# **Exercise 1: Ranking & 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.

CREATE & INSERT STATEMENT QUERIES:

CREATE TABLE Products (

ProductID INT PRIMARY KEY,

ProductName VARCHAR(100),

Category VARCHAR(50),

Price decimal(10,2)

);

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

(1, 'iPhone','Electronics', 1000),

(2, 'Samsung','Electronics', 900),

(3, 'OnePlus','Electronics', 900),

(4, 'Sony TV','Electronics', 800),

(5, 'Shirt','Fashion',50),

(6, 'Jacket','Fashion', 150),

(7, 'Shoes','Fashion',150),

(8, 'Watch','Fashion', 100.57);

Working Query:

SELECT ProductID,ProductName,Category,Price,

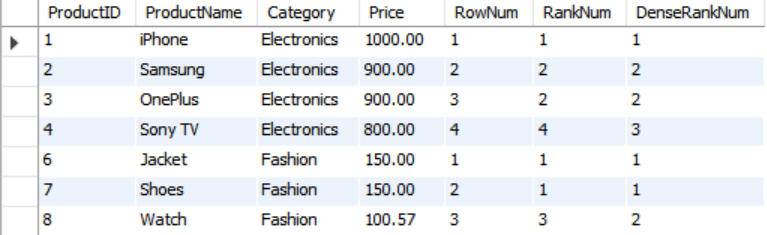
ROW\_NUMBER() OVER (PARTITION BY Category ORDER BY Price DESC) AS RowNum,

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

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

FROM Products;

Output:



# Exercise 1: Create a Stored Procedure

Goal: Create a stored procedure to retrieve employee details by department.

Steps:

1. Define the stored procedure with a parameter for DepartmentID.

2. Write the SQL query to select employee details based on the DepartmentID.

3. Create a stored procedure named `sp\_InsertEmployee` with the following code:

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

**QUERY:**

USE newdb;

GO

-- Drop existing procedures if they exist

IF OBJECT\_ID('sp\_GetEmployeesByDepartment', 'P') IS NOT NULL

DROP PROCEDURE sp\_GetEmployeesByDepartment;

IF OBJECT\_ID('sp\_InsertEmployee', 'P') IS NOT NULL

DROP PROCEDURE sp\_InsertEmployee;

GO

-- Drop tables if they exist

IF OBJECT\_ID('Employees', 'U') IS NOT NULL

DROP TABLE Employees;

IF OBJECT\_ID('Departments', 'U') IS NOT NULL

DROP TABLE Departments;

GO

-- Create Departments table

CREATE TABLE Departments (

DepartmentID INT PRIMARY KEY,

DepartmentName VARCHAR(100)

);

GO

-- Create Employees table with auto-increment EmployeeID

CREATE TABLE Employees (

EmployeeID INT IDENTITY(1,1) PRIMARY KEY,

FirstName VARCHAR(50),

LastName VARCHAR(50),

DepartmentID INT FOREIGN KEY REFERENCES Departments(DepartmentID),

Salary DECIMAL(10,2),

JoinDate DATE

);

GO

-- Insert sample departments

INSERT INTO Departments (DepartmentID, DepartmentName) VALUES

(1, 'HR'),

(2, 'Finance'),

(3, 'IT'),

(4, 'Marketing');

GO

-- Insert sample employees

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

('Suhana', 'Sharma', 1, 5000.00, '2020-01-15'),

('Jane', 'Smith', 2, 6000.00, '2019-03-22'),

('Rohit', 'Gaikwad', 3, 7000.00, '2018-07-30'),

('Shreyas', 'Shinde', 4, 5500.00, '2021-11-05');

GO

-- Create procedure to retrieve employees by department

CREATE PROCEDURE sp\_GetEmployeesByDepartment

@DeptID INT

AS

BEGIN

SELECT EmployeeID, FirstName, LastName, DepartmentID, Salary, JoinDate

FROM Employees

WHERE DepartmentID = @DeptID;

END;

GO

-- Create procedure to insert a new employee (auto-generates EmployeeID)

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;

GO

--TEST: Insert a new employee using the procedure

EXEC sp\_InsertEmployee

@FirstName = 'Alice',

@LastName = 'Williams',

@DepartmentID = 3,

@Salary = 7200.00,

@JoinDate = '2022-06-01';

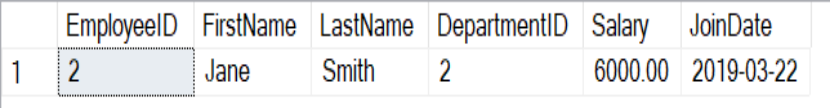
GO

-- TEST: Retrieve all employees in the Finance department (DepartmentID = 2)

EXEC sp\_GetEmployeesByDepartment @DeptID = 2;

GO

**OUTPUT:**



# Exercise 5: Return Data from a Stored Procedure

Goal: Create a stored procedure that returns the total number of employees in a

department.

Steps:

1. Define the stored procedure with a parameter for DepartmentID.

2. Write the SQL query to count the number of employees in the specified department.

3. Save the stored procedure by executing the Stored procedure content.

**QUERY:**

use sampledb;

CREATE TABLE Departments (

DepartmentID INT PRIMARY KEY,

DepartmentName VARCHAR(100)

);

CREATE TABLE Employees (

EmployeeID INT PRIMARY KEY,

FirstName VARCHAR(50),

LastName VARCHAR(50),

DepartmentID INT FOREIGN KEY REFERENCES Departments(DepartmentID),

Salary DECIMAL(10,2),

JoinDate DATE

);

INSERT INTO Departments (DepartmentID, DepartmentName) VALUES

(1, 'HR'),

(2, 'Finance'),

(3, 'IT'),

(4, 'Marketing');

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

('Suhana', 'Sharma', 1, 5000.00, '2020-01-15'),

('Jane', 'Smith', 2, 6000.00, '2019-03-22'),

('Rohit', 'Gaikwad', 3, 7000.00, '2018-07-30'),

('Shreyas', 'Shinde', 4, 5500.00, '2021-11-05');

CREATE PROCEDURE GetEmployeeCountByDepartment

@DeptID INT

AS

BEGIN

SELECT COUNT(\*) AS TotalEmployees

FROM Employees

WHERE DepartmentID = @DeptID;

END;

EXEC GetEmployeeCountByDepartment @DeptID = 3; -- Example for IT department

**OUTPUT**

