone	Address
1	John	Doe	john.doe@example.com	123-456-7890	123 Main St
2	Jane	Smith	varsh@example.com	NULL	25 street
3	Alice	Johnson	alice.johnson@example.com	555-123-4567	789 Oak St
4	David	Martinez	david.martinez@example.com	777-888-9999	987 Cedar St
5	Sarah	Anderson	sarah.anderson@example.com	NULL	741 Maple St
6	Christopher	Wilson	christopher.wilson@example.com	222-333-4444	852 Walnut St
7	Jessica	Thomas	jessica.thomas@example.com	999-888-7777	369 Oak St
8	Matthew	Lee	matthew.lee@example.com	111-222-3333	963 Elm St
9	Emma	Johnson	emma.johnson@example.com	NULL	456 Elm St, Springfield, USA

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



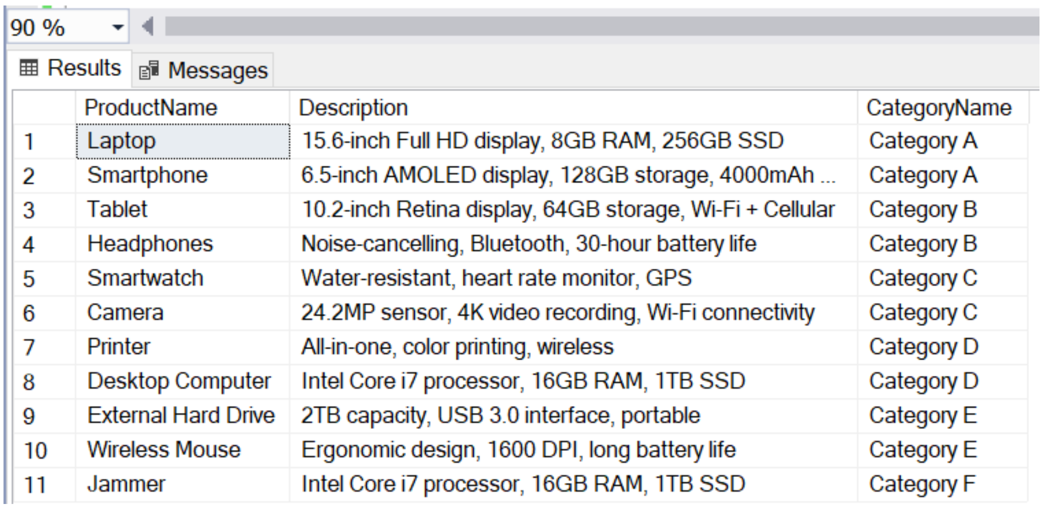
	ProductName	TotalQuantityOrdered
1	External Hard Drive	5

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

SELECT p.ProductName,p.Description,c.CategoryName

FROM Products p

JOIN Categories c ON p.ProductID = c.ProductID



	ProductName	Description	CategoryName
1	Laptop	15.6-inch Full HD display, 8GB RAM, 256GB SSD	Category A
2	Smartphone	6.5-inch AMOLED display, 128GB storage, 4000mAh ...	Category A
3	Tablet	10.2-inch Retina display, 64GB storage, Wi-Fi + Cellular	Category B
4	Headphones	Noise-cancelling, Bluetooth, 30-hour battery life	Category B
5	Smartwatch	Water-resistant, heart rate monitor, GPS	Category C
6	Camera	24.2MP sensor, 4K video recording, Wi-Fi connectivity	Category C
7	Printer	All-in-one, color printing, wireless	Category D
8	Desktop Computer	Intel Core i7 processor, 16GB RAM, 1TB SSD	Category D
9	External Hard Drive	2TB capacity, USB 3.0 interface, portable	Category E
10	Wireless Mouse	Ergonomic design, 1600 DPI, long battery life	Category E
11	Jammer	Intel Core i7 processor, 16GB RAM, 1TB SSD	Category F

