Foundation Technical Training

Assignment No. 1 - SQL - Electronic Gadgets - Task 4

Name: Niranjan Kolpe, Batch: C#-Batch 2

Tasks 4: Subquery and its type:

1. Write an SQL query to find out which customers have not placed any orders.
2. Write an SQL query to find the total number of products available for sale.
3. Write an SQL query to calculate the total revenue generated by TechShop.
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.
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.
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.
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.
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.
9. Write an SQL query to calculate the average order value (total revenue divided by the number of orders) for all customers.
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

Program Code in SQL:

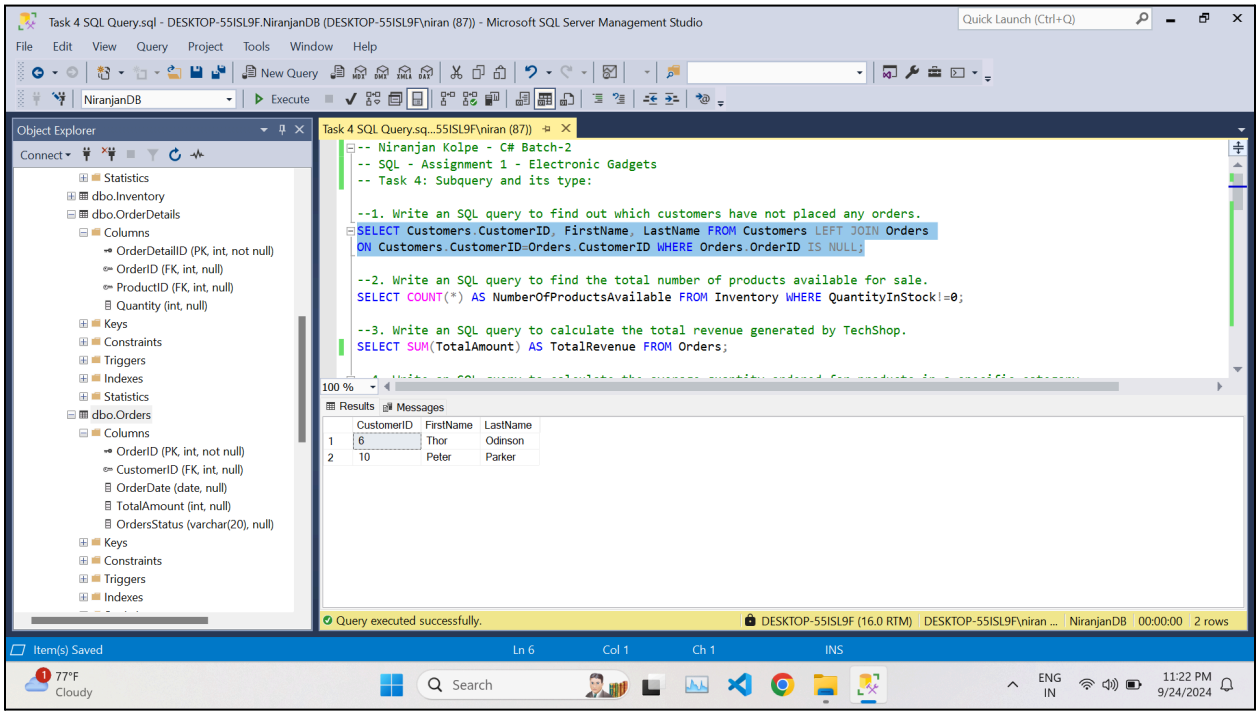
```
-- Niranjan Kolpe - C# Batch-2

-- SQL - Assignment 1 - Electronic Gadgets

-- Task 4: Subquery and its type:


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

SELECT Customers.CustomerID, FirstName, LastName FROM Customers LEFT JOIN Orders

ON Customers.CustomerID=Orders.CustomerID WHERE Orders.OrderID IS NULL;


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

```sql
SELECT COUNT(*) AS NumberOfProductsAvailable FROM Inventory WHERE QuantityInStock!=0;


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

SELECT SUM(TotalAmount) AS TotalRevenue FROM Orders;


--4. Write an SQL query to calculate the average quantity ordered for products in a specific category.

--Allow users to input the category name as a parameter.

