SQL Exercise: Advanced Ranking and Window Functions

- Objective:
  Use ROW_NUMBER(), RANK(), and DENSE_RANK() with OVER(PARTITION BY ...) to retrieve the top 3 priced products per category and explore how ranking functions behave with ties.

Step 1: Create and Use Database

```
CREATE DATABASE RetailStore;
GO

USE RetailStore;
GO
```

Step 2: Create Products Table

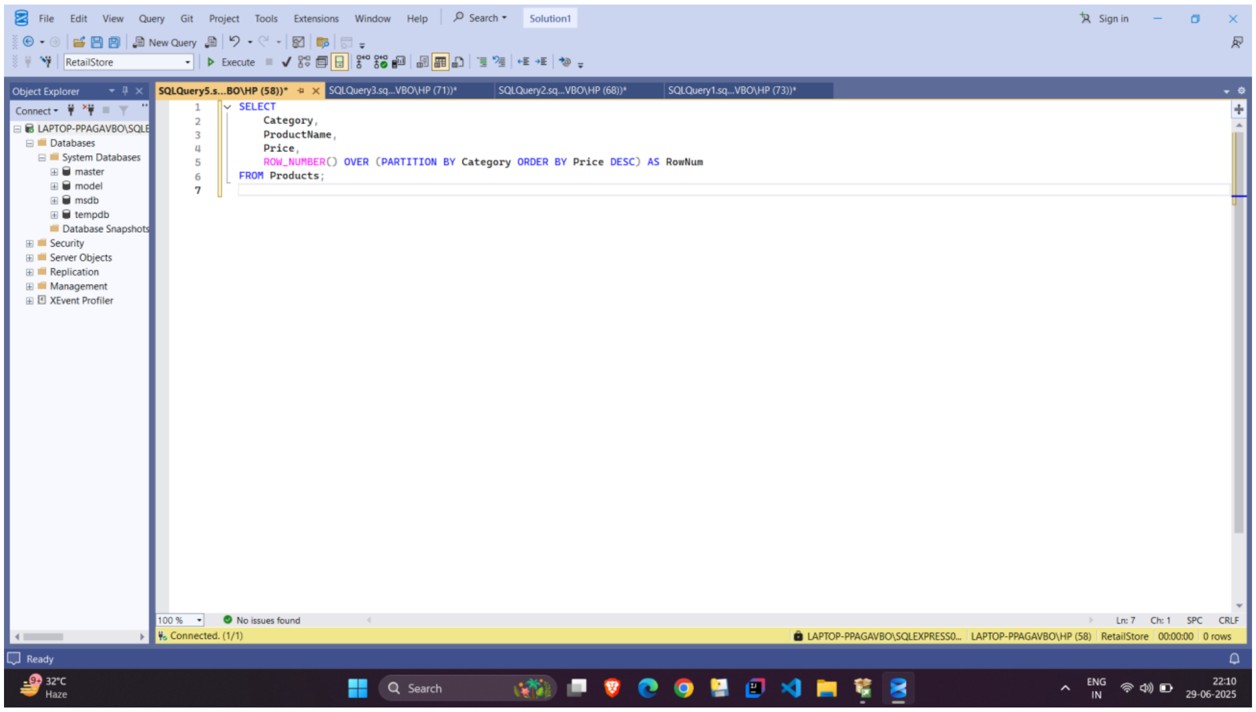
```
CREATE TABLE Products (
    ProductID INT PRIMARY KEY,
    ProductName VARCHAR(100),
    Category VARCHAR(50),
    Price DECIMAL(10, 2)
);
```

Step 3: Insert Sample Data

```
INSERT INTO Products VALUES
(1, 'Laptop', 'Electronics', 1000),
(2, 'Smartphone', 'Electronics', 700),
(3, 'Tablet', 'Electronics', 700),
(4, 'Headphones', 'Electronics', 150),
(5, 'Jeans', 'Clothing', 50),
(6, 'Jacket', 'Clothing', 120),
(7, 'Shoes', 'Clothing', 80),
(8, 'T-Shirt', 'Clothing', 20),
(9, 'Refrigerator', 'Appliances', 900),
(10, 'Microwave', 'Appliances', 300),
(11, 'Blender', 'Appliances', 300),
(12, 'Toaster', 'Appliances', 80);
```

Step 4: Ranking with ROW_NUMBER()

```
SELECT
    Category,
    ProductName,
    Price,
    ROW_NUMBER() OVER (PARTITION BY Category ORDER BY Price DESC) AS RowNum
FROM Products;
```

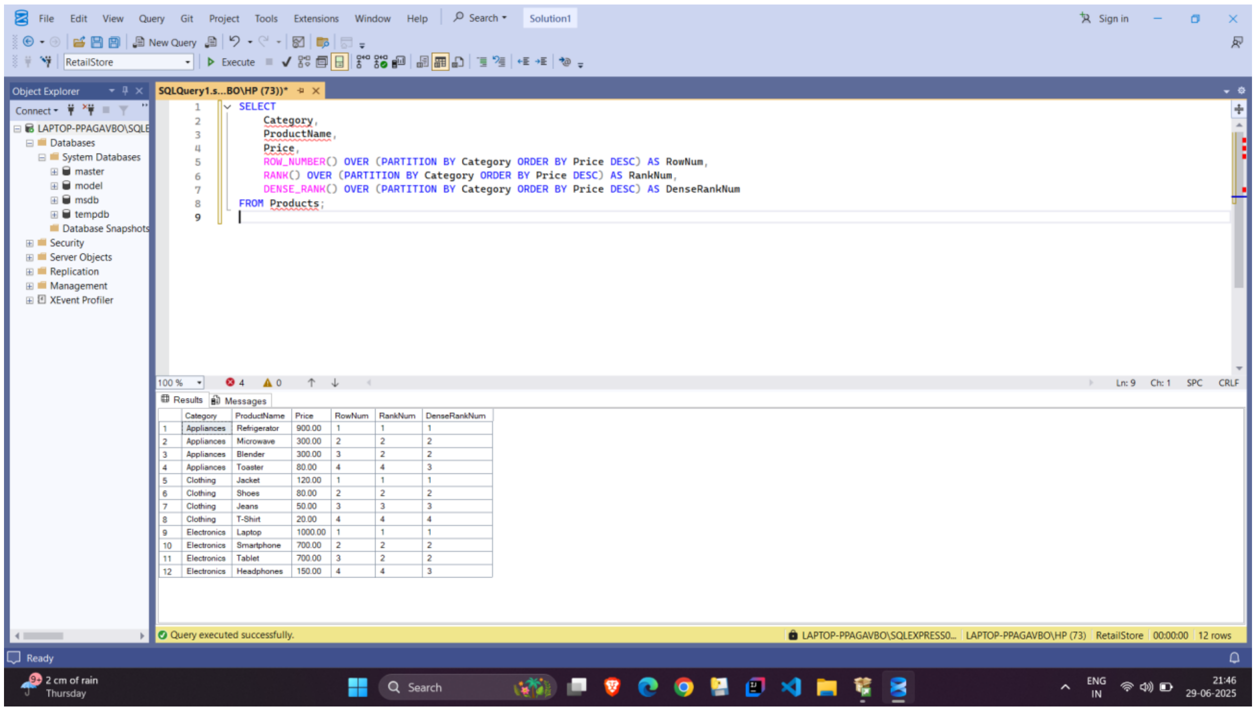


```
SELECT
    Category,
    ProductName,
    Price,
    ROW_NUMBER() OVER (PARTITION BY Category ORDER BY Price DESC) AS RowNum
FROM Products;
```

```
No issues found
Connected. (1/1)
LAPTOP-PPAGAVBO\SQLEXPRESS0...  LAPTOP-PPAGAVBO\HP (58)  RetailStore  00:00:00  0 rows
```

Step 5: Ranking with RANK() and DENSE_RANK()

```
SELECT
    Category,
    ProductName,
    Price,
    RANK() OVER (PARTITION BY Category ORDER BY Price DESC) AS RankNum,
    DENSE_RANK() OVER (PARTITION BY Category ORDER BY Price DESC) AS
DenseRankNum
FROM Products;
```



