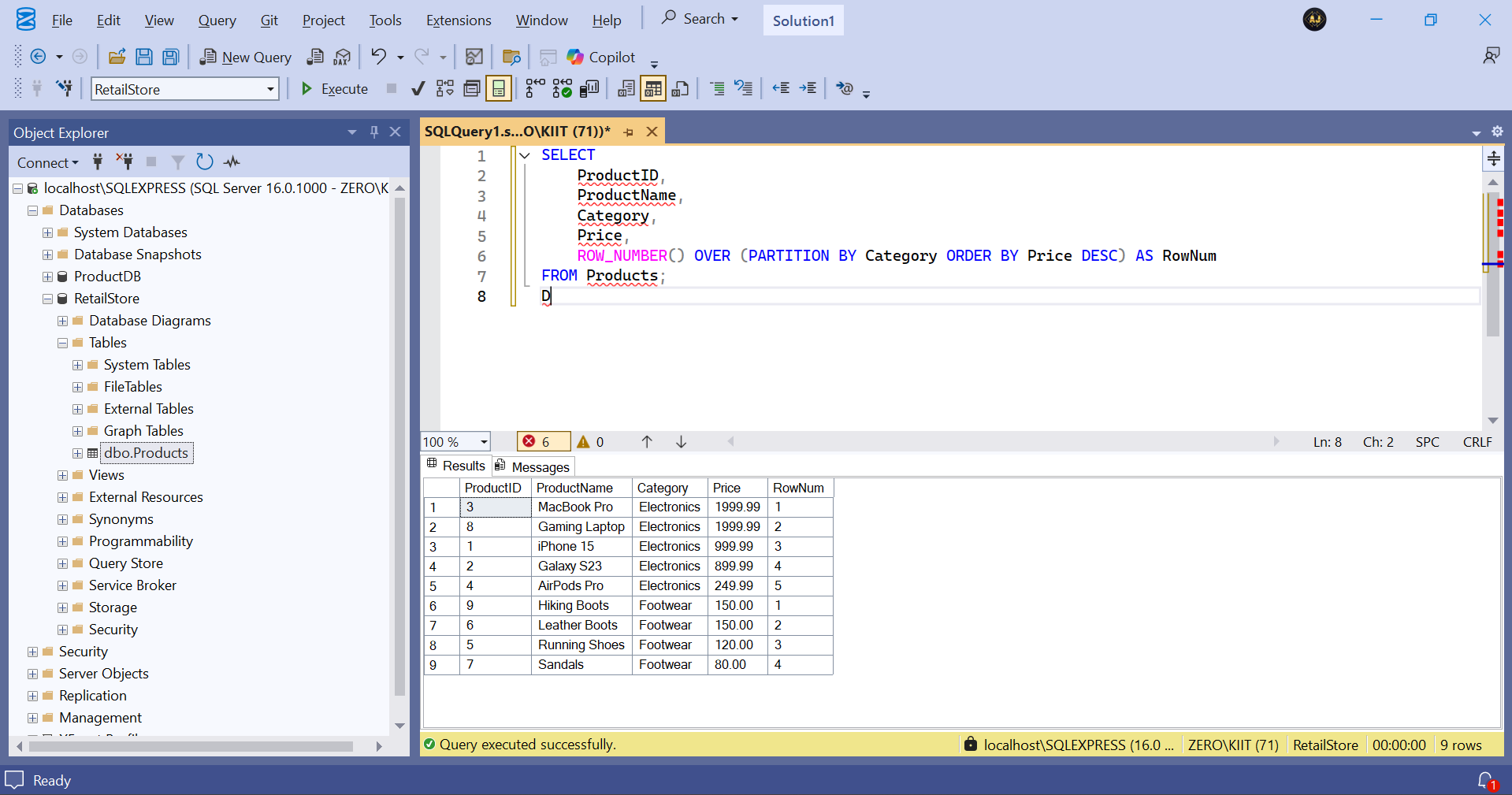
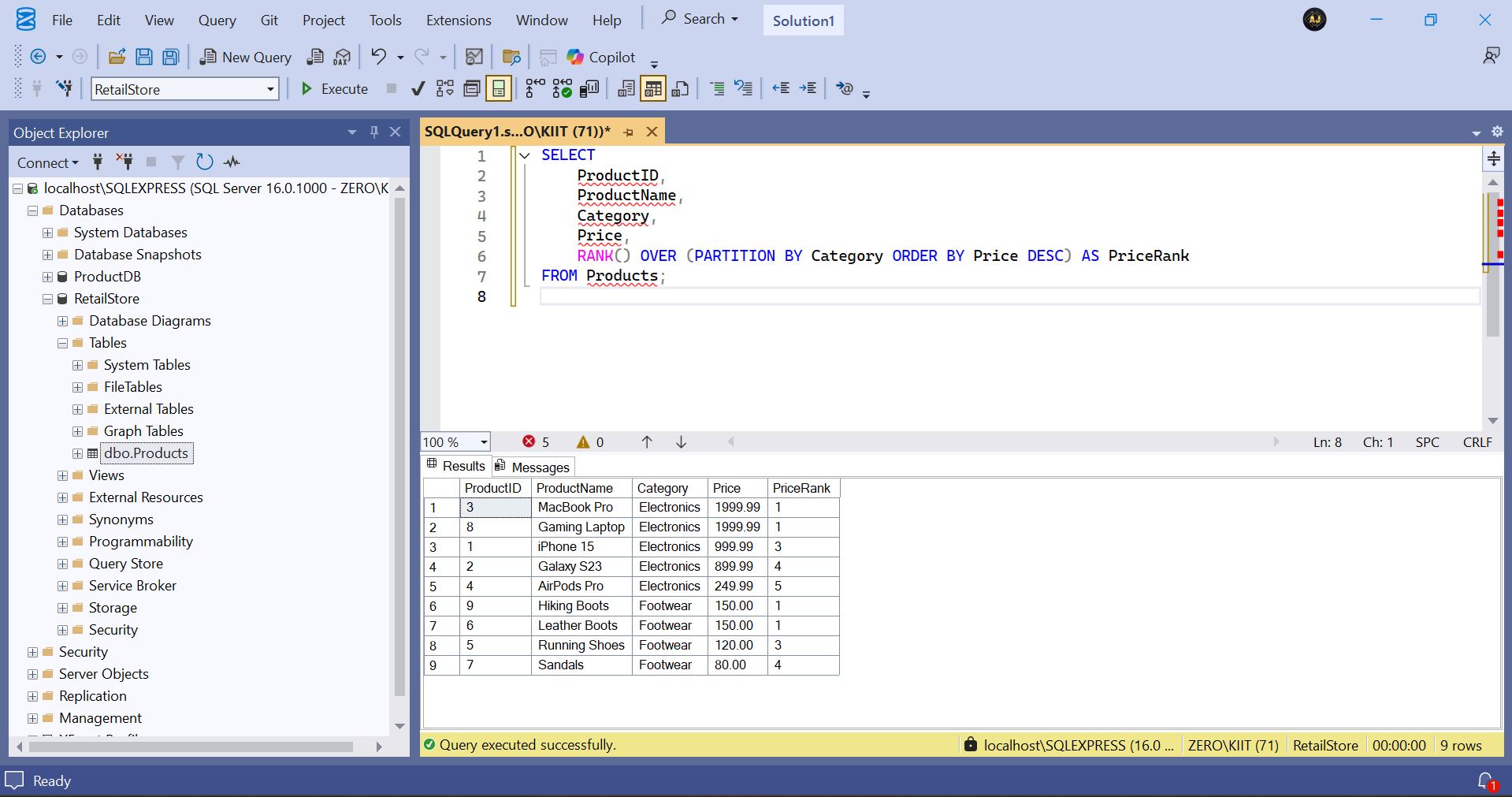
**1. SQL Exercise - Advanced concepts**

**Exercise 1: Ranking and Window Functions**

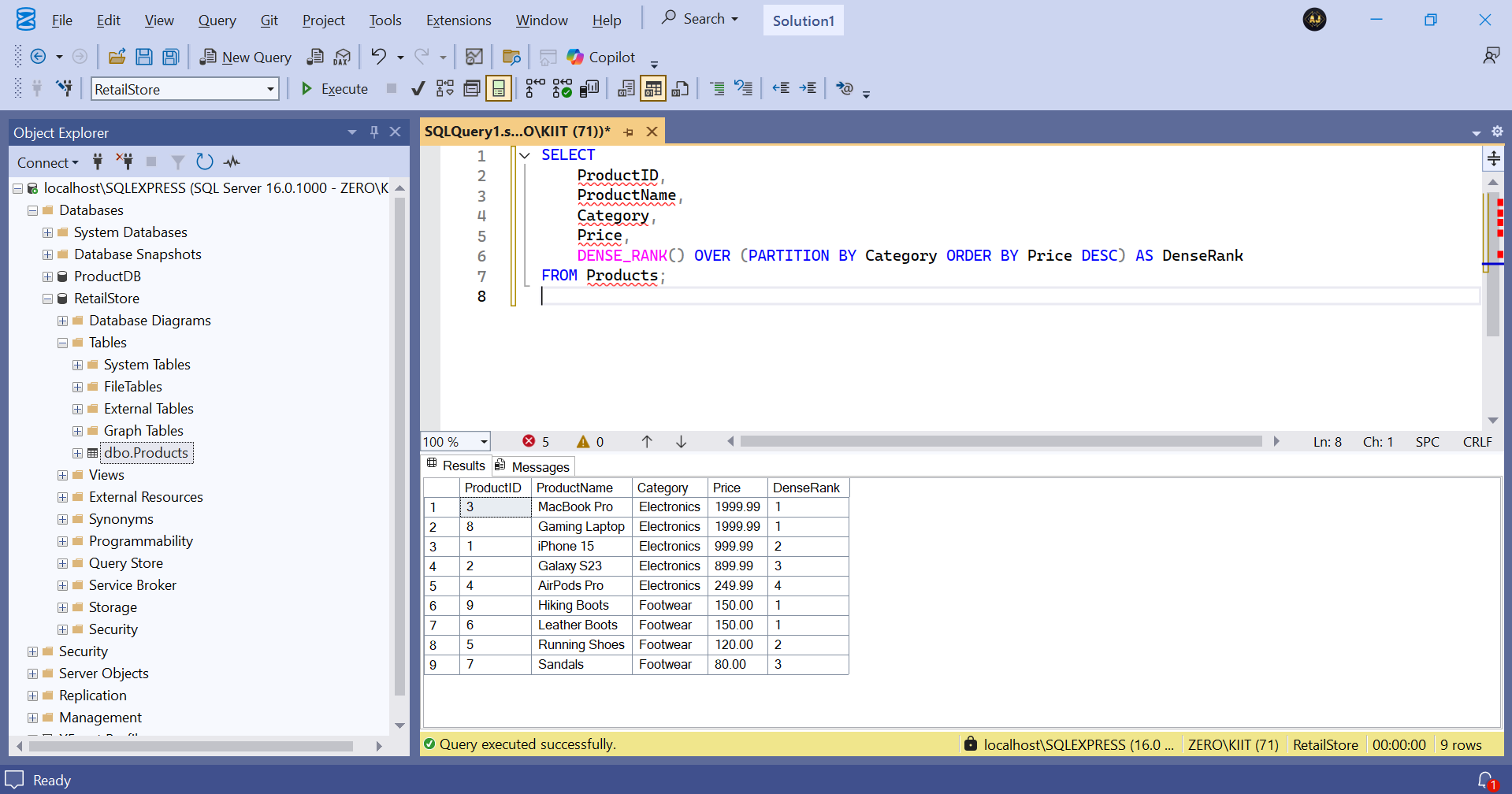
1. ROW\_NUMBER() — Unique Rank Per Category



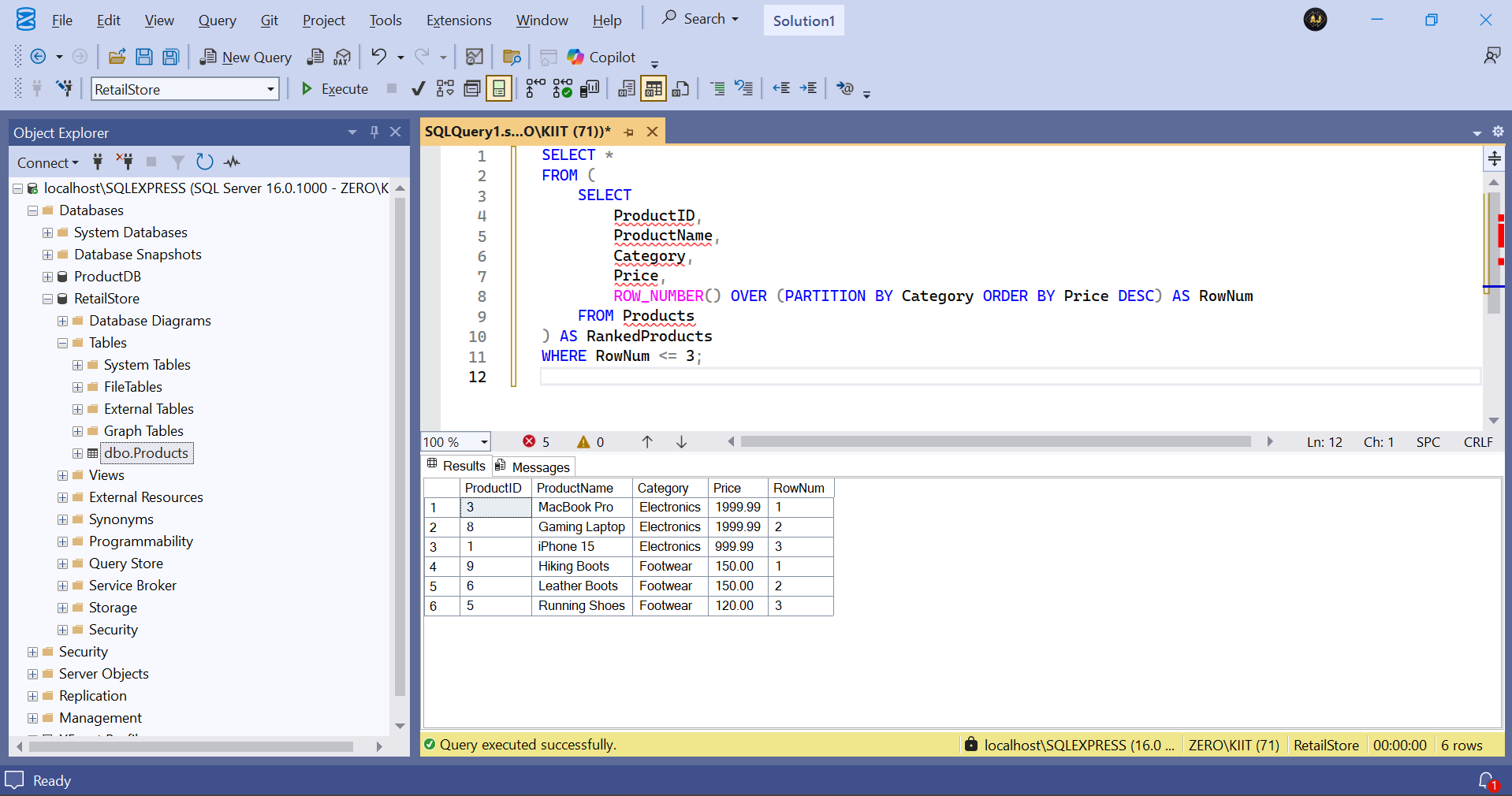
2. RANK() — Allows Ties but Skips Numbers



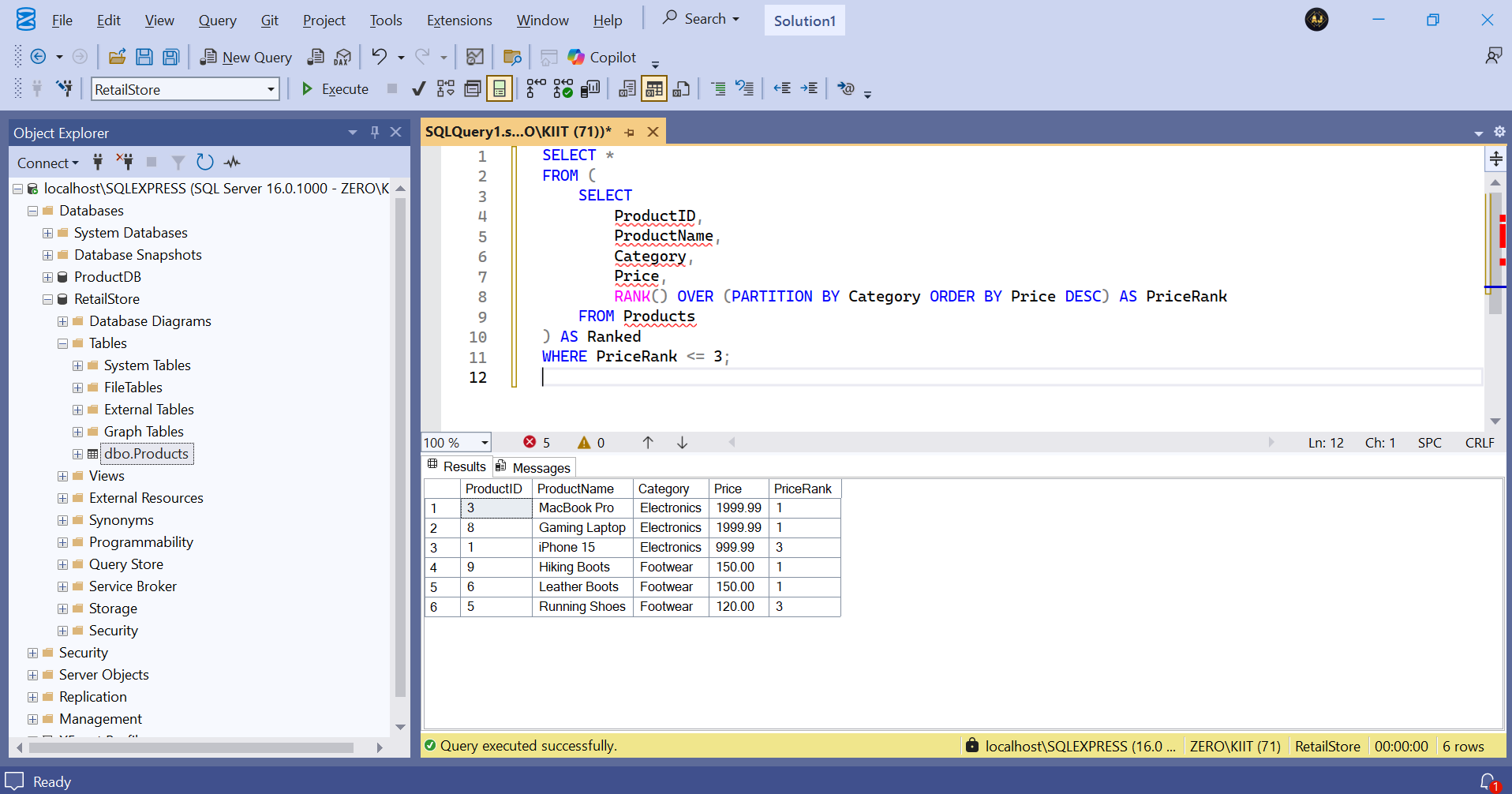
1. DENSE\_RANK() — Allows Ties with No Gaps



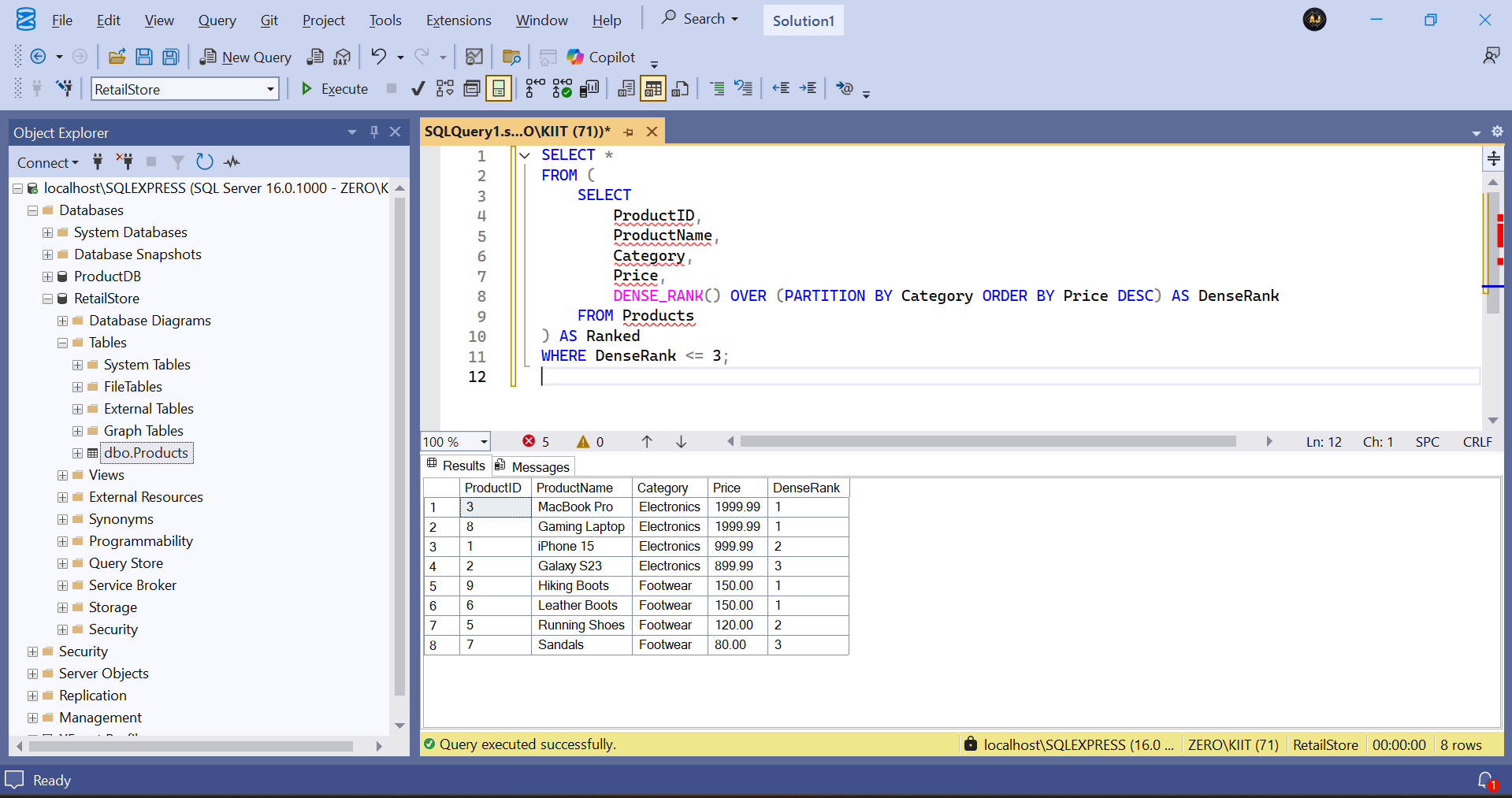
Using ROW\_NUMBER() — Strict Top 3



Using RANK() — Includes Tied Prices Even If More Than 3

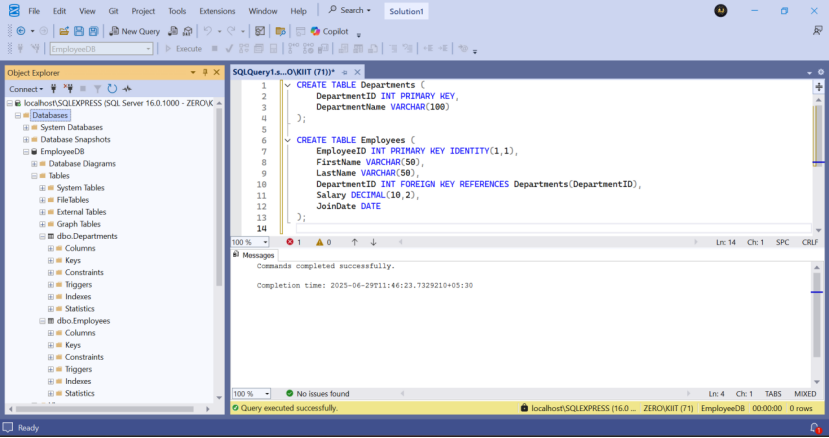


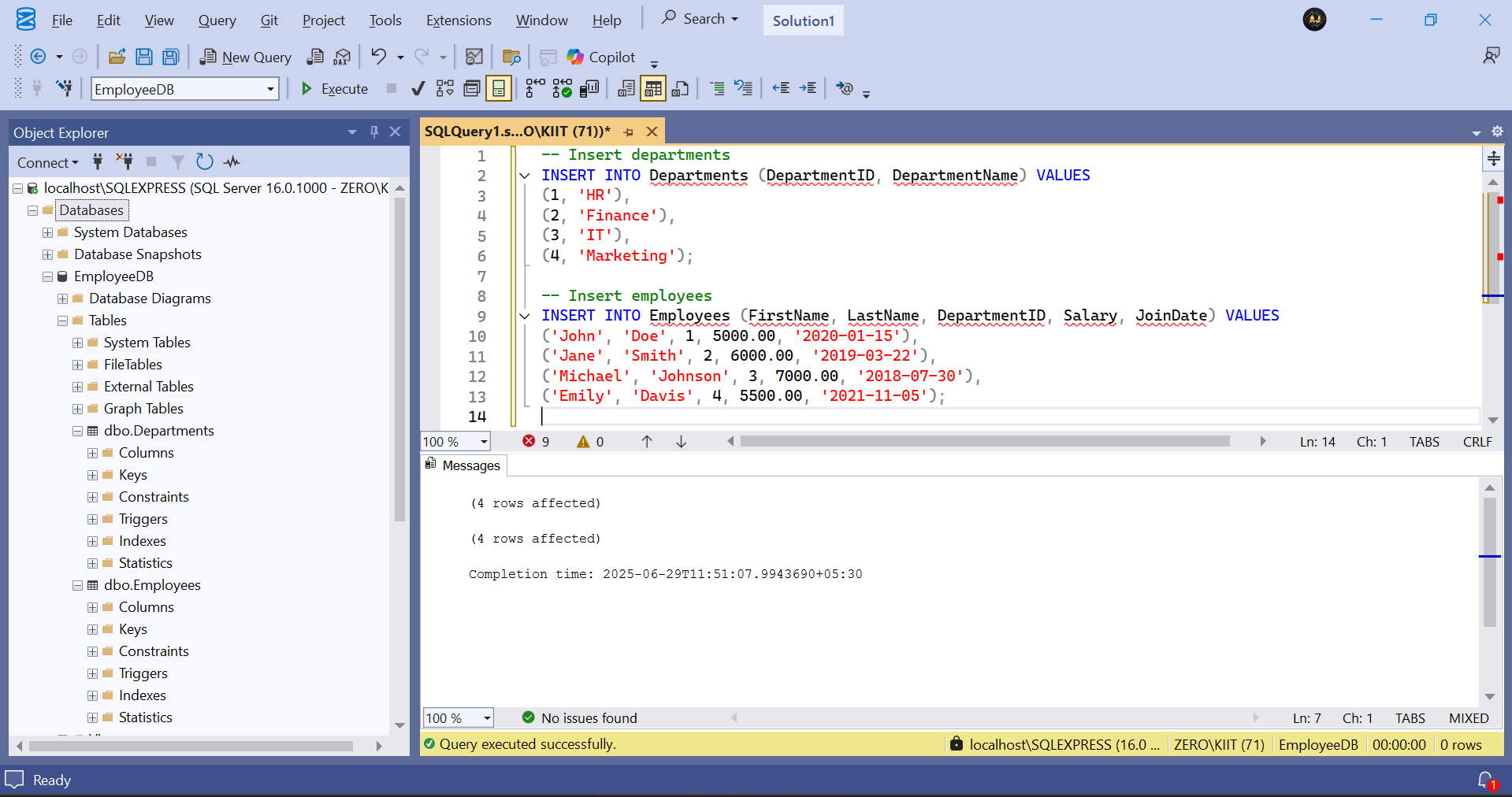
Using DENSE\_RANK() — Same as Above But Without Gaps



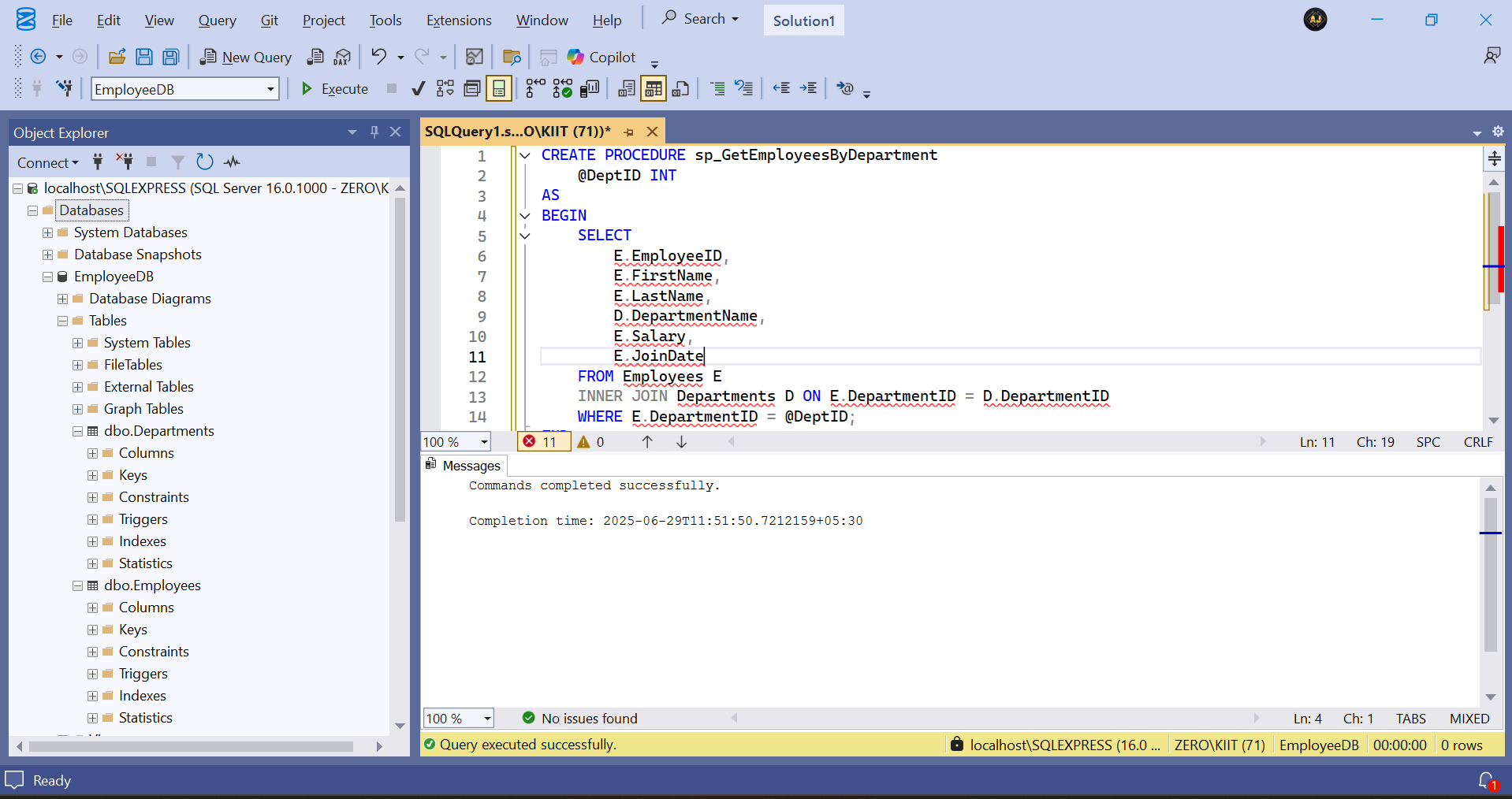
**4. SQL Exercise - Stored procedure**

**Exercise 1: Create a Stored Procedure**

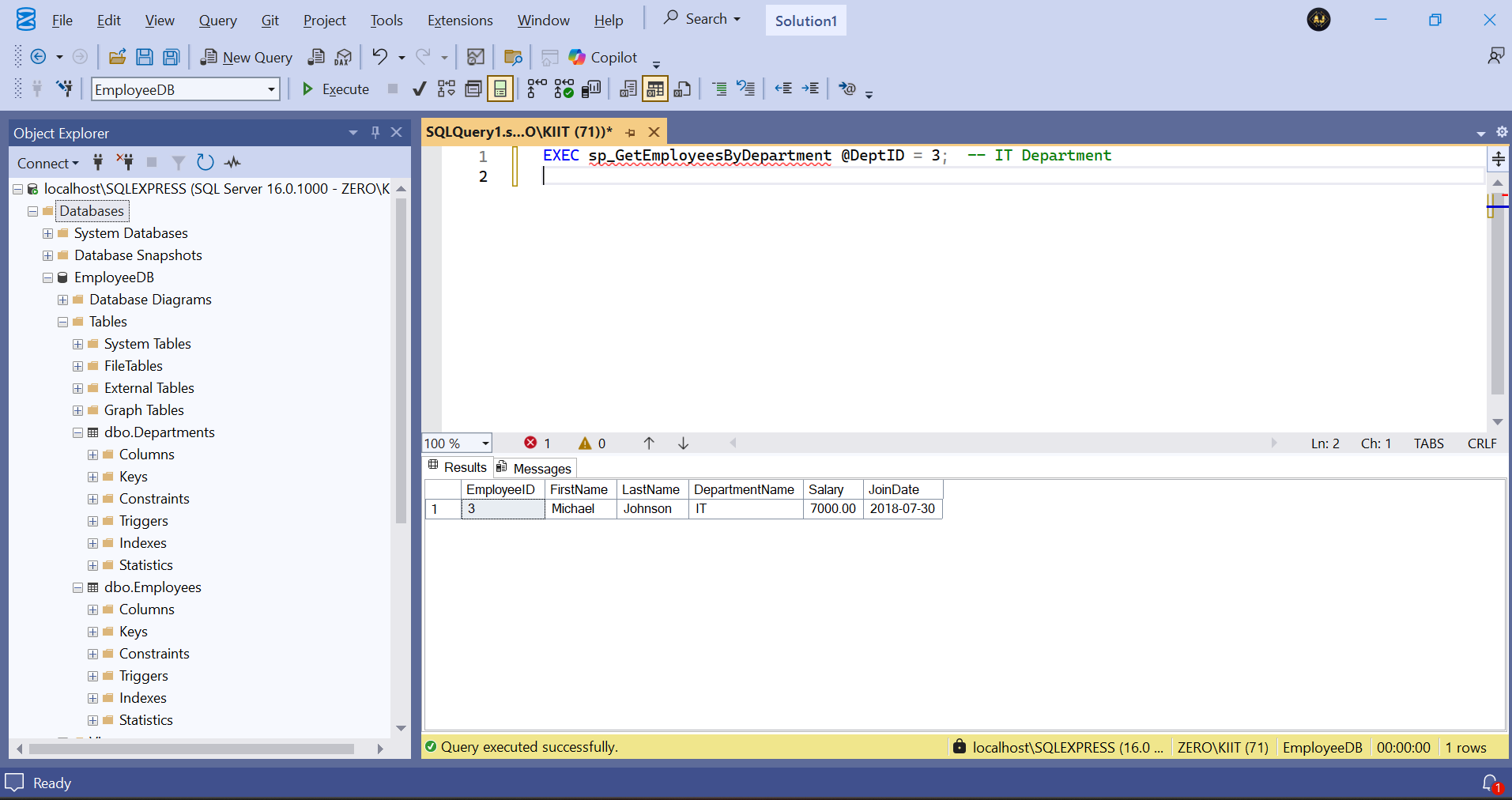
Creation of Employees and Department Tables:

Data Insertion:  


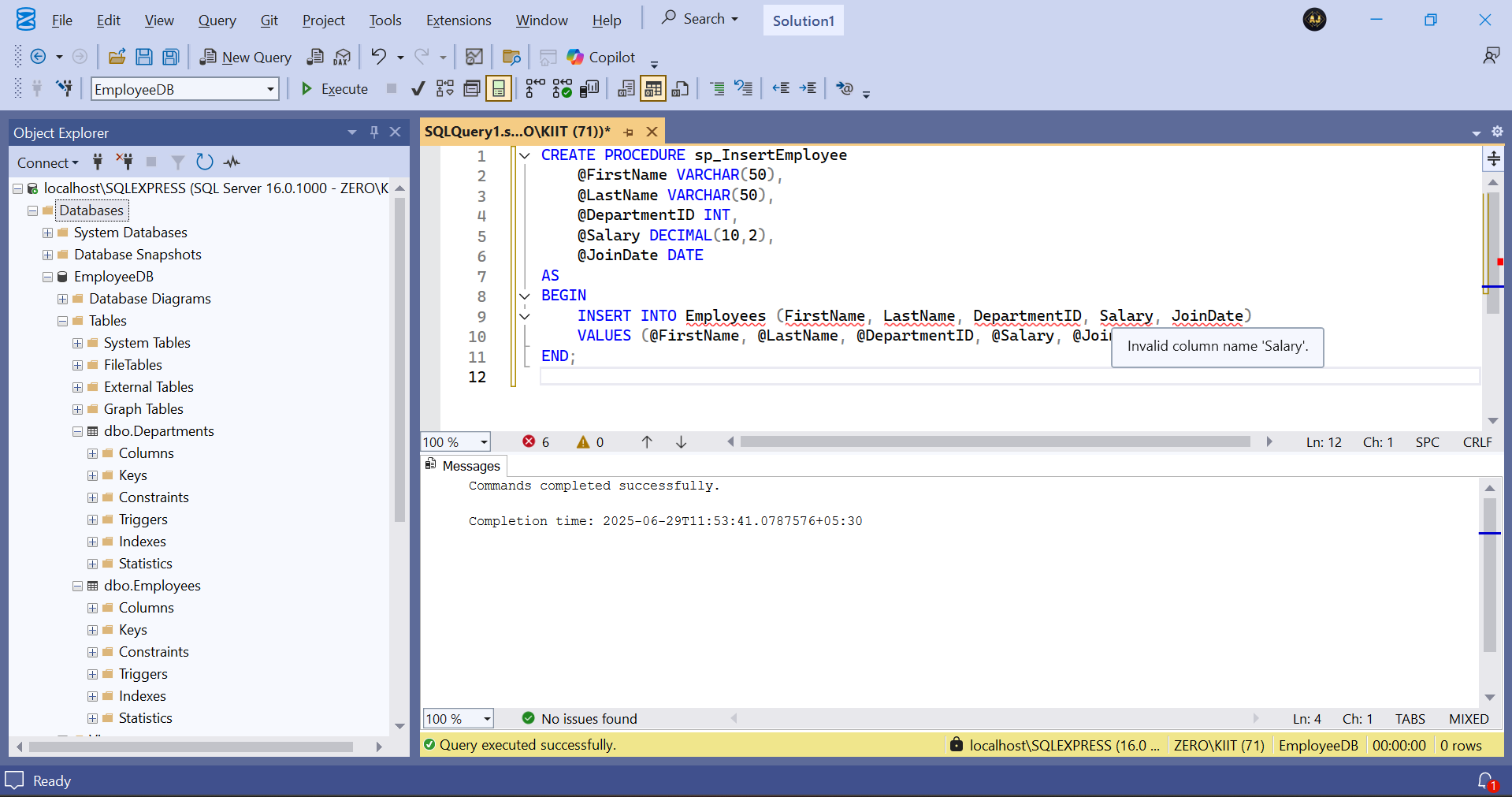
Created Procedure sp\_GetEmployeesByDepartment:



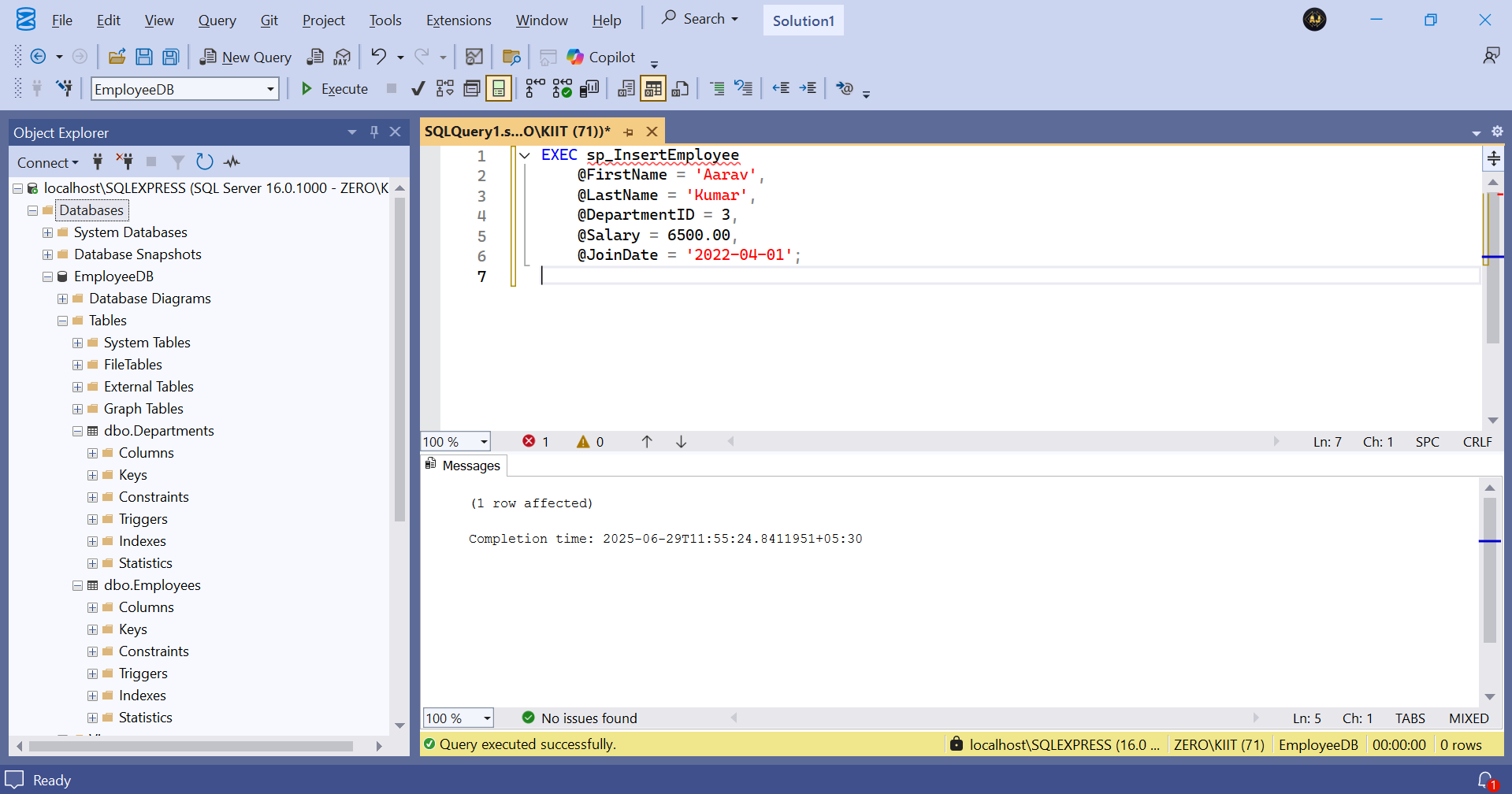
Procedure Execution:



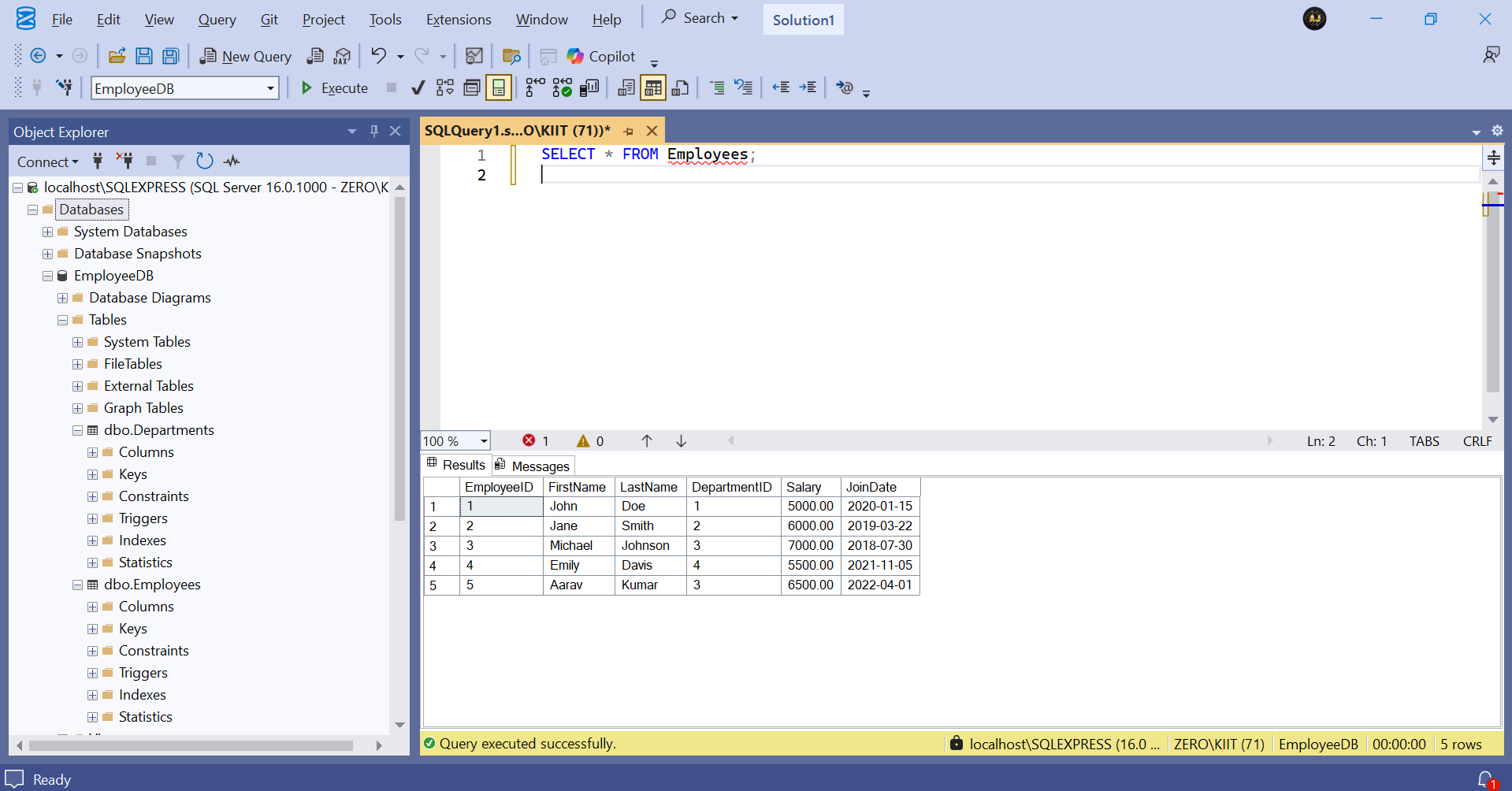
Created the sp\_InsertEmployee Procedure:



To insert a new employee using procedure:

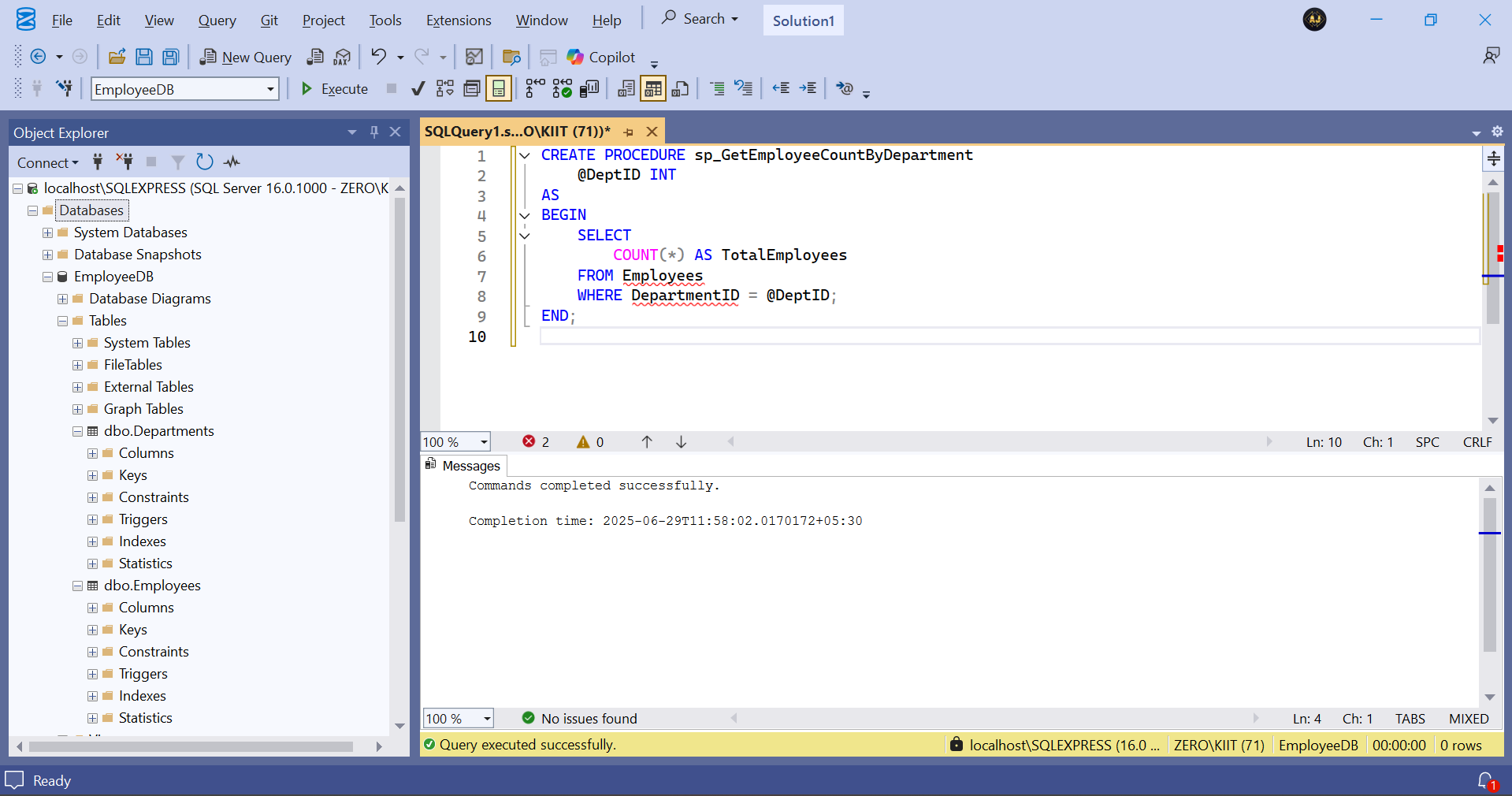


Employees Table:

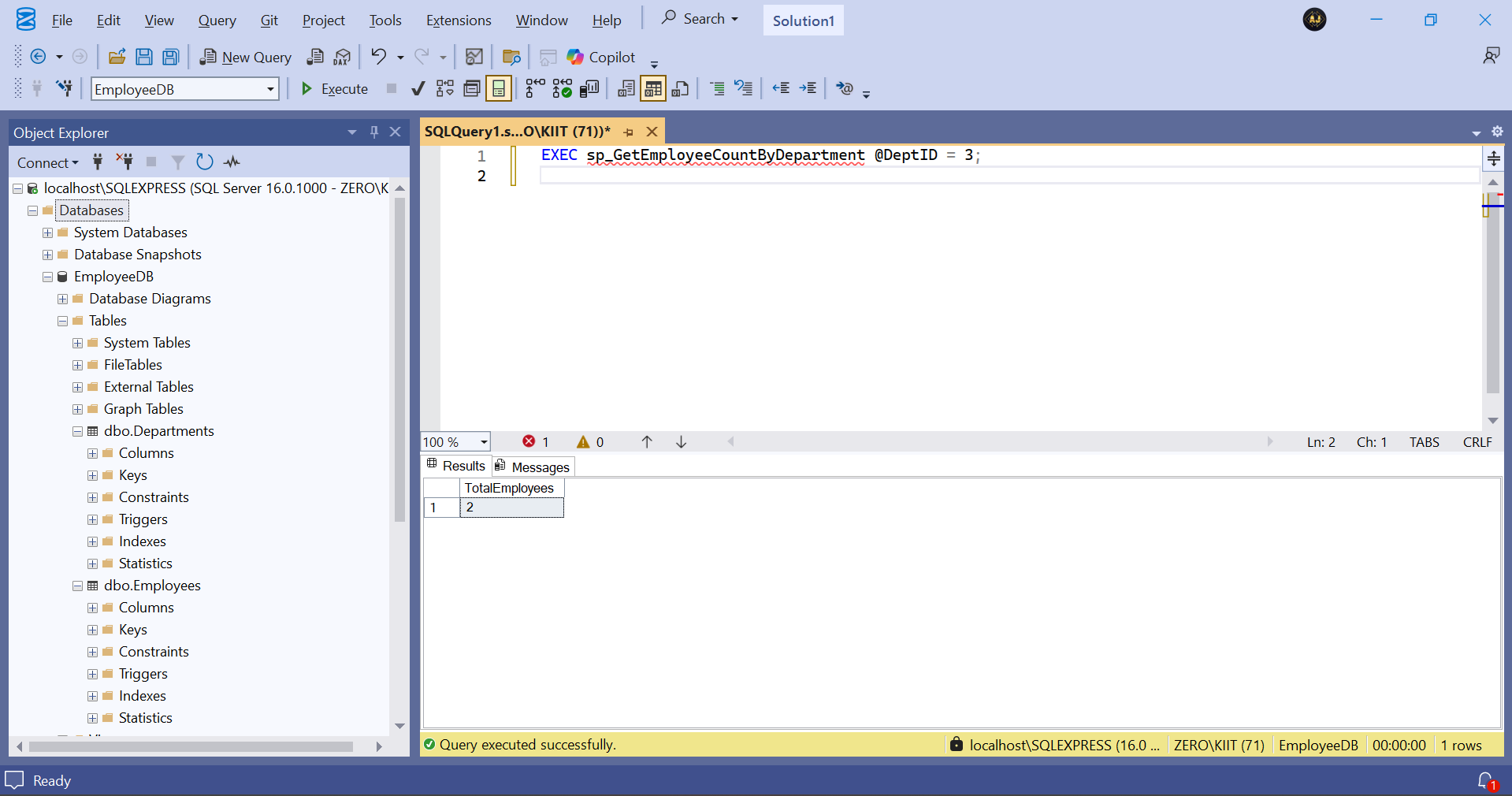


**Exercise 5: Return Data from a Stored Procedure**

Created a new procedure called sp\_GetEmployeeCountByDepartment:

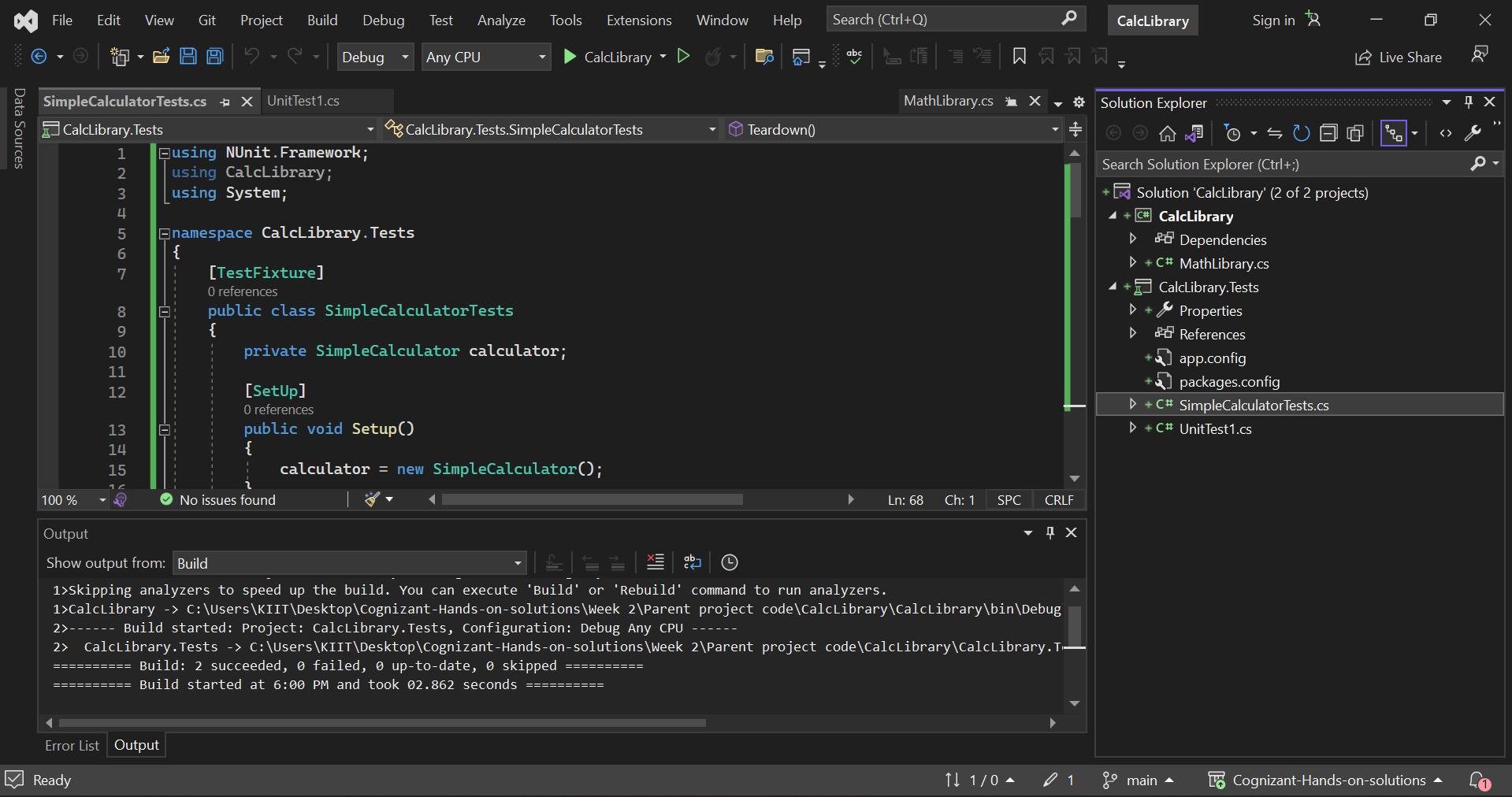


Execution of sp\_GetEmployeeCountByDepartment Procedure:

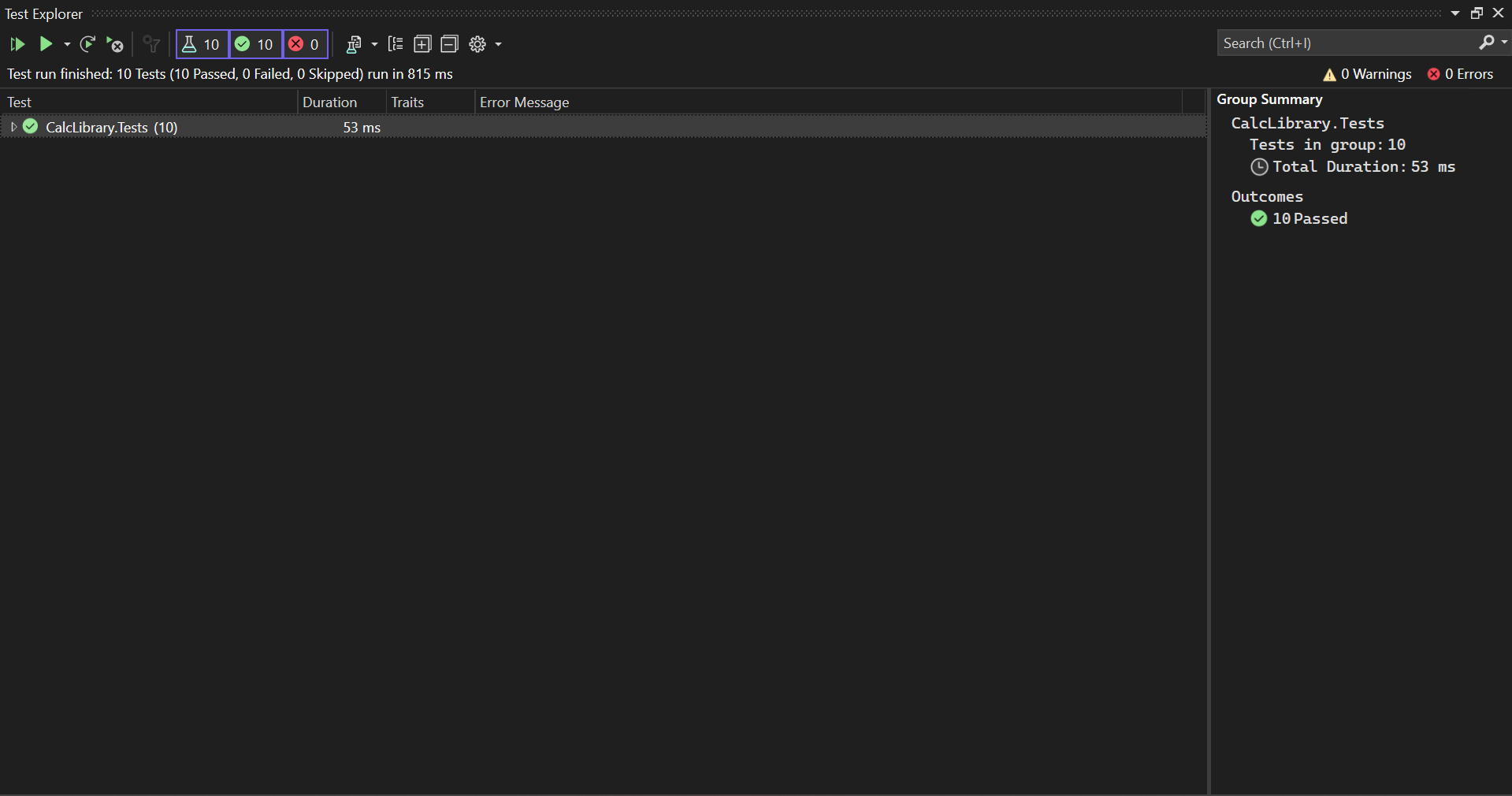


**1. NUnit-Handson**

Made a Unit Test project called CalcLibrary.Tests and added a reference to the CalcLibrary Project and made a new class called SimpleCalculatorTests.cs:



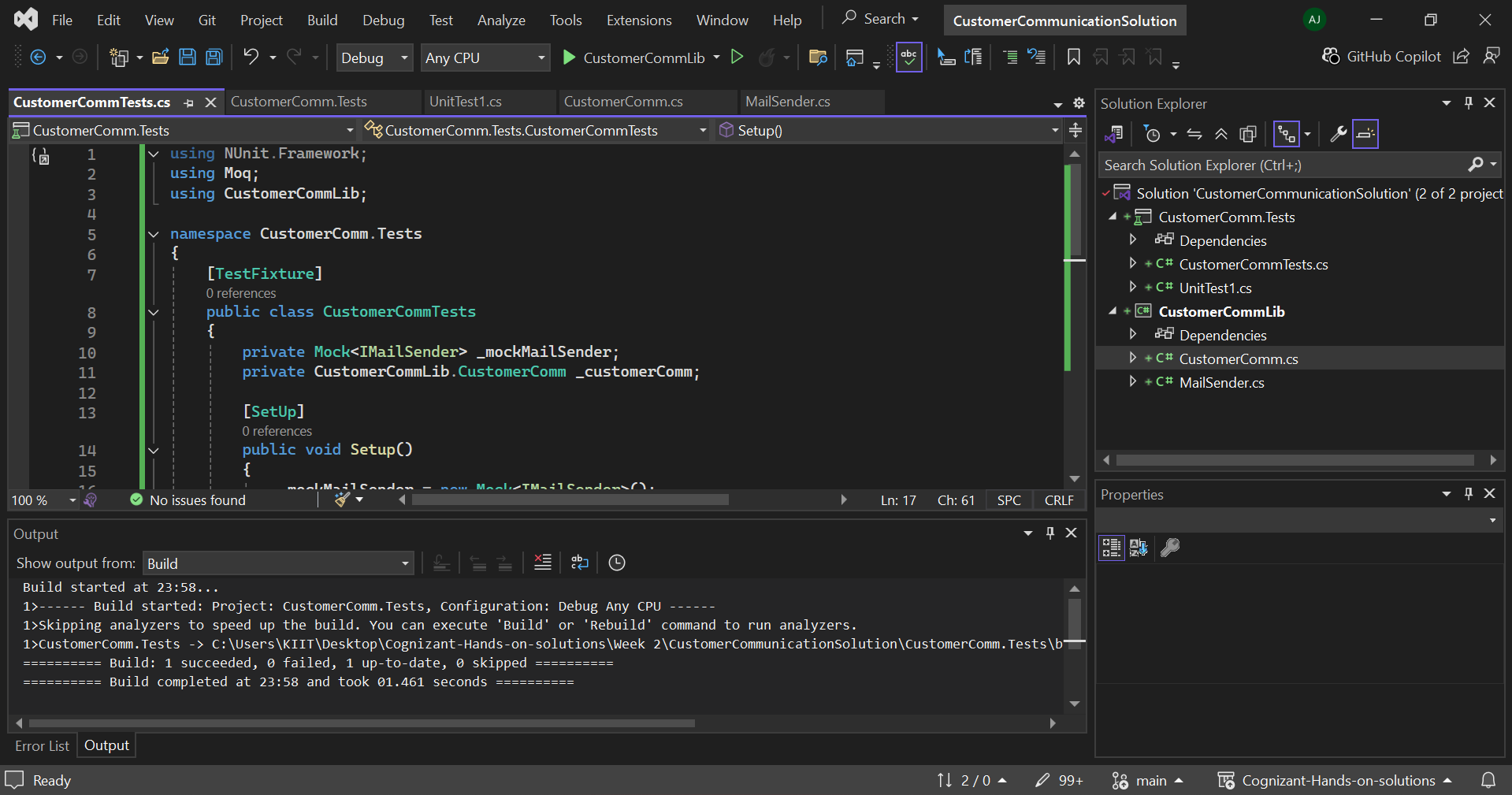
After running all the test cases:



**1.Moq-Handson**

1.Write Testable Code with Moq:

Made two project folders, one for Mail Sender logic and another for testing:



After running all the testcases:

