**WEEK -2**

**ADDITIONAL IMPORTANT HANDS – ON ADVANCED SQL  
2. SQL Exercise - Index**

**GIVEN QUESTION:**

-- Database Schema

CREATE TABLE Customers (

CustomerID INT PRIMARY KEY,

Name VARCHAR(100),

Region VARCHAR(50)

);

CREATE TABLE Products (

ProductID INT PRIMARY KEY,

ProductName VARCHAR(100),

Category VARCHAR(50),

Price DECIMAL(10, 2)

);

CREATE TABLE Orders (

OrderID INT PRIMARY KEY,

CustomerID INT,

OrderDate DATE,

FOREIGN KEY (CustomerID) REFERENCES Customers(CustomerID)

);

CREATE TABLE OrderDetails (

OrderDetailID INT PRIMARY KEY,

OrderID INT,

ProductID INT,

Quantity INT,

FOREIGN KEY (OrderID) REFERENCES Orders(OrderID),

FOREIGN KEY (ProductID) REFERENCES Products(ProductID)

);

-- Sample Data

INSERT INTO Customers (CustomerID, Name, Region) VALUES

(1, 'Alice', 'North'),

(2, 'Bob', 'South'),

(3, 'Charlie', 'East'),

(4, 'David', 'West');

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

(1, 'Laptop', 'Electronics', 1200.00),

(2, 'Smartphone', 'Electronics', 800.00),

(3, 'Tablet', 'Electronics', 600.00),

(4, 'Headphones', 'Accessories', 150.00);

INSERT INTO Orders (OrderID, CustomerID, OrderDate) VALUES

(1, 1, '2023-01-15'),

(2, 2, '2023-02-20'),

(3, 3, '2023-03-25'),

(4, 4, '2023-04-30');

INSERT INTO OrderDetails (OrderDetailID, OrderID, ProductID, Quantity) VALUES

(1, 1, 1, 1),

(2, 2, 2, 2),

(3, 3, 3, 1),

(4, 4, 4, 3);

**-- Exercise 1: Creating a Non-Clustered Index**

-- Goal: Create a non-clustered index on the ProductName column in the Products table and compare query execution time before and after index creation.

-- Step 1: Query to fetch product details before index creation

SELECT \* FROM Products WHERE ProductName = 'Laptop';

-- Step 2: Create a non-clustered index on ProductName

-- Step 3: Query to fetch product details after index creation

SELECT \* FROM Products WHERE ProductName = 'Laptop';

**SOLUTION:**

**1.Run query before index:**

SELECT \* FROM Products WHERE ProductName = 'Laptop';

**2.Create the non-clustered index:**

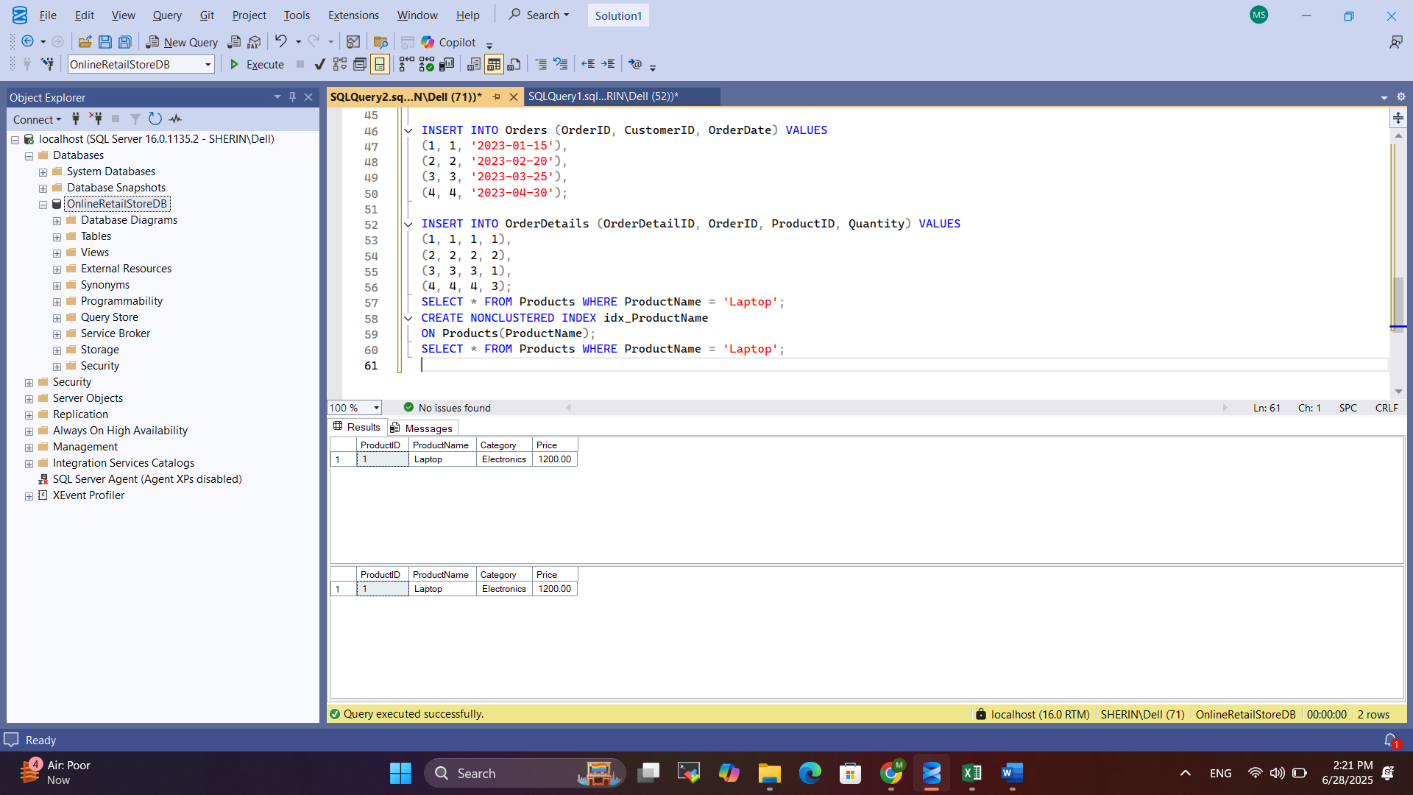
CREATE NONCLUSTERED INDEX idx\_ProductName

ON Products(ProductName);

**3.Run the same query again after index is created:**

SELECT \* FROM Products WHERE ProductName = 'Laptop';

**OUTPUT :**

****

**-- Exercise 2: Creating a Clustered Index**

**--** Goal: Create a clustered index on the OrderDate column in the Orders table and compare query execution time before and after index creation.

-- Step 1: Query to fetch orders before index creation

SELECT \* FROM Orders WHERE OrderDate = '2023-01-15';

-- Step 2: Create a clustered index on OrderDate

-- Step 3: Query to fetch orders after index creation

SELECT \* FROM Orders WHERE OrderDate = '2023-01-15';

**SOLUTION:**

**1.Check current indexes:**

EXEC sp\_helpindex 'Orders';

**2.Create clustered index on OrderDate:**

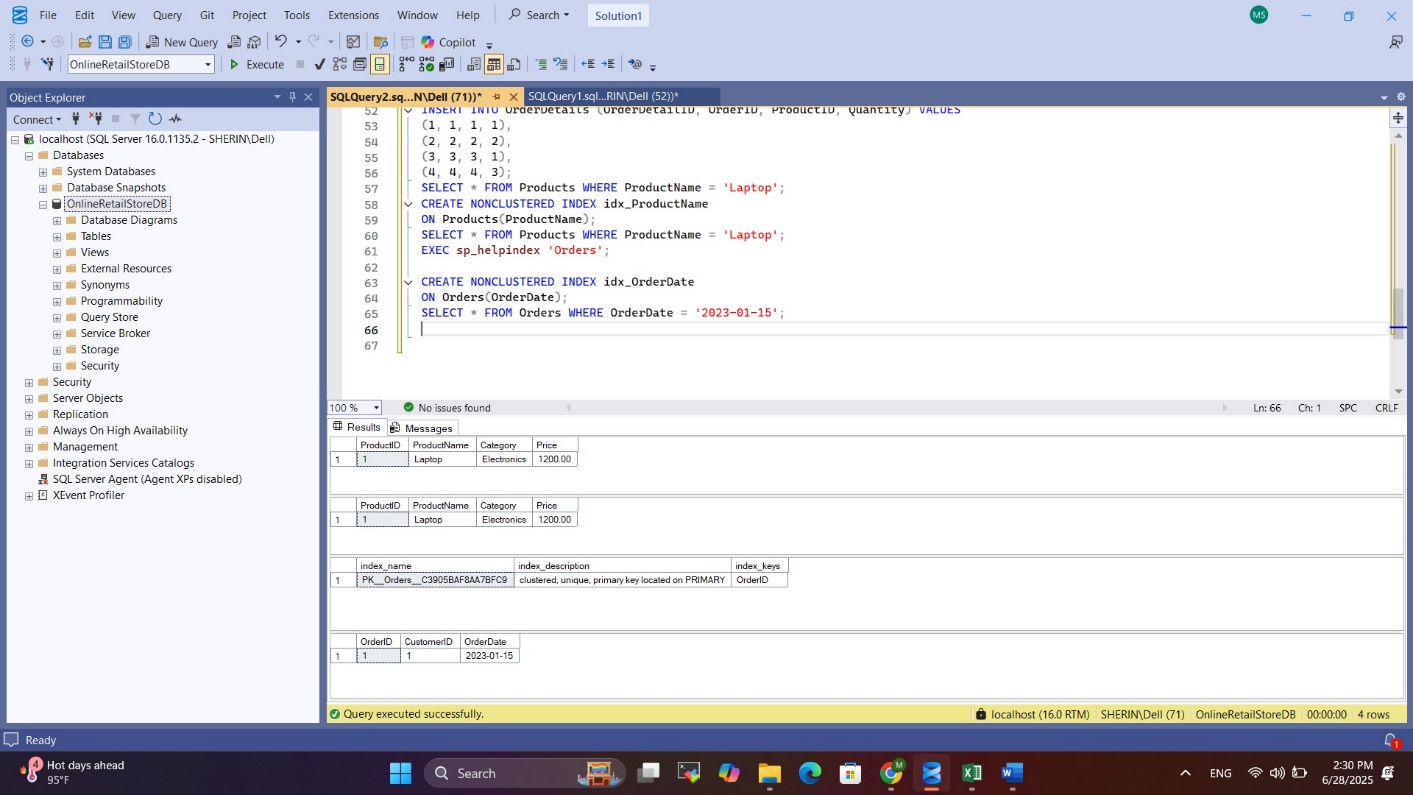
CREATE CLUSTERED INDEX idx\_OrderDate

ON Orders(OrderDate);

**3.Run query before/after index:**

SELECT \* FROM Orders WHERE OrderDate = '2023-01-15';

**OUTPUT :**

****

**-- Exercise 3: Creating a Composite Index**

-- Goal: Create a composite index on the CustomerID and OrderDate columns in the Orders table and compare query execution time before and after index creation.

-- Step 1: Query to fetch orders before index creation

SELECT \* FROM Orders WHERE CustomerID = 1 AND OrderDate = '2023-01-15';

-- Step 2: Create a composite index on CustomerID and OrderDate

-- Step 3: Query to fetch orders after index creation

SELECT \* FROM Orders WHERE CustomerID = 1 AND OrderDate = '2023-01-15';

**SOLUTION:**

**1.Run the query before index:**

SELECT \* FROM Orders

WHERE CustomerID = 1 AND OrderDate = '2023-01-15';

**2.Create composite index:**

CREATE NONCLUSTERED INDEX idx\_CustomerID\_OrderDate

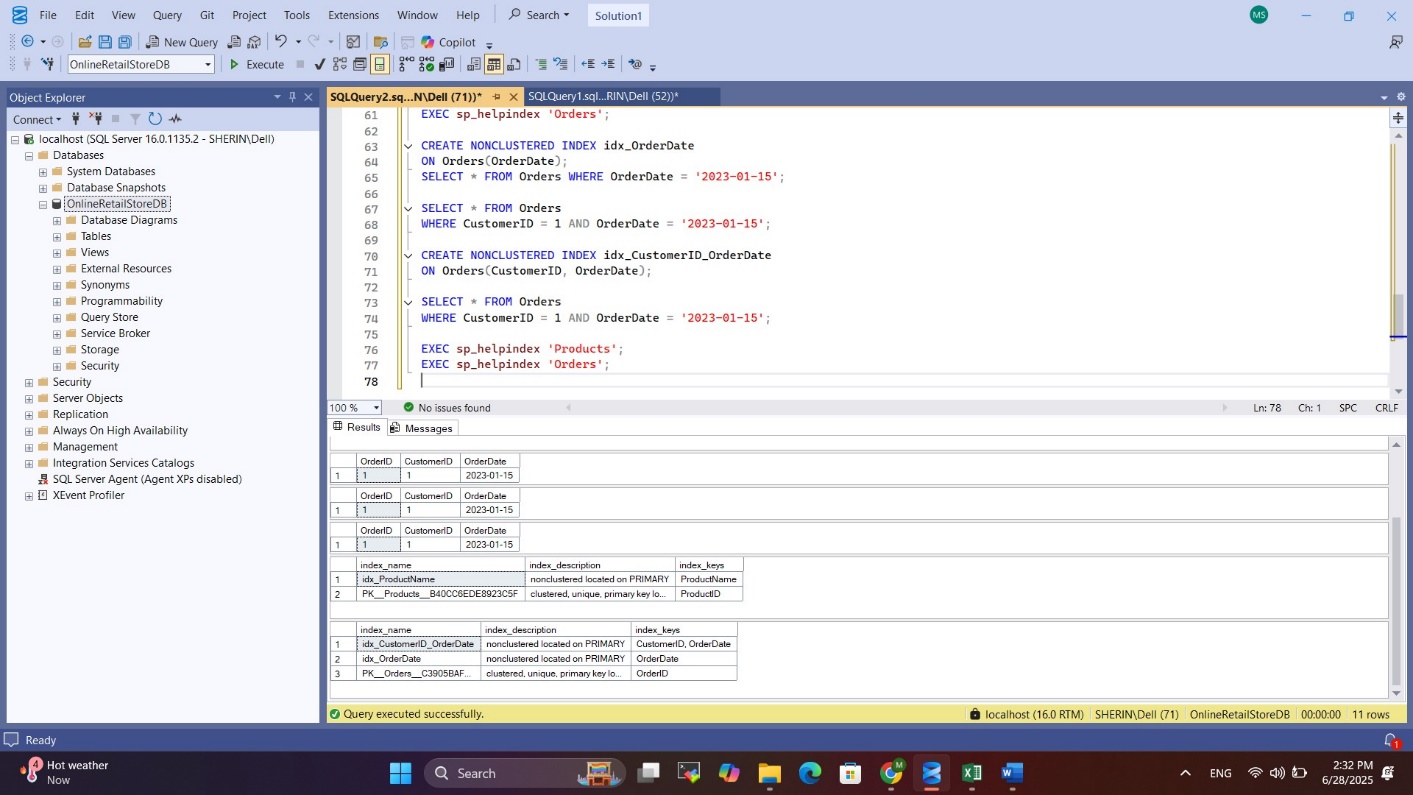
ON Orders(CustomerID, OrderDate);

**3. Run the same query again:**

SELECT \* FROM Orders

WHERE CustomerID = 1 AND OrderDate = '2023-01-15';

**OUTPUT :**

****

**5. SQL Exercise – Functions:**

**Given :**

Employee Management System - SQL Exercises

Database Schema

The Employee Management System database schema consists of the following tables:

1. Departments

| Column | Data Type | Description |

|---------------|---------------|------------------------------|

| DepartmentID | INT (PK) | Unique department ID |

| DepartmentName| VARCHAR(100) | Name of the department |

1. Employees

| Column | Data Type | Description |

|---------------|---------------|------------------------------|

| EmployeeID | INT (PK) | Unique employee ID |

| FirstName | VARCHAR(50) | Employee's first name | | LastName | VARCHAR(50) | Employee's last name |

| DepartmentID | INT (FK) | Linked to Departments |

| Salary | DECIMAL(10,2) | Monthly salary |

| JoinDate | DATE | Date of joining |

Sample Data

Sample data for testing:

Departments:

| DepartmentID | DepartmentName |

|--------------|----------------|

| 1 | HR |

| 2 | IT |

| 3 | Finance |

Employees:

| EmployeeID | FirstName | LastName | DepartmentID | Salary | JoinDate |

|------------|-----------|----------|--------------|---------|------------|

| 1 | John | Doe | 1 | 5000.00 | 2020-01-15 |

| 2 | Jane | Smith | 2 | 6000.00 | 2019-03-22 |

| 3 | Bob | Johnson | 3 | 5500.00 | 2021-07-01 |

**Exercise 7: Return Data from a Scalar Function**

Goal: Return the annual salary for a specific employee using `fn\_CalculateAnnualSalary`.

Steps:

1. Execute the `fn\_CalculateAnnualSalary` function for an employee with `EmployeeID = 1`.
2. Verify the result.

**SOLUTION :**

**1. Create A 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, 'IT'),

(3, 'Finance');

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, 'Bob', 'Johnson', 3, 5500.00, '2021-07-01');

**3. Create Scalar Function :**

CREATE FUNCTION fn\_CalculateAnnualSalary (@EmpID INT)

RETURNS DECIMAL(12,2)

AS

BEGIN

DECLARE @AnnualSalary DECIMAL(12,2)

SELECT @AnnualSalary = Salary \* 12

FROM Employees

WHERE EmployeeID = @EmpID

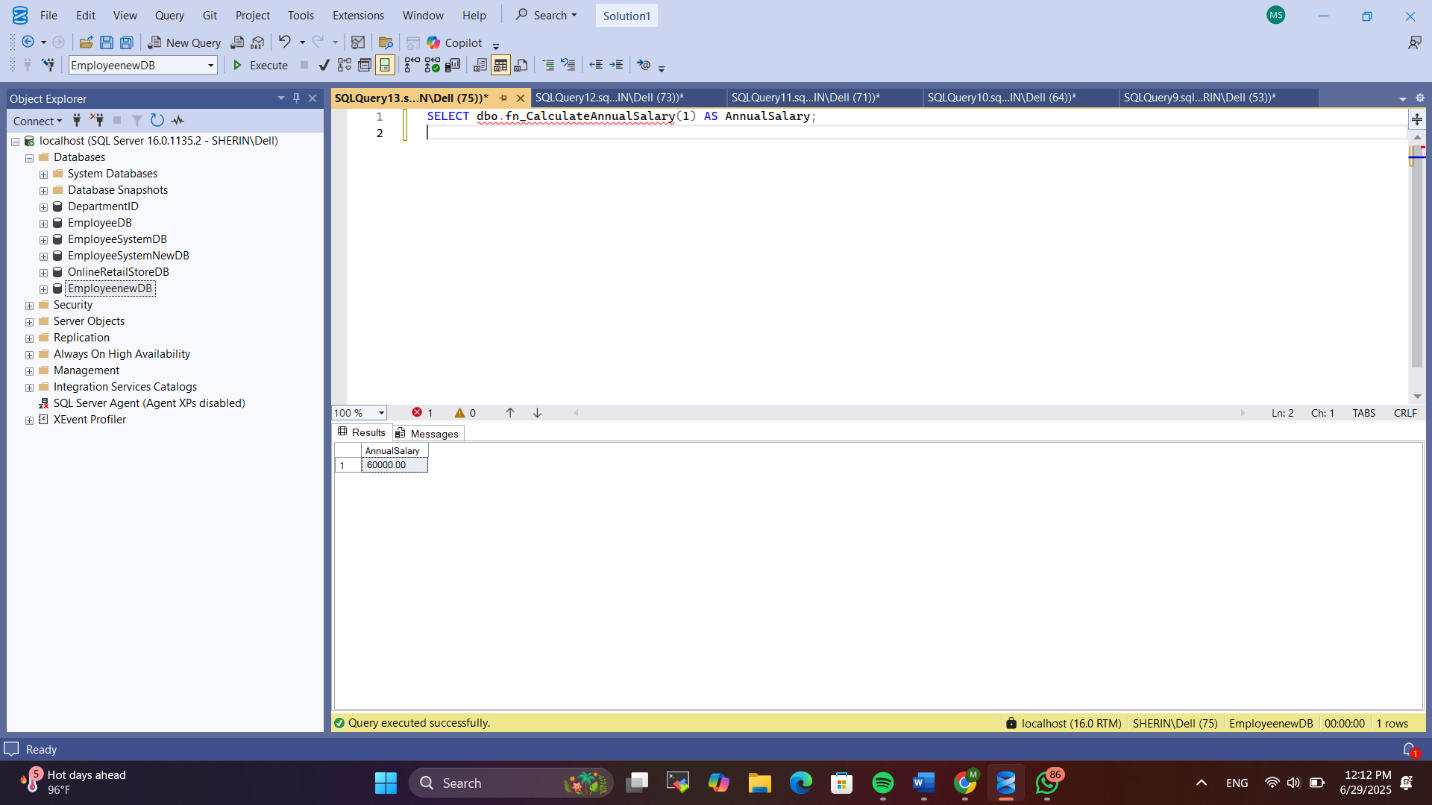
RETURN @AnnualSalary

END

**4. Call The Function :**

SELECT dbo.fn\_CalculateAnnualSalary(1) AS AnnualSalary;

**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');

**Exercise 4: Execute a Stored Procedure**

Goal: Execute the stored procedure to retrieve employee details for a specific department.

Steps:

1. Write the SQL command to execute the stored procedure with a DepartmentID parameter.
2. Execute the command and review the results.

**SOLUTION:**

**1. Create 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

);

**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');

**3. Create Stored Procedure :**

CREATE PROCEDURE sp\_GetEmployeesByDepartment

@DeptID INT

AS

BEGIN

SELECT

EmployeeID,

FirstName,

LastName,

DepartmentID,

Salary,

JoinDate

FROM Employees

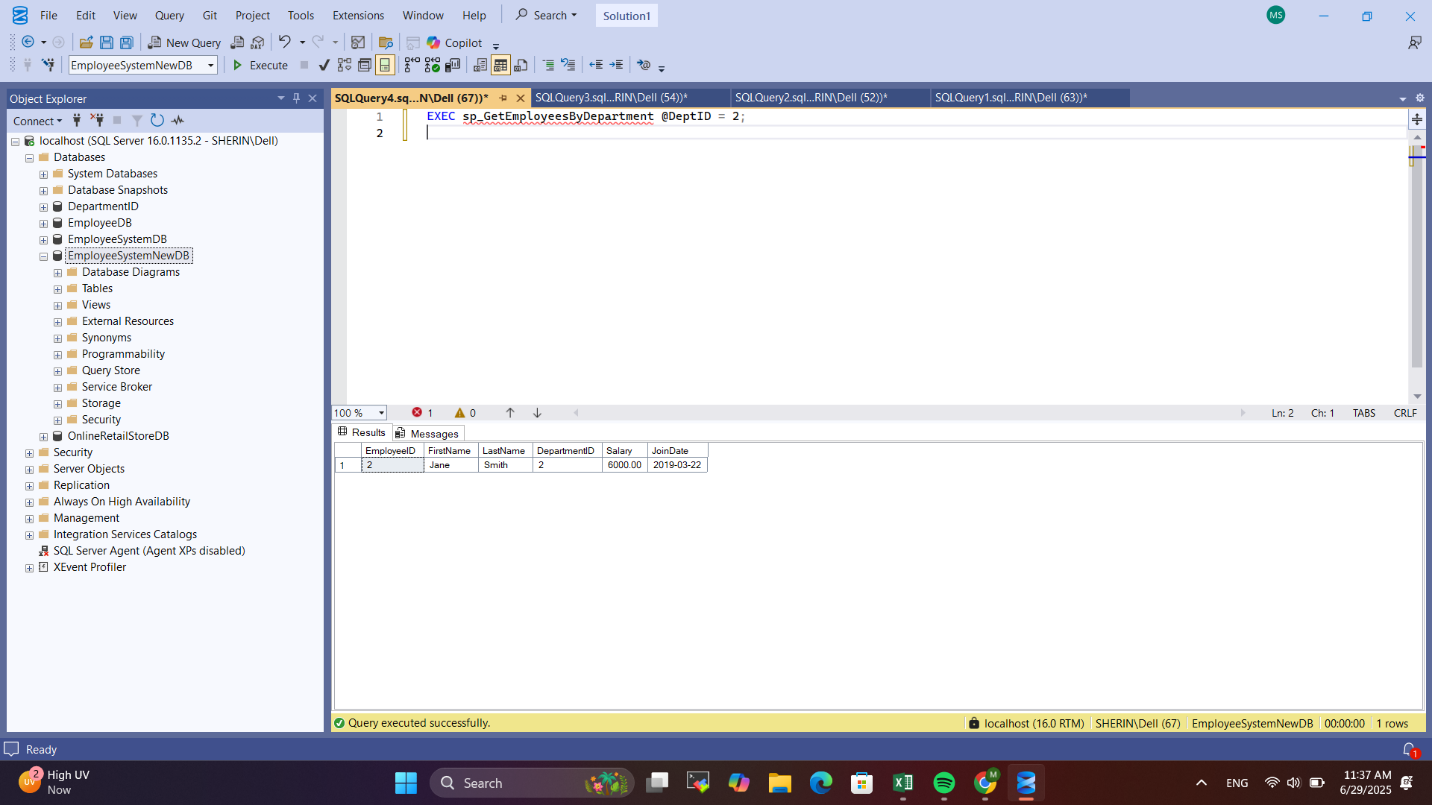
WHERE DepartmentID = @DeptID;

END;

**4. Execute the Stored Procedure:**

EXEC sp\_GetEmployeesByDepartment @DeptID = 2;

**OUTPUT:**

****