Advanced SQL

Exercise 1: Ranking and Window Functions

*-- Creating sample products table*

CREATE TABLE products (

product\_id INT IDENTITY(1,1) PRIMARY KEY,

product\_name NVARCHAR(100),

category NVARCHAR(50),

price DECIMAL(10,2),

stock\_quantity INT,

brand NVARCHAR(50)

);

*-- Inserting sample data*

*INSERT INTO products (product\_id, product\_name, category, price, stock\_quantity, brand) VALUES*

*(1, N'iPhone 15 Pro', N'Electronics', 999.99, 50, N'Apple'),*

*(2, N'Samsung Galaxy S24', N'Electronics', 899.99, 45, N'Samsung'),*

*(3, N'MacBook Pro 16"', N'Electronics', 2499.99, 20, N'Apple'),*

*(4, N'Dell XPS 13', N'Electronics', 1299.99, 30, N'Dell'),*

*(5, N'Sony WH-1000XM5', N'Electronics', 399.99, 100, N'Sony'),*

*(6, N'Levi''s 501 Jeans', N'Clothing', 89.99, 200, N'Levi''s'),*

*(7, N'Nike Air Max 270', N'Clothing', 149.99, 150, N'Nike'),*

*(8, N'Adidas Ultraboost', N'Clothing', 179.99, 120, N'Adidas'),*

*(9, N'Patagonia Jacket', N'Clothing', 299.99, 80, N'Patagonia'),*

*(10, N'North Face Backpack', N'Clothing', 129.99, 90, N'North Face'),*

*(11, N'Cuisinart Coffee Maker', N'Home & Kitchen', 149.99, 60, N'Cuisinart'),*

*(12, N'KitchenAid Mixer', N'Home & Kitchen', 379.99, 40, N'KitchenAid'),*

*(13, N'Dyson V15 Vacuum', N'Home & Kitchen', 749.99, 25, N'Dyson'),*

*(14, N'Instant Pot Duo', N'Home & Kitchen', 99.99, 80, N'Instant Pot'),*

*(15, N'Ninja Blender', N'Home & Kitchen', 199.99, 70, N'Ninja');*

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

SELECT

product\_id,

product\_name,

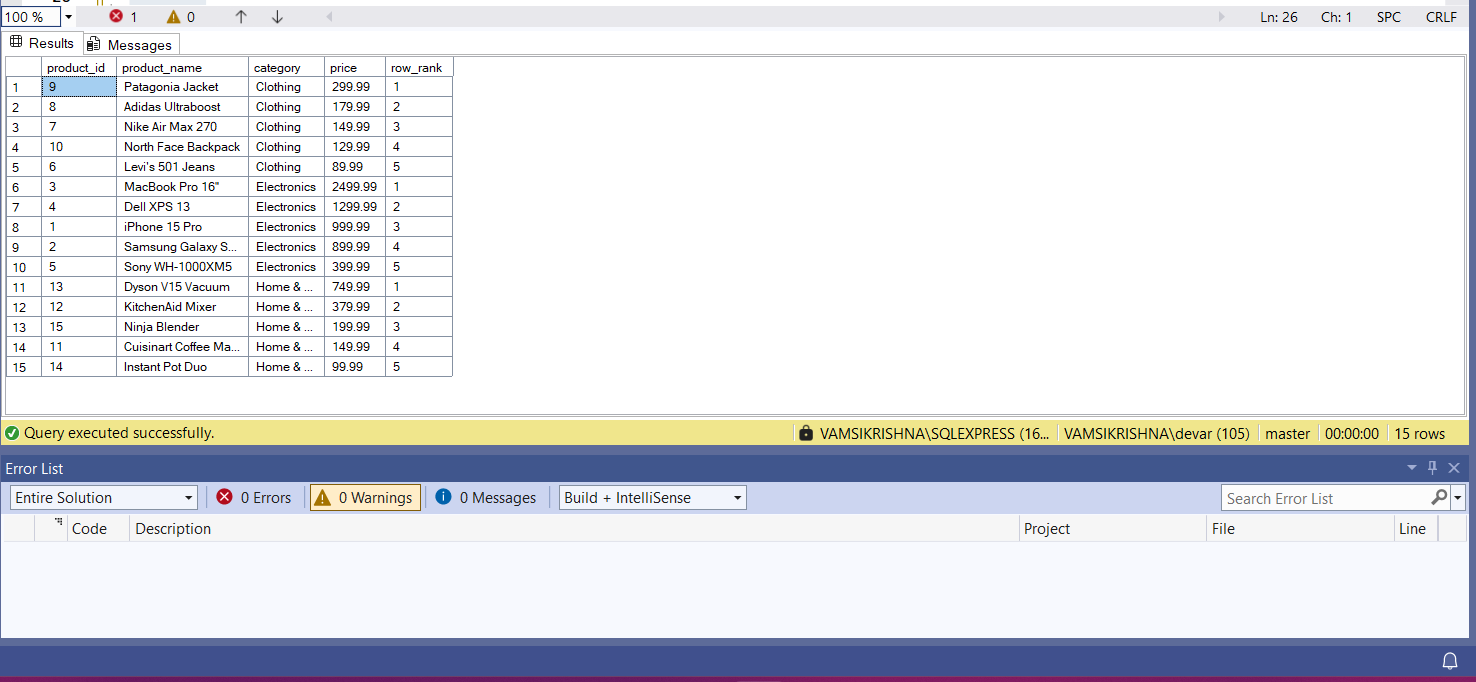
category,

price,

ROW\_NUMBER() OVER (PARTITION BY category ORDER BY price DESC) as row\_rank

FROM products

ORDER BY category, price DESC;



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

SELECT

product\_name,

category,

price,

RANK() OVER (PARTITION BY category ORDER BY price DESC) as rank\_with\_gaps,

DENSE\_RANK() OVER (PARTITION BY category ORDER BY price DESC) as dense\_rank\_no\_gaps,

ROW\_NUMBER() OVER (PARTITION BY category ORDER BY price DESC) as row\_number

FROM products

WHERE category = 'Electronics'

ORDER BY price DESC;

A screenshot of a computer

AI-generated content may be incorrect.

3. Use PARTITION BY Category and ORDER BY Price DESC.

WITH ranked\_products AS (

SELECT

product\_id,

product\_name,

category,

price,

brand,

DENSE\_RANK() OVER (PARTITION BY category ORDER BY price DESC) as price\_rank

FROM products

)

SELECT

category,

product\_name,

price,

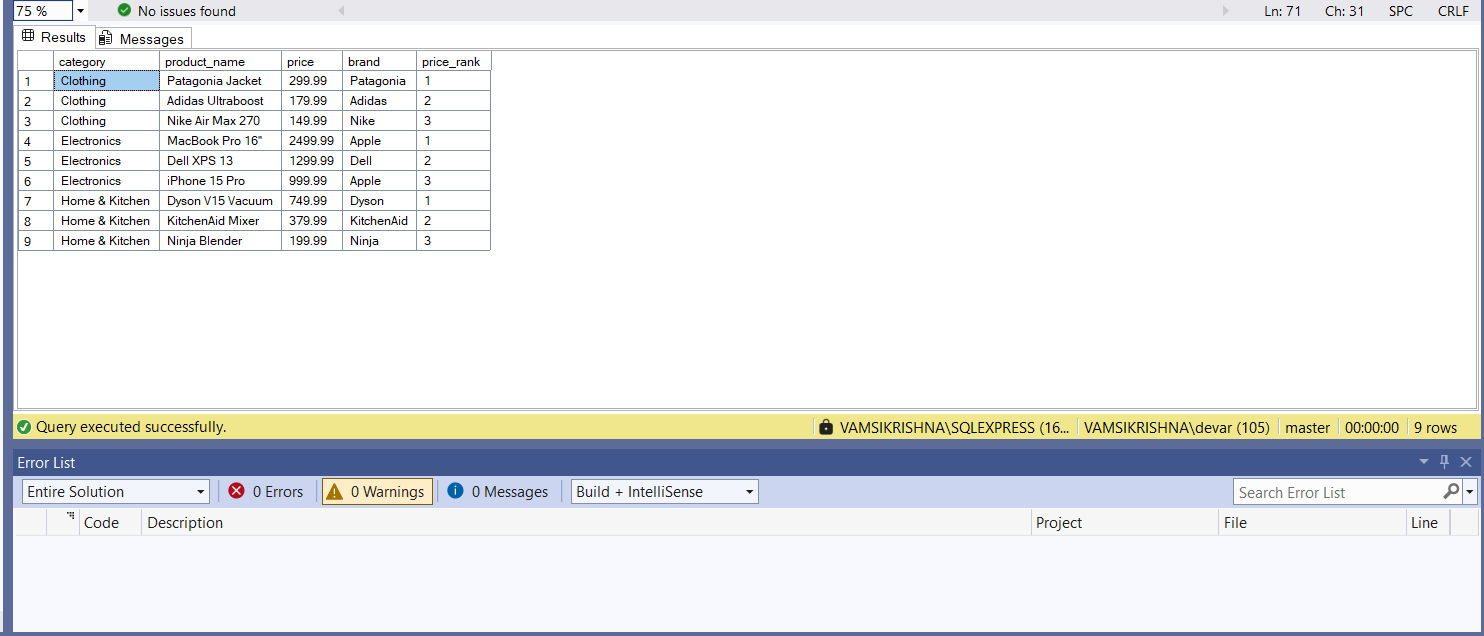
brand,

price\_rank

FROM ranked\_products

WHERE price\_rank <= 3

ORDER BY category, price\_rank;



SQL Exercise - Stored procedure

Exercise 1: Create a Stored Procedure

CREATE PROCEDURE sp\_GetEmployeesByDepartment

@DepartmentID INT

AS

BEGIN.

SET NOCOUNT ON;

SELECT

EmployeeID,

FirstName,

LastName,

DepartmentID,

Salary,

JoinDate

FROM

Employees

WHERE

DepartmentID = @DepartmentID;

END;

GO

CREATE PROCEDURE sp\_InsertEmployee

@FirstName VARCHAR(50),

@LastName VARCHAR(50),

@DepartmentID INT,

@Salary DECIMAL(10, 2),

@JoinDate DATE

AS

BEGIN

SET NOCOUNT ON;

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

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

END;

GO

Exercise 5: Return Data from a Stored Procedure

CREATE PROCEDURE sp\_GetEmployeeCountByDepartment

    @DepartmentID INT,

    -- Output parameter

    @EmployeeCount INT OUTPUT

AS

BEGIN

    SET NOCOUNT ON;

    -- SQL query to count employees in the given department

    SELECT @EmployeeCount = COUNT(EmployeeID)

    FROM Employees

    WHERE DepartmentID = @DepartmentID;

END;

GO

