**DAY 6**

**ANSWERS**

DROP DATABASE IF EXISTS HexawareDB;

CREATE DATABASE HexawareDB;

USE HexawareDB;

-- Customers Table

CREATE TABLE Customers (

CustomerID INT PRIMARY KEY,

Name VARCHAR(100),

City VARCHAR(100)

);

-- Products Table

CREATE TABLE Products (

ProductID INT PRIMARY KEY,

ProductName VARCHAR(100),

Price DECIMAL(10,2)

);

-- Orders Table

CREATE TABLE Orders (

OrderID INT PRIMARY KEY,

CustomerID INT,

OrderDate DATE,

Amount DECIMAL(10,2),

FOREIGN KEY (CustomerID) REFERENCES Customers(CustomerID)

);

-- OrderDetails Table

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 Sample Data

INSERT INTO Customers VALUES

(1, 'Customer1', 'Kolkata'),

(2, 'Customer2', 'Delhi'),

(3, 'Customer3', 'Delhi'),

(4, 'Customer4', 'Delhi'),

(5, 'Customer5', 'Mumbai'),

(6, 'Customer6', 'Chennai'),

(7, 'Customer7', 'Kolkata'),

(8, 'Customer8', 'Delhi'),

(9, 'Customer9', 'Mumbai'),

(10, 'Customer10', 'Mumbai');

INSERT INTO Products VALUES

(1, 'Product1', 1208.09),

(2, 'Product2', 1071.75),

(3, 'Product3', 1121.13),

(4, 'Product4', 1635.29),

(5, 'Product5', 827.73),

(6, 'Product6', 500.00),

(7, 'Product7', 900.00),

(8, 'Product8', 1300.00),

(9, 'Product9', 1500.00),

(10, 'Product10', 999.99);

INSERT INTO Orders VALUES

(1, 10, '2025-04-13', 6740.06),

(2, 10, '2025-01-05', 3543.48),

(3, 4, '2025-02-23', 8883.38),

(4, 3, '2025-05-10', 3446.25),

(5, 7, '2025-01-25', 7519.55),

(6, 2, '2025-03-05', 5123.00),

(7, 1, '2025-03-10', 4589.00),

(8, 5, '2025-04-14', 2480.00),

(9, 6, '2025-06-01', 3200.00),

(10, 8, '2025-04-20', 5700.00);

INSERT INTO OrderDetails VALUES

(1, 1, 1, 8),

(2, 1, 6, 4),

(3, 2, 7, 1),

(4, 2, 6, 7),

(5, 3, 6, 3),

(6, 4, 4, 2),

(7, 5, 5, 1),

(8, 6, 1, 2),

(9, 7, 2, 6),

(10, 8, 3, 4),

(11, 9, 4, 5),

(12, 10, 10, 2);

**Part A – Subqueries (20 marks)**

1. **Write a query to find customers who have placed orders in every month of the current year.**

INSERT INTO Orders (OrderID, CustomerID, OrderDate, Amount) VALUES

(2001, 1, '2025-01-10', 1000.00),

(2002, 1, '2025-02-10', 1000.00),

(2003, 1, '2025-03-10', 1000.00),

(2004, 1, '2025-04-10', 1000.00),

(2005, 1, '2025-05-10', 1000.00),

(2006, 1, '2025-06-10', 1000.00),

(2007, 1, '2025-07-10', 1000.00),

(2008, 1, '2025-08-10', 1000.00),

(2009, 1, '2025-09-10', 1000.00),

(2010, 1, '2025-10-10', 1000.00),

(2011, 1, '2025-11-10', 1000.00),

(2012, 1, '2025-12-10', 1000.00);

SELECT c.Name

FROM Customers c

WHERE NOT EXISTS (

SELECT DISTINCT MONTH(d) FROM (

SELECT DATE(OrderDate) AS d FROM Orders

WHERE YEAR(OrderDate) = 2025

) AS months

WHERE NOT EXISTS (

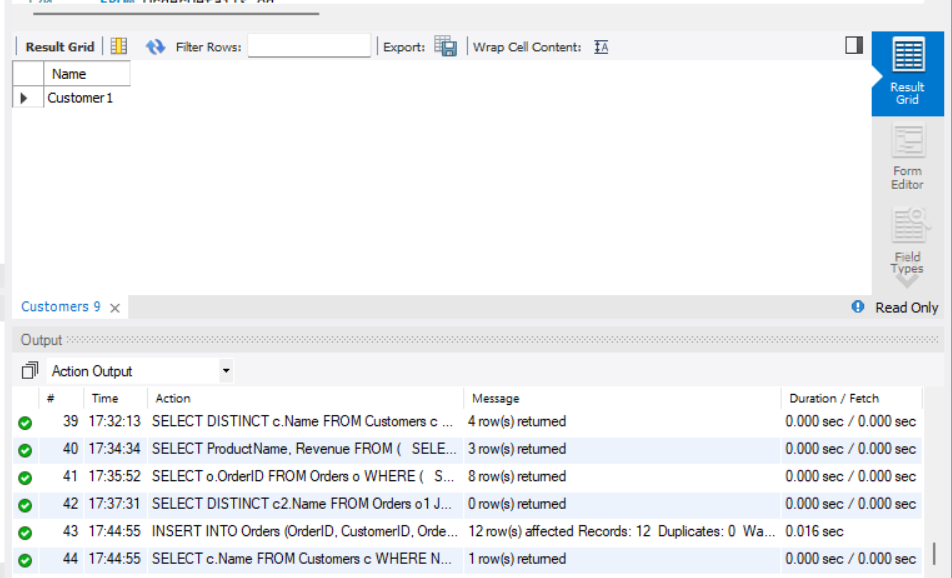
SELECT 1 FROM Orders o

WHERE o.CustomerID = c.CustomerID AND MONTH(o.OrderDate) = MONTH(d)

)

);

**OUTPUT**

****

1. **Retrieve the names of products that have been ordered more than the average quantity across all products.**

