**Name:** Manu Rajesh Naik

**Superset ID:** 6365342

***Advanced SQL Exercises***

**Exercise 1: Ranking and Window Functions**

**Using the Below sample Data (from Exercise 2)**

-- 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

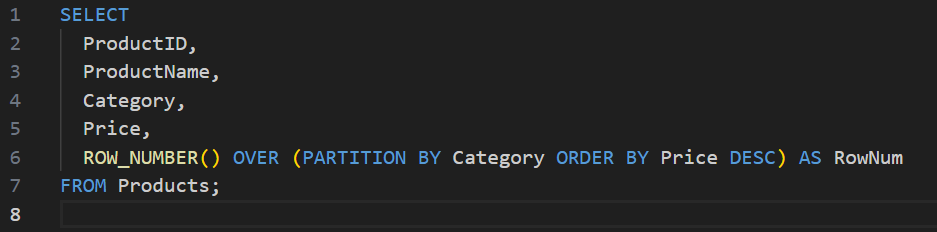
(1, 1, 1, 1),

(2, 2, 2, 2),

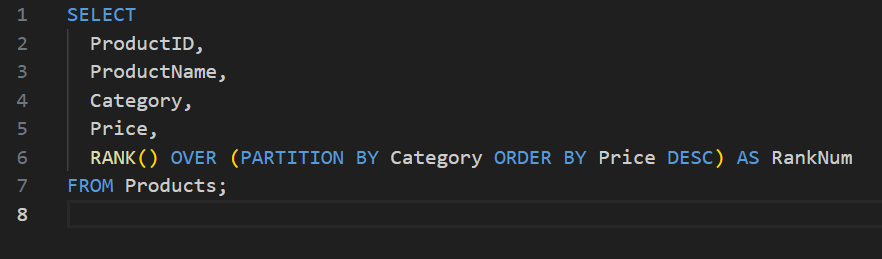
(3, 3, 3, 1),

(4, 4, 4, 3);

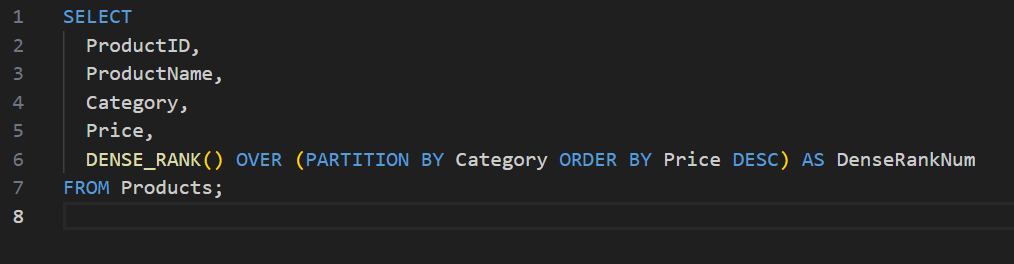
1. Using ROW\_NUMBER() to assign unique rank per category



1. Using RANK() to handle ties (same price → same rank, skips numbers)

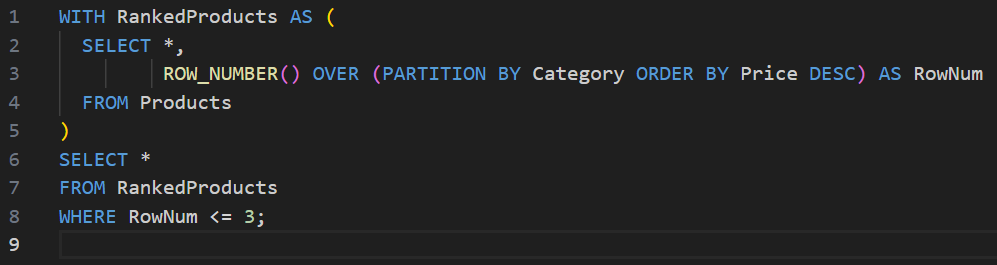


1. Using DENSE\_RANK() to handle ties (same price → same rank, no skips)

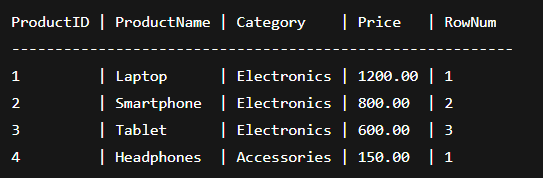


**Now, to filter Top 3 products in each category**

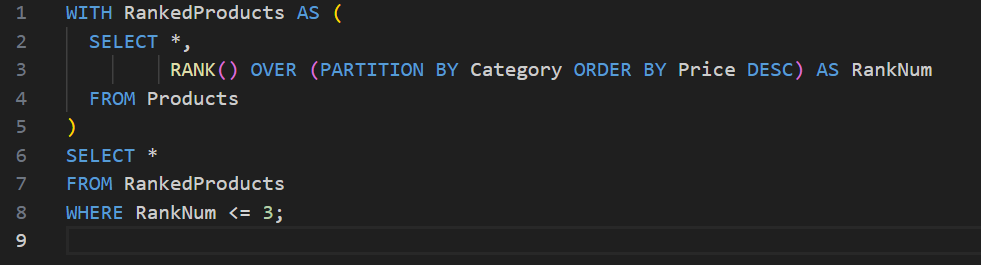
1. ROW\_NUMBER() – Top 3 Products per Category



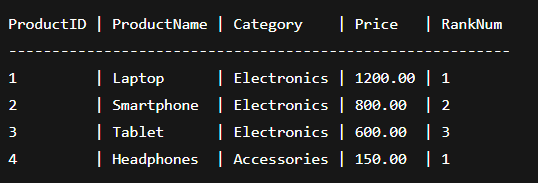
Output:



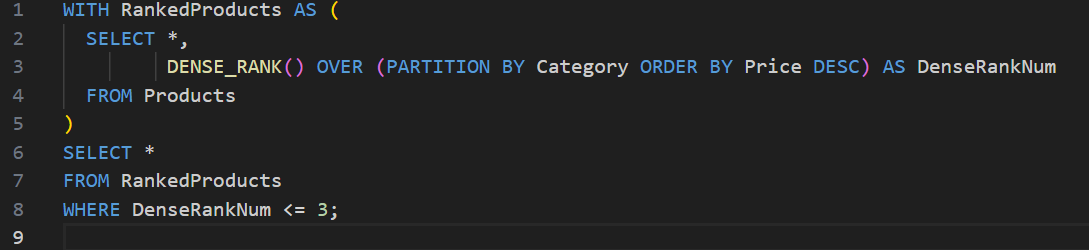
1. RANK() – Top 3 Products per Category (with gap on ties)



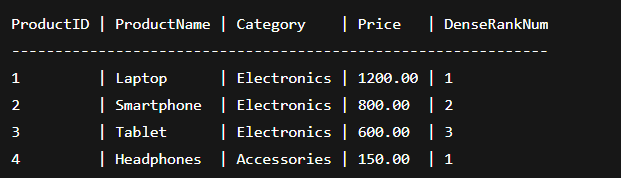
Output:



1. DENSE\_RANK() – Top 3 Products per Category (no gap on ties)

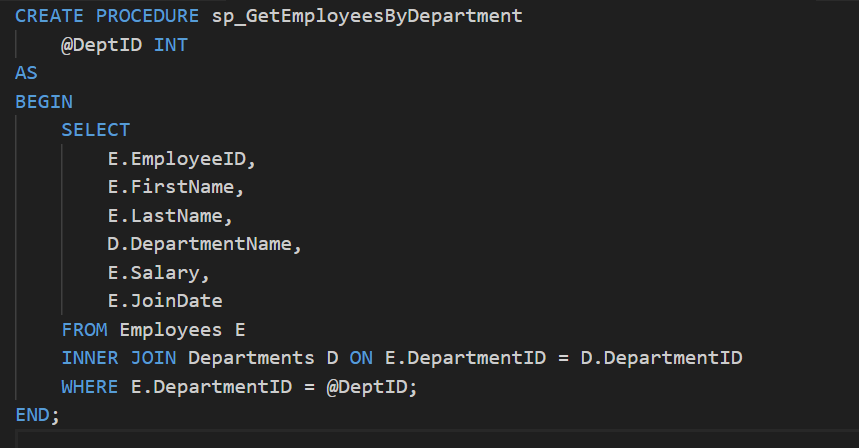


Output:



*SQL stored procedure*

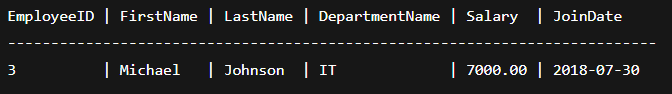
**Exercise 1: Create a Stored Procedure**



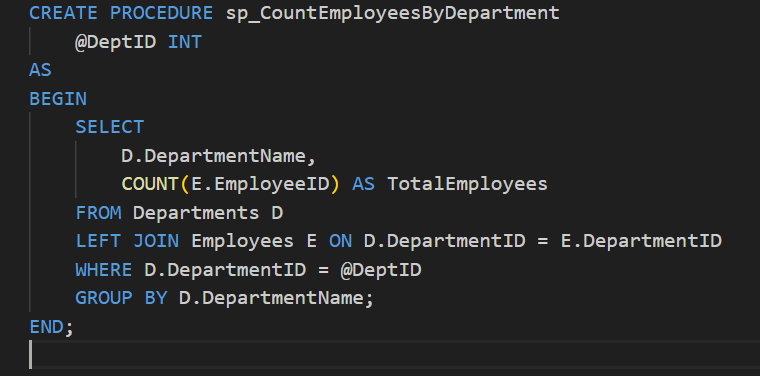
Input:



Output:



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



Input:



Output:

