ASSIGNMENT

TECH SHOP, AN ELECTRONIC GADGETS SHOP

NAME: POORVANSH VYAS (P118)

TOPIC: TECH SHOP, AN ELECTRONIC GADGETS SHOP

TASK -1:

1. Create the database named "TechShop"

create database TECHSHOP;

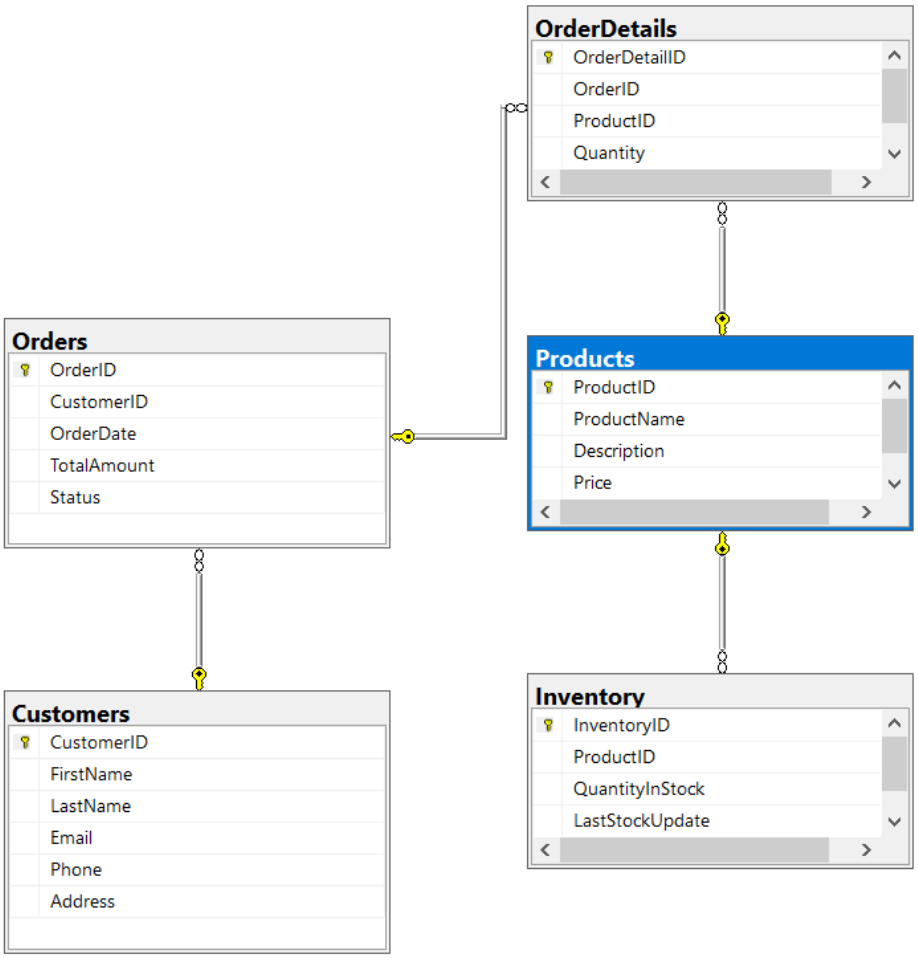
use TECHSHOP;

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

```
create table Customers(
CustomerID int identity Primary Key,
FirstName varchar(20),
LastName  varchar(20),
Email varchar(40),
Phone int,
Address varchar(40)
)
create table Products(
ProductID int identity Primary Key,
ProductName varchar(20),
Description varchar(20),
Price int)

create table Orders(
```

```
OrderID int identity Primary Key,
CustomerID int,
Foreign Key(CustomerID) references Customers(CustomerID),
OrderDate date,
TotalAmount int,
)

create table OrderDetails(
OrderDetailID int identity Primary Key,
OrderID int,
Foreign Key (OrderID) references Orders(OrderID),
ProductID int,
Foreign Key (ProductID) references Products(ProductID),
Quantity int
)
create table Inventory(
InventoryID int identity Primary Key,
ProductID int,
Foreign Key (ProductID) references Products(ProductID),
QuantityInStock int,
LastStockUpdate int
)
```

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



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

- All primary and foreign keys are inserted while creating the table.

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
('John', 'Doe', 'john.doe@example.com', 555-1234, '123 Elm St'),
('Jane', 'Smith', 'jane.smith@example.com', 555-5678, '456 Oak St'),
('Alice', 'Johnson', 'alice.j@example.com', 555-8765, '789 Pine St'),
('Bob', 'Brown', 'bob.brown@example.com', 555-4321, '101 Maple St'),
('Charlie', 'Davis', 'charlie.davis@example.com', 555-9876, '102 Cedar
St'),
('David', 'Wilson', 'david.wilson@example.com', 555-1122, '222 Birch St'),
('Eva', 'Taylor', 'eva.taylor@example.com', 555-3344, '333 Walnut St'),
('Frank', 'Thomas', 'frank.thomas@example.com', 555-5566, '444 Chestnut
St'),
('Grace', 'Moore', 'grace.moore@example.com', 555-7788, '555 Spruce St'),
('Harry', 'Anderson', 'harry.anderson@example.com', 555-9911, '666 Aspen
St');

select * from Customers;

INSERT INTO Products VALUES
('Laptop' , '15-inch display', 799),
('Smartphone' ,  '5G enabled', 599),
('Tablet', '10-inch screen', 299),
('Headphones', 'Noise-canceling', 199),
('Smartwatch', 'Fitness tracker', 149),
('Gaming Console', 'Next-gen gaming', 499),
('Wireless Mouse', 'Bluetooth mouse', 29),
('Mechanical Keyboard', 'RGB backlit', 99),
('Monitor', '27-inch 4K display', 399),
('External SSD', '1TB', 149);

select * from Products;

INSERT INTO Orders values
(1, '2024-09-01', 1500),
(2, '2024-09-02', 2500),
(3, '2024-09-05', 1000),
(4, '2024-09-06', 3000),
(1, '2024-09-07', 2200),
(2, '2024-09-08', 1800),
(5, '2024-09-10', 2700),
(3, '2024-09-11', 3200);

INSERT INTO Orders values
(4, '2024-09-06', 3000);

select * from Orders;

select * from products;
```

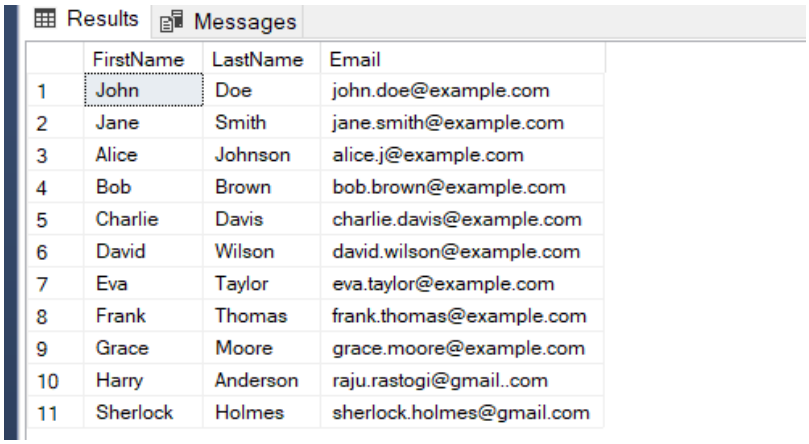
```sql
select * from OrderDetails;

insert into OrderDetails values
(1, 10, 2),
(2, 9, 1),
(3, 12, 5),
(4, 11, 3),
(5, 16, 4),
(6, 15, 6),
(7, 17, 2),
(8, 13, 1),
(7, 14, 8),
(8, 18, 10);

INSERT INTO Inventory (ProductID, QuantityInStock, LastStockUpdate)
VALUES
(9, 50, 20230901),
(10, 100, 20230902),
(13, 200, 20230905),
(14, 30, 20230828),
(15, 75, 20230903),
(16, 120, 20230910),
(17, 60, 20230830),
(18, 90, 20230912),
(11, 150, 20230908),
(12, 10, 20230911);
```

TASK -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;
```

OUTPUT:



Results | Messages

	FirstName	LastName	Email
1	John	Doe	john.doe@example.com
2	Jane	Smith	jane.smith@example.com
3	Alice	Johnson	alice.j@example.com
4	Bob	Brown	bob.brown@example.com
5	Charlie	Davis	charlie.davis@example.com
6	David	Wilson	david.wilson@example.com
7	Eva	Taylor	eva.taylor@example.com
8	Frank	Thomas	frank.thomas@example.com
9	Grace	Moore	grace.moore@example.com
10	Harry	Anderson	raju.rastogi@gmail..com
11	Sherlock	Holmes	sherlock.holmes@gmail.com

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

```
SELECT
    OrderID,
    OrderDate,
    (SELECT FirstName
     FROM Customers
     WHERE Customers.CustomerID = Orders.CustomerID)AS Customer_Name
FROM
    Orders;
```

