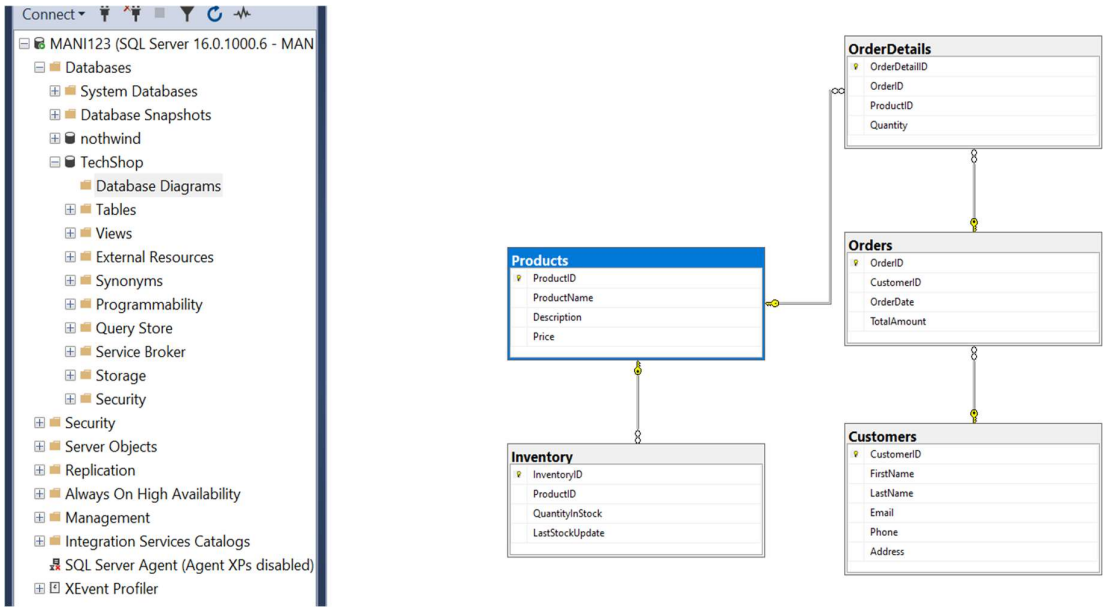
ASSIGNMENT 1:

TASK 1: Database Design

1. Creating Database **Techshop:**

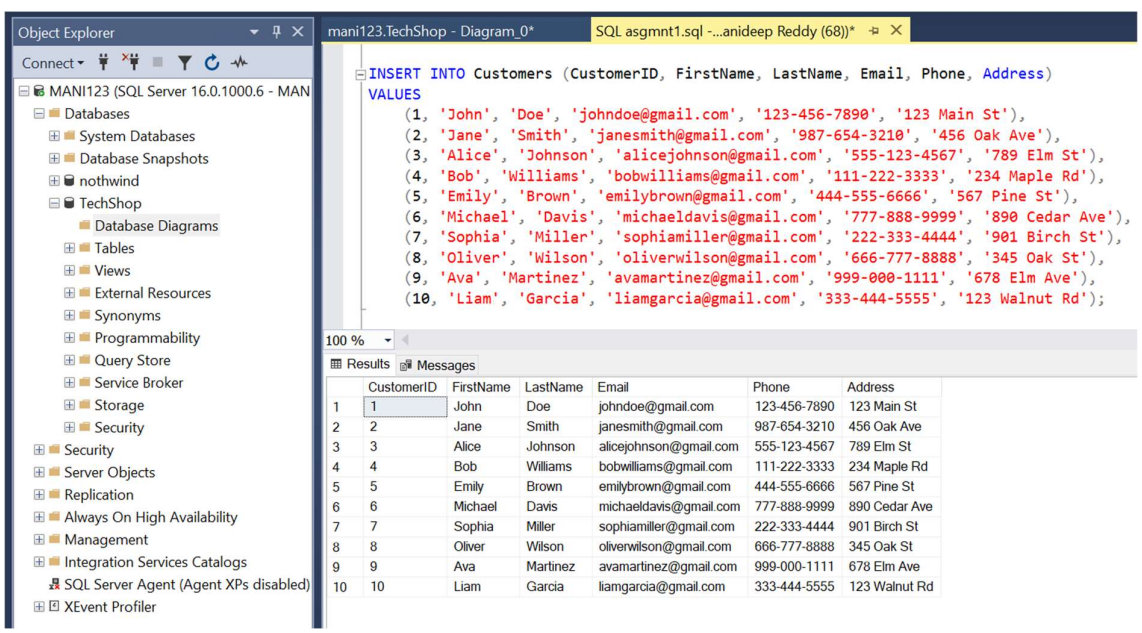
```
create database TechShop
use TechShop

create Table Customers
(CustomerID int primary key,
FirstName varchar(50),
LastName varchar(50),
Email varchar(50),
Phone varchar(20),
Address varchar(100)
)

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


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


CREATE TABLE OrderDetails (
OrderDetailID INT PRIMARY KEY,
OrderID INT,
ProductID INT,
Quantity INT,
FOREIGN KEY (OrderID) REFERENCES Orders(OrderID),
FOREIGN KEY (ProductID) REFERENCES Products(ProductID)
);


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

2. Entity Relationship Diagram:



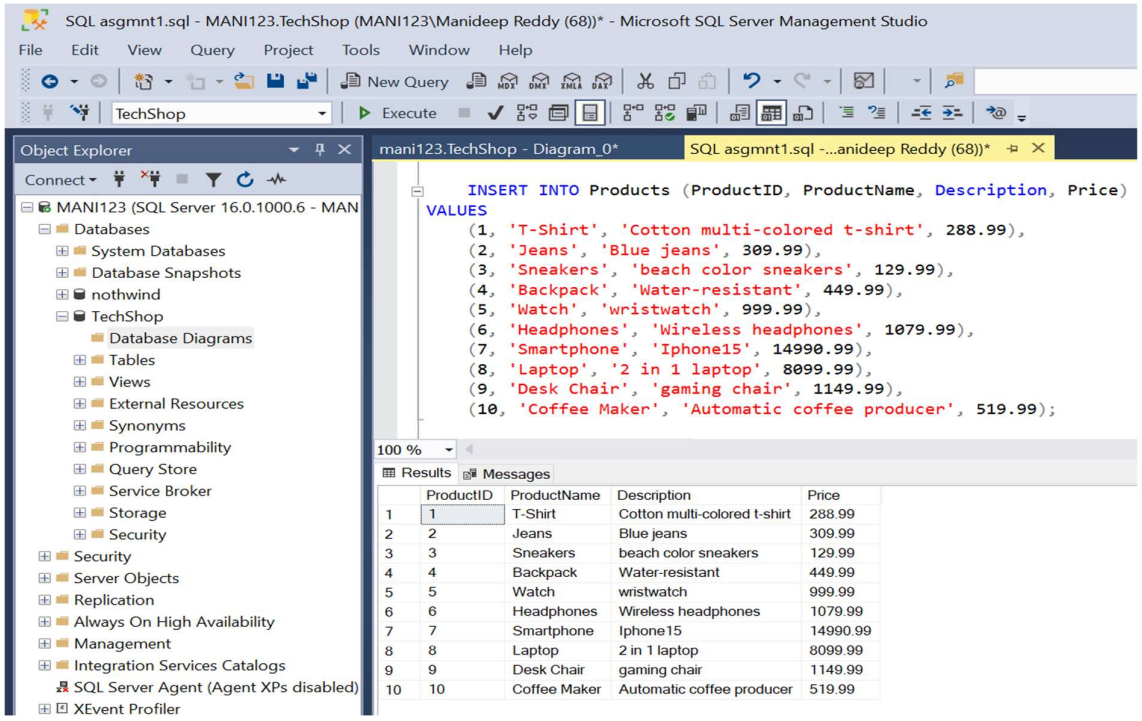
3. Inserting sample records into the tables:

A. Customers table:

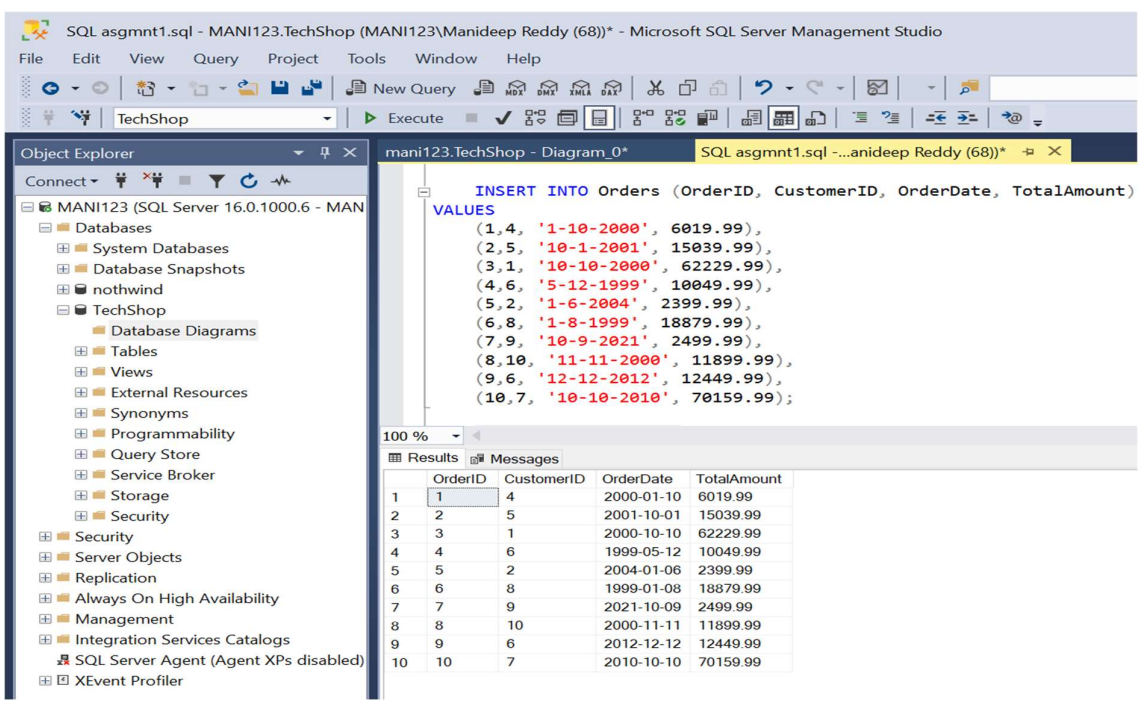


	CustomerID	FirstName	LastName	Email	Phone	Address
1	1	John	Doe	johndoe@gmail.com	123-456-7890	123 Main St
2	2	Jane	Smith	janesmith@gmail.com	987-654-3210	456 Oak Ave
3	3	Alice	Johnson	alicejohnson@gmail.com	555-123-4567	789 Elm St
4	4	Bob	Williams	bobwilliams@gmail.com	111-222-3333	234 Maple Rd
5	5	Emily	Brown	emilybrown@gmail.com	444-555-6666	567 Pine St
6	6	Michael	Davis	michaeldavis@gmail.com	777-888-9999	890 Cedar Ave
7	7	Sophia	Miller	sophiamiller@gmail.com	222-333-4444	901 Birch St
8	8	Oliver	Wilson	oliverwilson@gmail.com	666-777-8888	345 Oak St