SELECT p.ProductName

FROM OrderDetails od

JOIN Products p ON od.ProductID = p.ProductID

GROUP BY p.ProductID

HAVING SUM(Quantity) > (

SELECT AVG(TotalQty) FROM (

SELECT SUM(Quantity) AS TotalQty

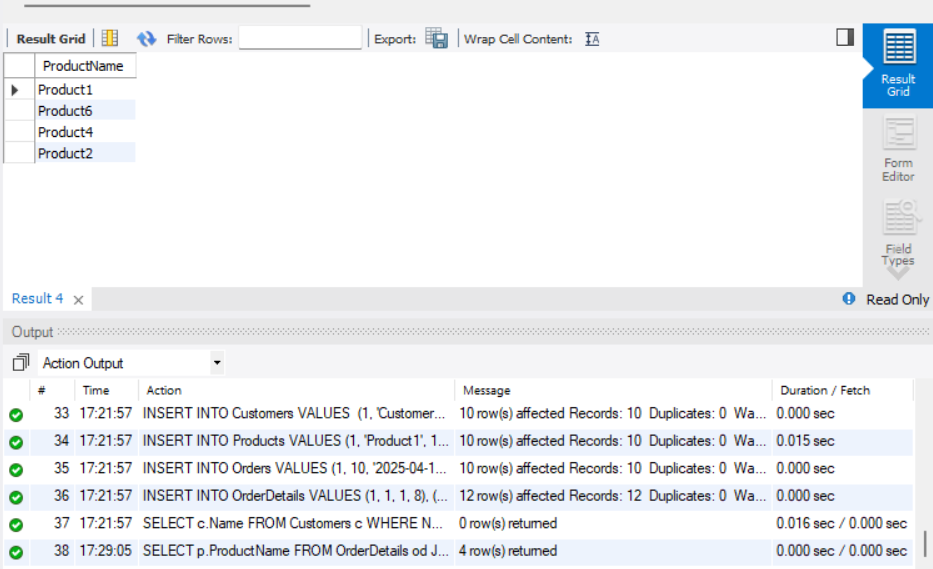
FROM OrderDetails

GROUP BY ProductID

) AS avgqty

);

**OUTPUT**

****

1. **Find customers who have never ordered a product priced above ₹1000.**

SELECT DISTINCT c.Name

FROM Customers c

WHERE CustomerID NOT IN (

SELECT DISTINCT o.CustomerID

FROM Orders o

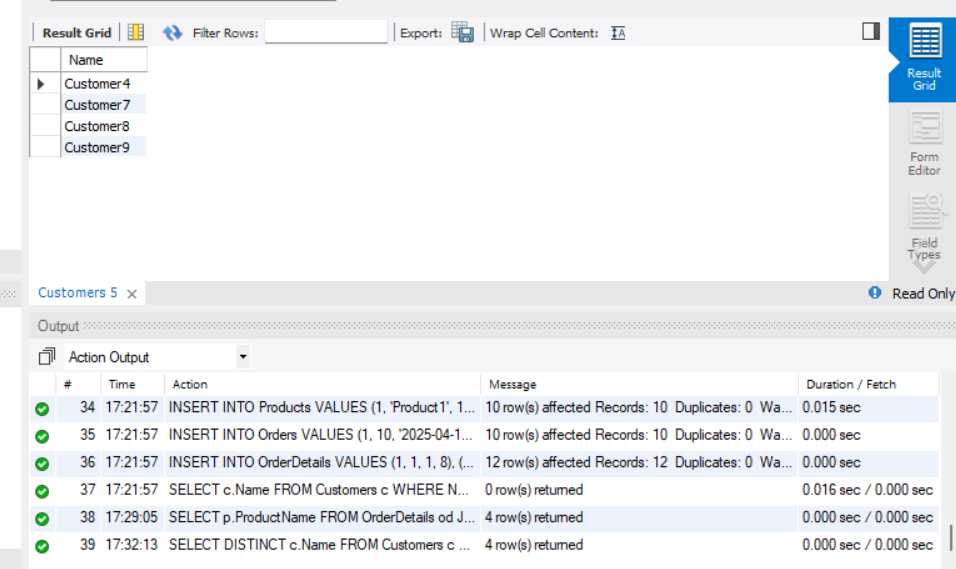
JOIN OrderDetails od ON o.OrderID = od.OrderID

JOIN Products p ON od.ProductID = p.ProductID

WHERE p.Price > 1000

);

**OUTPUT**

****

1. **List the top 3 products by total revenue using a subquery.**

SELECT ProductName, Revenue

FROM (

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

FROM OrderDetails od

JOIN Products p ON od.ProductID = p.ProductID

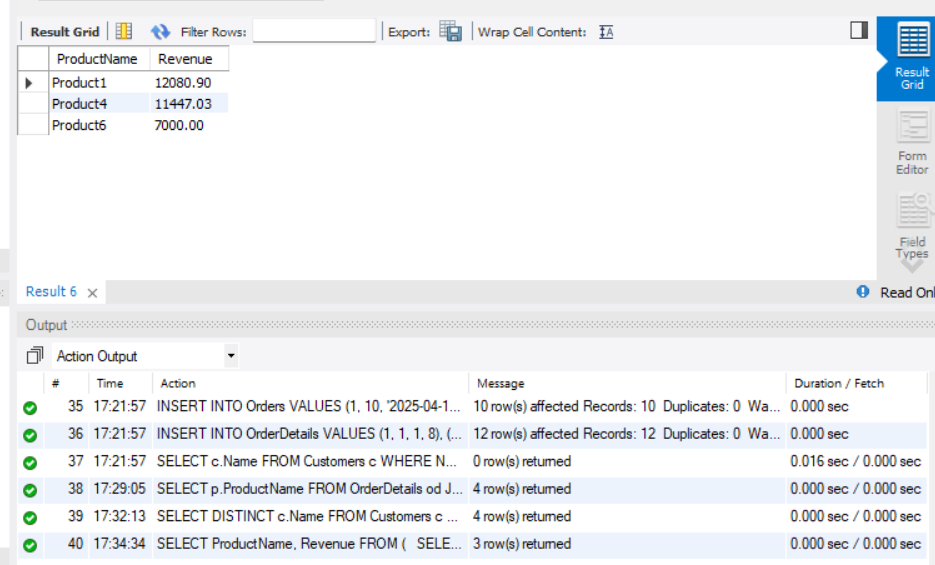
GROUP BY p.ProductID

ORDER BY Revenue DESC

LIMIT 3

) AS top3;

**OUTPUT**

****

1. **Find orders that contain only one product using a correlated subquery**.

SELECT o.OrderID

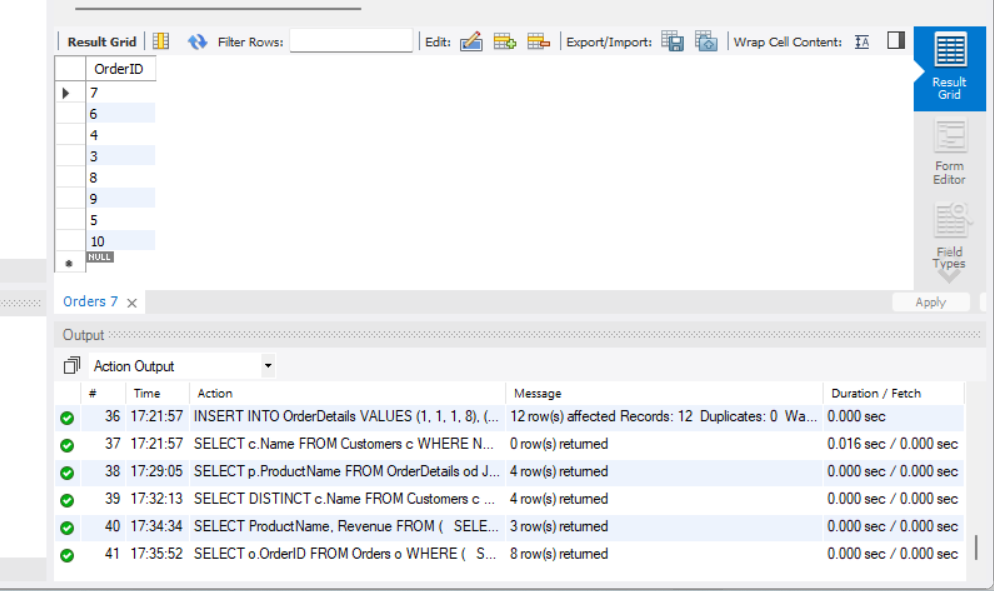
FROM Orders o

WHERE (

SELECT COUNT(\*) FROM OrderDetails od WHERE od.OrderID = o.OrderID

) = 1;

