**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.

CREATE TABLE Products (

ProductID INT PRIMARY KEY,

ProductName VARCHAR(100),

Category VARCHAR(100),

Price DECIMAL(10, 2)

);

INSERT INTO Products (ProductID, ProductName, Category, Price) VALUES

(1, 'iPhone', 'Electronics', 999.99),

(2, 'Galaxy S', 'Electronics', 899.99),

(3, 'MacBook', 'Electronics', 1999.99),

(4, 'Dell XPS', 'Electronics', 1999.99),

(5, 'Shampoo', 'PersonalCare', 8.99),

(6, 'Lotion', 'PersonalCare', 8.99),

(7, 'Perfume', 'PersonalCare', 19.99);

SELECT \*

FROM (

SELECT \*,

ROW\_NUMBER() OVER (PARTITION BY Category ORDER BY Price DESC) AS RowNum

FROM Products

) AS Ranked

WHERE RowNum <= 3;

SELECT \*

FROM (

SELECT \*,

RANK() OVER (PARTITION BY Category ORDER BY Price DESC) AS PriceRank

FROM Products

) AS Ranked

WHERE PriceRank <= 3;

SELECT \*

FROM (

SELECT \*,

DENSE\_RANK() OVER (PARTITION BY Category ORDER BY Price DESC) AS DenseRank

FROM Products

) AS Ranked

WHERE DenseRank <= 3;

SELECT \*,

ROW\_NUMBER() OVER (PARTITION BY Category ORDER BY Price DESC) AS RowNum,

RANK() OVER (PARTITION BY Category ORDER BY Price DESC) AS PriceRank,

DENSE\_RANK() OVER (PARTITION BY Category ORDER BY Price DESC) AS DenseRank

FROM Products

ORDER BY Category, Price DESC;