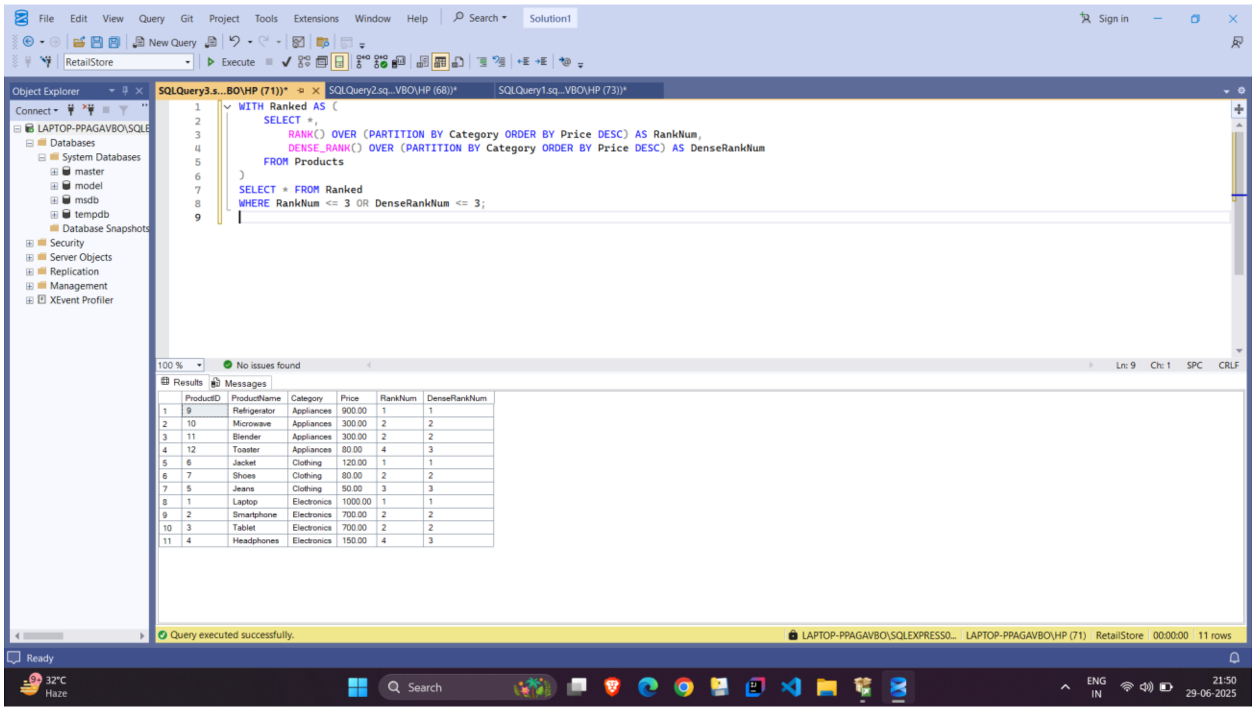
```sql
SELECT
    Category,
    ProductName,
    Price,
    ROW_NUMBER() OVER (PARTITION BY Category ORDER BY Price DESC) AS RowNum,
    RANK() OVER (PARTITION BY Category ORDER BY Price DESC) AS RankNum,
    DENSE_RANK() OVER (PARTITION BY Category ORDER BY Price DESC) AS DenseRankNum
FROM Products;
```

	Category	ProductName	Price	RowNum	RankNum	DenseRankNum
1	Appliances	Refrigerator	900.00	1	1	1
2	Appliances	Microwave	300.00	2	2	2
3	Appliances	Blender	300.00	3	2	2
4	Appliances	Toaster	80.00	4	4	3
5	Clothing	Jacket	120.00	1	1	1
6	Clothing	Shoes	80.00	2	2	2
7	Clothing	Jeans	50.00	3	3	3
8	Clothing	T-Shirt	20.00	4	4	4
9	Electronics	Laptop	1000.00	1	1	1
10	Electronics	Smartphone	700.00	2	2	2
11	Electronics	Tablet	700.00	3	2	2
12	Electronics	Headphones	150.00	4	4	3

Query executed successfully.

Step 6: Get Top 3 Products Using ROW_NUMBER()

```
WITH RankedProducts AS (
    SELECT *,
        ROW_NUMBER() OVER (PARTITION BY Category ORDER BY Price DESC) AS
RowNum
    FROM Products
)
SELECT * FROM RankedProducts
WHERE RowNum <= 3;
```



```sql
WITH Ranked AS (
    SELECT *,
        RANK() OVER (PARTITION BY Category ORDER BY Price DESC) AS RankNum,
        DENSE_RANK() OVER (PARTITION BY Category ORDER BY Price DESC) AS DenseRankNum
    FROM Products
)
SELECT * FROM Ranked
WHERE RankNum <= 3 OR DenseRankNum <= 3;
```

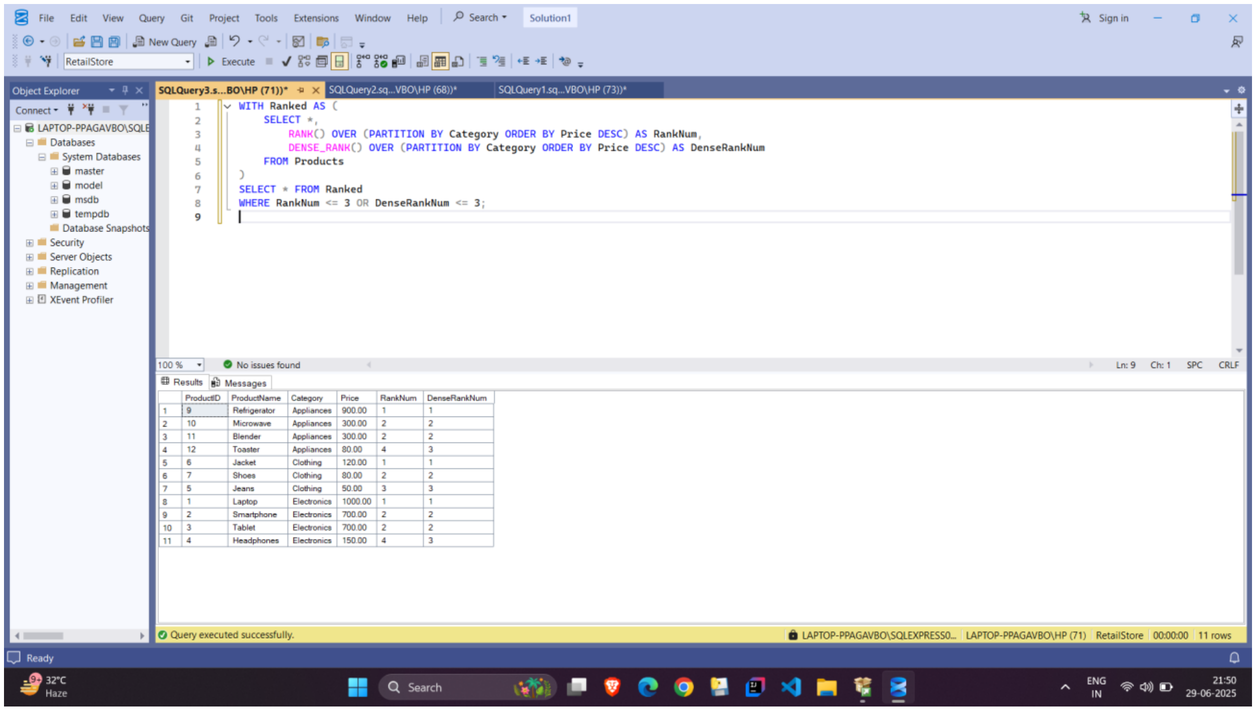
	ProductID	ProductName	Category	Price	RankNum	DenseRankNum
1	9	Refrigerator	Appliances	900.00	1	1
2	10	Microwave	Appliances	300.00	2	2
3	11	Blender	Appliances	300.00	2	2
4	12	Toaster	Appliances	80.00	4	3
5	6	Jacket	Clothing	120.00	1	1
6	7	Shoes	Clothing	80.00	2	2
7	5	Jeans	Clothing	50.00	3	3
8	1	Laptop	Electronics	1000.00	1	1
9	2	Smartphone	Electronics	700.00	2	2
10	3	Tablet	Electronics	700.00	2	2
11	4	Headphones	Electronics	150.00	4	3

```
Query executed successfully.
```

Step 7: Get Top 3 Products Using RANK() and DENSE_RANK()

```
WITH Ranked AS (
    SELECT *,
        RANK() OVER (PARTITION BY Category ORDER BY Price DESC) AS RankNum,
        DENSE_RANK() OVER (PARTITION BY Category ORDER BY Price DESC) AS 
DenseRankNum
    FROM Products
)
SELECT * FROM Ranked
```

```
WHERE RankNum <= 3 OR DenseRankNum <= 3;
```



```sql
WITH Ranked AS (
    SELECT *,
        RANK() OVER (PARTITION BY Category ORDER BY Price DESC) AS RankNum,
        DENSE_RANK() OVER (PARTITION BY Category ORDER BY Price DESC) AS DenseRankNum
    FROM Products
)
SELECT * FROM Ranked
WHERE RankNum <= 3 OR DenseRankNum <= 3;
```

	ProductID	ProductName	Category	Price	RankNum	DenseRankNum
1	9	Refrigerator	Appliances	900.00	1	1
2	10	Microwave	Appliances	300.00	2	2
3	11	Blender	Appliances	300.00	2	2
4	12	Toaster	Appliances	80.00	4	3
5	6	Jacket	Clothing	120.00	1	1
6	7	Shoes	Clothing	80.00	2	2
7	5	Jeans	Clothing	50.00	3	3
8	1	Laptop	Electronics	1000.00	1	1
9	2	Smartphone	Electronics	700.00	2	2
10	3	Tablet	Electronics	700.00	2	2
11	4	Headphones	Electronics	150.00	4	3

Query executed successfully.

Summary

- ROW_NUMBER() always gives unique ranks, no matter if prices are tied.
- RANK() skips rank numbers when there are ties.
- DENSE_RANK() does not skip rank numbers when there are ties.

The queries return top 3 items per category based on these differences.