**Exercise: 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.

**Creating the table:**

CREATE TABLE RetailProducts (

ProductID INT,

ProductName VARCHAR(100),

Category VARCHAR(50),

Brand VARCHAR(50),

Stock INT,

Price DECIMAL(10, 2)

);

**Inserting values:**

INSERT INTO RetailProducts (ProductID, ProductName, Category, Brand, Stock, Price) VALUES

(1, 'Gaming Laptop', 'Computers', 'XtremeGear', 12, 1700.00),

(2, 'Business Laptop', 'Computers', 'OfficeMax', 20, 1450.00),

(3, 'Convertible Laptop', 'Computers', 'FlexTech', 10, 1450.00),

(4, 'Mechanical Keyboard', 'Accessories', 'KeyStorm', 50, 120.00),

(5, 'Wireless Mouse', 'Accessories', 'ClickPro', 70, 60.00),

(6, 'Ergonomic Chair', 'Furniture', 'ComfySit', 15, 300.00),

(7, 'Standing Desk', 'Furniture', 'WorkFlow', 10, 420.00),

(8, 'Bookshelf', 'Furniture', 'HomeLine', 25, 180.00),

(9, 'Noise Cancelling Headset', 'Accessories', 'AudioZen', 30, 120.00),

(10, 'Gaming Monitor', 'Computers', 'XtremeGear', 8, 1500.00),

(11, 'Desk Organizer', 'Accessories', 'Neatify', 60, 40.00),

(12, 'Laptop Backpack', 'Accessories', 'GearPack', 40, 120.00);

**Use ROW\_NUMBER to assign unique rank within each category**

SELECT

ProductID,

ProductName,

Category,

Brand,

Stock,

Price,

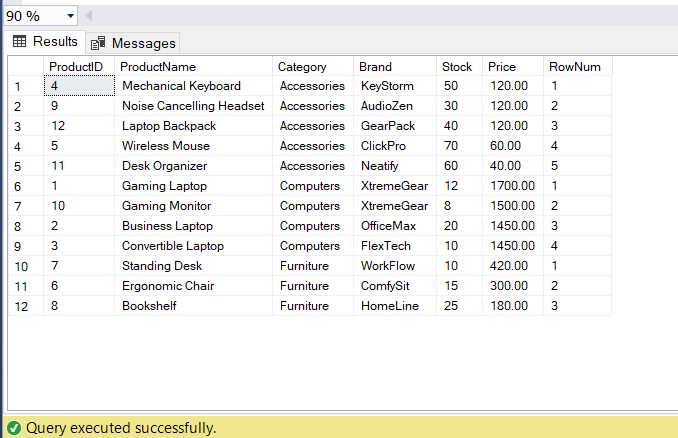
ROW\_NUMBER() OVER (

PARTITION BY Category

ORDER BY Price DESC

) AS RowNum

FROM RetailProducts;

****

**Use RANK to handle ties with gaps**

SELECT

ProductID,

ProductName,

Category,

Brand,

Stock,

Price,

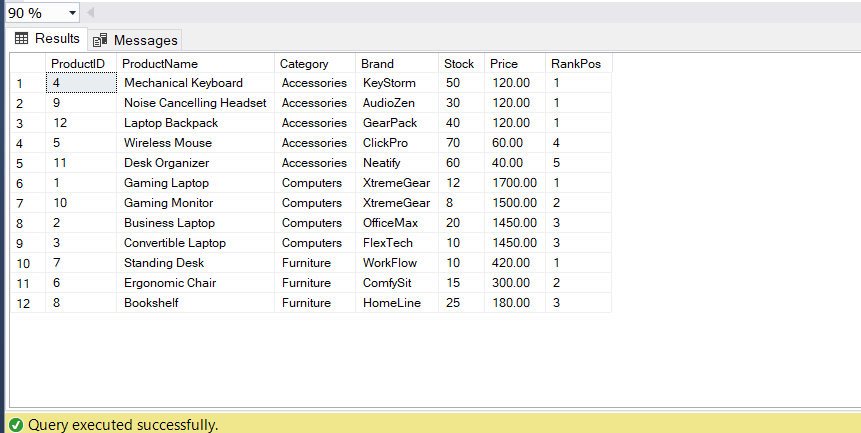
RANK() OVER (

PARTITION BY Category

ORDER BY Price DESC

) AS RankPos

FROM RetailProducts;



**Use DENSE\_RANK to handle ties without gaps**

SELECT

ProductID,

ProductName,

Category,

Brand,

Stock,

Price,

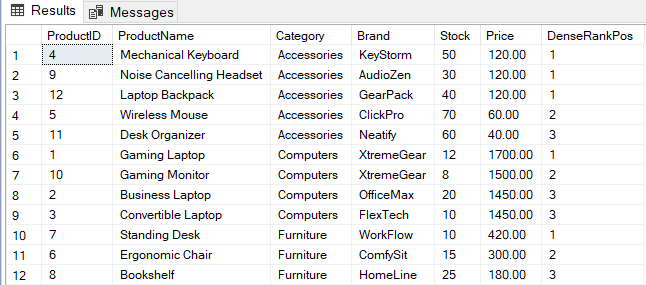
DENSE\_RANK() OVER (

PARTITION BY Category

ORDER BY Price DESC

) AS DenseRankPos

FROM RetailProducts;



**Top 3 most expensive products per category using ROW\_NUMBER**

WITH RankedProducts AS (

SELECT

\*,

ROW\_NUMBER() OVER (

PARTITION BY Category

ORDER BY Price DESC

) AS RowNum

FROM RetailProducts

)

SELECT

ProductID,

ProductName,

Category,

Brand,

Stock,

Price

FROM RankedProducts

WHERE RowNum <= 3;