9	9	Ava	Martinez	avamartinez@gmail.com	999-000-1111	678 Elm Ave
10	10	Liam	Garcia	liamgarcia@gmail.com	333-444-5555	123 Walnut Rd

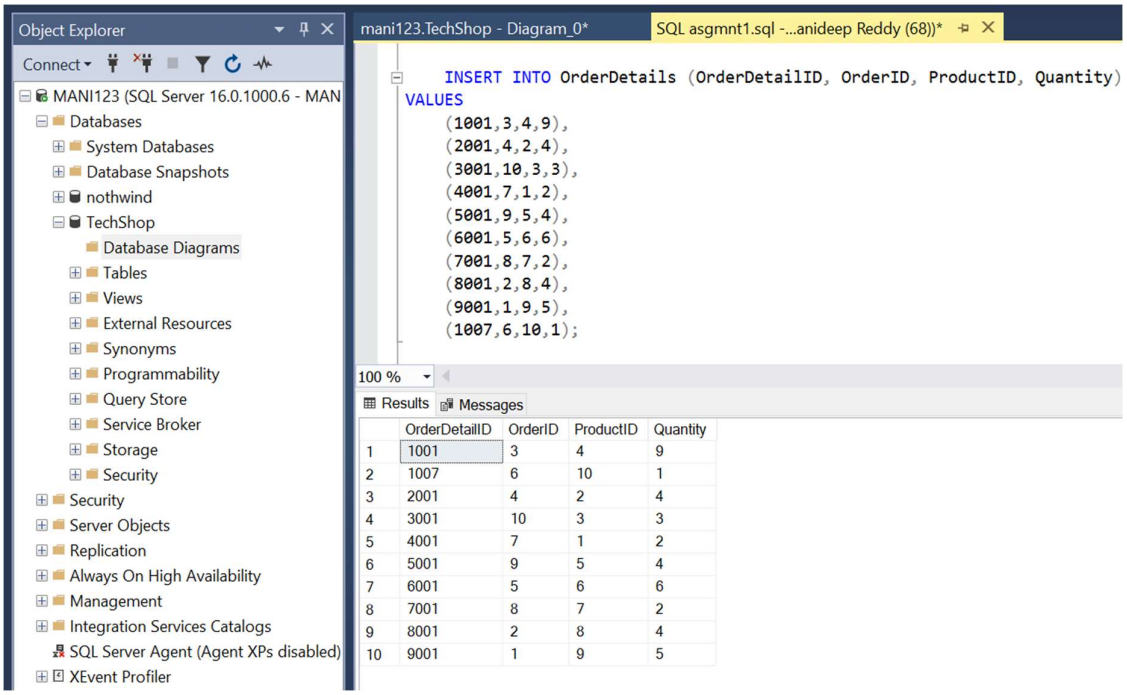
B. Products table:



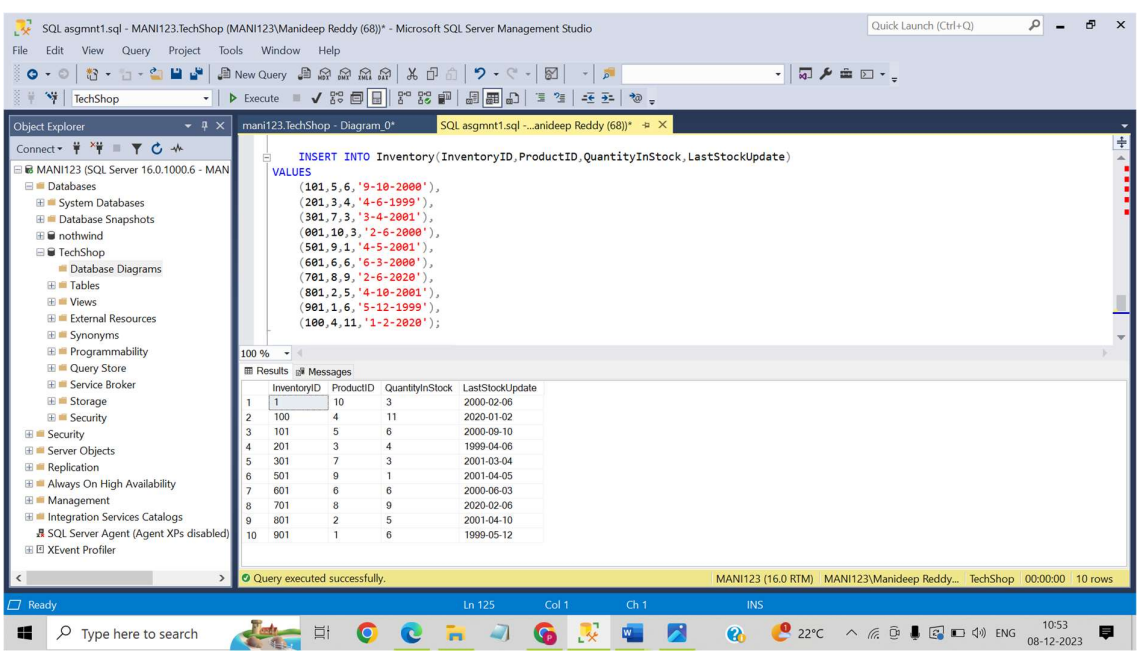
C. Orders table:



D. OrderDetails table:

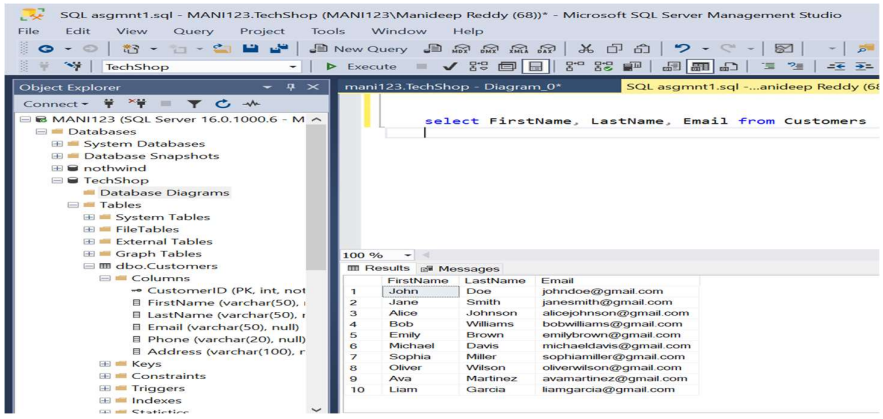


E. Inventory table:

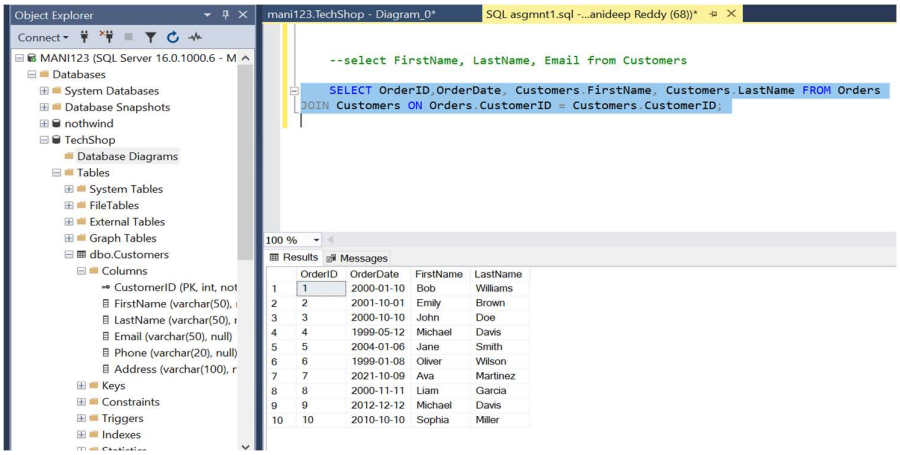


TASK 2: Select, Where, Between and Like

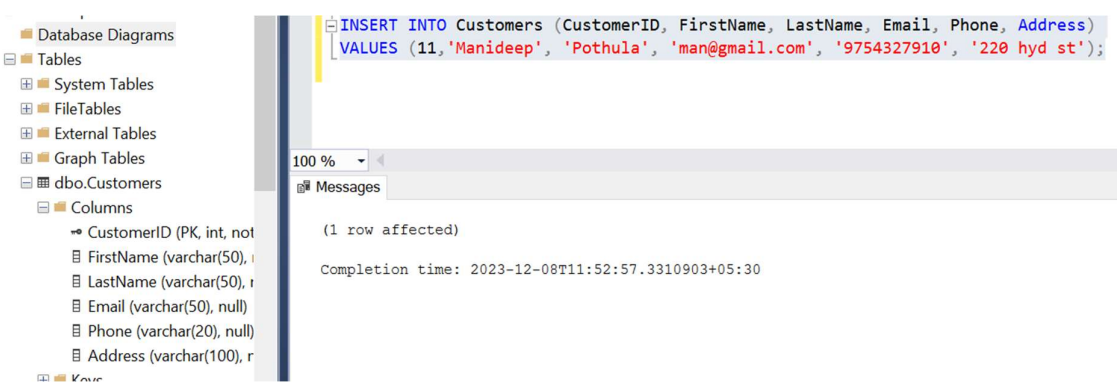
1.Write an SQL query to retrieve the names and emails of all customers



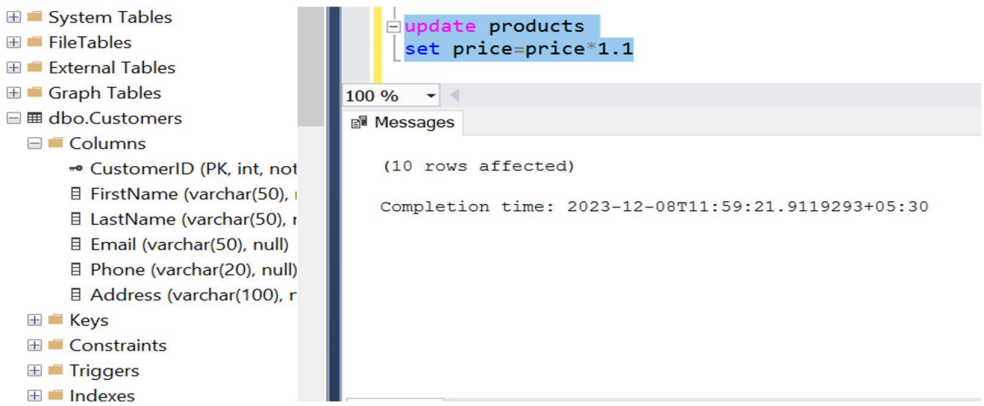
2. Write SQL query to list all orders with their order dates and corresponding customer name.



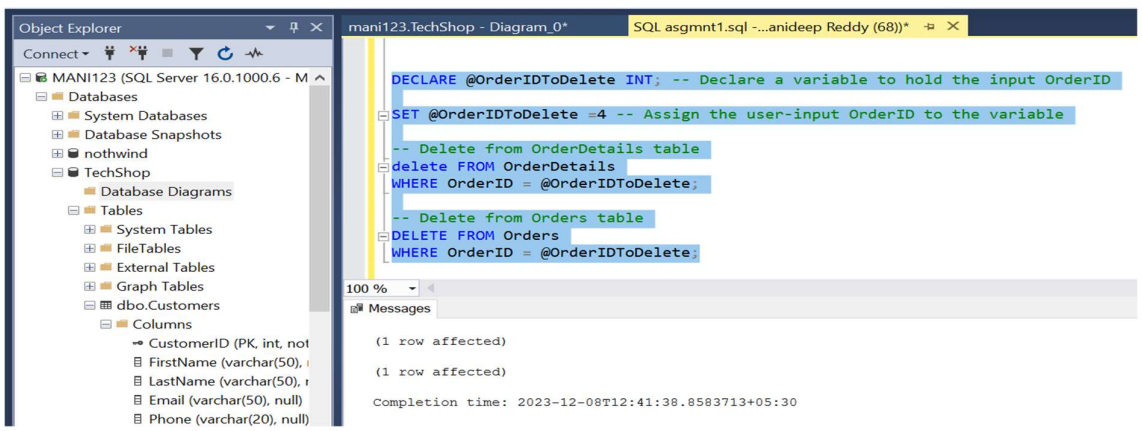