OUTPUT:

Results Messages

	OrderID	OrderDate	Customer_Name
1	1	2024-09-01	John
2	2	2024-09-02	Jane
3	3	2024-09-05	Alice
4	5	2024-09-07	John
5	6	2024-09-08	Jane
6	7	2024-09-10	Charlie
7	8	2024-09-11	Alice
8	10	2024-09-04	Alice
9	11	2024-09-04	Alice

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 values
('Sherlock','Holmes','sherlock.holmes@gmail.com',NULL,'221B Baker
Street');

select*from Customers;
```

OUTPUT:

Results Messages

	CustomerID	FirstName	LastName	Email	Phone	Address
1	1	John	Doe	john.doe@example.com	-679	123 Elm St
2	2	Jane	Smith	jane.smith@example.com	-5123	456 Oak St
3	3	Alice	Johnson	alice.j@example.com	-8210	789 Pine St
4	4	Bob	Brown	bob.brown@example.com	-3766	101 Maple St
5	5	Charlie	Davis	charlie.davis@example.com	-9321	102 Cedar St
6	6	David	Wilson	david.wilson@example.com	-567	222 Birch St
7	7	Eva	Taylor	eva.taylor@example.com	-2789	333 Walnut St
8	8	Frank	Thomas	frank.thomas@example.com	-5011	444 Chestnut St
9	9	Grace	Moore	grace.moore@example.com	-7233	555 Spruce St
10	10	Harry	Anderson	raju.rastogi@gmail..com	-9356	221b, ST 12345
11	11	Sherlock	Holmes	sherlock.holmes@gmail.com	NULL	221B Baker Street

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

```
select * from Products
update Products
set Price = price*1.1;
select * from Products;
```

OUTPUT:

Results | Messages

ProductID	ProductName	Description	Price
9	Laptop	15-inch display	878
10	Smartphone	5G enabled	658
11	Tablet	10-inch screen	328
12	Headphones	Noise-canceling	218
13	Smartwatch	Fitness tracker	163
14	Gaming Console	Next-gen gaming	548
15	Wireless Mouse	Bluetooth mouse	31
16	Mechanical Keyboard	RGB backlit	108
17	Monitor	27-inch 4K display	438
18	External SSD	1TB	163

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;
select * from OrderDetails;
```

OUTPUT:

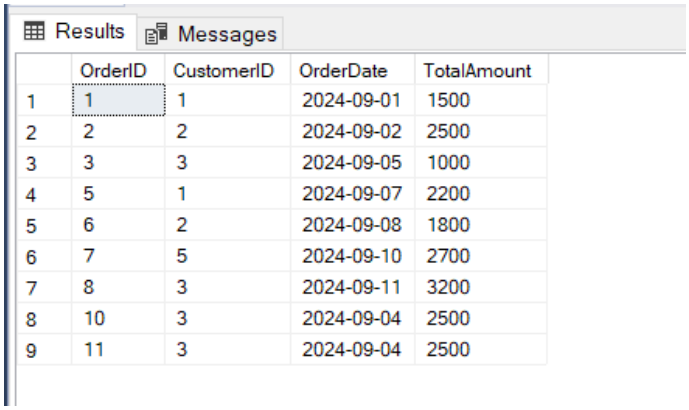
Results | Messages

	OrderDetailID	OrderID	ProductID	Quantity
1	6	1	10	2
2	7	1	10	2
3	118	1	10	2
4	119	2	9	1
5	120	3	12	5
6	122	5	16	4
7	123	6	15	6
8	124	7	17	2
9	125	8	13	1

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 values
(3,'2024-09-04',2500);
select* from orders;
```

OUTPUT:

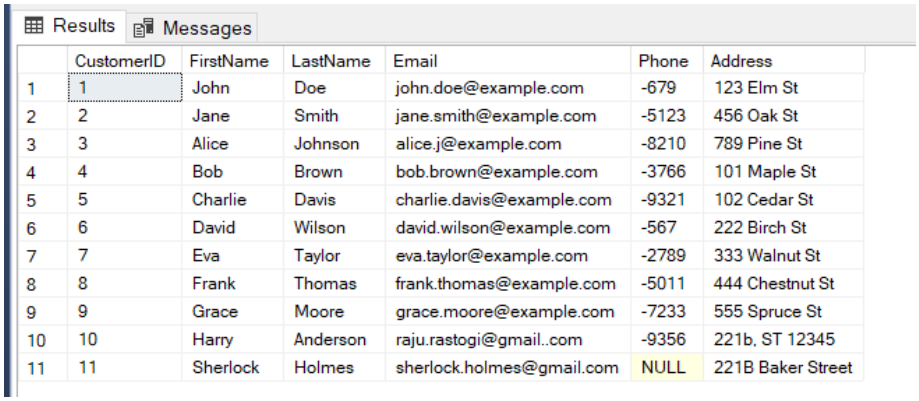


	OrderID	CustomerID	OrderDate	TotalAmount
1	1	1	2024-09-01	1500
2	2	2	2024-09-02	2500
3	3	3	2024-09-05	1000
4	5	1	2024-09-07	2200
5	6	2	2024-09-08	1800
6	7	5	2024-09-10	2700
7	8	3	2024-09-11	3200
8	10	3	2024-09-04	2500
9	11	3	2024-09-04	2500

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.

```
UPDATE Customers
SET
    Email = 'raju.rastogi@gmail..com',
    Address = '221b, ST 12345'
WHERE
    CustomerID = 10;


select*from Customers;
```

OUTPUT:

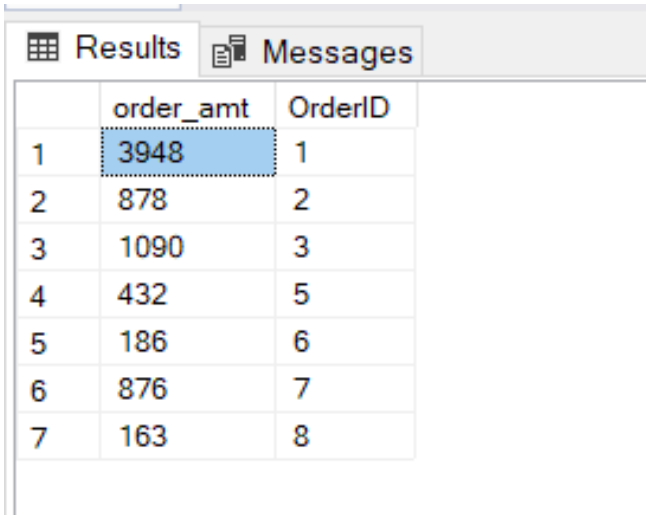


	CustomerID	FirstName	LastName	Email	Phone	Address
1	1	John	Doe	john.doe@example.com	-679	123 Elm St
2	2	Jane	Smith	jane.smith@example.com	-5123	456 Oak St
3	3	Alice	Johnson	alice.j@example.com	-8210	789 Pine St
4	4	Bob	Brown	bob.brown@example.com	-3766	101 Maple St
5	5	Charlie	Davis	charlie.davis@example.com	-9321	102 Cedar St
6	6	David	Wilson	david.wilson@example.com	-567	222 Birch St
7	7	Eva	Taylor	eva.taylor@example.com	-2789	333 Walnut St
8	8	Frank	Thomas	frank.thomas@example.com	-5011	444 Chestnut St
9	9	Grace	Moore	grace.moore@example.com	-7233	555 Spruce St
10	10	Harry	Anderson	raju.rastogi@gmail..com	-9356	221b, ST 12345
11	11	Sherlock	Holmes	sherlock.holmes@gmail.com	NULL	221B Baker Street

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
select sum(p.price * o.Quantity) as order_amt, o.OrderID
from Products p, OrderDetails o
where p.ProductID = o.ProductID
group by o.OrderID;
```

OUTPUT:



Results | Messages

	order_amt	OrderID
1	3948	1
2	878	2
3	1090	3
4	432	5
5	186	6
6	876	7
7	163	8

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 = 4;


delete from Orders

where CustomerID=@CustomerId;
select * from Orders;

delete from OrderDetails
where OrderID = (select OrderID from Orders where CustomerID =
@CustomerId);
```

OUTPUT:

Results | Messages

	OrderID	CustomerID	OrderDate	TotalAmount
1	1	1	2024-09-01	1500
2	2	2	2024-09-02	2500
3	3	3	2024-09-05	1000
4	5	1	2024-09-07	2200
5	6	2	2024-09-08	1800
6	7	5	2024-09-10	2700
7	8	3	2024-09-11	3200
8	10	3	2024-09-04	2500
9	11	3	2024-09-04	2500

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 values
('PS5','Console',1055);
```

OUTPUT:

```
(1 row affected)

Completion time: 2024-09-24T17:56:26.8434592+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.

```
Alter table Orders add Status varchar(20);

update Orders
set Status = 'Pending';

Declare @OrderId int = 3;

update Orders
set Status = 'Shipped'
where OrderID=@OrderId;
```

OUTPUT:

Results Messages

	OrderID	CustomerID	OrderDate	TotalAmount	Status
1	1	1	2024-09-01	1500	Pending
2	2	2	2024-09-02	2500	Pending
3	3	3	2024-09-05	1000	Shipped
4	5	1	2024-09-07	2200	Pending
5	6	2	2024-09-08	1800	Pending
6	7	5	2024-09-10	2700	Pending
7	8	3	2024-09-11	3200	Pending
8	10	3	2024-09-04	2500	Pending
9	11	3	2024-09-04	2500	Pending

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

```
SELECT C.*,
       (SELECT COUNT(*)
        FROM Orders O
        WHERE O.CustomerID = C.CustomerID) AS NumberOfOrders
FROM Customers C;
```

OUTPUT:

Results Messages

	CustomerID	FirstName	LastName	Email	Phone	Address	NumberOfOrders
1	1	John	Doe	john.doe@example.com	-679	123 Elm St	2
2	2	Jane	Smith	jane.smith@example.com	-5123	456 Oak St	2
3	3	Alice	Johnson	alice.j@example.com	-8210	789 Pine St	4
4	4	Bob	Brown	bob.brown@example.com	-3766	101 Maple St	0
5	5	Charlie	Davis	charlie.davis@example.com	-9321	102 Cedar St	1
6	6	David	Wilson	david.wilson@example.com	-567	222 Birch St	0
7	7	Eva	Taylor	eva.taylor@example.com	-2789	333 Walnut St	0
8	8	Frank	Thomas	frank.thomas@example.com	-5011	444 Chestnut St	0
9	9	Grace	Moore	grace.moore@example.com	-7233	555 Spruce St	0
10	10	Harry	Anderson	raju.rastogi@gmail..com	-9356	221b, ST 12345	0
11	11	Sherlock	Holmes	sherlock.holmes@gmail.com	NULL	221B Baker Street	0

TASK -3: Aggregate functions, Having, Order By, GroupBy and Joins:

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

```
select Orders.*,
      (select FirstName from Customers
      where Customers.CustomerID=Orders.CustomerID) as Names
 from Orders;
```

OUTPUT:

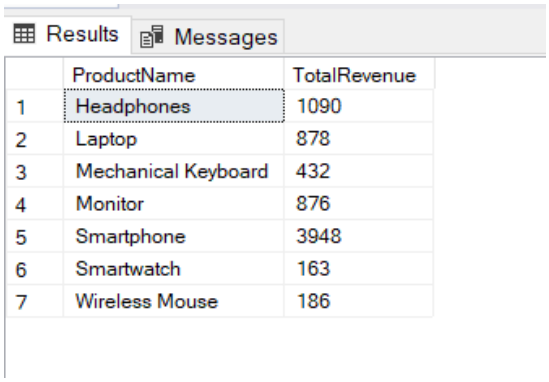
Results | Messages

	OrderID	CustomerID	OrderDate	TotalAmount	Status	Names
1	1	1	2024-09-01	1500	Pending	John
2	2	2	2024-09-02	2500	Pending	Jane
3	3	3	2024-09-05	1000	Shipped	Alice
4	5	1	2024-09-07	2200	Pending	John
5	6	2	2024-09-08	1800	Pending	Jane
6	7	5	2024-09-10	2700	Pending	Charlie
7	8	3	2024-09-11	3200	Pending	Alice
8	10	3	2024-09-04	2500	Pending	Alice
9	11	3	2024-09-04	2500	Pending	Alice

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
GROUP BY P.ProductName;
```

OUTPUT:

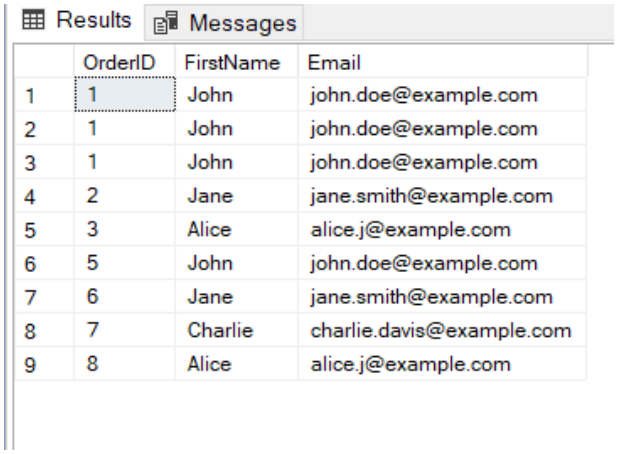


	ProductName	TotalRevenue
1	Headphones	1090
2	Laptop	878
3	Mechanical Keyboard	432
4	Monitor	876
5	Smartphone	3948
6	Smartwatch	163
7	Wireless Mouse	186

