**WEEK – 2**

**MANDATORY HANDS-ON ADVANCE SQL**

**1.SQL Exercise – Advanced Concepts**

**Advanced SQL Exercises for Online Retail Store**

**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.Sample Data :**

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, 'Laptop A', 'Electronics', 1000),

(2, 'Laptop B', 'Electronics', 1200),

(3, 'Laptop C', 'Electronics', 1200),

(4, 'Phone A', 'Electronics', 800),

(5, 'Shirt A', 'Clothing', 50),

(6, 'Shirt B', 'Clothing', 60),

(7, 'Shirt C', 'Clothing', 60),

(8, 'Shirt D', 'Clothing', 55);

**2. Run Ranking 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 DenseRank

FROM Products;

**3. Get Top 3 Products per Category:**

WITH RankedProducts AS (

SELECT \*,

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

FROM Products

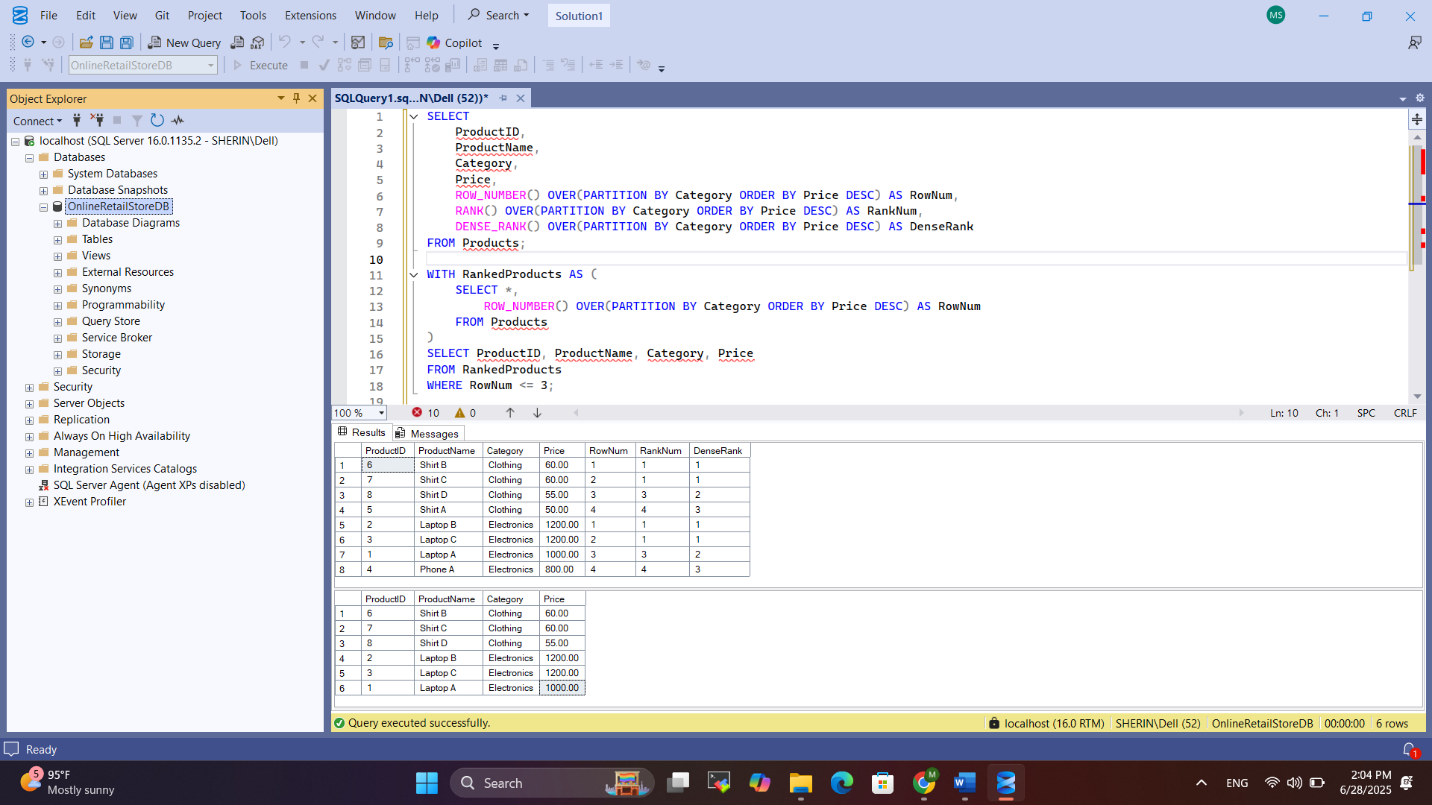
)

