**Assignment Week 2**

**Exercise 1: Ranking and Window Functions**

Goal: Use 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.

**Solution**

SELECT \*

FROM (

SELECT \*,

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

FROM Products

) ranked

WHERE row\_num <= 3;

**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 ssp\_InsertEmployee' with the following code:

CREATE PROCEDURE sp\_1nsertEmployee

@FirstName VARCHAR(50),

@LastName VARCHAR(50),

@DepartmentlD INT,

@Sa1ary

@J0inDate DATE

AS

BEGIN

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

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

END;

**Solution**

1. CREATE PROCEDURE sp\_GetEmployeesByDepartment

@DeptID INT

AS

BEGIN

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;

END;

1. 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;

1. 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;

**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.

**Solution**

1. CREATE PROCEDURE sp\_GetEmployeeCountByDepartment

@DeptID INT

AS

BEGIN

-- Body will go here

END;

1. CREATE PROCEDURE sp\_GetEmployeeCountByDepartment

@DeptID INT

AS

BEGIN

SELECT COUNT(\*) AS TotalEmployees

FROM Employees

WHERE DepartmentID = @DeptID;

END;

**How to Execute the Stored Procedure**

EXEC sp\_GetEmployeeCountByDepartment @DeptID = 3;