**Advanced SQL Exercises**

**Exercise 1: Ranking and Window Functions**

Goal: Use ROW\_NUMBER(), RANK(), DENSE\_RANK(), OVER(), and PARTITION BY. Scenario:

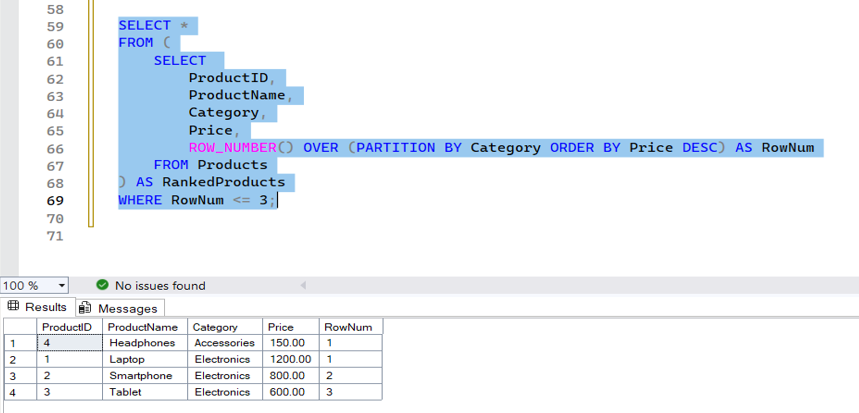
Find the top 3 most expensive products in each category using different ranking functions. Steps:

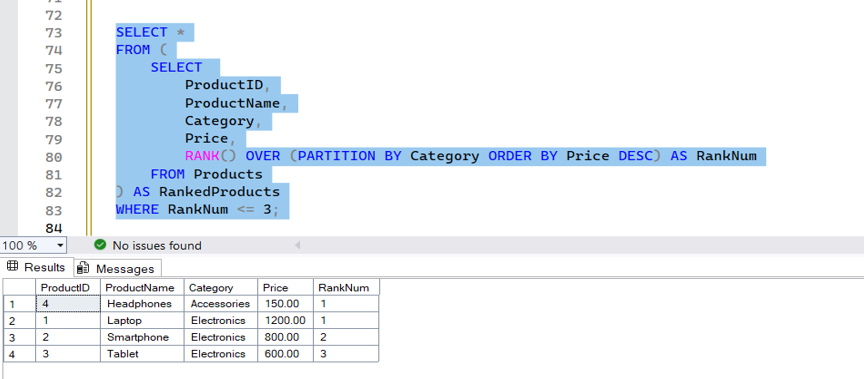
1. Use ROW\_NUMBER() to assign a unique rank within each category.

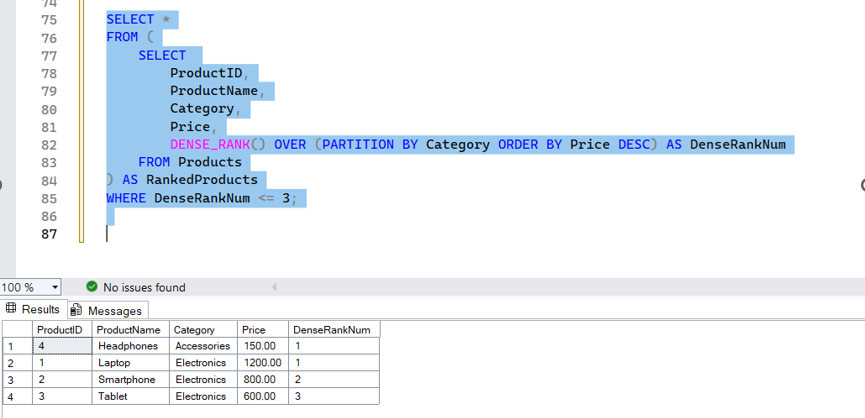
2. Use RANK() and DENSE\_RANK() to compare how ties are handled.

3. Use PARTITION BY Category and ORDER BY Price DESC

**Implementation**







**Exercise 2: Create a Stored Procedure**

Goal: Create a stored procedure to retrieve employee details by department.

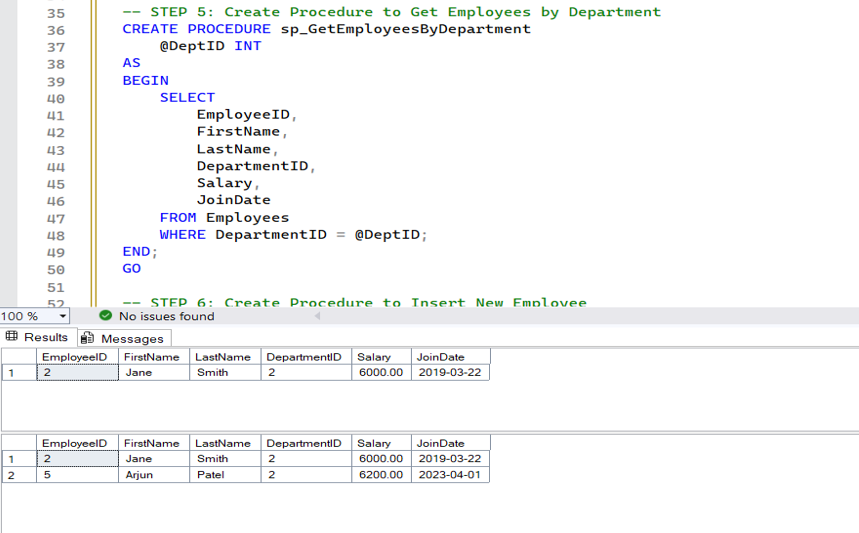
Steps:

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;

**Implementation:**



**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

**Implementation:**