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

```
SELECT OrderDetails.OrderID, Customers.FirstName, Customers.Email
FROM OrderDetails
JOIN Orders ON OrderDetails.OrderID = Orders.OrderID
JOIN Customers ON Orders.CustomerID = Customers.CustomerID
WHERE OrderDetails.Quantity >= 1;
```

OUTPUT:



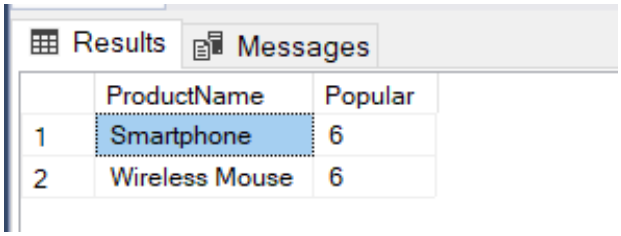
	OrderID	FirstName	Email
1	1	John	john.doe@example.com
2	1	John	john.doe@example.com
3	1	John	john.doe@example.com
4	2	Jane	jane.smith@example.com
5	3	Alice	alice.j@example.com
6	5	John	john.doe@example.com
7	6	Jane	jane.smith@example.com
8	7	Charlie	charlie.davis@example.com
9	8	Alice	alice.j@example.com

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 with ties a.ProductName , sum(b.Quantity) as Popular
FROM Products a , OrderDetails b
WHERE a.ProductID = b.ProductID
GROUP BY a.ProductName
```

```
ORDER BY Popular DESC;
```

OUTPUT:



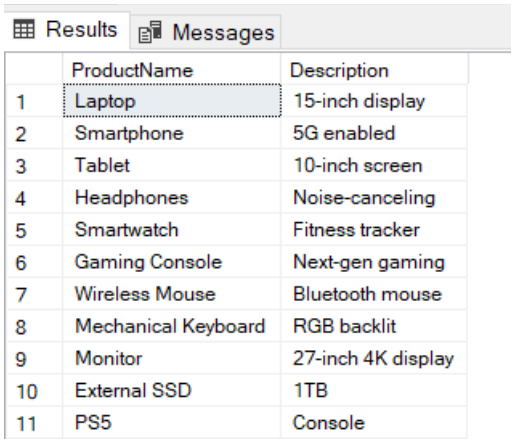
Results | Messages

	ProductName	Popular
1	Smartphone	6
2	Wireless Mouse	6

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

```
select ProductName,Description from Products;
```

OUTPUT:



Results | Messages

	ProductName	Description
1	Laptop	15-inch display
2	Smartphone	5G enabled
3	Tablet	10-inch screen
4	Headphones	Noise-canceling
5	Smartwatch	Fitness tracker
6	Gaming Console	Next-gen gaming
7	Wireless Mouse	Bluetooth mouse
8	Mechanical Keyboard	RGB backlit
9	Monitor	27-inch 4K display
10	External SSD	1TB
11	PS5	Console

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 a.FirstName ,AVG(b.TotalAmount)
from Customers a , Orders b
where a.CustomerID=b.CustomerID
group by a.FirstName;
```

OUTPUT:

Results Messages

	FirstName	(No column name)
1	Alice	2300
2	Charlie	2700
3	Jane	2150
4	John	1850

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 sum(a.Price * b.Quantity) as revenue, b.OrderID , FirstName
from Products a, OrderDetails b, Orders c, Customers d
where a.ProductID = b.ProductID
  and c.OrderID=b.OrderID
  and c.CustomerID=d.CustomerID
group by b.OrderID, FirstName
order by  revenue desc;
```

OUTPUT:

Results Messages

	revenue	OrderID	FirstName
1	3948	1	John

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

```
select a.ProductName , sum(b.Quantity ) as Number_of_Orders
from Products a , OrderDetails b
where a.ProductID = b.ProductID
group by a.ProductName;
```

OUTPUT:

Results | Messages

	ProductName	Number_of_Orders
1	Headphones	5
2	Laptop	1
3	Mechanical Keyboard	4
4	Monitor	2
5	Smartphone	6
6	Smartwatch	1
7	Wireless Mouse	6

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 NVARCHAR(100);

SET @ProductName = 'Laptop';

select a.FirstName , b.ProductName
from Products b , Customers a , Orders c , OrderDetails d
where b.ProductID = d.ProductID
  and d.OrderID = c.OrderID
  and a.CustomerID = c.CustomerID
  and ProductName = @ProductName;
```