DECLARE @ProductName VARCHAR(20) = 'Keyboard';

SELECT ProductName, AVG(Quantity) AS AverageQuantity

FROM OrderDetails JOIN Products ON OrderDetails.ProductID=Products.ProductID

GROUP BY ProductName HAVING ProductName=@ProductName;


--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 = 3;

SELECT Customers.CustomerID, FirstName, SUM(TotalAmount) AS Revenue

FROM Customers JOIN Orders ON Customers.CustomerID=Orders.CustomerID

GROUP BY Customers.CustomerID, FirstName HAVING Customers.CustomerID=@CustomerID;



--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 Customers.CustomerID, FirstName, Count(*) AS NumberOfOrdersPlaced

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

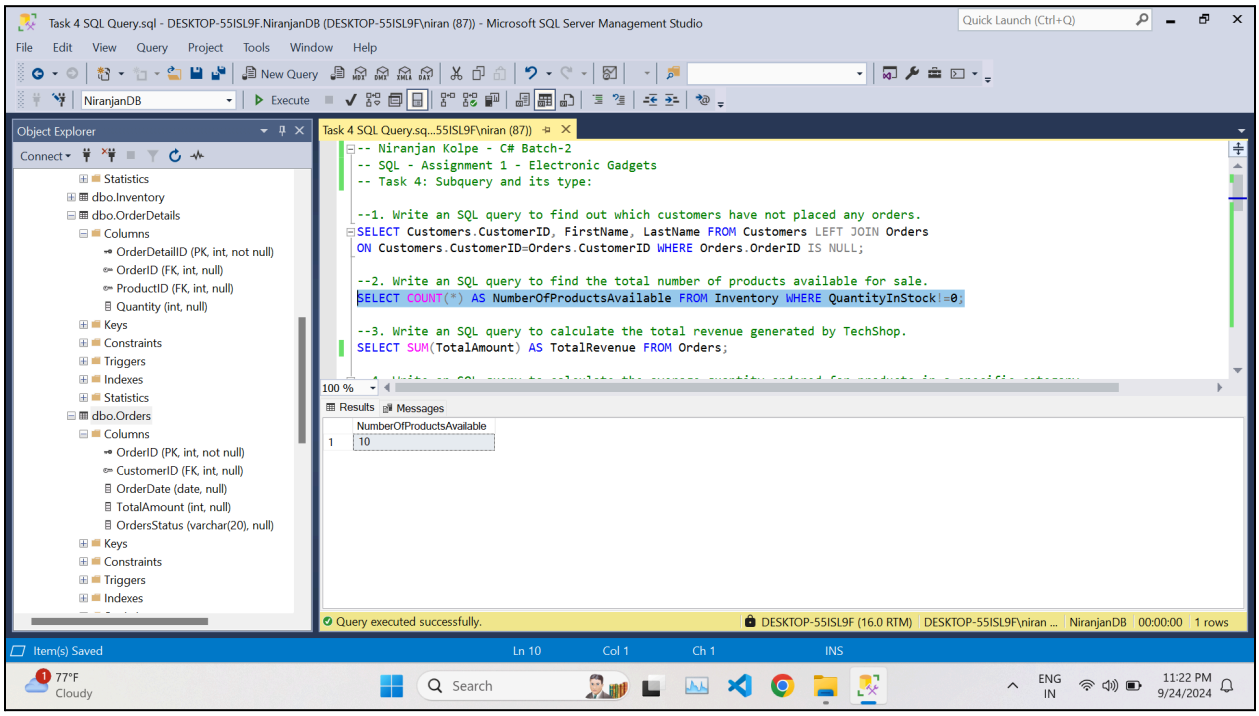
```
GROUP BY Customers.CustomerID, FirstName ORDER BY NumberOfOrdersPlaced DESC;


--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 Products.ProductID, Products.ProductName, SUM(Quantity) AS TotalQuantity

FROM Products JOIN OrderDetails ON Products.ProductID=OrderDetails.ProductID

GROUP BY Products.ProductID, ProductName ORDER BY TotalQuantity DESC;


--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 Customers.CustomerID, FirstName, SUM(TotalAmount) AS Revenue

FROM Customers JOIN Orders ON Customers.CustomerID=Orders.CustomerID

GROUP BY Customers.CustomerID, FirstName ORDER BY Revenue DESC;


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

-- for all customers.

SELECT Customers.CustomerID, FirstName, SUM(Orders.TotalAmount/Customers.NumberOfOrders) AS AverageOrderValue

FROM Customers JOIN Orders ON Customers.CustomerID=Orders.CustomerID

GROUP BY Customers.CustomerID, FirstName ORDER BY AverageOrderValue DESC;


--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 Customers.CustomerID, FirstName, COUNT(*) AS NumberOfOrders
```