**OUTPUT:**



**Part B – Correlated & Nested Subqueries (25 marks)**

1. **Retrieve the names of customers who placed an order on the same date as 'John'.**

DELETE FROM Orders WHERE OrderID BETWEEN 3000 AND 3005;

INSERT INTO Orders (OrderID, CustomerID, OrderDate, Amount) VALUES

(3000, 1, '2025-07-15', 2000.00),

(3001, 2, '2025-07-15', 2500.00), -- same date

(3002, 3, '2025-08-01', 3000.00); -- different date

SELECT DISTINCT c2.Name

FROM Orders o1

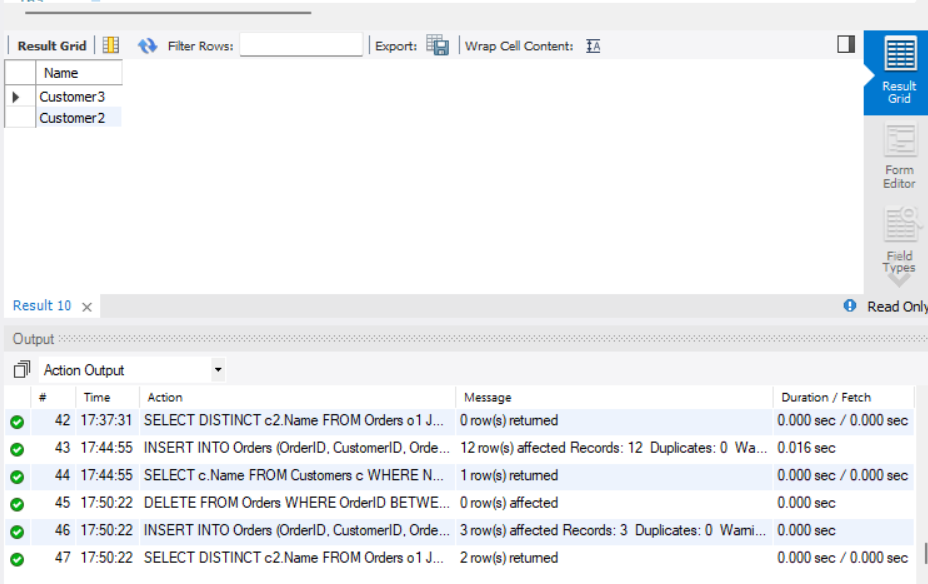
JOIN Customers c1 ON o1.CustomerID = c1.CustomerID

JOIN Orders o2 ON o1.OrderDate = o2.OrderDate

JOIN Customers c2 ON o2.CustomerID = c2.CustomerID

WHERE c1.Name = 'Customer1' AND c2.Name != 'Customer1';

**OUTPUT**

****

1. **Find the name of the customer who placed the most recent order.**

INSERT INTO Orders (OrderID, CustomerID, OrderDate, Amount) VALUES

(3010, 5, '2025-12-31', 4000.00); -- Most recent order by Customer5

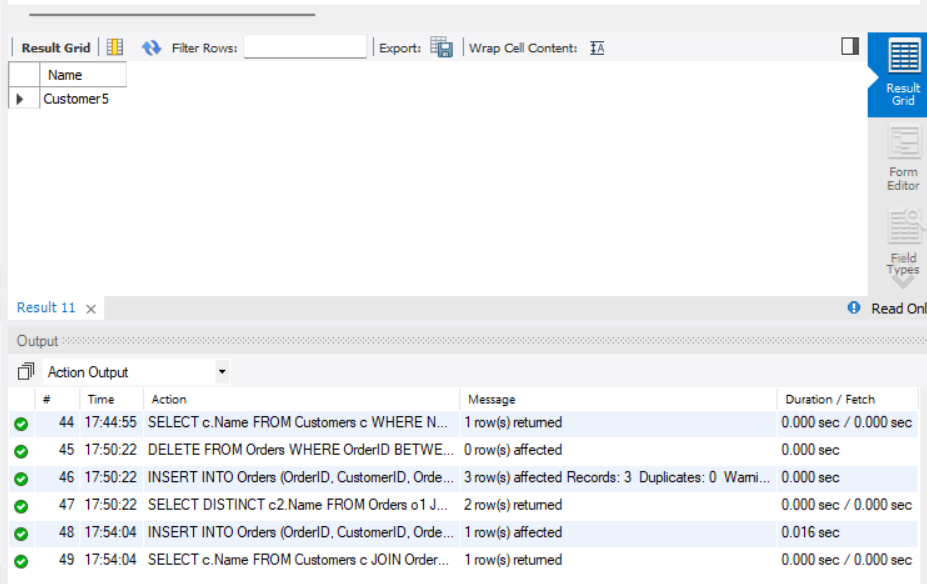
SELECT c.Name

FROM Customers c

JOIN Orders o ON c.CustomerID = o.CustomerID

WHERE o.OrderDate = (SELECT MAX(OrderDate) FROM Orders);

**OUTPUT**

****

1. **Write a query to find the product that has the second lowest price using a subquery.**

UPDATE Products SET Price = 100.00 WHERE ProductID = 1;

UPDATE Products SET Price = 150.00 WHERE ProductID = 2;

UPDATE Products SET Price = 500.00 WHERE ProductID = 3;

UPDATE Products SET Price = 1000.00 WHERE ProductID = 4;

