**WEEK – 2 Assignments**

**Advanced SQL**

**Mandatory hands-on :-**

1. **SQL Exercise Advanced concepts**

**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

**SQL Codes :-**

**Creating Tables :-**

| USE AdvSQL; GO  CREATE TABLE Categories (  category\_id INT PRIMARY KEY IDENTITY(1,1),  category\_name VARCHAR(100) NOT NULL );  CREATE TABLE Products (  product\_id INT PRIMARY KEY IDENTITY(1,1),  product\_name VARCHAR(100) NOT NULL,  category\_id INT NOT NULL,  price DECIMAL(10,2) NOT NULL,  description TEXT,  FOREIGN KEY (category\_id) REFERENCES Categories(category\_id) ); |
| --- |

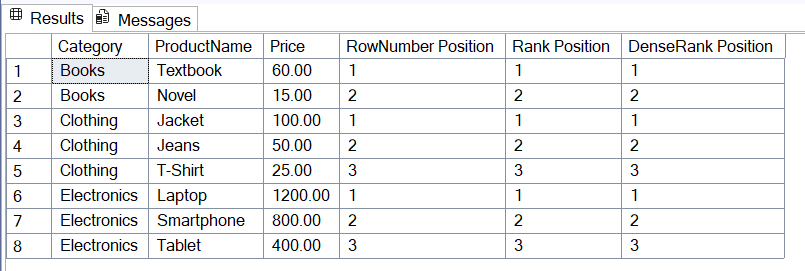
**Inserts Product :-**

| INSERT INTO Categories (category\_name) VALUES  ('Electronics'),  ('Clothing'),  ('Books');  INSERT INTO Products (product\_name, category\_id, price, description) VALUES  ('Smartphone', 1, 800.00, 'Latest model smartphone'),  ('Laptop', 1, 1200.00, 'High performance laptop'),  ('Headphones', 1, 200.00, 'Noise cancelling headphones'),  ('T-Shirt', 2, 25.00, '100% cotton t-shirt'),  ('Jeans', 2, 50.00, 'Slim fit jeans'),  ('Novel', 3, 15.00, 'Bestselling fiction novel'),  ('Textbook', 3, 60.00, 'University level textbook'),  ('Tablet', 1, 400.00, '10-inch tablet'),  ('Jacket', 2, 100.00, 'Waterproof jacket');  SELECT \* FROM Products; SELECT \* FROM Categories; |
| --- |

**Ranking Products :-**

| --Ranking products  WITH ranked\_products AS (  SELECT  c.category\_name AS Category,  p.product\_name AS ProductName,  p.price AS Price,  ROW\_NUMBER() OVER (  PARTITION BY c.category\_name   ORDER BY p.price DESC  ) AS row\_num,  RANK() OVER (  PARTITION BY c.category\_name   ORDER BY p.price DESC  ) AS rank,  DENSE\_RANK() OVER (  PARTITION BY c.category\_name   ORDER BY p.price DESC  ) AS dense\_rank  FROM Products p  INNER JOIN Categories c ON p.category\_id = c.category\_id  )  -- Results for all ranking methods  SELECT   Category,  ProductName,  Price,  row\_num AS "RowNumber Position",  rank AS "Rank Position",  dense\_rank AS "DenseRank Position" FROM ranked\_products WHERE   row\_num <= 3   OR rank <= 3   OR dense\_rank <= 3 ORDER BY Category, Price DESC; |
| --- |

**OUTPUT :-**

****