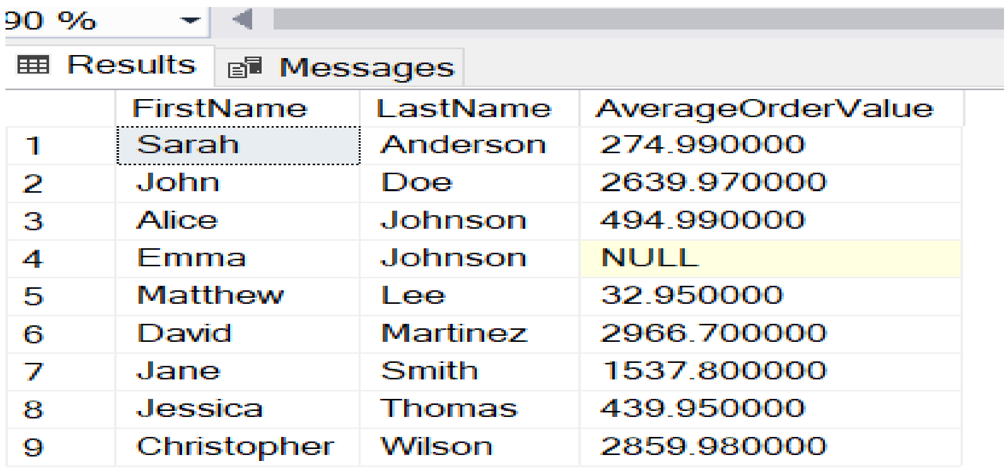
6. Write an SQL query to calculate the average order value for each customer. Include the customer's name and their average order value.

SELECT c.FirstName,c.LastName,AVG(o.TotalAmount) AS AverageOrderValue

FROM Customers c

JOIN Orders o ON c.CustomerID = o.CustomerID

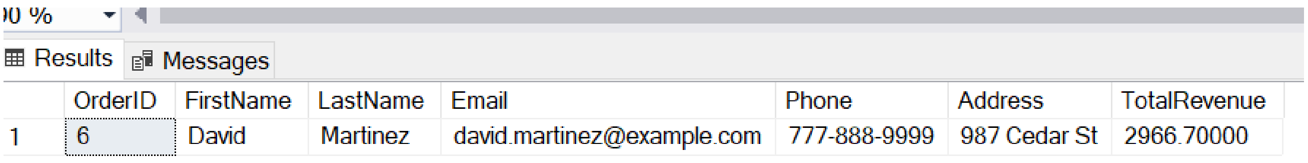
GROUP BY c.FirstName, c.LastName;



	FirstName	LastName	AverageOrderValue
1	Sarah	Anderson	274.990000
2	John	Doe	2639.970000
3	Alice	Johnson	494.990000
4	Emma	Johnson	NULL
5	Matthew	Lee	32.950000
6	David	Martinez	2966.700000
7	Jane	Smith	1537.800000
8	Jessica	Thomas	439.950000
9	Christopher	Wilson	2859.980000

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

```
SELECT TOP 1 o.OrderID,c.FirstName,c.LastName,c.Email,c.Phone,c.Address,
MAX(o.TotalAmount) AS TotalRevenue
FROM Orders o
JOIN Customers c ON o.CustomerID = c.CustomerID
GROUP BY o.OrderID,c.FirstName,c.LastName,c.Email,c.Phone,c.Address
ORDER BY TotalRevenue DESC;
```

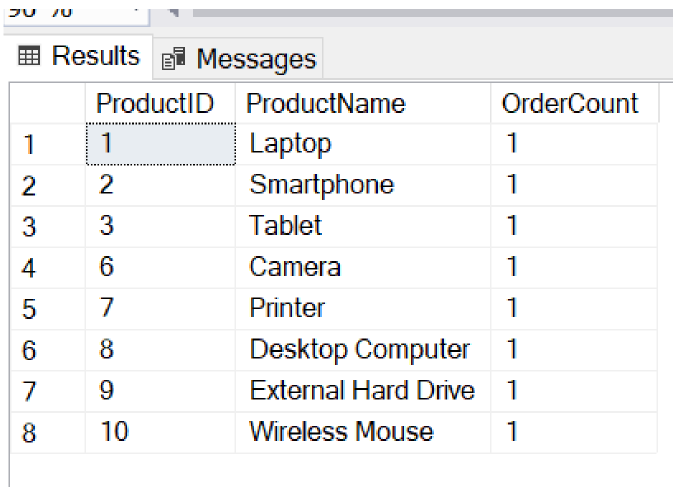


Results | Messages

	OrderID	FirstName	LastName	Email	Phone	Address	TotalRevenue
1	6	David	Martinez	david.martinez@example.com	777-888-9999	987 Cedar St	2966.70000