SELECT ProductName

FROM Products

WHERE Price = (

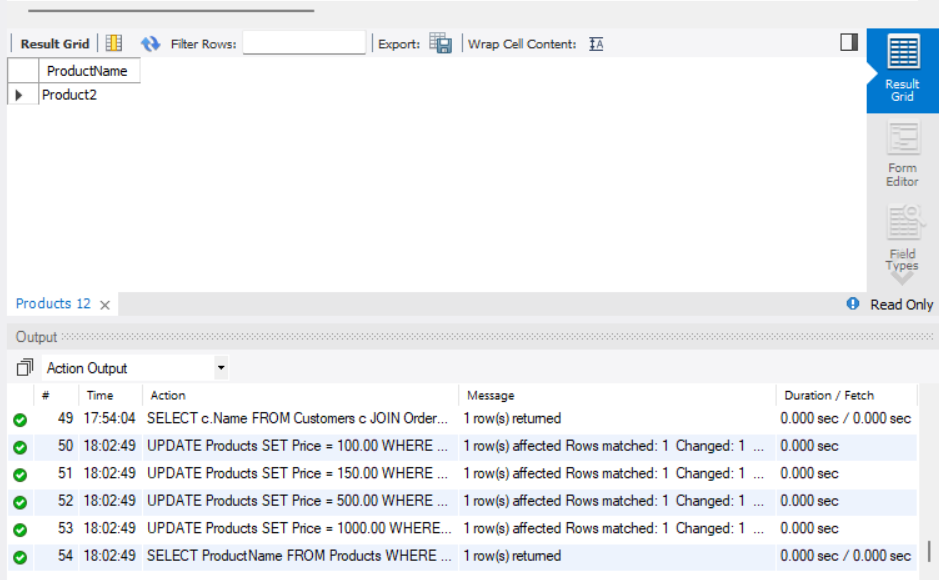
SELECT MIN(Price)

FROM Products

WHERE Price > (SELECT MIN(Price) FROM Products)

);

**OUTPUT**

****

1. **Display customer names who have spent more than double the average spending.**

DELETE FROM Orders WHERE OrderID BETWEEN 3020 AND 3025;

INSERT INTO Orders (OrderID, CustomerID, OrderDate, Amount) VALUES

(3020, 6, '2025-10-01', 1000.00), -- Low spender

(3021, 7, '2025-10-02', 1100.00), -- Low spender

(3022, 8, '2025-10-03', 1200.00), -- Low spender

(3023, 9, '2025-10-04', 1300.00), -- Low spender

(3024, 10, '2025-10-05', 9000.00); -- High spender

SELECT c.Name

FROM Customers c

JOIN Orders o ON c.CustomerID = o.CustomerID

GROUP BY c.CustomerID

HAVING SUM(o.Amount) > 2 \* (

SELECT AVG(TotalSpent)

FROM (

SELECT SUM(Amount) AS TotalSpent

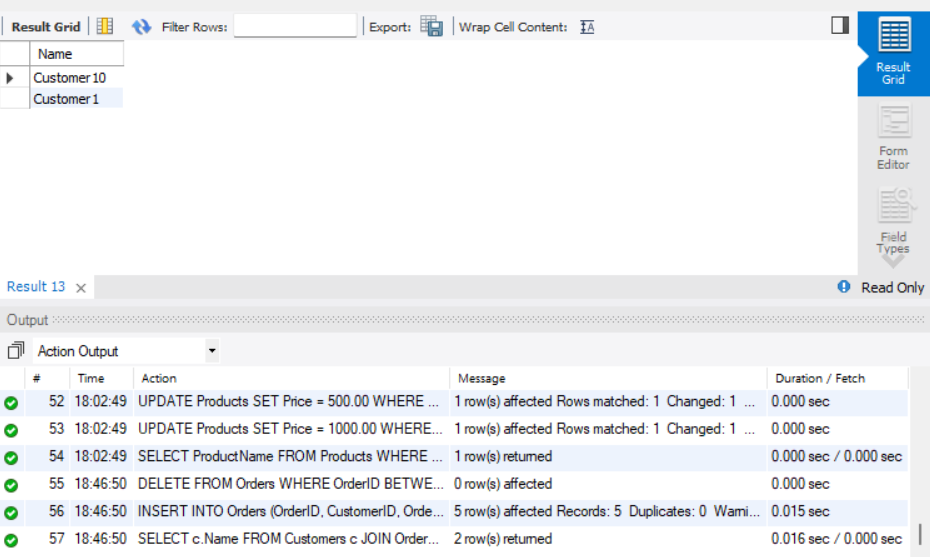
FROM Orders

GROUP BY CustomerID

) AS spending

);

**OUTPUT**

****

1. **List customers whose total order amount is more than the total order amount of any customer from 'Delhi'.**

DELETE FROM Orders WHERE OrderID BETWEEN 3030 AND 3035;

INSERT INTO Orders (OrderID, CustomerID, OrderDate, Amount) VALUES

(3030, 2, '2025-10-10', 1000.00),

(3031, 3, '2025-10-11', 800.00),

(3032, 4, '2025-10-12', 900.00),

(3033, 8, '2025-10-13', 1100.00);

INSERT INTO Orders (OrderID, CustomerID, OrderDate, Amount) VALUES

(3034, 5, '2025-10-14', 6000.00);

SELECT c.Name

FROM Customers c

JOIN Orders o ON c.CustomerID = o.CustomerID

GROUP BY c.CustomerID

HAVING SUM(o.Amount) > ALL (

SELECT SUM(o2.Amount)

FROM Orders o2

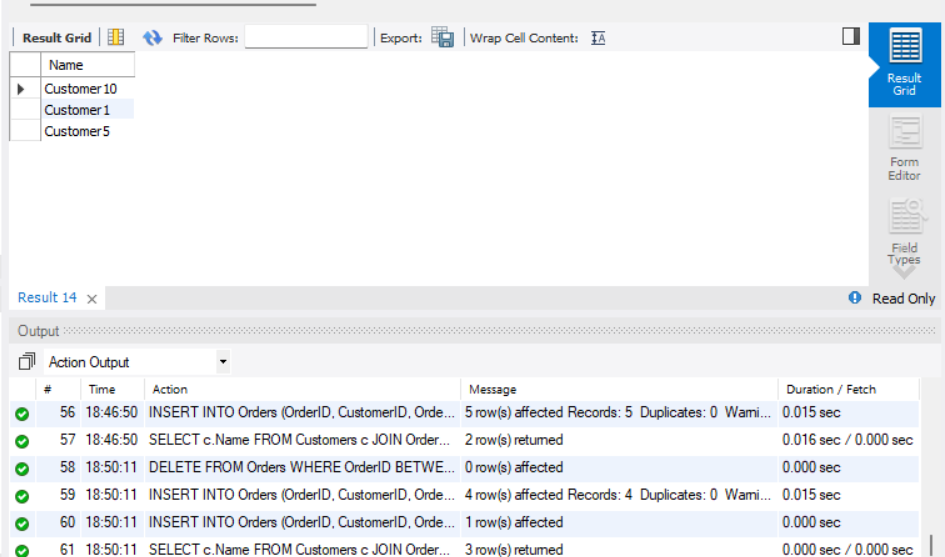
JOIN Customers c2 ON o2.CustomerID = c2.CustomerID

WHERE c2.City = 'Delhi'

GROUP BY c2.CustomerID

);

**OUTPUT**

****

**Part C – Join + Subquery Mix (30 marks)**

1. **Use a correlated subquery to find customers who have placed more orders than the average number of orders placed by all customers.**

DELETE FROM Orders WHERE OrderID BETWEEN 3040 AND 3050;

