**WEEK 2**

**ADVANCED SQL ASSIGNMENT**

**-Dipanshu Modi**

**Superset ID: 6373703**

**Roll Number : 2205032**

**College : Kalinga Institute of Industrial Technology (KIIT), Bhubaneswar**

**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.

**Solution:**

1. **1\_row\_number\_query.sql**

WITH RowNumberRanked AS (

    SELECT

        ProductID,

        ProductName,

        Category,

        Price,

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

    FROM Products

)

SELECT

    ProductID,

    ProductName,

    Category,

    Price,

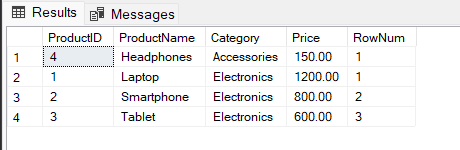
    RowNum

FROM RowNumberRanked

WHERE RowNum <= 3

ORDER BY Category, RowNum;

**Output:**



1. **1\_rank\_query.sql**

WITH RankBased AS (

    SELECT

        ProductID,

        ProductName,

        Category,

        Price,

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

    FROM Products

)

SELECT

    ProductID,

    ProductName,

    Category,

    Price,

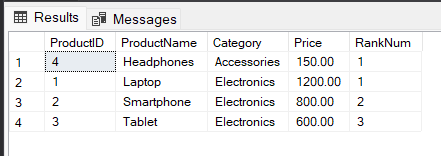
    RankNum

FROM RankBased

WHERE RankNum <= 3

ORDER BY Category, RankNum;

**Output:**



1. **1\_dense\_rank\_query.sql**

WITH DenseRanked AS (

    SELECT

        ProductID,

        ProductName,

        Category,

        Price,

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

    FROM Products

)

SELECT

    ProductID,

    ProductName,

    Category,

    Price,

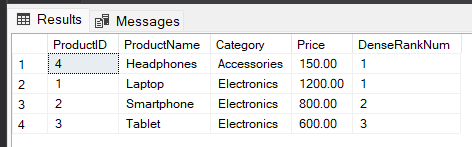
    DenseRankNum

FROM DenseRanked

WHERE DenseRankNum <= 3

ORDER BY Category, DenseRankNum;

**Output:**



**Exercise 2: 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;

**Solution:**

**2\_employee\_procedures.sql**

CREATE PROCEDURE sp\_GetEmployeesByDepartment

    @DepartmentID INT

AS

BEGIN

    SELECT

        EmployeeID,

        FirstName,

        LastName,

        DepartmentID,

        Salary,

        JoinDate

    FROM Employees

    WHERE DepartmentID = @DepartmentID;

END;

GO

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

EXEC sp\_InsertEmployee

    @FirstName = 'Dipanshu',

    @LastName = 'Modi',

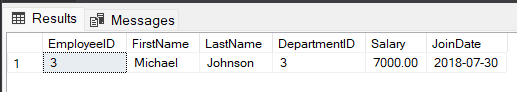
    @DepartmentID = 3,

    @Salary = 7200.00,

    @JoinDate = '2025-06-28';

EXEC sp\_GetEmployeesByDepartment @DepartmentID = 3;

**Output:**



**Exercise 3: 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:**

**3\_count\_employees\_by\_department.sql**

CREATE PROCEDURE sp\_CountEmployeesByDepartment

    @DepartmentID INT

AS

BEGIN

    SELECT

        COUNT(\*) AS TotalEmployees

    FROM

        Employees

    WHERE

        DepartmentID = @DepartmentID;

END;

GO

EXEC sp\_CountEmployeesByDepartment @DepartmentID = 3;

**Output:**