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

GROUP BY Customers.CustomerID, FirstName;
```

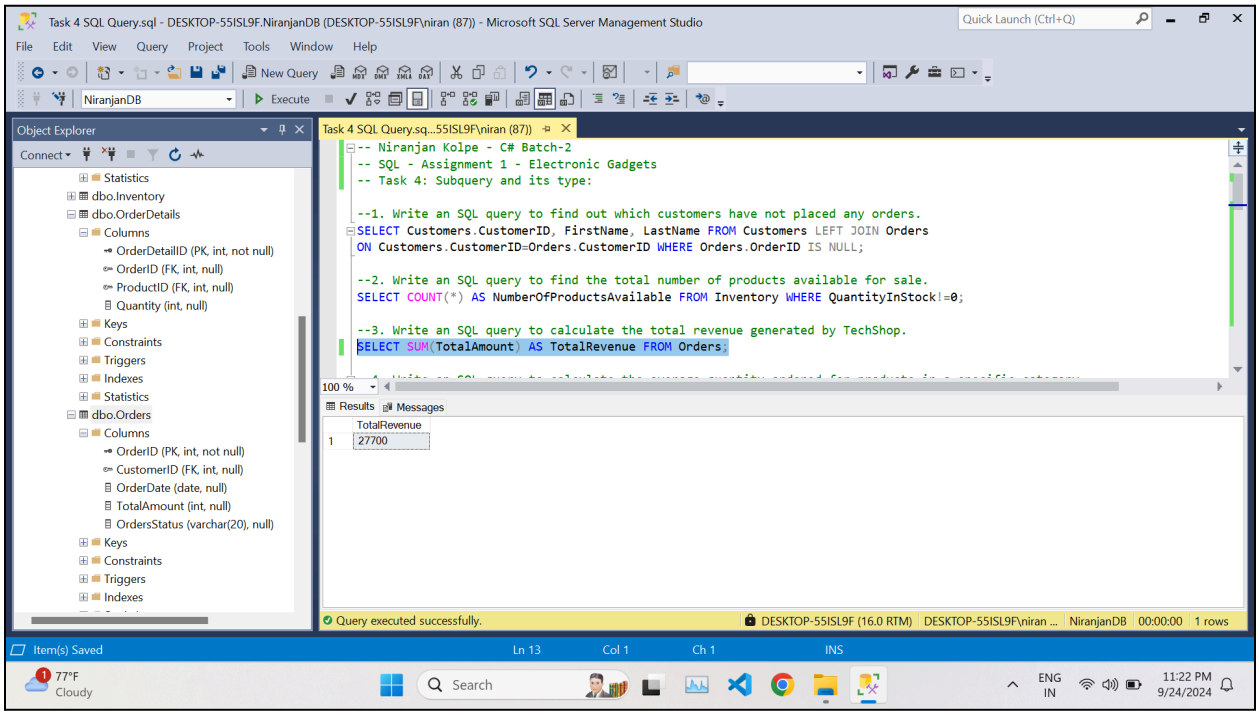
Output 1:



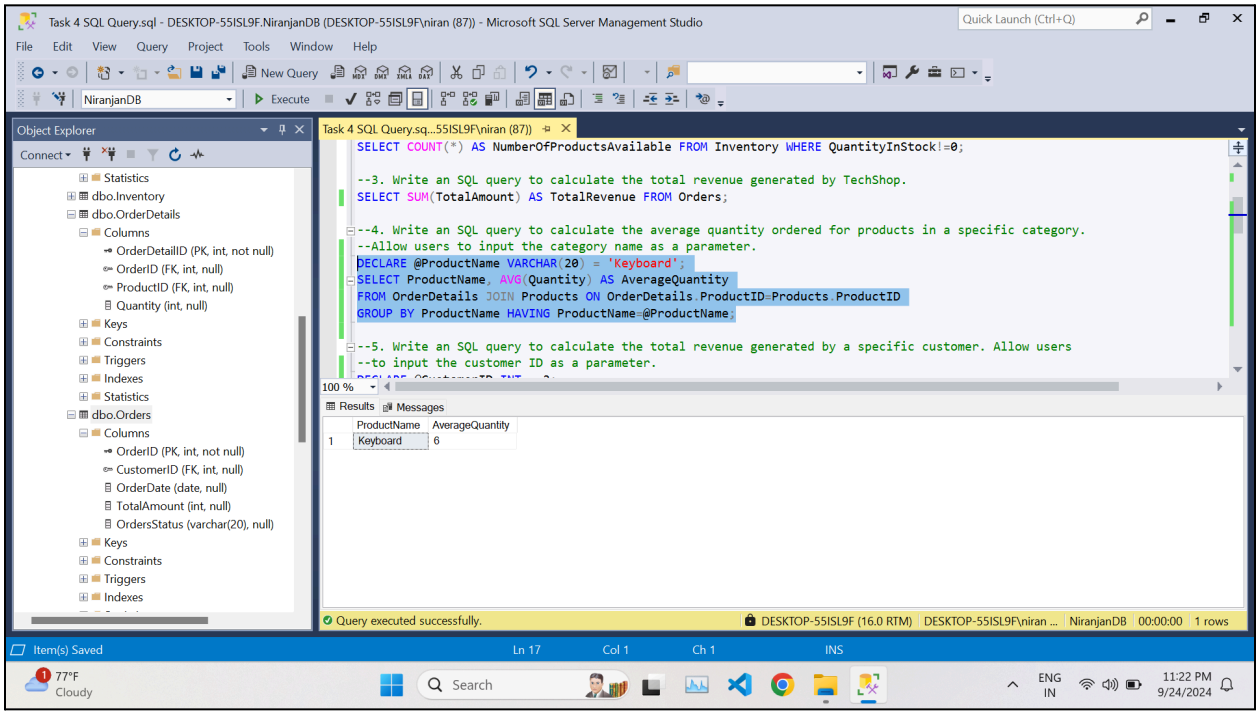
Output 2:



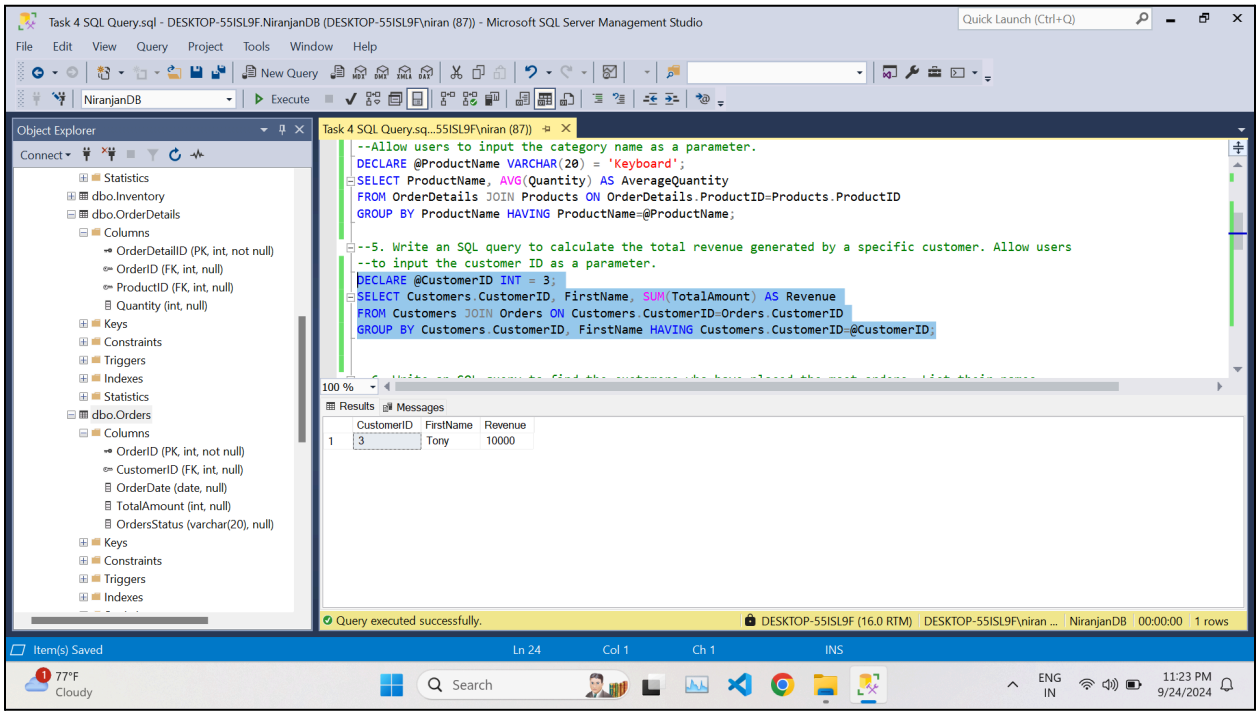
Output 3:



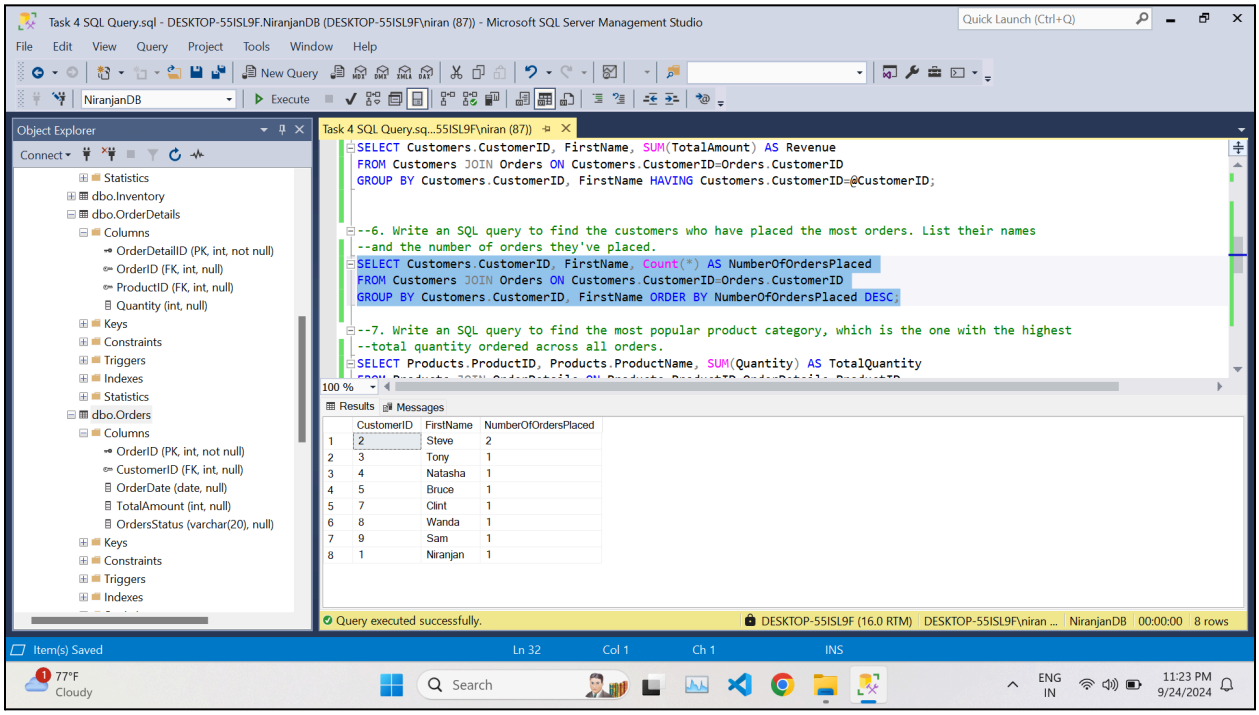