3. Write an SQL query to insert a new customer record into the "Customers" table. Include customer information such as name, email, and address.



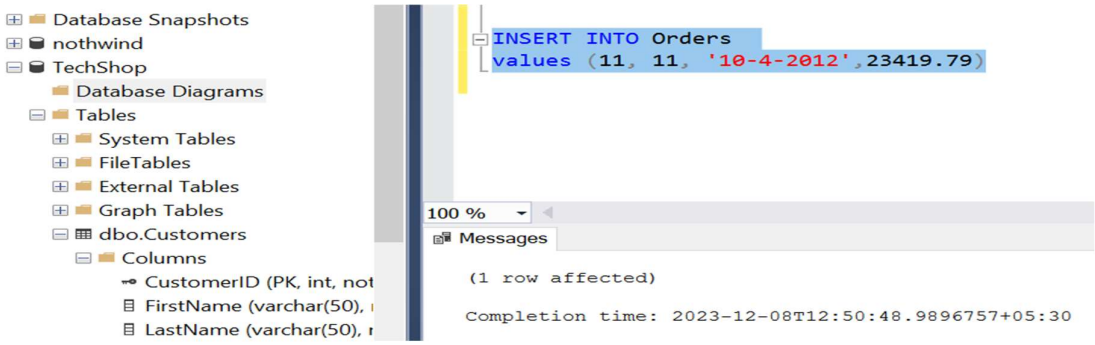
4. Write an SQL query to update the prices of all electronic gadgets in the "Products" table by increasing them by 10%.



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



6. Write an SQL query to insert a new order into the "Orders" table. Include the customer ID, order date, and any other necessary information.



7. Write an SQL query to update the contact information (e.g., email and address) of a specific customer in the "Customers" table. Allow users to input the customer ID and new contact information.

