**1: Ranking and Window Functions**

Created a random database for the exercise

OUTPUT:-

1. Use ROW\_NUMBER() to assign a unique rank within each category.

SELECT

Category,

ProductName,

Price,

ROW\_NUMBER() OVER (PARTITION BY Category ORDER BY Price DESC) AS RowNum

FROM Products;

****

2. Use RANK() and DENSE\_RANK() to compare how ties are handled.

SELECT

Category,

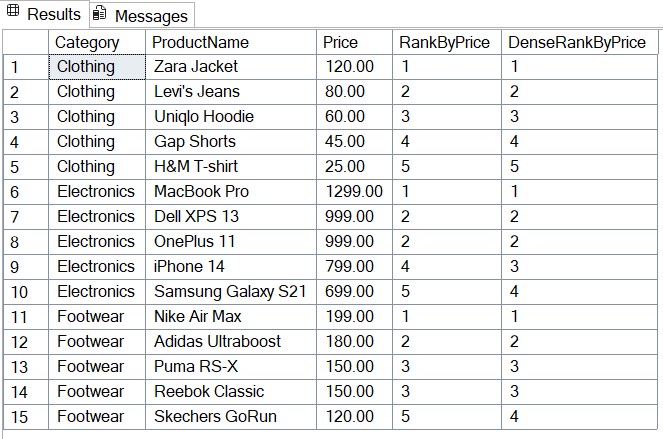
ProductName,

Price,

RANK() OVER (PARTITION BY Category ORDER BY Price DESC) AS RankByPrice,

DENSE\_RANK() OVER (PARTITION BY Category ORDER BY Price DESC) AS DenseRankByPrice

FROM Products;

****

3. Use PARTITION BY Category and ORDER BY Price DESC.

WITH RankedProducts AS (

SELECT

Category,

ProductName,

Price,

ROW\_NUMBER() OVER (PARTITION BY Category ORDER BY Price DESC) AS RowNum

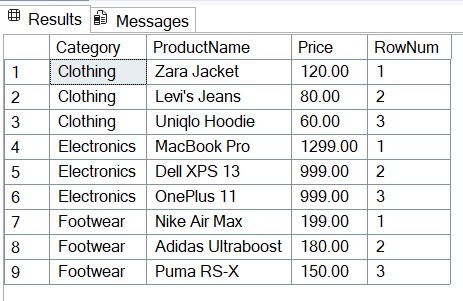
FROM Products

)

SELECT \*

FROM RankedProducts

WHERE RowNum <= 3;

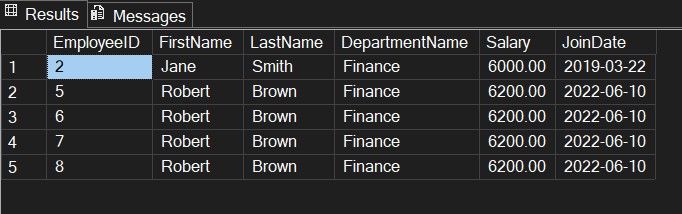
****

**4. Stored procedure**

**1.**

Query

EXEC sp\_GetEmployeesByDepartment @DeptID = 2;

****

Query

DECLARE @DeptID INT;

SET @DeptID = 3;

SELECT

E.EmployeeID,

E.FirstName,

E.LastName,

D.DepartmentName,

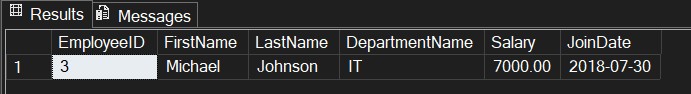
E.Salary,

E.JoinDate

FROM Employees E

INNER JOIN Departments D ON E.DepartmentID = D.DepartmentID

WHERE E.DepartmentID = @DeptID;

****

**3.**

Query

EXEC sp\_InsertEmployee

@FirstName = 'Alice',

@LastName = 'Walker',

@DepartmentID = 2,

@Salary = 6200.00,

@JoinDate = '2023-05-10';



**5: Return Data from a Stored Procedure**

Query to Count Employees:-

CREATE PROCEDURE sp\_GetEmployeeCountByDepartment

@DeptID INT

AS

BEGIN

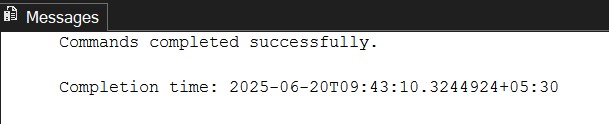
SELECT

COUNT(\*) AS TotalEmployees

FROM Employees

WHERE DepartmentID = @DeptID;

END;

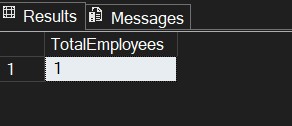


OUTPUT:-

Query

EXEC sp\_GetEmployeeCountByDepartment @DeptID = 3;

Snap of Output:-

****