Output 4:



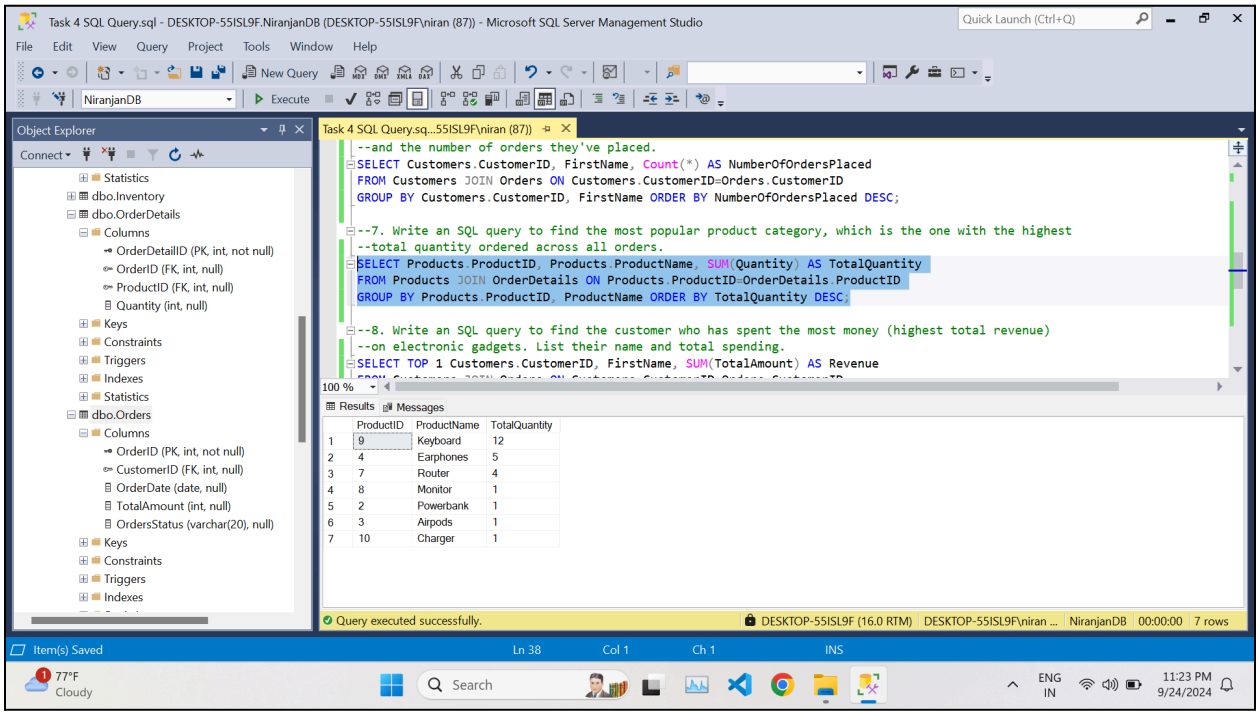
Output 5:



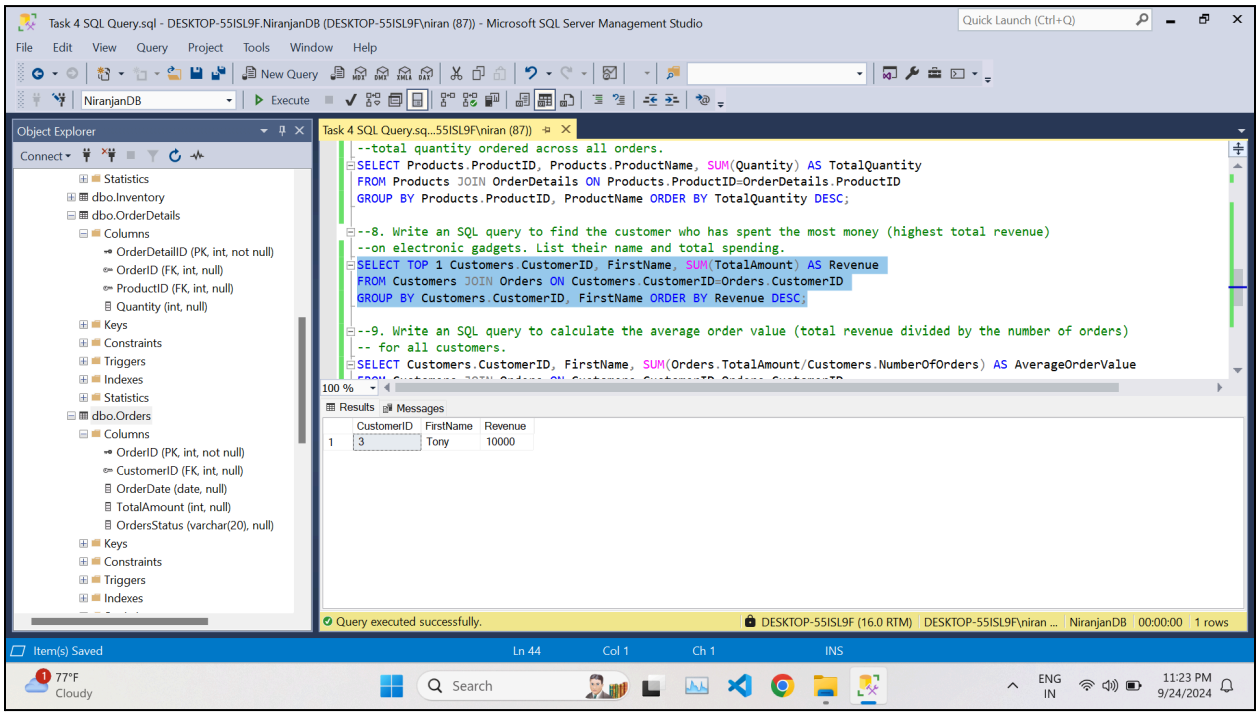
Output 6:



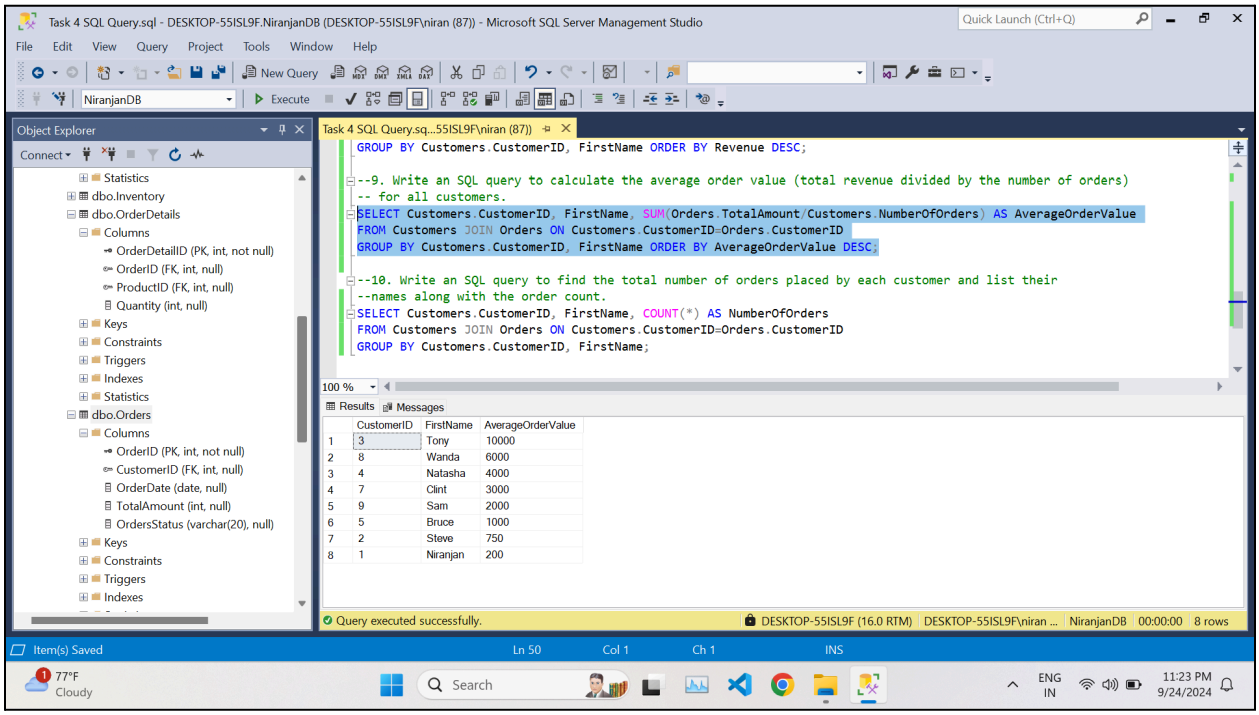
Output 7:



Output 8:



Output 9:



Output 10: