# ****WEEK 2 – ADVANCED SQL TASKS****

## 0. ****TABLE CREATION + SAMPLE DATA****

-- Week 2: Initial Setup — Create Employees Table

USE DN4\_DemoDB;

GO

-- Drop old table if exists

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

DROP TABLE Employees;

GO

-- Create table with auto-incremented EmployeeID

CREATE TABLE Employees (

EmployeeID INT IDENTITY(1,1) PRIMARY KEY,

Name VARCHAR(100),

Department VARCHAR(50),

Salary INT

);

GO

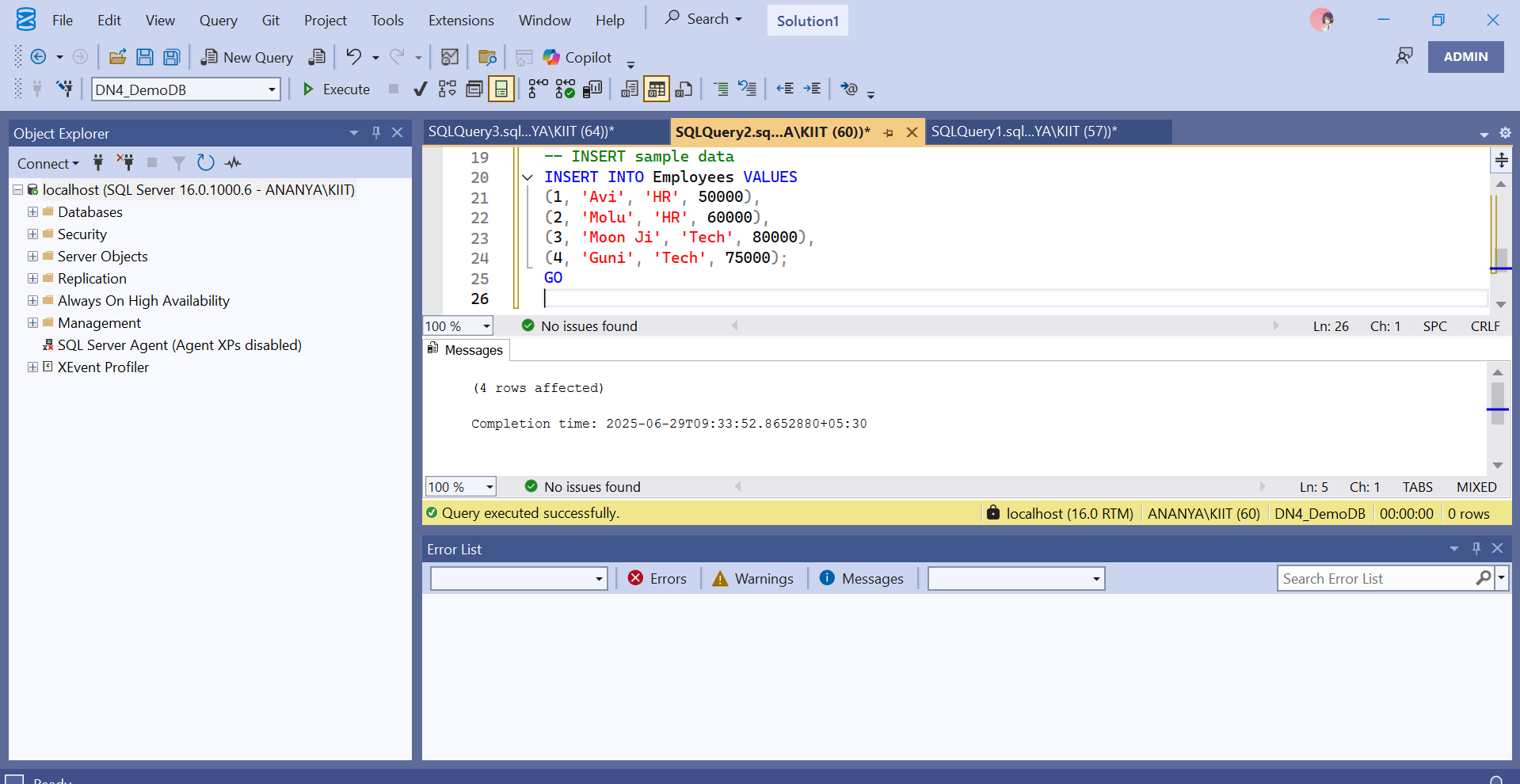
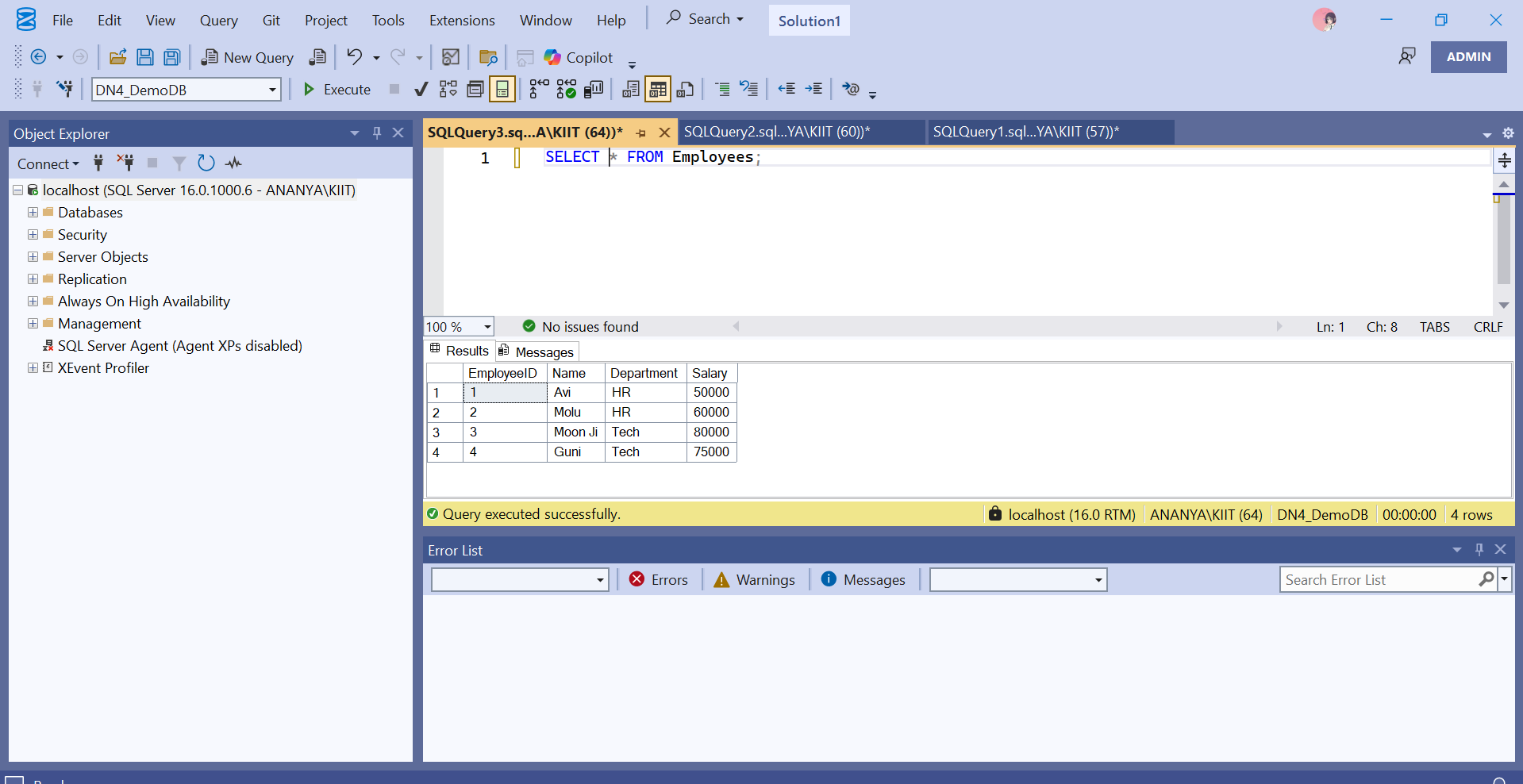
-- Insert sample data

INSERT INTO Employees (Name, Department, Salary) VALUES

('Avi', 'HR', 50000),

('Molu', 'HR', 60000),

('Moon Ji', 'Tech', 80000),

('Guni', 'Tech',75000);Go

## 1. ****TASK 1 — WINDOW FUNCTIONS****

-- Task 1: Use RANK(), DENSE\_RANK(), and ROW\_NUMBER()

USE DN4\_DemoDB;

GO

SELECT

EmployeeID,

Name,

Department,

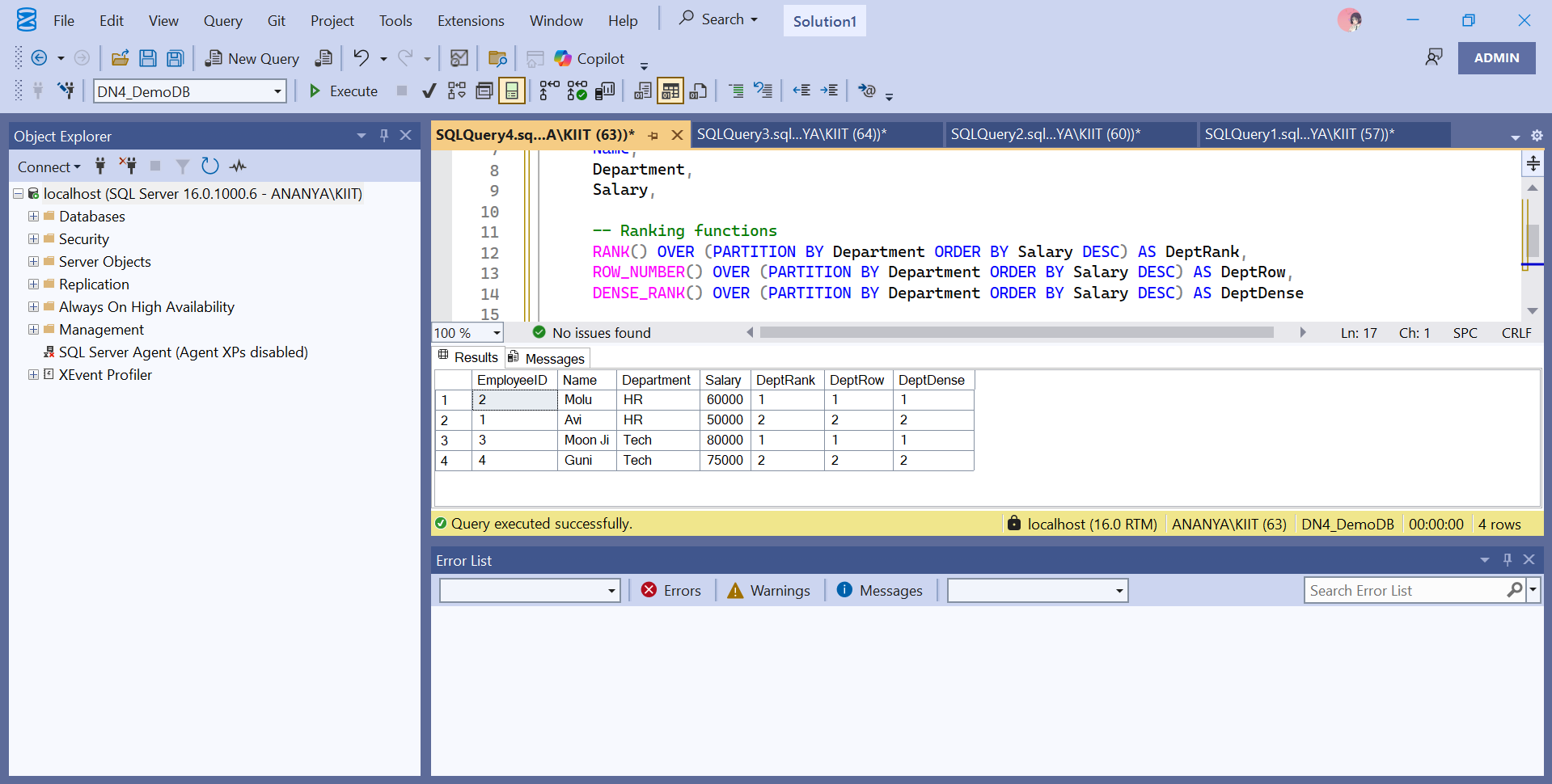
Salary,

RANK() OVER (PARTITION BY Department ORDER BY Salary DESC) AS DeptRank,

ROW\_NUMBER() OVER (PARTITION BY Department ORDER BY Salary DESC) AS DeptRow,

DENSE\_RANK() OVER (PARTITION BY Department ORDER BY Salary DESC) AS DeptDense

FROM Employees;



## 2. ****TASK 2 — STORED PROCEDURE (INSERT EMPLOYEE)****

-- Task 2: Stored Procedure to insert employee data

USE DN4\_DemoDB;

GO

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

DROP PROCEDURE InsertEmployee;

GO

CREATE PROCEDURE InsertEmployee

@Name VARCHAR(100),

@Department VARCHAR(50),

@Salary INT

AS

BEGIN

INSERT INTO Employees (Name, Department, Salary)

VALUES (@Name, @Department, @Salary);

END;

GO

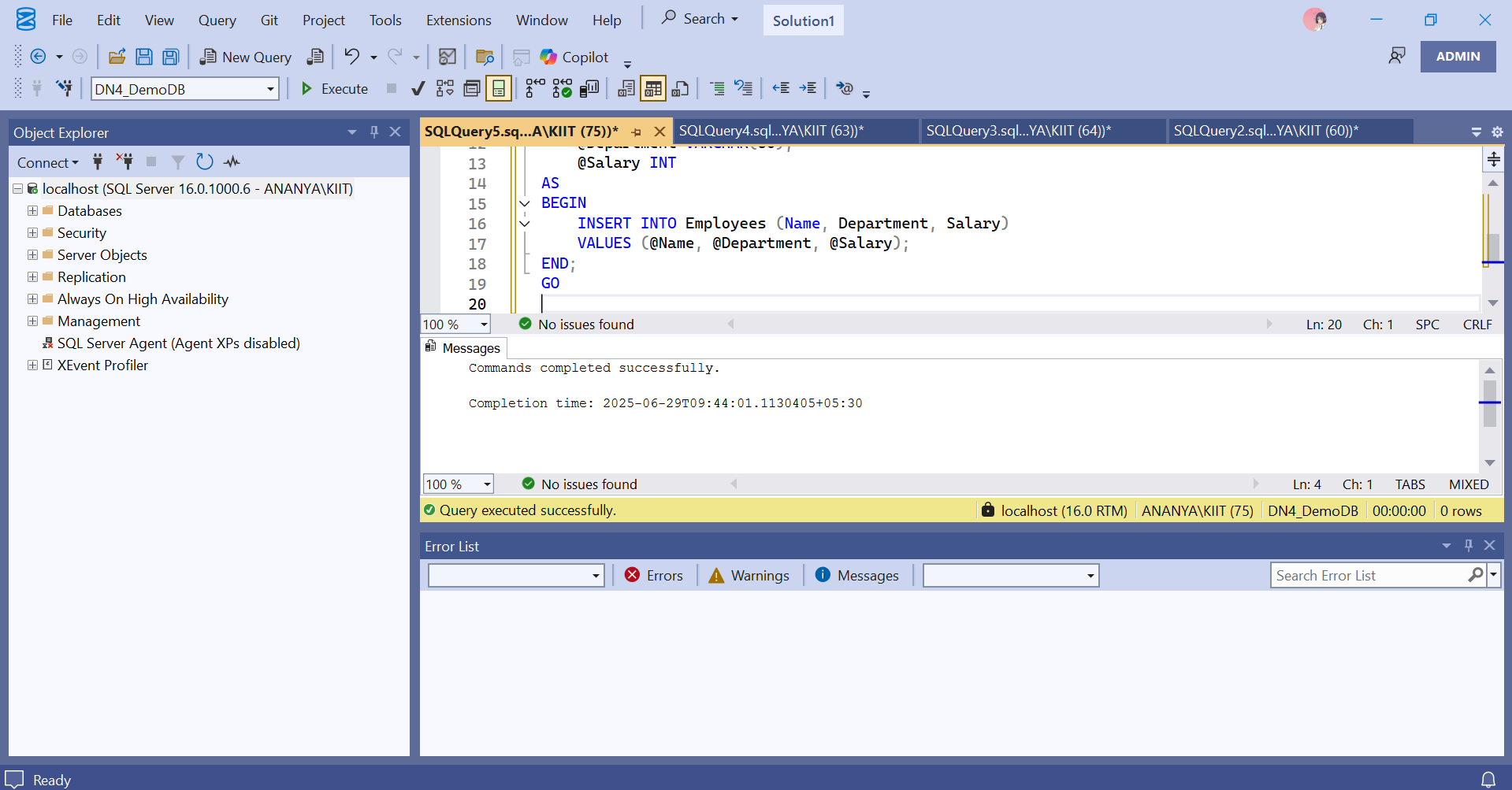
-- Test Insert

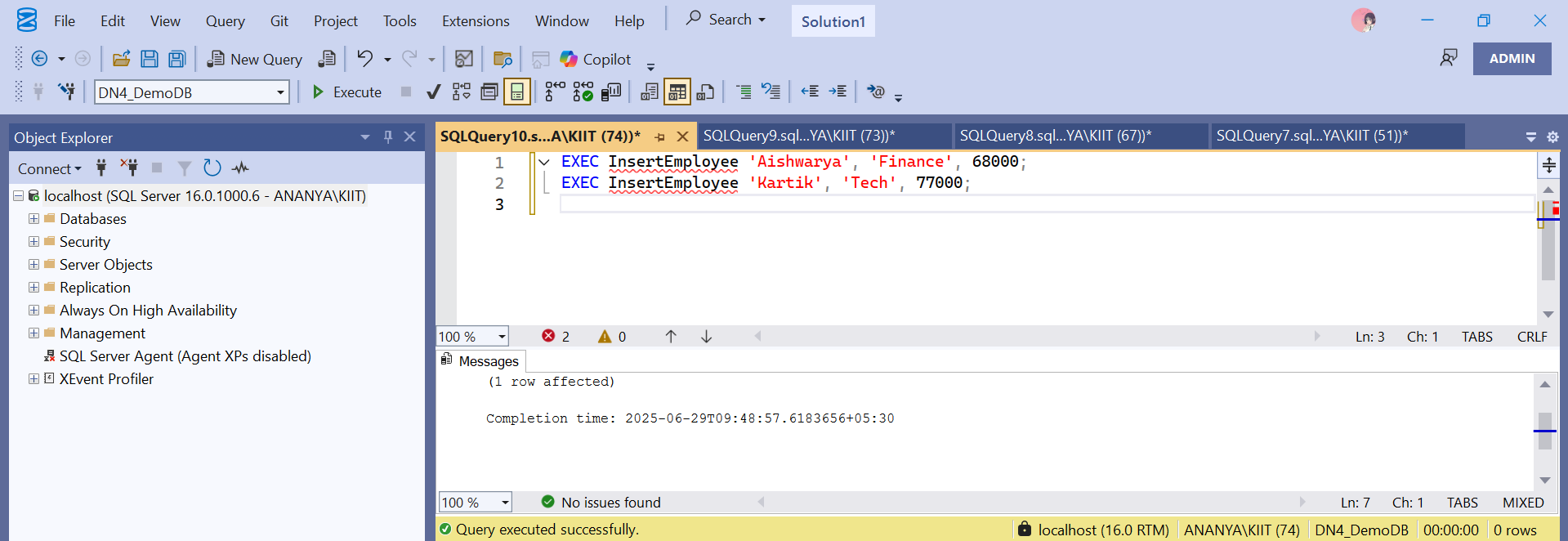
EXEC InsertEmployee 'Aishwarya', 'Finance', 68000;

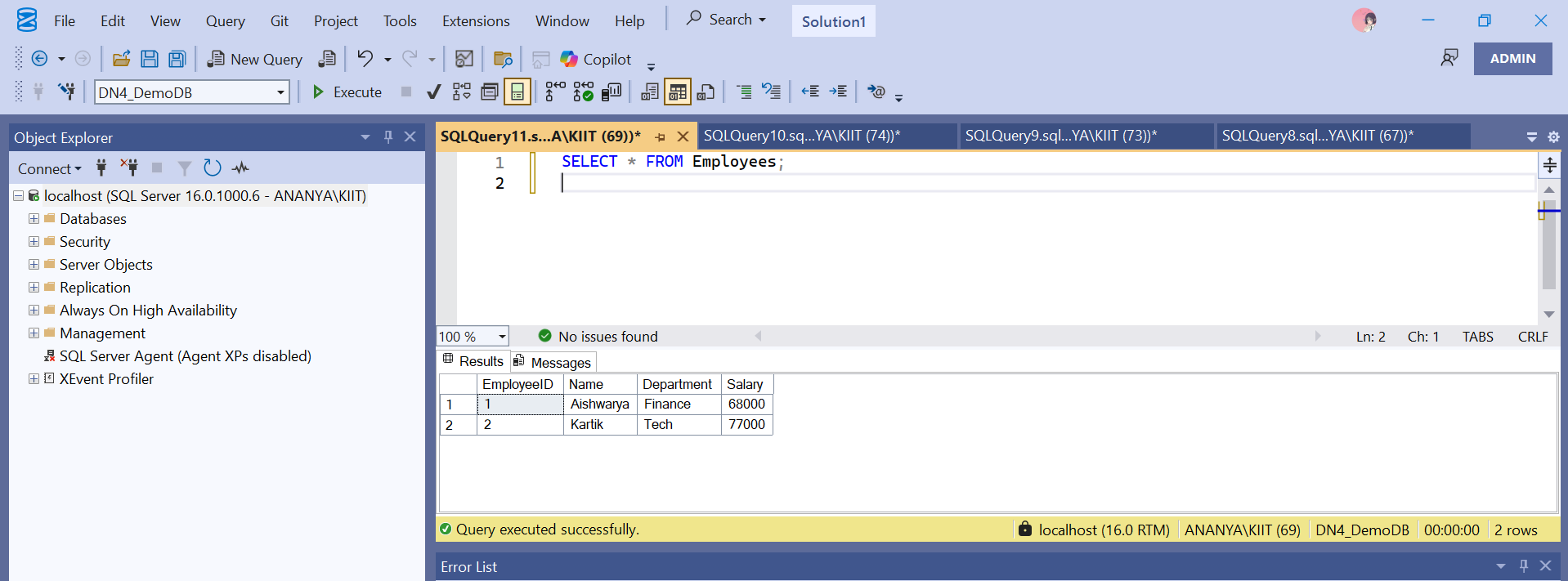
EXEC InsertEmployee 'Kartik', 'Tech', 77000;

-- Check Table

SELECT \* FROM Employees;







## 3. ****TASK 3 — STORED PROCEDURE (RETURN DATA BY DEPT)****

-- Task 3: Stored Procedure to return employees by department

USE DN4\_DemoDB;

GO

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

DROP PROCEDURE GetEmployeesByDept;

GO

CREATE PROCEDURE GetEmployeesByDept

@Department VARCHAR(50)

AS

BEGIN

SELECT \* FROM Employees

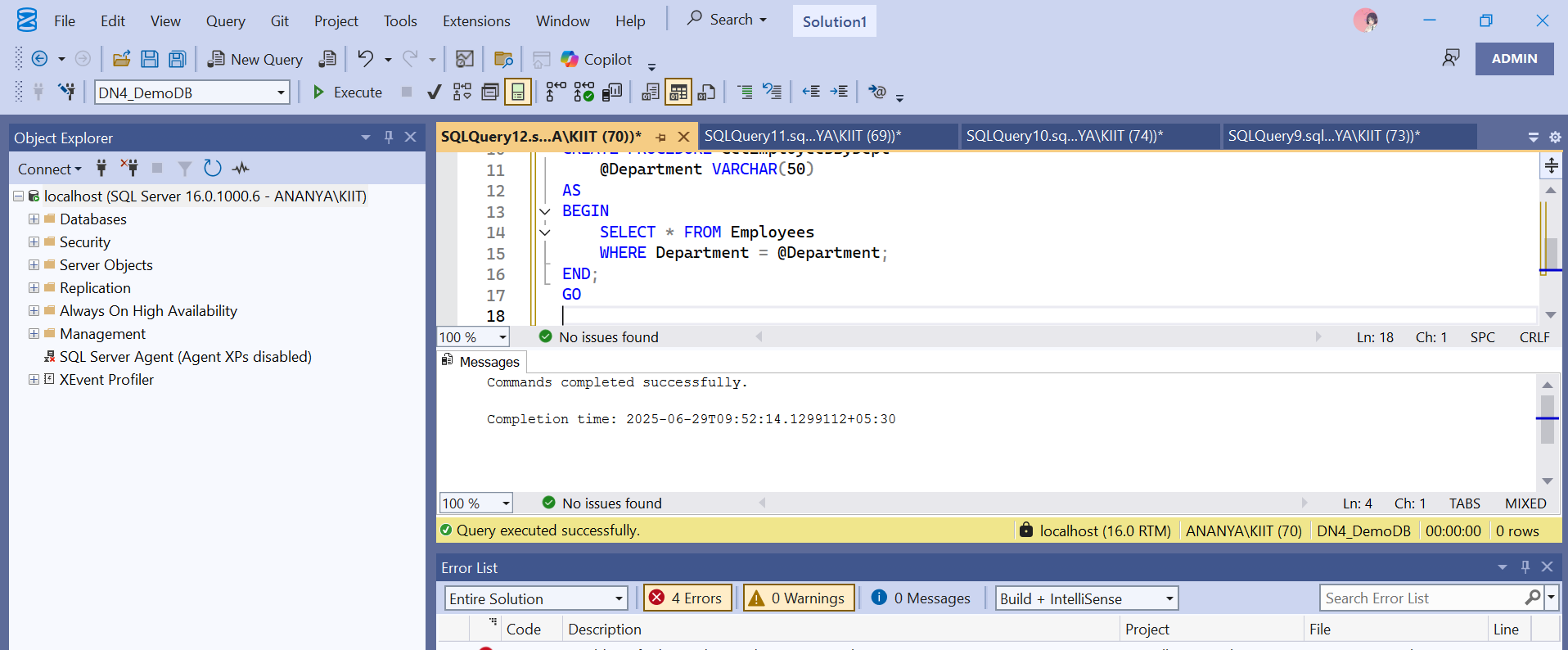
WHERE Department = @Department;

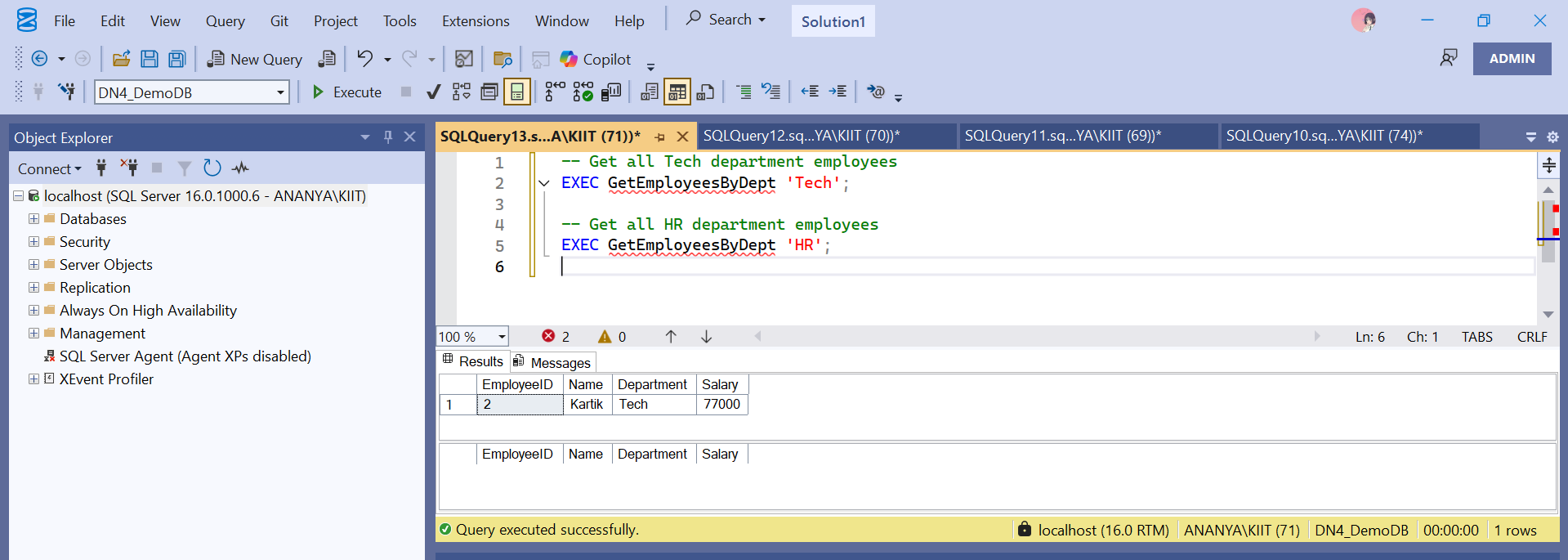
END;

GO

-- Test Procedure

EXEC GetEmployeesByDept 'Tech';





## 4. ****OPTIONAL TASK — SCALAR FUNCTION****

-- Scalar Function: Calculate Yearly Salary from Monthly Salary

USE DN4\_DemoDB;

GO

IF OBJECT\_ID('dbo.GetYearlySalary', 'FN') IS NOT NULL

DROP FUNCTION dbo.GetYearlySalary;

GO

CREATE FUNCTION dbo.GetYearlySalary (@MonthlySalary INT)

RETURNS INT

AS

BEGIN

RETURN @MonthlySalary \* 12;

END;

GO

-- Test Function

SELECT dbo.GetYearlySalary(50000) AS YearlySalary;

-- Use in Table

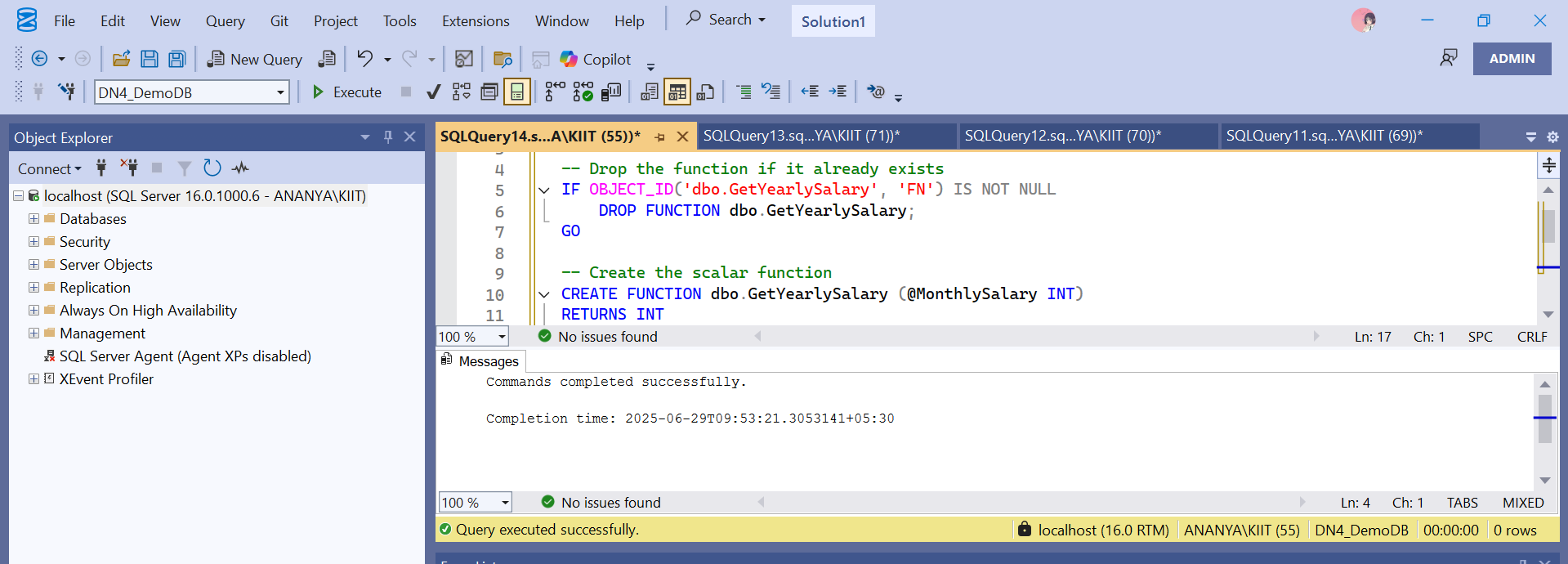
SELECT

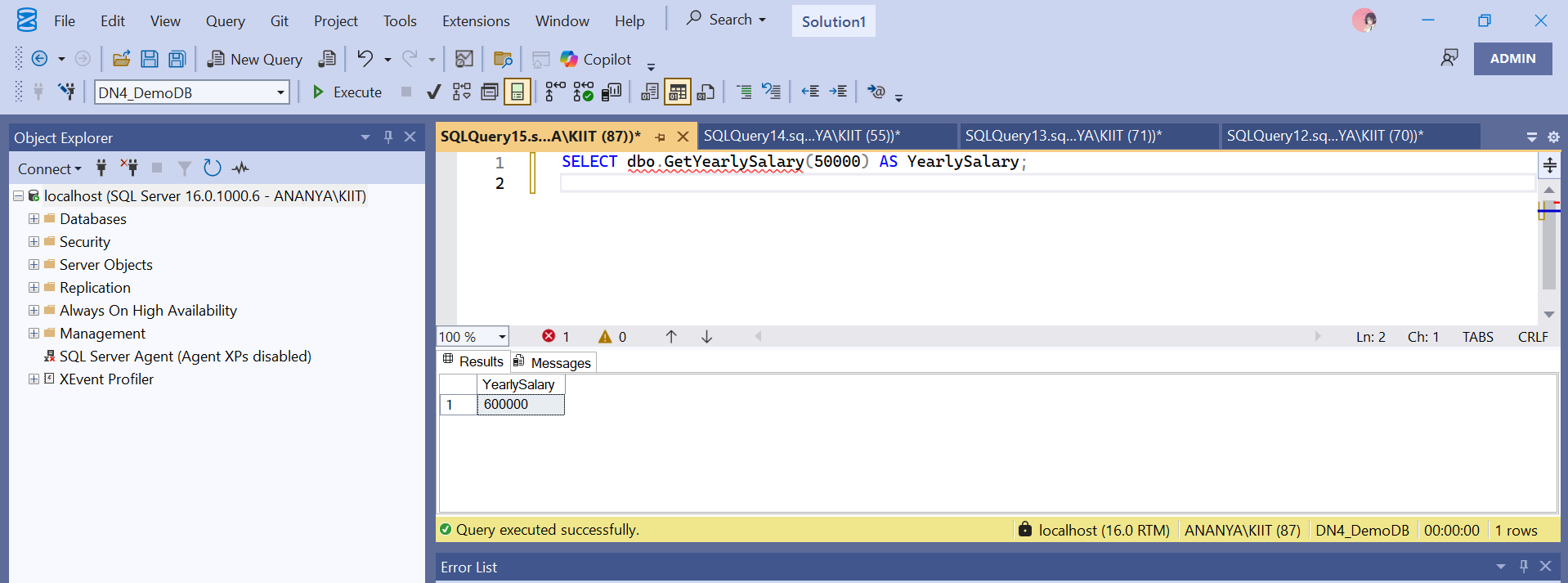
Name,

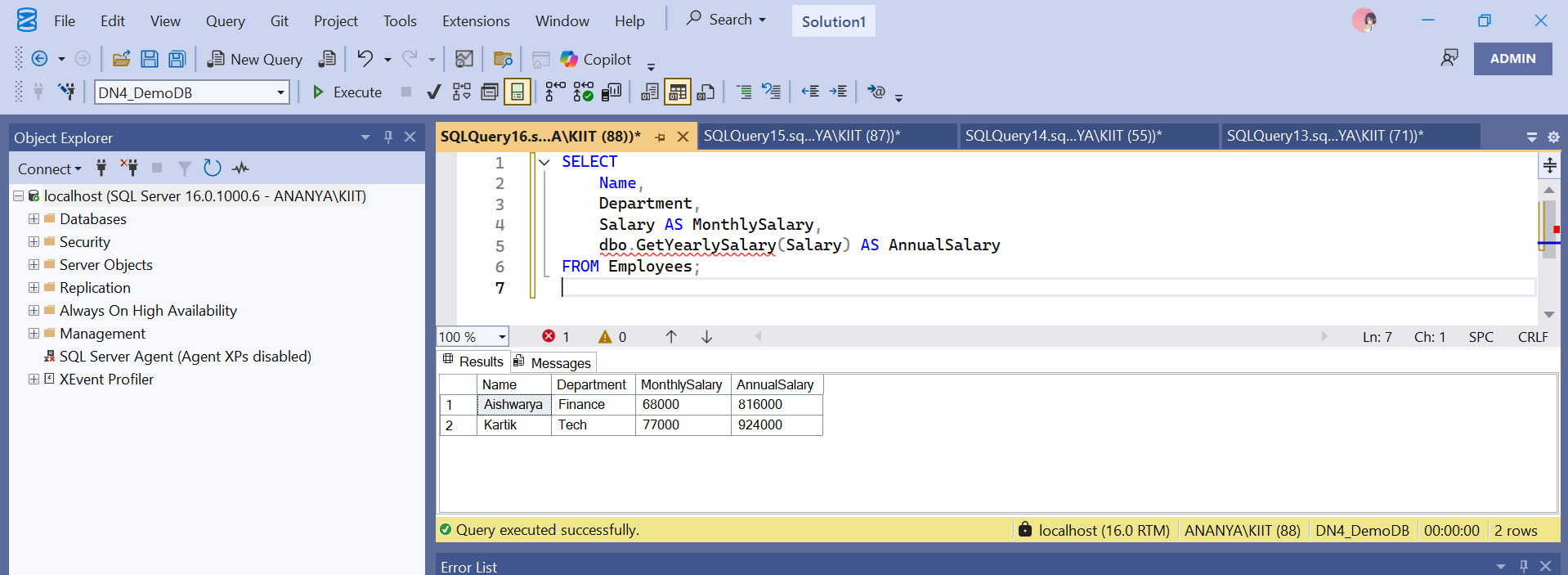
Salary AS MonthlySalary,

dbo.GetYearlySalary(Salary) AS AnnualSalary

FROM Employees;







## 5. ****OPTIONAL TASK — CREATE INDEX****

-- Create non-clustered index on Department column

USE DN4\_DemoDB;

GO

IF EXISTS (

SELECT name

FROM sys.indexes

WHERE name = 'IX\_Employees\_Department'

)

DROP INDEX IX\_Employees\_Department ON Employees;

GO

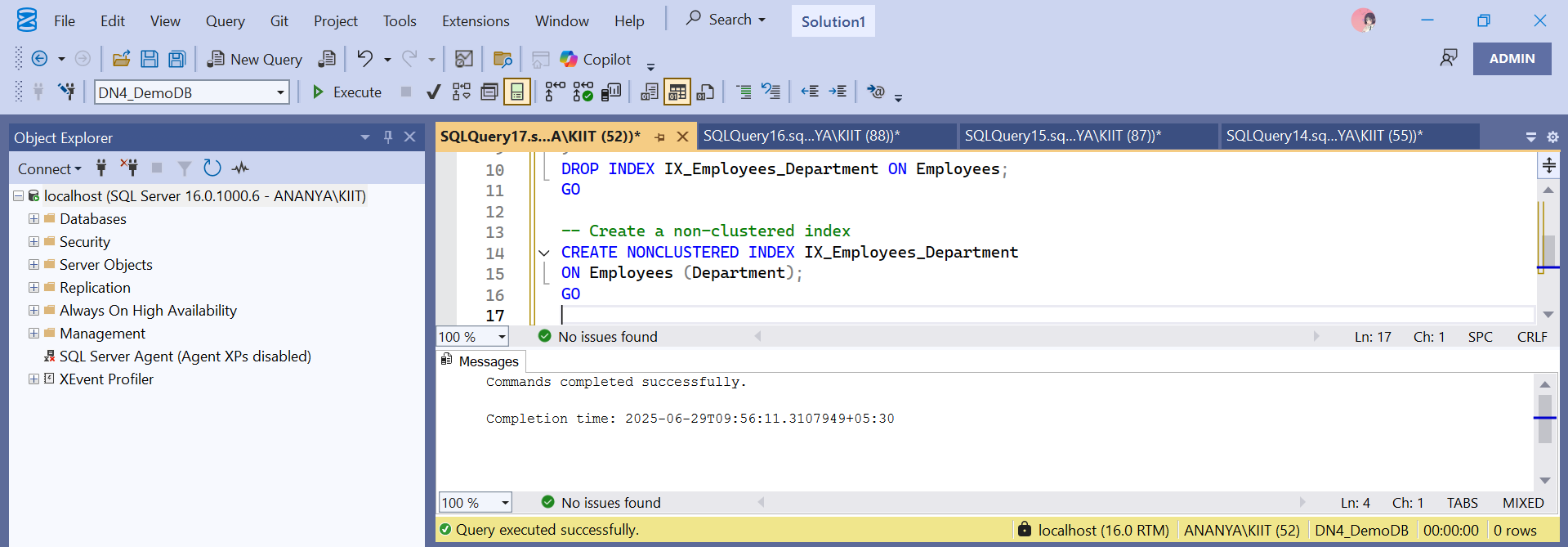
CREATE NONCLUSTERED INDEX IX\_Employees\_Department

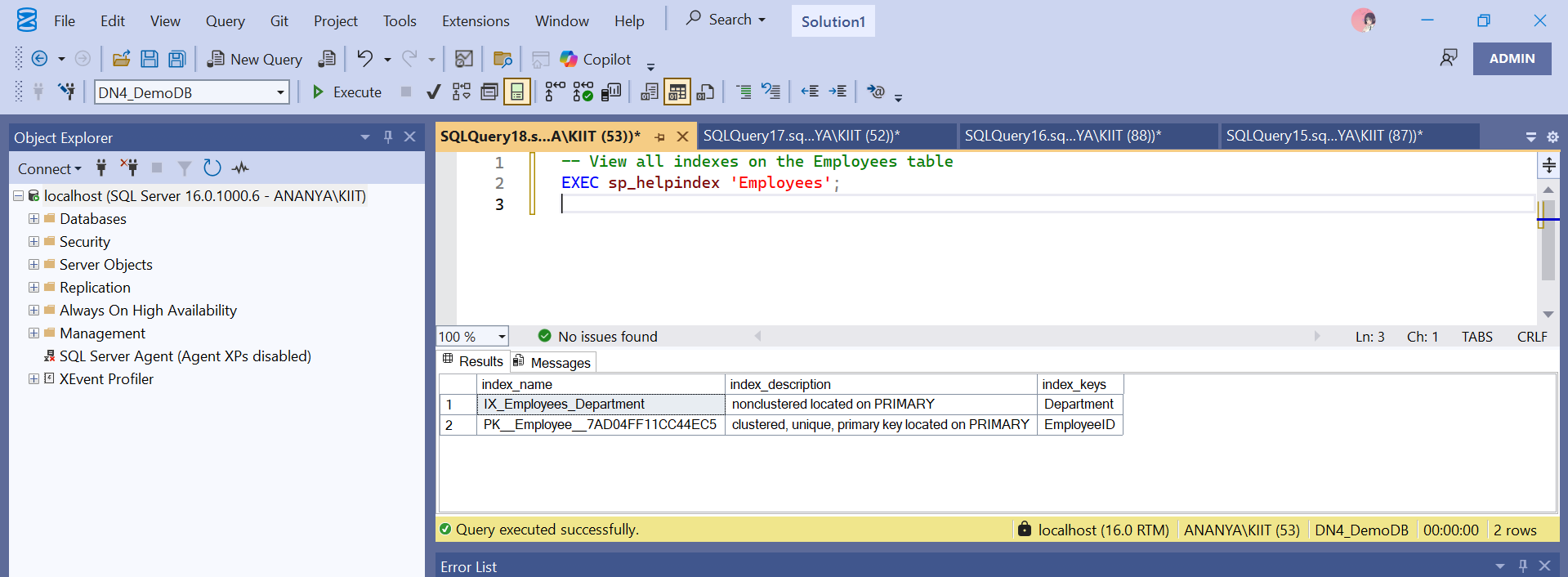
ON Employees (Department);

GO

-- Check Index

EXEC sp\_helpindex 'Employees';





## Submission Format

Submitted by: Ananya Upadhyay

Week 2 — Advanced SQL

Cognizant DN4.0 DotNet FSE Track