```sql
DECLARE @newemail varchar(50),@newaddress varchar(50),@newID int;
SET @newemail = 'man123@gmail.com'
SET @newaddress = 'hyd 125'
SET @newID = 4

update Customers
SET Email = @newemail ,Address = @newaddress
where CustomerId = @newID
```

100 %

Messages

```
(1 row affected)

Completion time: 2023-12-08T13:02:38.1639849+05:30
```

100 %

Query executed successfully.

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

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

150 %

Messages

```
(10 rows affected)

Completion time: 2023-12-07T20:26:33.3855310+05:30
```

150 %

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 @newID int;
SET @newID = 5
--first delete from oderdetails table
delete from OrderDetails
where OrderID in (select OrderID from Orders where CustomerId = @newID)
--next delete from orders table
delete from Orders
where CustomerID = @newID
```

100 %

Messages

```
(1 row affected)

(1 row affected)

Completion time: 2023-12-08T13:27:57.4685400+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.

```sql
INSERT INTO Products
values (11, 'Smart cooker', 'Uses electricity',41.79)


DECLARE @newID int;
SET @newID = 5
--first delete from oderdetails table
delete from OrderDetails
```

%

Messages

```
(1 row affected)

Completion time: 2023-12-08T13:40:39.7016997+05:30
```

11. 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.

```sql
SELECT Customers.CustomerID, COUNT(*) as totalOrders
FROM Orders join Customers on Customers.CustomerID = Orders.CustomerID group by Customers.CustomerID
```

100 %

Results | Messages

	CustomerID	totalOrders
1	1	1
2	2	1
3	4	1
4	6	1
5	7	1
6	8	1
7	9	1
8	10	1
9	11	1

TASK 3: Aggregate Functions, having, order by, group by 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.

```sql
SELECT OrderID, OrderDate, concat(FirstName, LastName) as CustomerName
FROM Orders
JOIN Customers ON Orders.CustomerID = Customers.CustomerID;
```

100 %

Results | Messages

	OrderID	OrderDate	CustomerName
1	1	2000-01-10	BobWilliams
2	3	2000-10-10	JohnDoe
3	5	2004-01-06	JaneSmith
4	6	1999-01-08	OliverWilson
5	7	2021-10-09	AvaMartinez
6	8	2000-11-11	LiamGarcia
7	9	2012-12-12	MichaelDavis
8	10	2010-10-10	SophiaMiller
9	11	2012-10-04	ManideepPothula

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

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

100 %

Results | Messages

	ProductName	TotalRevenue
1	Backpack	4454.91
2	Coffee Maker	571.99
3	Desk Chair	6324.95
4	Headphones	7127.94
5	Smartphone	32980.18
6	Sneakers	428.97
7	T-Shirt	635.78
8	Watch	4399.96

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

```sql
SELECT DISTINCT Concat(FirstName,LastName) as CustomerName, Email,Phone,Address
FROM Customers JOIN Orders  ON Customers.CustomerID = Orders.CustomerID;
```

%

esults | Messages

CustomerName	Email	Phone	Address
AvaMartinez	avamartinez@gmail.com	999-000-1111	678 Elm Ave
BobWilliams	man123@gmail.com	111-222-3333	hyd 125
JaneSmith	janesmith@gmail.com	987-654-3210	456 Oak Ave
JohnDoe	johndoe@gmail.com	123-456-7890	123 Main St
LiamGarcia	liamgarcia@gmail.com	333-444-5555	123 Walnut Rd
ManideepPothula	man@gmail.com	9754327910	220 hyd st
MichaelDavis	michaeldavis@gmail.com	777-888-9999	890 Cedar Ave
OliverWilson	oliverwilson@gmail.com	666-777-8888	345 Oak St
SophiaMiller	sophiamiller@gmail.com	222-333-4444	901 Birch St

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

```sql
SELECT TOP 1 ProductName, MAX(Quantity) AS TotalQuantity
FROM Products
JOIN OrderDetails OD ON Products.ProductID = OD.ProductID
Group by ProductName
```

100 %

Results | Messages

	ProductName	TotalQuantity
1	Backpack	9

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

```sql
select ProductName, category from Products
```

100 %

Results | Messages

	ProductName	category
1	T-Shirt	NULL
2	Jeans	NULL
3	Sneakers	NULL
4	Backpack	NULL
5	Watch	NULL
6	Headphones	NULL
7	Smartphone	NULL
8	Laptop	NULL
9	Desk Chair	NULL
10	Coffee Maker	NULL
11	Smart cooker	NULL

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

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

%

Results | Messages

CustomerName	AverageOrderValue
MichaelDavis	12449.990000
JohnDoe	62229.990000
LiamGarcia	11899.990000
AvaMartinez	2499.990000
SophiaMiller	70159.990000
ManideepPothula	23419.790000
JaneSmith	2399.990000
BobWilliams	6019.990000
OliverWilson	18879.990000

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

```sql
Select top 1 Orders.OrderID, c.FirstName, c.LastName, sum(Orders.TotalAmount) as TotalRevenue
 from Orders join Customers c on Orders.CustomerID=C.CustomerID
 group by Orders.OrderID,c.FirstName,c.LastName
 Order by TotalRevenue desc
```

100 %

Results | Messages

	OrderID	FirstName	LastName	TotalRevenue
1	10	Sophia	Miller	70159.99

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

```sql
select ProductName, count(OrderDetails.OrderID) as OrderCount from Products
join OrderDetails on Products.ProductID= OrderDetails.ProductID
group by Products.ProductName
```

100 %

Results | Messages

	ProductName	OrderCount
1	Backpack	1
2	Coffee Maker	1
3	Desk Chair	1
4	Headphones	1
5	Smartphone	1
6	Sneakers	1
7	T-Shirt	1
8	Watch	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.

```sql
DECLARE @newName varchar(50)
SET @newName = 'Smartphone'

select c.FirstName,c.LastName,c.Email,c.Phone,c.Address from Customers c join Orders on c.CustomerID=Orders.CustomerID
join OrderDetails on Orders.OrderID=OrderDetails.OrderID
join Products on OrderDetails.ProductID=Products.ProductID
where ProductName=@newName
```

%

esults | Messages

FirstName	LastName	Email	Phone	Address
Liam	Garcia	liamgarcia@gmail.com	333-444-5555	123 Walnut Rd

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.

```sql
select sum(TotalAmount) as TotalRevenue
from Orders where OrderDate between '1-1-2000' and '10-10-2000'
```

100 %

Results | Messages

	TotalRevenue
1	68249.98

TASK 4: Subquery and its Types

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

```sql
SELECT C.CustomerID, C.FirstName, C.LastName
FROM Customers C
LEFT JOIN Orders O ON C.CustomerID = O.CustomerID
WHERE O.CustomerID IS NULL;
```

100 %

Results | Messages

	CustomerID	FirstName	LastName
1	3	Alice	Johnson
2	5	Emily	Brown

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

```sql
select count(ProductID) as TotalAvailableProducts from Products
```

100 %

Results | Messages

	TotalAvailableProducts
1	11

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

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

100 %

Results | Messages

	TotalRevenueGenerated
1	209959.71

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.

```sql
DECLARE @newName varchar(50)
 SET @newName = 'Smartphone'

 select avg(OrderDetails.Quantity) as AvgQuantityOrdered from OrderDetails
 join Products on OrderDetails.ProductID=Products.ProductID
 where ProductName=@newName
```

100 %

Results | Messages

	AvgQuantityOrdered
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.

```sql
DECLARE @newIDZ int
set @newIDZ=4

select sum(TotalAmount) as TotalRevenue from Orders
where CustomerID=@newIDZ
```

100 %

Results | Messages

	TotalRevenue
1	6019.99

6. Write an SQL query to find the customers who have placed the most orders. List their names and the number of orders they have placed.

```sql
Select top 1 c.FirstName, c.LastName, count(Orders.OrderID) as OrderCount from Customers c
join Orders on c.CustomerID=Orders.CustomerID
group by c.FirstName,c.LastName
order by OrderCount desc
```

100 %

Results | Messages

	FirstName	LastName	OrderCount
1	John	Doe	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.

```sql
Select top 1 P.ProductName, sum(OrderDetails.Quantity) as TotalOrderCount from Products P
join OrderDetails on P.ProductID=OrderDetails.ProductID
group by P.ProductName
order by TotalOrderCount desc
```

100 %

Results | Messages

	ProductName	TotalOrderCount
1	Backpack	9

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.

```sql
Select top 1 c.FirstName, c.LastName, sum(Orders.TotalAmount) as TotalSpent from Customers c
join Orders on c.CustomerID=Orders.CustomerID
join OrderDetails on Orders.OrderID = OrderDetails.OrderID
join Products on OrderDetails.ProductID= Products.ProductID
group by c.FirstName,c.LastName
order by TotalSpent desc
```

100 %

Results | Messages

	FirstName	LastName	TotalSpent
1	Sophia	Miller	70159.99

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

```sql
Select c.FirstName, c.LastName, avg(Orders.TotalAmount) as AvgOrderValue from Customers c
join Orders on c.CustomerID=Orders.CustomerID
group by c.FirstName,c.LastName
```

100 %

Results | Messages

	FirstName	LastName	AvgOrderValue
1	Michael	Davis	12449.990000
2	John	Doe	62229.990000
3	Liam	Garcia	11899.990000
4	Ava	Martinez	2499.990000
5	Sophia	Miller	70159.990000
6	Manideep	Pothula	23419.790000
7	Jane	Smith	2399.990000
8	Bob	Williams	6019.990000
9	Oliver	Wilson	18879.990000

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.

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

100 %

Results | Messages

	CustomerID	FirstName	LastName	OrderCount
1	1	John	Doe	1
2	2	Jane	Smith	1
3	3	Alice	Johnson	0
4	4	Bob	Williams	1
5	5	Emily	Brown	0
6	6	Michael	Davis	1
7	7	Sophia	Miller	1
8	8	Oliver	Wilson	1
9	9	Ava	Martinez	1
10	10	Liam	Garcia	1
11	11	Manideep	Pothula	1