INSERT INTO Orders (OrderID, CustomerID, OrderDate, Amount) VALUES

(3040, 1, '2025-10-20', 1000.00),

(3041, 1, '2025-10-21', 1100.00),

(3042, 1, '2025-10-22', 1200.00),

(3043, 1, '2025-10-23', 1300.00),

(3044, 1, '2025-10-24', 1400.00);

INSERT INTO Orders (OrderID, CustomerID, OrderDate, Amount) VALUES

(3045, 2, '2025-10-25', 1000.00);

INSERT INTO Orders (OrderID, CustomerID, OrderDate, Amount) VALUES

(3046, 3, '2025-10-26', 1500.00),

(3047, 3, '2025-10-27', 1600.00);

SELECT c.Name

FROM Customers c

WHERE (

SELECT COUNT(\*)

FROM Orders o

WHERE o.CustomerID = c.CustomerID

) > (

SELECT AVG(order\_count)

FROM (

SELECT COUNT(\*) AS order\_count

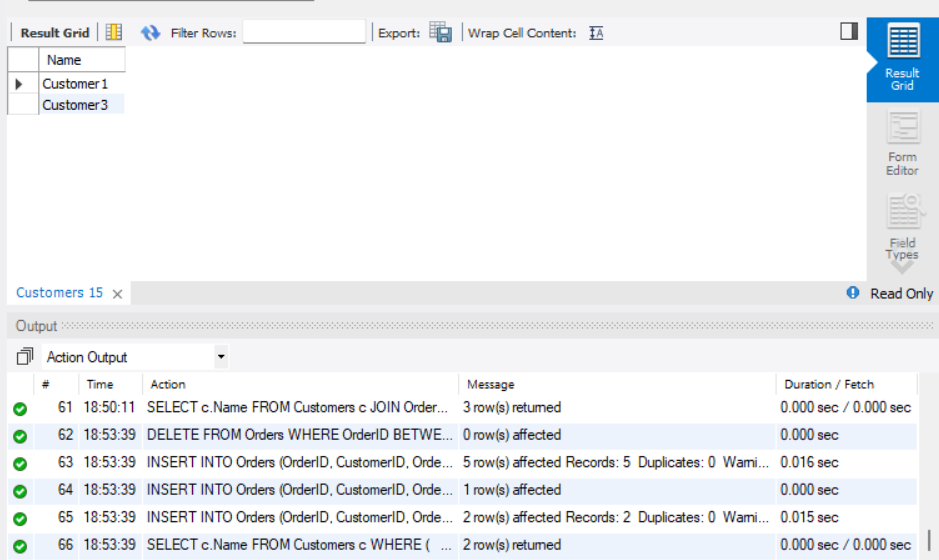
FROM Orders

GROUP BY CustomerID

) AS avg\_orders

);

**OUTPUT**

****

1. **Find all products whose total sales quantity is higher than the average total quantity sold per product.**

DELETE FROM OrderDetails WHERE OrderDetailID BETWEEN 3050 AND 3060;

INSERT INTO OrderDetails VALUES (3050, 1, 1, 10); -- High

INSERT INTO OrderDetails VALUES (3051, 2, 2, 5); -- Medium

INSERT INTO OrderDetails VALUES (3052, 3, 3, 3); -- Low

INSERT INTO OrderDetails VALUES (3053, 4, 4, 2); -- Very Low

INSERT INTO OrderDetails VALUES (3054, 5, 5, 7); -- Medium-High

SELECT p.ProductName

FROM OrderDetails od

JOIN Products p ON od.ProductID = p.ProductID

GROUP BY p.ProductID

HAVING SUM(Quantity) > (

SELECT AVG(total\_qty)

FROM (

SELECT SUM(Quantity) AS total\_qty

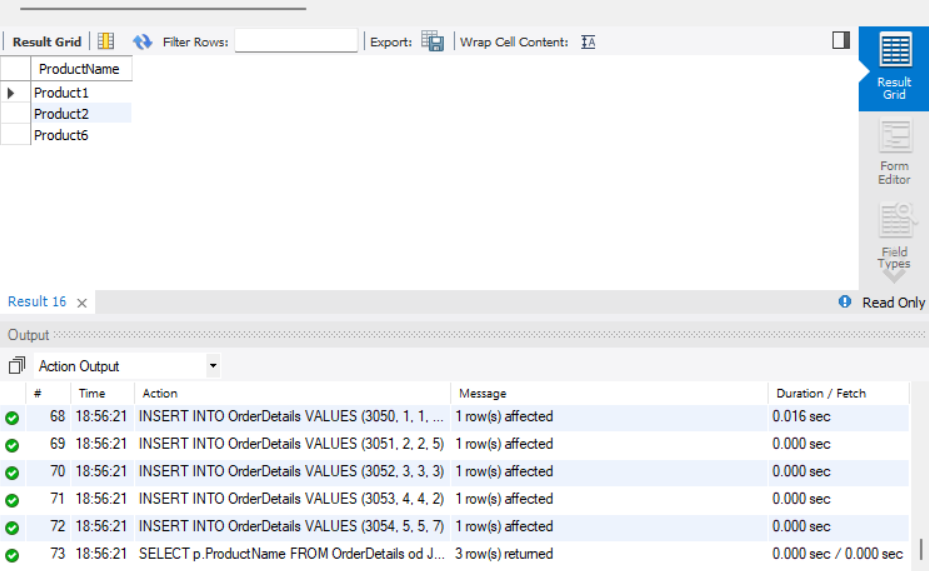
FROM OrderDetails

GROUP BY ProductID

) AS avg\_qty

);

**OUTPUT**

****

1. **Get customers who have ordered at least one product that no one else has ordered.**

DELETE FROM OrderDetails WHERE OrderDetailID BETWEEN 3060 AND 3070;

INSERT INTO OrderDetails VALUES (3060, 1, 6, 1);

INSERT INTO OrderDetails VALUES (3061, 2, 7, 2);

INSERT INTO OrderDetails VALUES (3062, 3, 7, 2);

SELECT DISTINCT c.Name

FROM Customers c

JOIN Orders o ON c.CustomerID = o.CustomerID

JOIN OrderDetails od ON o.OrderID = od.OrderID

WHERE od.ProductID IN (

SELECT ProductID

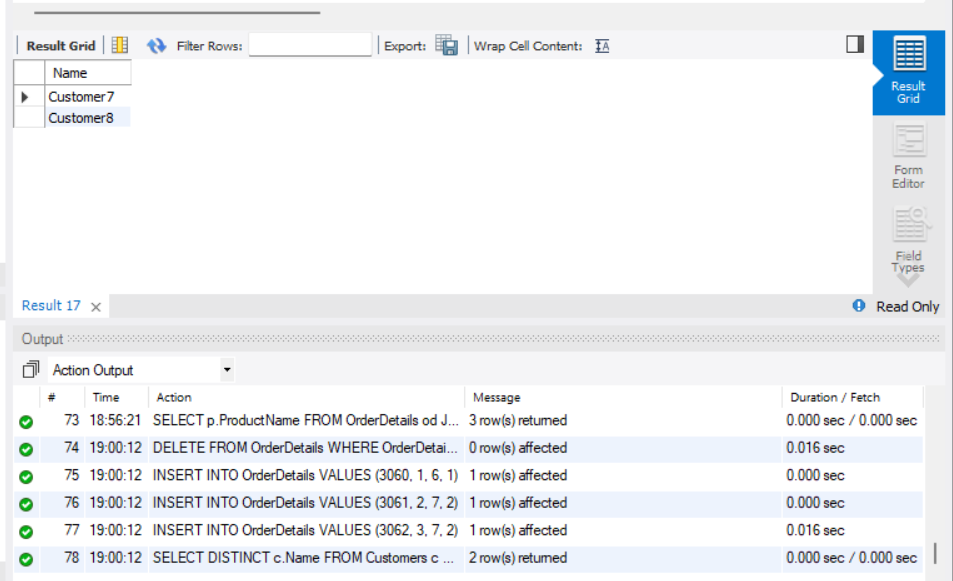
FROM OrderDetails

GROUP BY ProductID

HAVING COUNT(DISTINCT OrderID) = 1

);

**OUTPUT**

****

1. **Retrieve all orders where the total order amount is equal to the maximum order amount for that customer.**

DELETE FROM Orders WHERE CustomerID = 4 AND OrderID BETWEEN 3070 AND 3072;

INSERT INTO Orders (OrderID, CustomerID, OrderDate, Amount) VALUES

(3070, 4, '2025-11-01', 2000.00),

(3071, 4, '2025-11-02', 2500.00), -- max

(3072, 4, '2025-11-03', 1800.00);

SELECT o.OrderID, c.Name, o.Amount

FROM Orders o

JOIN Customers c ON o.CustomerID = c.CustomerID

WHERE o.Amount = (

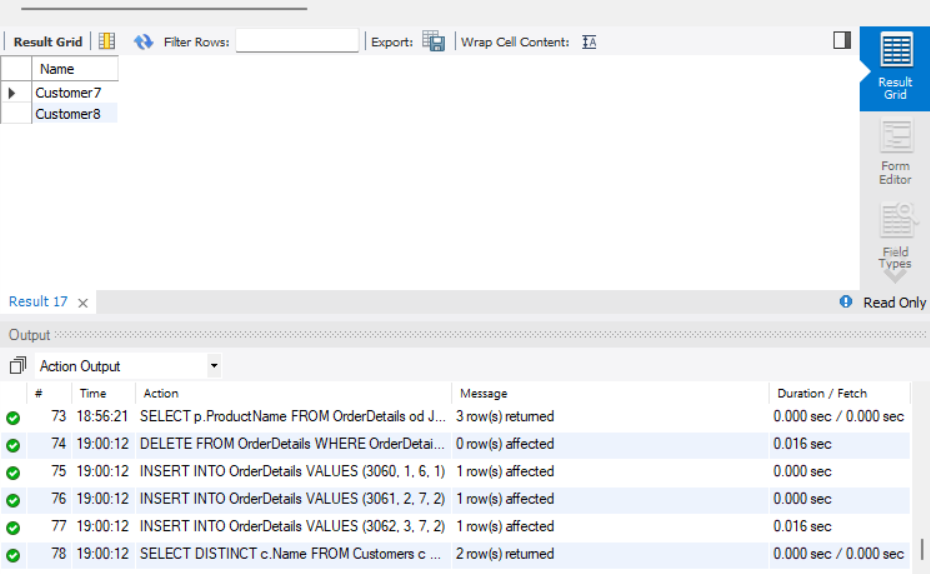
SELECT MAX(o2.Amount)

FROM Orders o2

WHERE o2.CustomerID = o.CustomerID

);

**OUTPUT**

****

1. **Write a query to list customers who have never placed an order with a quantity greater than 5**.

DELETE FROM OrderDetails WHERE OrderDetailID BETWEEN 3080 AND 3085;

INSERT INTO OrderDetails VALUES (3080, 9, 1, 3);

INSERT INTO OrderDetails VALUES (3081, 9, 2, 4);

INSERT INTO OrderDetails VALUES (3082, 5, 3, 6);

SELECT DISTINCT c.Name

FROM Customers c

WHERE c.CustomerID NOT IN (

SELECT o.CustomerID

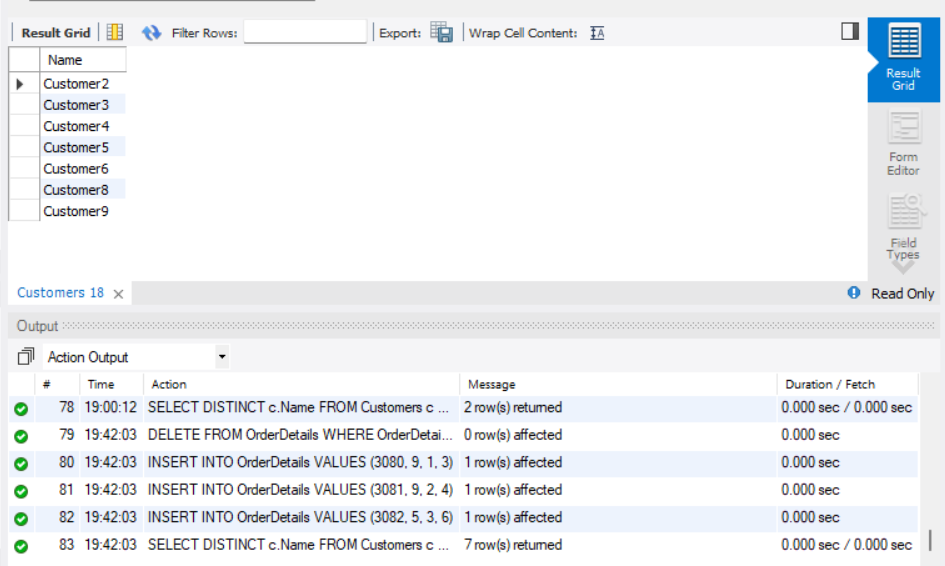
FROM Orders o

JOIN OrderDetails od ON o.OrderID = od.OrderID

WHERE od.Quantity > 5

);

**OUTPUT**

****

**Part D – Joins & Set Operations (25 marks)**

1. **Use a subquery to list the top 5 customers by total spending.**

DELETE FROM Orders WHERE OrderID BETWEEN 3090 AND 3099;

INSERT INTO Orders (OrderID, CustomerID, OrderDate, Amount) VALUES

(3090, 1, '2025-11-10', 2000.00),

(3091, 2, '2025-11-11', 1000.00),

(3092, 3, '2025-11-12', 1500.00),

(3093, 4, '2025-11-13', 2500.00),

(3094, 5, '2025-11-14', 3500.00),

(3095, 6, '2025-11-15', 4000.00),

(3096, 7, '2025-11-16', 300.00),

(3097, 8, '2025-11-17', 500.00),

(3098, 9, '2025-11-18', 4500.00),

(3099, 10, '2025-11-19', 5500.00);

SELECT c.Name, SUM(o.Amount) AS TotalSpent

FROM Customers c

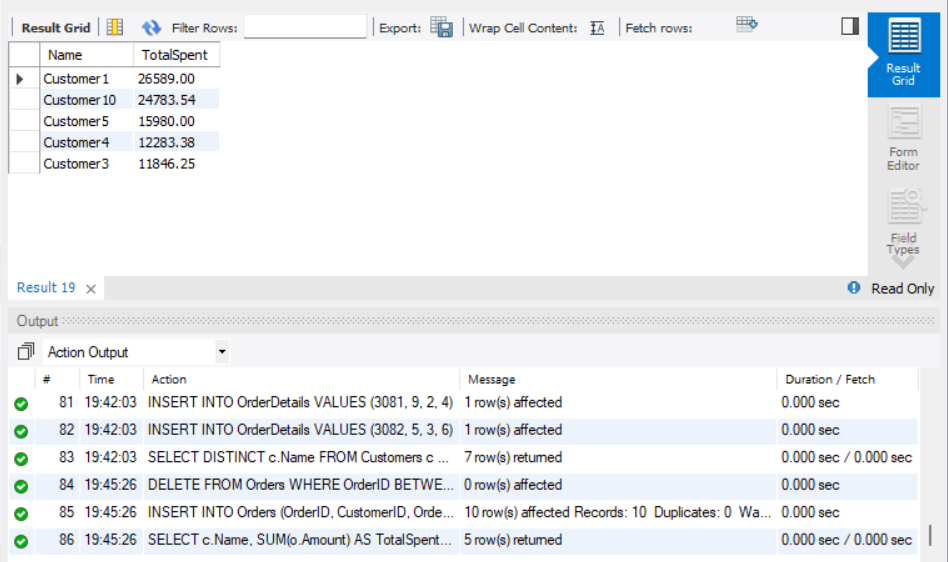
JOIN Orders o ON c.CustomerID = o.CustomerID

GROUP BY c.CustomerID

ORDER BY TotalSpent DESC

LIMIT 5;

**OUTPUT**

****

1. **Find all customers who have only ordered one unique product using subqueries.**

DELETE FROM OrderDetails WHERE OrderDetailID BETWEEN 3100 AND 3110;

INSERT INTO OrderDetails VALUES (3100, 1, 1, 2);

INSERT INTO OrderDetails VALUES (3101, 1, 1, 3);

INSERT INTO OrderDetails VALUES (3102, 2, 2, 1);

INSERT INTO OrderDetails VALUES (3103, 2, 3, 1);

INSERT INTO OrderDetails VALUES (3104, 3, 4, 5);

SELECT c.Name

FROM Customers c

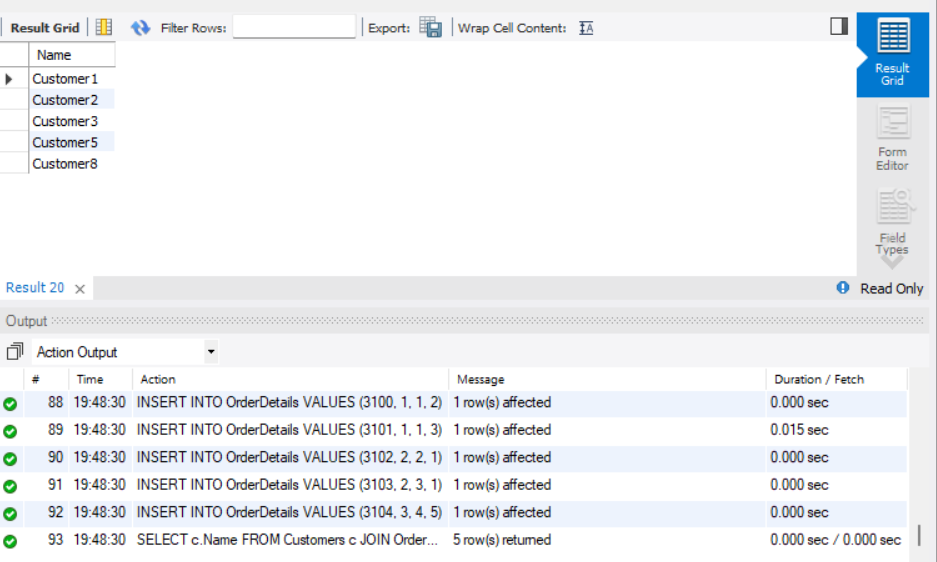
JOIN Orders o ON c.CustomerID = o.CustomerID

JOIN OrderDetails od ON o.OrderID = od.OrderID

GROUP BY c.CustomerID

HAVING COUNT(DISTINCT od.ProductID) = 1;

**OUTPUT**

****

1. **List all orders where the amount is not in the top 10 highest order amounts.**

DELETE FROM Orders WHERE OrderID BETWEEN 3200 AND 3215;

INSERT INTO Orders (OrderID, CustomerID, OrderDate, Amount) VALUES

(3200, 1, '2025-12-01', 100.00),

(3201, 2, '2025-12-02', 200.00),

(3202, 3, '2025-12-03', 300.00),

(3203, 4, '2025-12-04', 400.00),

(3204, 5, '2025-12-05', 500.00),

(3205, 6, '2025-12-06', 600.00),

(3206, 7, '2025-12-07', 700.00),

(3207, 8, '2025-12-08', 800.00),

(3208, 9, '2025-12-09', 900.00),

(3209, 10, '2025-12-10', 1000.00),

(3210, 1, '2025-12-11', 1100.00),

(3211, 2, '2025-12-12', 1200.00),

(3212, 3, '2025-12-13', 1300.00);

SELECT o.\*

FROM Orders o

LEFT JOIN (

SELECT Amount

FROM Orders

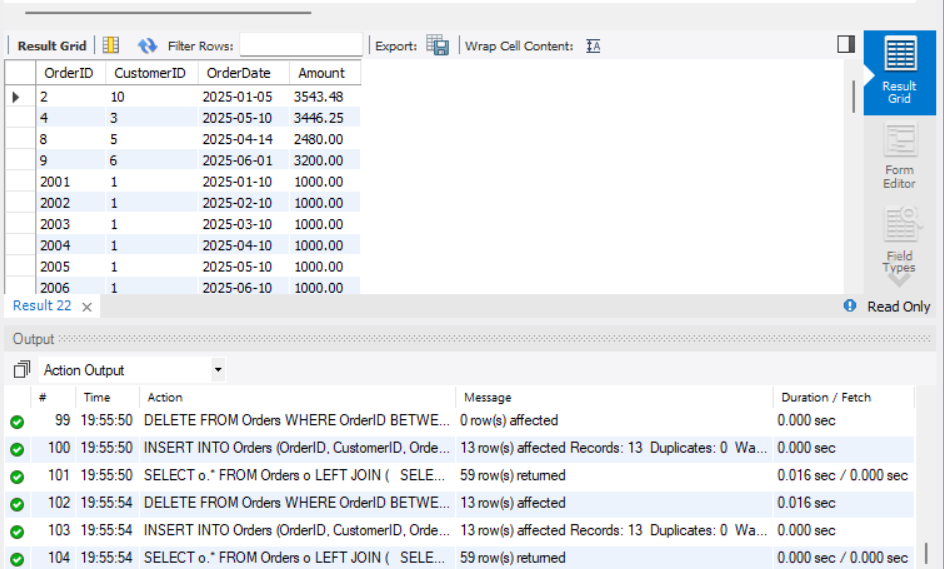
ORDER BY Amount DESC

LIMIT 10

) AS top10 ON o.Amount = top10.Amount

WHERE top10.Amount IS NULL;

