1. Advanced Concepts

**Exercise 1: Ranking and Window Functions**

Goal: Use ROW\_NUMBER(), RANK(), DENSE\_RANK(), OVER(), and PARTITION BY.

Scenario:

Find the top 3 most expensive products in each category using different ranking functions.

Steps:

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

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

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

**QUERIES:**

**STEP 1: Create Schema**

CREATE TABLE Categories (

CategoryID INT PRIMARY KEY,

CategoryName VARCHAR(100)

);

CREATE TABLE Products (

ProductID INT PRIMARY KEY,

ProductName VARCHAR(100),

Price DECIMAL(10, 2),

CategoryID INT FOREIGN KEY REFERENCES Categories(CategoryID)

);

**STEP 2: Insert Sample Data**

INSERT INTO Categories (CategoryID, CategoryName) VALUES

(1, 'Electronics'),

(2, 'Books'),

(3, 'Clothing');

INSERT INTO Products (ProductID, ProductName, Price, CategoryID) VALUES

(1, 'Smartphone', 800.00, 1),

(2, 'Laptop', 1200.00, 1),

(3, 'Tablet', 600.00, 1),

(4, 'Smartwatch', 600.00, 1),

(5, 'Novel A', 15.00, 2),

(6, 'Novel B', 25.00, 2),

(7, 'Collector Edition', 25.00, 2),

(8, 'T-shirt', 20.00, 3),

(9, 'Jacket', 60.00, 3),

(10, 'Sneakers', 60.00, 3),

(11, 'Cap', 10.00, 3);

**STEP 3: Use Ranking Functions (Main query)**

SELECT

c.CategoryName,

p.ProductName,

p.Price,

ROW\_NUMBER() OVER (PARTITION BY c.CategoryID ORDER BY p.Price DESC) AS RowNum,

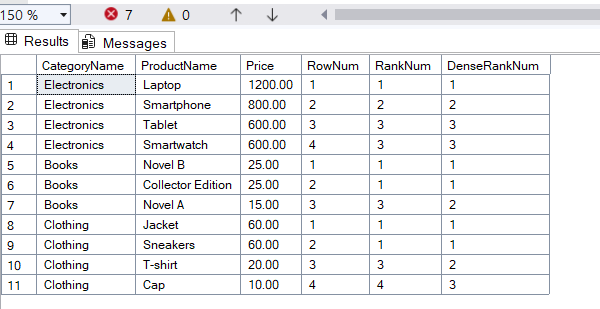
RANK() OVER (PARTITION BY c.CategoryID ORDER BY p.Price DESC) AS RankNum,

DENSE\_RANK() OVER (PARTITION BY c.CategoryID ORDER BY p.Price DESC) AS DenseRankNum

FROM Products p

JOIN Categories c ON p.CategoryID = c.CategoryID;

**OUTPUT:**

****

4. Stored Procedures

**Exercise 1: Create a Stored Procedure**

**QUERIES:**

**Creating Table:**

CREATE TABLE Employees (

EmployeeID INT IDENTITY(1,1) PRIMARY KEY,

FirstName VARCHAR(50),

LastName VARCHAR(50),

DepartmentID INT,

Salary DECIMAL(10,2),

JoinDate DATE

);

**Creating Procedure:**

CREATE PROCEDURE sp\_InsertEmployee

@FirstName VARCHAR(50),

@LastName VARCHAR(50),

@DepartmentID INT,

@Salary DECIMAL(10,2),

@JoinDate DATE

AS

BEGIN

INSERT INTO Employees (FirstName, LastName, DepartmentID, Salary, JoinDate)

VALUES (@FirstName, @LastName, @DepartmentID, @Salary, @JoinDate);

END;

CREATE PROCEDURE sp\_InsertEmployee

@FirstName VARCHAR(50),

@LastName VARCHAR(50),

@DepartmentID INT,

@Salary DECIMAL(10,2),

@JoinDate DATE

AS

BEGIN

INSERT INTO Employees (FirstName, LastName, DepartmentID, Salary, JoinDate)

VALUES (@FirstName, @LastName, @DepartmentID, @Salary, @JoinDate);

END;

**Insert rows:**

EXEC sp\_InsertEmployee @FirstName = 'Ravi', @LastName = 'Mohan', @DepartmentID = 1, @Salary = 72000.00, @JoinDate = '2023-11-01';

EXEC sp\_InsertEmployee @FirstName = 'Lucia', @LastName = 'Martinez', @DepartmentID = 3, @Salary = 68000.00, @JoinDate = '2024-02-15';

EXEC sp\_InsertEmployee @FirstName = 'Cristiano', @LastName = 'Ronaldo', @DepartmentID = 4, @Salary = 51000.00, @JoinDate = '2022-08-21';

EXEC sp\_InsertEmployee @FirstName = 'Mei', @LastName = 'Wang', @DepartmentID = 1, @Salary = 85000.00, @JoinDate = '2023-06-30';

EXEC sp\_InsertEmployee @FirstName = 'Fatima', @LastName = 'Al-Farsi', @DepartmentID = 2, @Salary = 74000.00, @JoinDate = '2023-10-18';

EXEC sp\_InsertEmployee @FirstName = 'Ethan', @LastName = 'Nguyen', @DepartmentID = 5, @Salary = 56000.00, @JoinDate = '2024-04-12';

EXEC sp\_InsertEmployee @FirstName = 'Zara', @LastName = 'Khan', @DepartmentID = 3, @Salary = 69500.00, @JoinDate = '2024-01-07';

EXEC sp\_InsertEmployee @FirstName = 'Tobias', @LastName = 'Baldoni', @DepartmentID = 4, @Salary = 62000.00, @JoinDate = '2022-09-25';

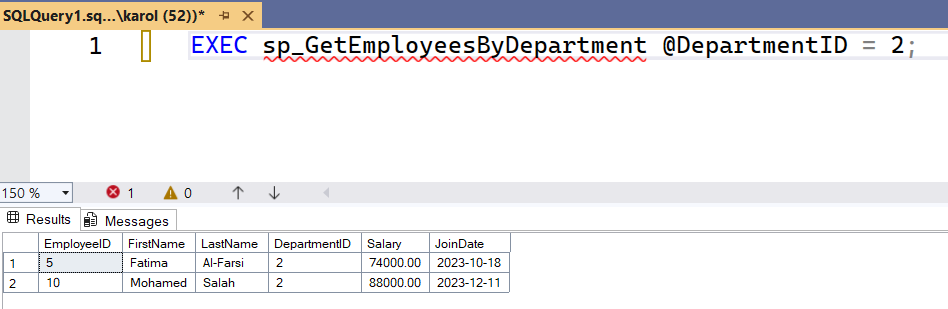
EXEC sp\_InsertEmployee @FirstName = 'Anika', @LastName = 'Johanson', @DepartmentID = 5, @Salary = 79000.00, @JoinDate = '2023-03-19';

EXEC sp\_InsertEmployee @FirstName = 'Mohamed', @LastName = 'Salah', @DepartmentID = 2, @Salary = 88000.00, @JoinDate = '2023-12-11';

**Main Query:**

EXEC sp\_GetEmployeesByDepartment @DepartmentID = 2;

**OUTPUT:**

****

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

**QUERIES:**

CREATE PROCEDURE sp\_CountEmployeesByDepartment

@DepartmentID INT

AS

BEGIN

SELECT COUNT(\*) AS TotalEmployees

FROM Employees

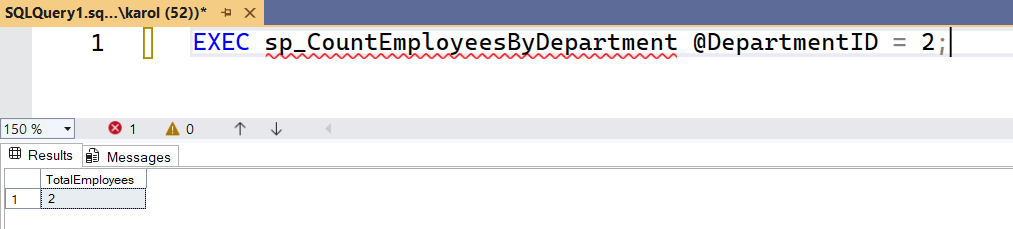
WHERE DepartmentID = @DepartmentID;

END;

**Main Query:**

EXEC sp\_CountEmployeesByDepartment @DepartmentID = 2;

**OUTPUT:**

****