SELECT ProductID, ProductName, Category, Price

FROM RankedProducts

WHERE RowNum <= 3;

**OUTPUT:**

****

**4. SQL EXERCISE – Stored Procedure**

**GIVEN :**

**Employee Management System SQL Exercises**

**Database Schema**

The following schema defines the structure for an Employee Management System:

**Departments Table**

CREATE TABLE Departments (

DepartmentID INT PRIMARY KEY,

DepartmentName VARCHAR(100)

);

**Employees Table**

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

);

**Sample Data**

The following sample data can be used for testing:

**Departments Sample Data**

INSERT INTO Departments (DepartmentID, DepartmentName) VALUES

(1, 'HR'),

(2, 'Finance'),

(3, 'IT'),

(4, 'Marketing');

**Employees Sample Data**

INSERT INTO Employees (EmployeeID, FirstName, LastName, DepartmentID, Salary,

JoinDate) VALUES

(1, 'John', 'Doe', 1, 5000.00, '2020-01-15'),

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

(3, 'Michael', 'Johnson', 3, 7000.00, '2018-07-30'),

(4, 'Emily', 'Davis', 4, 5500.00, '2021-11-05');

**Exercises**

**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;

**SOLUTION :**

**1. Create the Table:**

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

);

**2. Insert Sample Data :**

INSERT INTO Departments (DepartmentID, DepartmentName) VALUES

(1, 'HR'),

(2, 'Finance'),

(3, 'IT'),

(4, 'Marketing');

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

(21, 'John', 'Doe', 1, 5000.00, '2020-01-15'),

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

(23, 'Michael', 'Johnson', 3, 7000.00, '2018-07-30'),

(24, 'Emily', 'Davis', 4, 5500.00, '2021-11-05');

**3. Create Stored Procedure :**

CREATE PROCEDURE sp\_InsertEmployee

@FirstName VARCHAR(50),

@LastName VARCHAR(50),

@DepartmentID INT,

@Salary DECIMAL(10,2),

@JoinDate DATE

AS

BEGIN

DECLARE @NewID INT

SELECT @NewID = ISNULL(MAX(EmployeeID), 20) + 1 FROM Employees;

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

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

END;