OUTPUT:

Results | Messages

	FirstName	ProductName
1	Jane	Laptop

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

```
select a.OrderDate, sum(b.price * c.Quantity) as revenue
from products b, Orders a, OrderDetails c
where a.OrderID = c.OrderID
  and c.ProductID = b.ProductID
  and OrderDate = '2024-09-08'
group by OrderDate;
```

OUTPUT:

Results Messages

	OrderDate	revenue
1	2024-09-08	186

TASK -4: Subquery and its type:

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

```
SELECT Customers.CustomerID, Customers.FirstName, Customers.LastName,
Customers.Email
FROM Customers
LEFT JOIN Orders ON Customers.CustomerID = Orders.CustomerID
WHERE Orders.OrderID IS NULL;
```

OUTPUT:

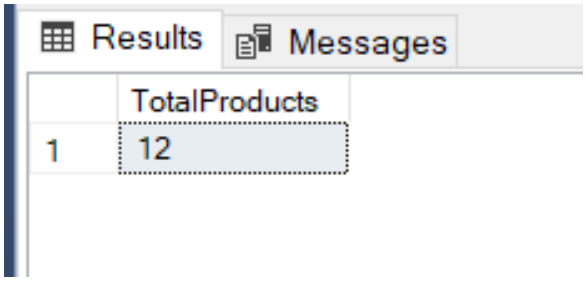
Results Messages

	CustomerID	FirstName	LastName	Email
1	4	Bob	Brown	bob.brown@example.com
2	6	David	Wilson	david.wilson@example.com
3	7	Eva	Taylor	eva.taylor@example.com
4	8	Frank	Thomas	frank.thomas@example.com
5	9	Grace	Moore	grace.moore@example.com
6	10	Harry	Anderson	raju.rastogi@gmail..com
7	11	Sherlock	Holmes	sherlock.holmes@gmail.com

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

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

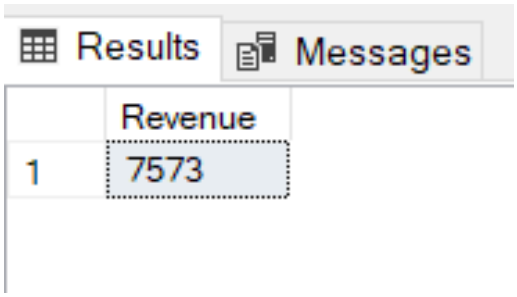
OUTPUT:



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

```
SELECT SUM(a.Quantity*b.Price) as Revenue
FROM OrderDetails a , Products b
WHERE a.ProductID = b.ProductID
```

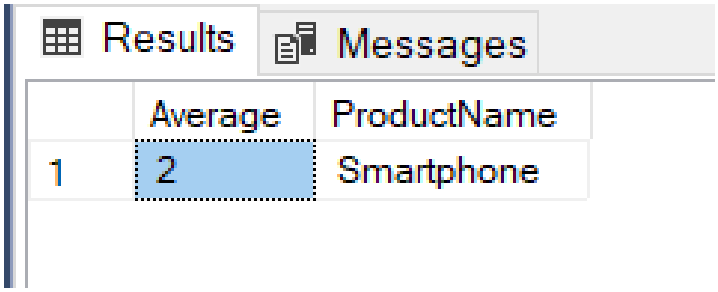
OUTPUT:



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 NVARCHAR(100);

SET @ProductName = 'Smartphone';

SELECT AVG(a.Quantity) Average  , b.ProductName
FROM OrderDetails a , Products b
WHERE a.ProductID = b.ProductID
  AND ProductName = @ProductName
GROUP BY b.ProductName
```

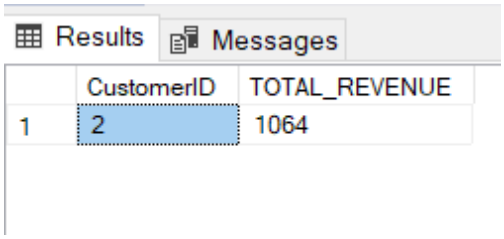
OUTPUT:



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 = 2 ;

SELECT a.CustomerID , SUM(c.Quantity * b.Price) TOTAL_REVENUE
FROM Customers a , Products b , OrderDetails c , Orders d
WHERE b.ProductID  = c.ProductID
  AND c.OrderID    = d.OrderID
  AND d.CustomerID = a.CustomerID
  AND a.CustomerID = @CustomerID
GROUP BY a.CustomerID;
```

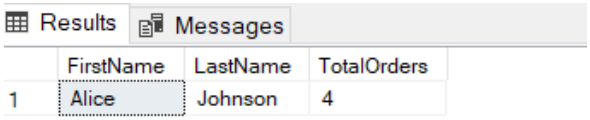
OUTPUT:



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 WITH TIES C.FirstName, C.LastName, COUNT(O.OrderID) AS
TotalOrders
FROM Customers C
JOIN Orders O ON C.CustomerID = O.CustomerID
GROUP BY C.FirstName, C.LastName
ORDER BY TotalOrders DESC;
```

OUTPUT:



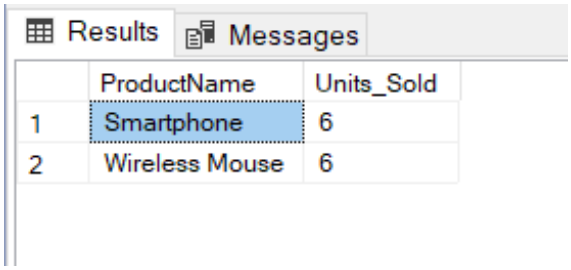
Results | Messages

	FirstName	LastName	TotalOrders
1	Alice	Johnson	4

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 WITH TIES a.ProductName , SUM(b.Quantity) Units_Sold
FROM Products a , OrderDetails b
WHERE a.ProductID = b.ProductID
GROUP BY a.ProductName
ORDER BY Units_Sold DESC;
```

OUTPUT:



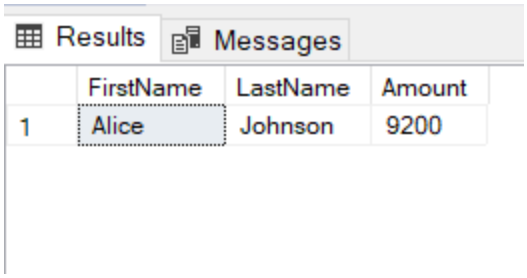
Results | Messages

	ProductName	Units_Sold
1	Smartphone	6
2	Wireless Mouse	6

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 WITH TIES a.FirstName , a.LastName , SUM(b.TotalAmount)
Amount
FROM Customers a , Orders b
WHERE a.CustomerID = b.CustomerID
GROUP BY a.FirstName , a.LastName
ORDER BY Amount DESC;
```

OUTPUT:



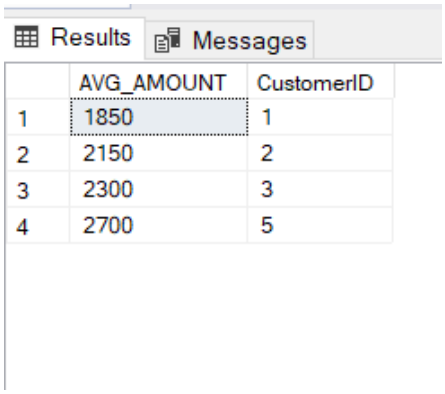
Results Messages

	FirstName	LastName	Amount
1	Alice	Johnson	9200

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

```
SELECT avg(o.TotalAmount) AVG_AMOUNT , o.CustomerID
FROM Orders o
GROUP BY o.CustomerID;
```

OUTPUT:



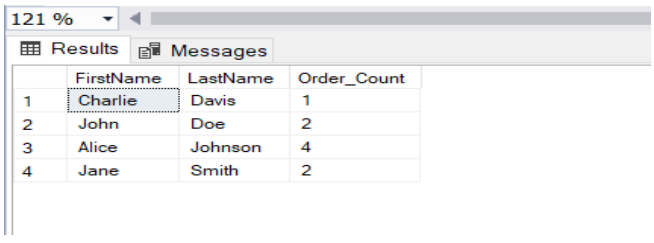
Results Messages

	AVG_AMOUNT	CustomerID
1	1850	1
2	2150	2
3	2300	3
4	2700	5

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 a.FirstName , a.LastName , COUNT(b.OrderID) Order_Count
FROM Customers a , Orders b
WHERE a.CustomerID = b.CustomerID
GROUP BY a.FirstName , a.LastName
```

OUTPUT:



121 % Results Messages

	FirstName	LastName	Order_Count
1	Charlie	Davis	1
2	John	Doe	2
3	Alice	Johnson	4
4	Jane	Smith	2