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

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



Results | Messages

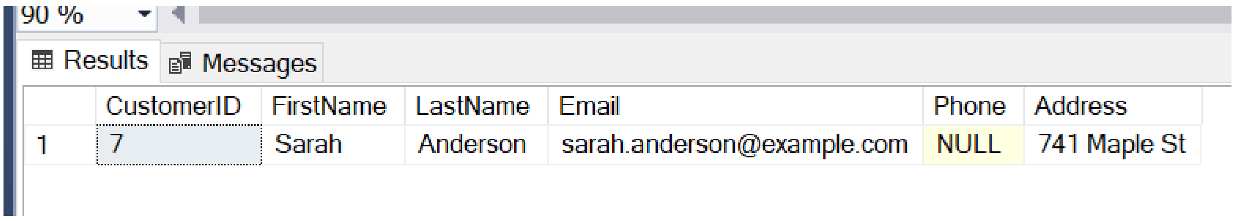
	ProductID	ProductName	OrderCount
1	1	Laptop	1
2	2	Smartphone	1
3	3	Tablet	1
4	6	Camera	1
5	7	Printer	1
6	8	Desktop Computer	1
7	9	External Hard Drive	1
8	10	Wireless Mouse	1

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.

```
DECLARE @ProductName VARCHAR(100);
```

```
SET @ProductName = 'Printer';

SELECT DISTINCT c.CustomerID,c.FirstName,c.LastName,c.Email,c.Phone,c.Address
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;
```



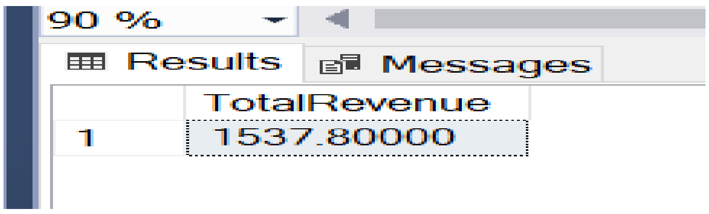
```
   CustomerID  FirstName  LastName  Email                       Phone  Address
1  7           Sarah      Anderson  sarah.anderson@example.com  NULL   741 Maple St
```

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.

```
DECLARE @StartDate DATE;
DECLARE @EndDate DATE;

SET @StartDate = '2024-02-01';
SET @EndDate = '2024-02-03';

SELECT SUM(o.TotalAmount) AS TotalRevenue
FROM Orders o
WHERE o.OrderDate >= @StartDate
AND o.OrderDate <= @EndDate;
```



```
   TotalRevenue
1  1537.80000
```

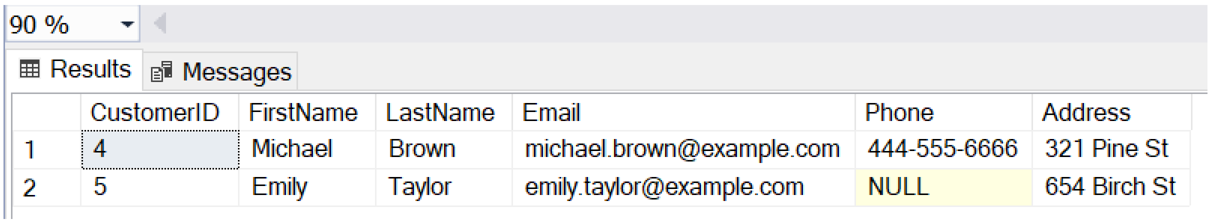
Task 4. Subquery and its type:

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

SELECT CustomerID,FirstName,LastName,Email,Phone,Address

FROM Customers

WHERE CustomerID NOT IN (SELECT CustomerID FROM Orders);