GO

**4. Insert a New Employee Using Procedure :**

EXEC sp\_InsertEmployee

@FirstName = 'Sara',

@LastName = 'Thomas',

@DepartmentID = 2,

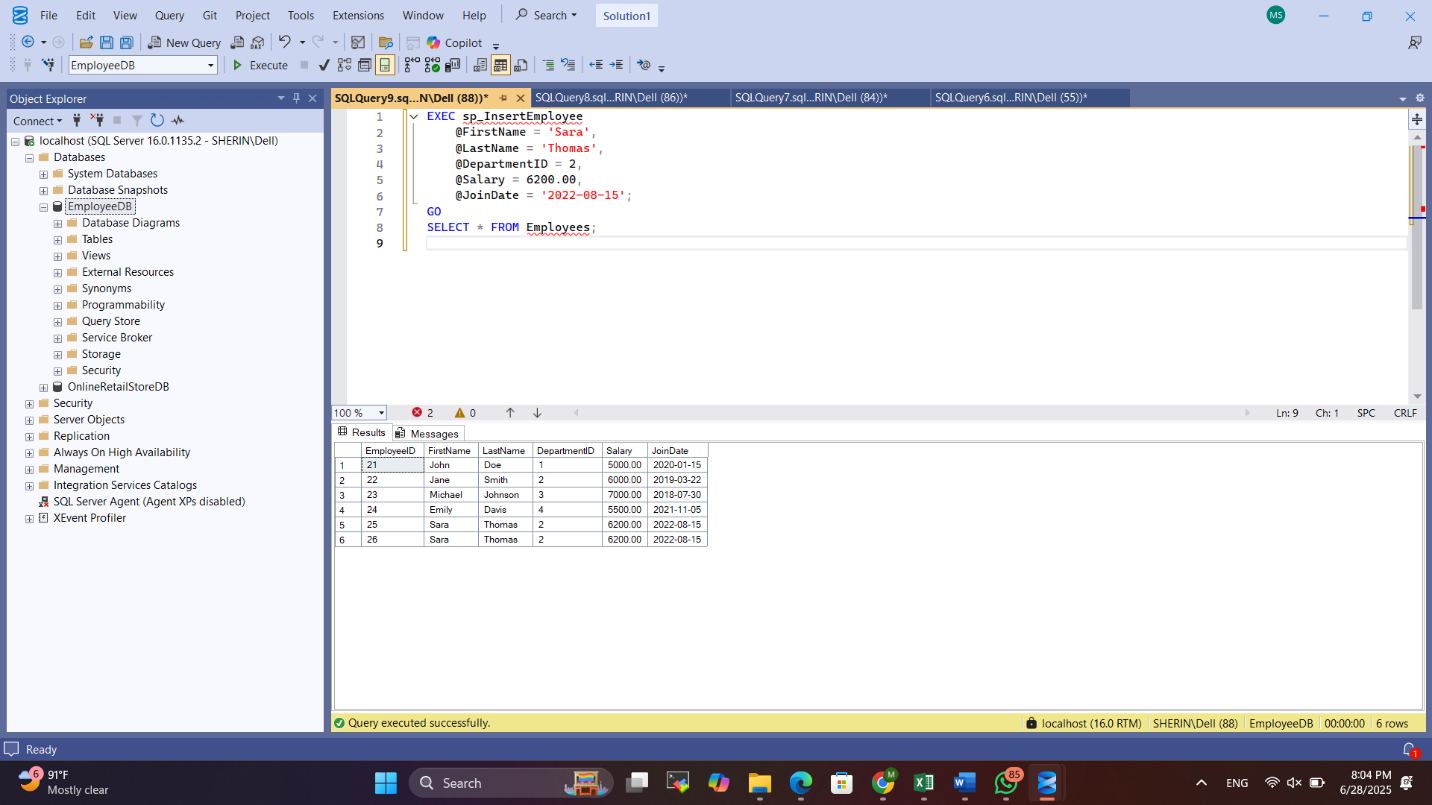
@Salary = 6200.00,

@JoinDate = '2022-08-15';

GO

SELECT \* FROM Employees;

**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.

**SOLUTION :**

**1. Create the Tables :**

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

);

GO

**2. Insert Sample Data :**

INSERT INTO Departments (DepartmentID, DepartmentName) VALUES

(1, 'HR'),

(2, 'Finance'),

(3, 'IT'),

(4, 'Marketing');

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

(1, 'John', 'Doe', 1, 5000.00, '2020-01-15'),

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

(3, 'Michael', 'Johnson', 3, 7000.00, '2018-07-30'),

(4, 'Emily', 'Davis', 4, 5500.00, '2021-11-05');

GO

**3. Create Stored Procedure :**

CREATE PROCEDURE sp\_InsertEmployee

@FirstName VARCHAR(50),

@LastName VARCHAR(50),

@DepartmentID INT,

@Salary DECIMAL(10,2),

@JoinDate DATE

AS

BEGIN

DECLARE @NewID INT;

SELECT @NewID = ISNULL(MAX(EmployeeID), 0) + 1 FROM Employees;

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

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

END;

GO

CREATE PROCEDURE sp\_GetEmployeeCountByDepartment

@DeptID INT

AS

BEGIN

SELECT COUNT(\*) AS EmployeeCount

FROM Employees

WHERE DepartmentID = @DeptID;

END;

GO

**4. Test the Procedures:**

EXEC sp\_InsertEmployee

@FirstName = 'Sara',

@LastName = 'Thomas',

@DepartmentID = 2,

@Salary = 6200.00,

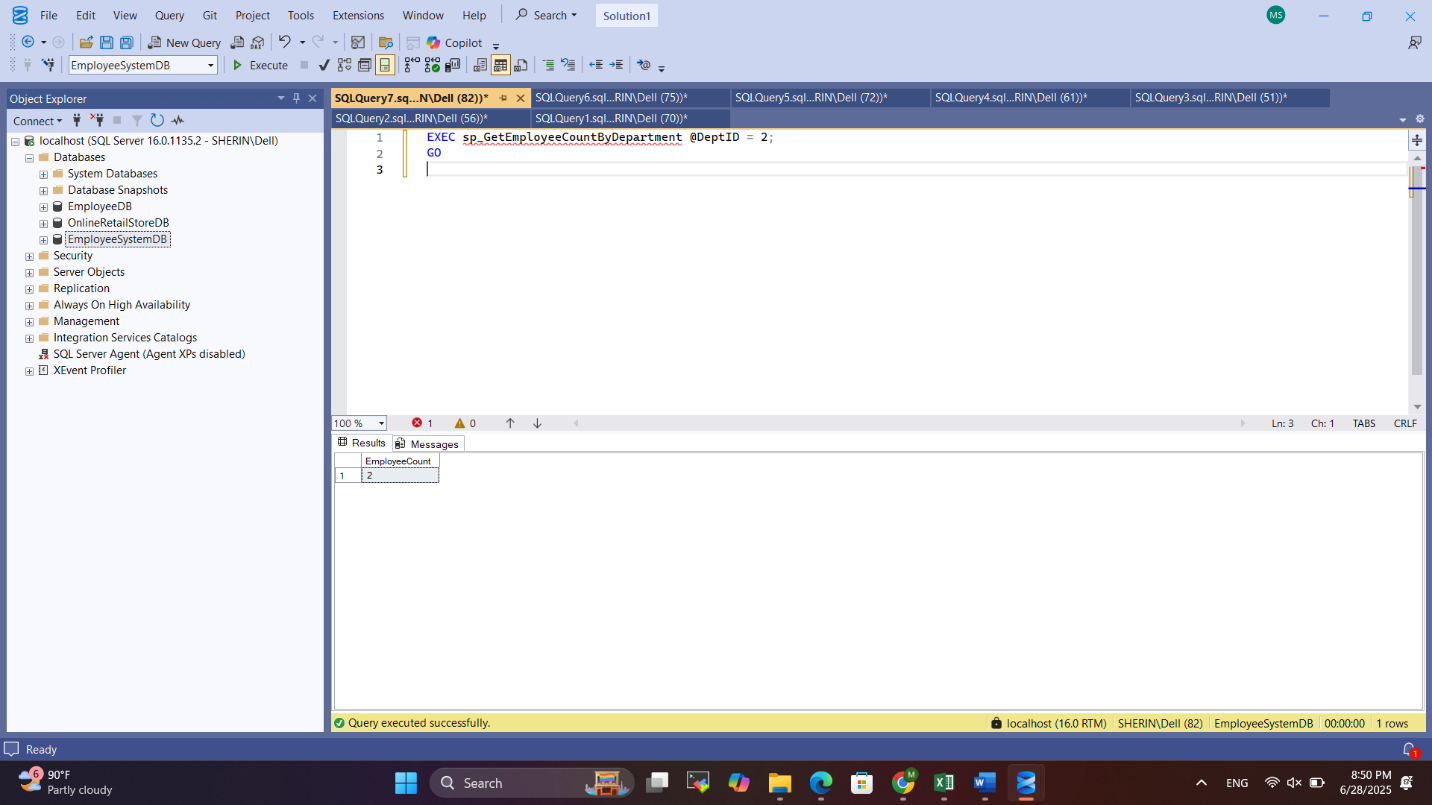
@JoinDate = '2022-08-15';

GO

EXEC sp\_GetEmployeeCountByDepartment @DeptID = 2;

GO

**OUTPUT :**

****