**OUTPUT**

****

1. **Retrieve customer names who placed an order in the last 7 days but not in the previous 30 days before that.**

DELETE FROM Customers WHERE CustomerID BETWEEN 101 AND 104;

DELETE FROM Orders WHERE OrderID BETWEEN 9001 AND 9006;

INSERT INTO Customers (CustomerID, Name, City) VALUES

(101, 'Customer\_RecentOnly', 'Chennai'),

(102, 'Customer\_BothPeriods', 'Mumbai'),

(103, 'Customer\_OnlyEarlier', 'Delhi'),

(104, 'Customer\_OnlyOld', 'Kolkata');

INSERT INTO Orders VALUES (9001, 101, '2025-12-30', 999.00);

INSERT INTO Orders VALUES (9002, 102, '2025-12-28', 999.00);

INSERT INTO Orders VALUES (9003, 102, '2025-12-10', 500.00);

INSERT INTO Orders VALUES (9004, 103, '2025-12-15', 300.00);

INSERT INTO Orders VALUES (9005, 104, '2025-11-20', 400.00);

SELECT DISTINCT c.Name

FROM Customers c

JOIN Orders o ON c.CustomerID = o.CustomerID

WHERE o.OrderDate BETWEEN '2025-12-24' AND '2025-12-31'

AND c.CustomerID NOT IN (

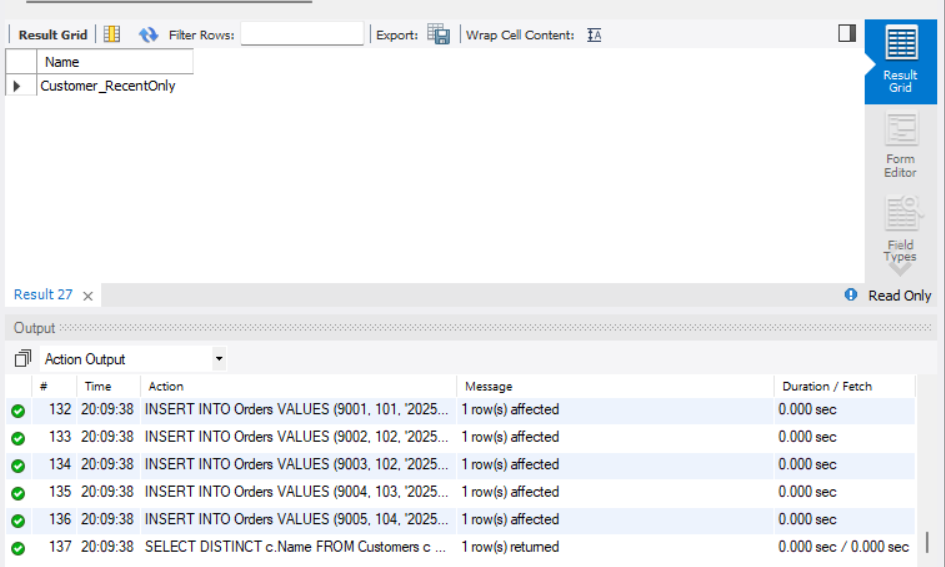
SELECT CustomerID

FROM Orders

WHERE OrderDate BETWEEN '2025-11-24' AND '2025-12-23'

);

**OUTPUT**

****

1. **Write a query to list all products ordered in the highest number of distinct orders.**

DELETE FROM OrderDetails WHERE OrderDetailID BETWEEN 9501 AND 9510;

INSERT INTO OrderDetails VALUES (9501, 1, 1, 2); -- OrderID 1

INSERT INTO OrderDetails VALUES (9502, 2, 1, 3); -- OrderID 2

INSERT INTO OrderDetails VALUES (9503, 3, 1, 1); -- OrderID 3

INSERT INTO OrderDetails VALUES (9504, 4, 2, 1);

INSERT INTO OrderDetails VALUES (9505, 5, 2, 1);

INSERT INTO OrderDetails VALUES (9506, 6, 3, 1);

SELECT p.ProductName

FROM OrderDetails od

JOIN Products p ON od.ProductID = p.ProductID

GROUP BY p.ProductID

HAVING COUNT(DISTINCT od.OrderID) = (

SELECT MAX(order\_count)

FROM (

SELECT COUNT(DISTINCT OrderID) AS order\_count

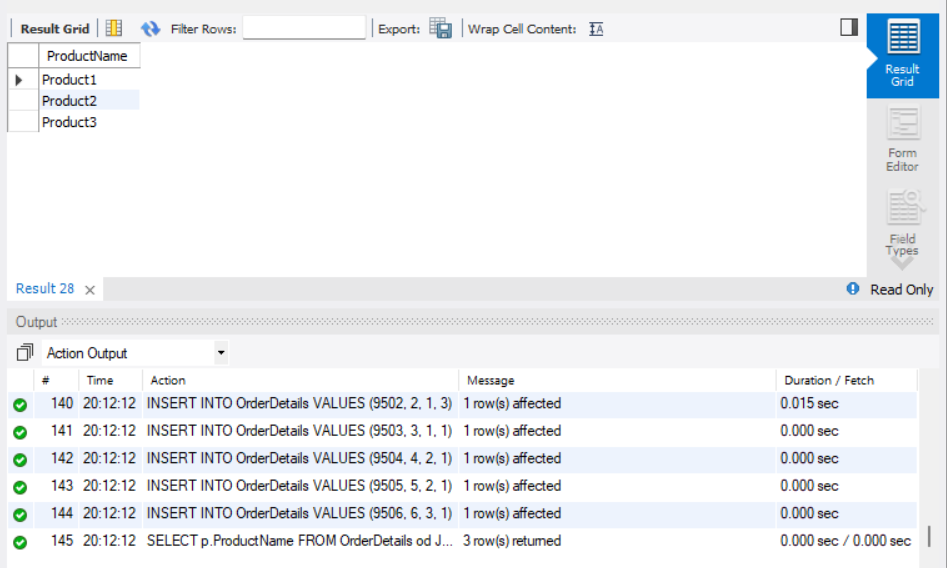
FROM OrderDetails

GROUP BY ProductID

) AS counts

);

**OUTPUT**

****