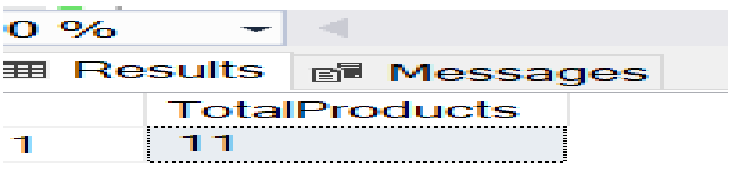

90 %
Results
Messages

	CustomerID	FirstName	LastName	Email	Phone	Address
1	4	Michael	Brown	michael.brown@example.com	444-555-6666	321 Pine St
2	5	Emily	Taylor	emily.taylor@example.com	NULL	654 Birch St

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

SELECT COUNT(*) AS TotalProducts

FROM Products;

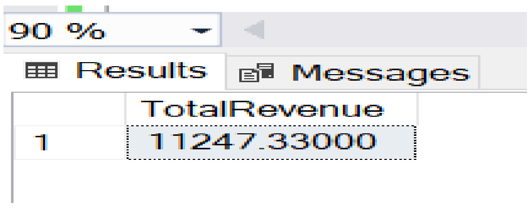

0 %
Results
Messages

	TotalProducts
1	11

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

SELECT SUM(TotalAmount) AS TotalRevenue

FROM Orders;


90 %
Results
Messages

	TotalRevenue
1	11247.33000

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.

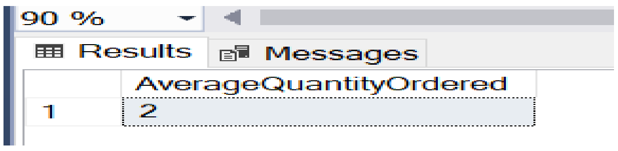
DECLARE @CategoryName VARCHAR(100);

SET @CategoryName = 'Category A';

SELECT AVG(od.Quantity) AS AverageQuantityOrdered

FROM OrderDetails od

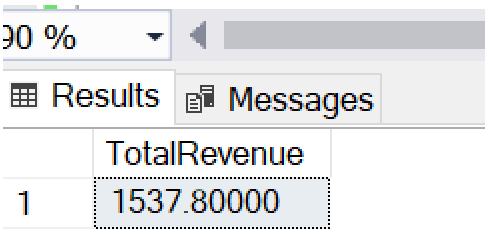
```
WHERE od.ProductID IN (SELECT p.ProductID
FROM Products p
JOIN Categories c ON p.ProductID = c.ProductID
WHERE c.CategoryName = @CategoryName);
```



```
   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.

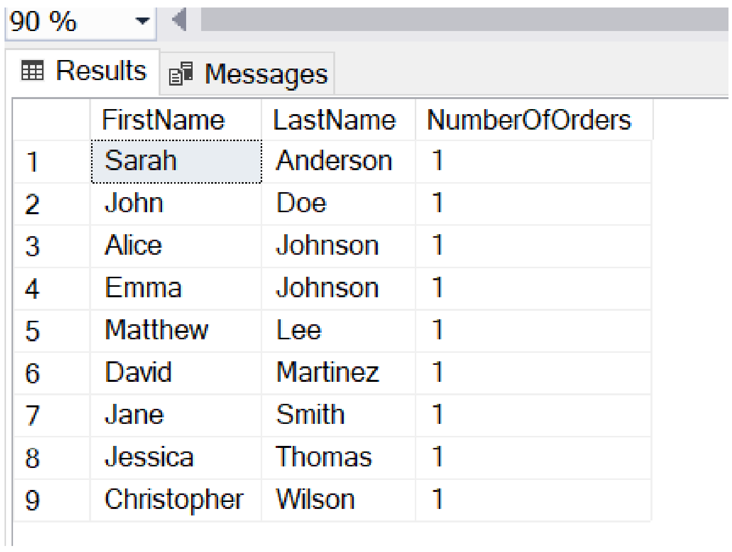
```
DECLARE @CustomerID INT;
SET @CustomerID = 2;
SELECT SUM(TotalAmount) AS TotalRevenue
FROM Orders
WHERE CustomerID = @CustomerID;
```



```
   TotalRevenue
1  1537.80000
```

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.

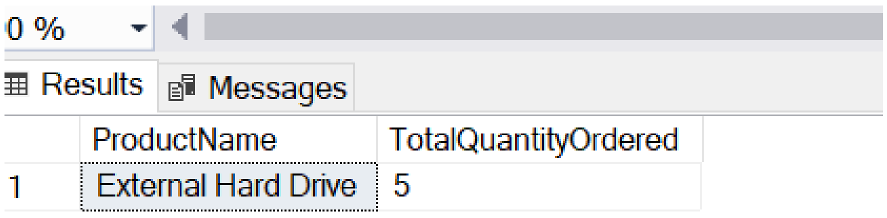
```
SELECT c.FirstName,c.LastName,
COUNT(o.OrderID) AS NumberOfOrders
FROM Customers c
JOIN Orders o ON c.CustomerID = o.CustomerID
GROUP BY c.FirstName,c.LastName
HAVING COUNT(o.OrderID) = (SELECT MAX(OrdersCount)
FROM (SELECT COUNT(OrderID) AS OrdersCount
FROM Orders GROUP BY CustomerID) AS OrderCounts);
```



	FirstName	LastName	NumberOfOrders
1	Sarah	Anderson	1
2	John	Doe	1
3	Alice	Johnson	1
4	Emma	Johnson	1
5	Matthew	Lee	1
6	David	Martinez	1
7	Jane	Smith	1
8	Jessica	Thomas	1
9	Christopher	Wilson	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 TOP 1 p.ProductName,
SUM(od.Quantity) AS TotalQuantityOrdered
FROM OrderDetails od
JOIN Products p ON od.ProductID = p.ProductID
GROUP BY p.ProductName
ORDER BY TotalQuantityOrdered DESC;
```



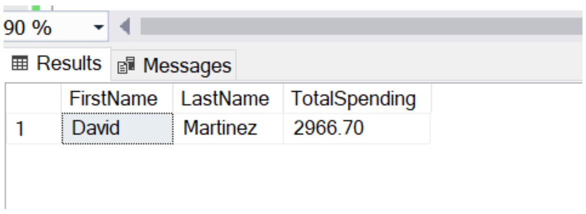
	ProductName	TotalQuantityOrdered
1	External Hard Drive	5

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 TOP 1 c.FirstName,c.LastName,
SUM(od.Quantity * p.Price) AS TotalSpending
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
```

GROUP BY c.FirstName,c.LastName

ORDER BY TotalSpending DESC;

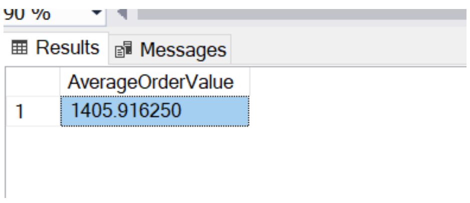


	FirstName	LastName	TotalSpending
1	David	Martinez	2966.70

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

SELECT AVG(TotalAmount) AS AverageOrderValue

FROM Orders;



	AverageOrderValue
1	1405.916250

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.

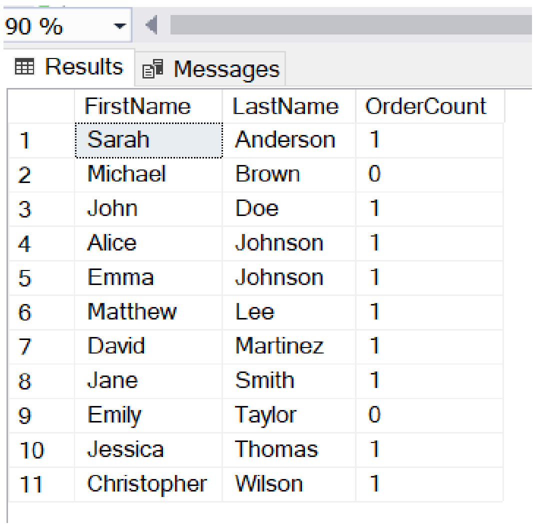
SELECT c.FirstName,c.LastName,

COUNT(o.OrderID) AS OrderCount

FROM Customers c

LEFT JOIN Orders o ON c.CustomerID = o.CustomerID

GROUP BY c.FirstName,c.LastName;



	FirstName	LastName	OrderCount
1	Sarah	Anderson	1
2	Michael	Brown	0
3	John	Doe	1
4	Alice	Johnson	1
5	Emma	Johnson	1
6	Matthew	Lee	1
7	David	Martinez	1
8	Jane	Smith	1
9	Emily	Taylor	0
10	Jessica	Thomas	1
11	Christopher	Wilson	1