

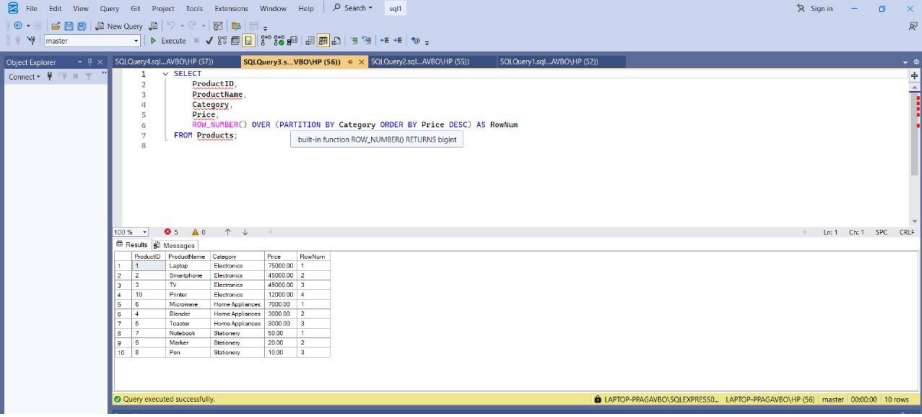
# SQL Window Functions: Ranking and Top-N Queries

---

## 1. Use ROW\_NUMBER() to Assign a Unique Rank Within Each Category

The ROW\_NUMBER() function assigns a unique number to each row within a partition. Even if values are tied, each row receives a distinct number.

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



The screenshot shows the SQL Server Enterprise Manager interface. The query window displays the SQL code for the ROW\_NUMBER() query. The results window shows the output of the query, which is a table with 5 columns: ProductID, ProductName, Category, Price, and RowNum. The results are sorted by Category and then by Price in descending order. The RowNum column shows the unique rank assigned to each row within each category.

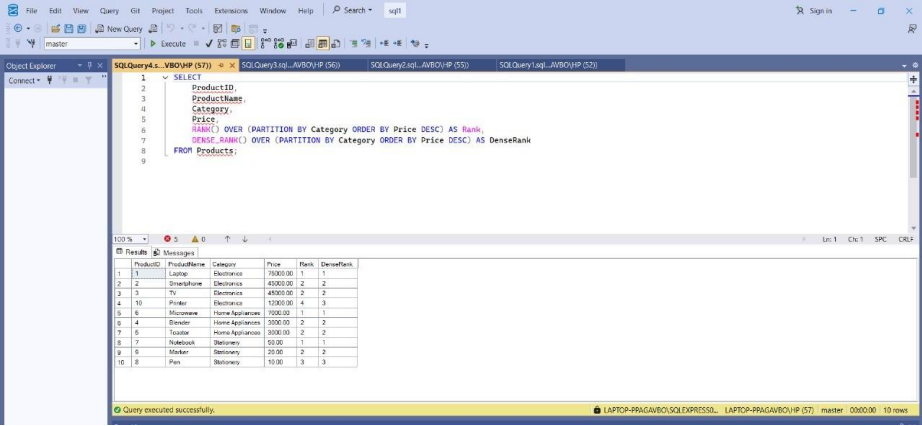
ProductID	ProductName	Category	Price	RowNum
1	Laptop	Electronics	7500.00	1
2	Smartphone	Electronics	4500.00	2
3	TV	Electronics	4500.00	3
4	Printer	Electronics	1200.00	4
5	Microwave	Home Appliances	700.00	1
6	Blender	Home Appliances	500.00	2
7	Toaster	Home Appliances	300.00	3
8	Headset	Peripherals	50.00	1
9	Mouse	Peripherals	20.00	2
10	Pen	Peripherals	10.00	3

ROW\_NUMBER() gives a unique number to each row — even if prices are the same (no ties allowed).

## 2. Use RANK() and DENSE\_RANK() to Compare Tie Handling

```
SELECT  
    ProductID,  
    ProductName,
```

*Category,*  
*Price,*  
***RANK() OVER (PARTITION BY Category ORDER BY Price DESC) AS***  
*Rank,*  
***DENSE\_RANK() OVER (PARTITION BY Category ORDER BY Price***  
***DESC) AS DenseRank***  
***FROM Products;***



The screenshot shows a SQL Server Enterprise Manager window with a query editor and a results pane. The query is as follows:

```

1 SELECT
2     ProductID,
3     ProductName,
4     Category,
5     Price,
6     RANK() OVER (PARTITION BY Category ORDER BY Price DESC) AS Rank,
7     DENSE_RANK() OVER (PARTITION BY Category ORDER BY Price DESC) AS DenseRank
8 FROM Products;

```

The results pane displays the following data:

ProductID	ProductName	Category	Price	Rank	DenseRank
1	Laptop	Electronics	7500.00	1	1
2	Smartphone	Electronics	4500.00	2	2
3	TV	Electronics	4000.00	2	2
4	Printer	Electronics	1200.00	4	3
5	Blender	Home Appliances	7000.00	1	1
6	Washer	Home Appliances	5000.00	2	2
7	Toaster	Home Appliances	3000.00	2	2
8	Headset	Peripherals	60.00	1	1
9	Mouse	Peripherals	20.00	2	2
10	Pen	Peripherals	10.00	3	3

Difference:

RANK(): Skips numbers after ties.

Example: If two products tie at 1st place, the next one is 3rd.

DENSE\_RANK(): No skipping.

Example: If two products tie at 1st, next one is 2nd.

### 3. Find Top 3 Most Expensive Products in Each Category Using ROW\_NUMBER()

***WITH RankedProducts AS (***  
***SELECT***  
***ProductID,***  
***ProductName,***  
***Category,***  
***Price,***  
***ROW\_NUMBER() OVER (PARTITION BY Category ORDER BY Price***  
***DESC) AS RowNum***  
***FROM Products***  
***)***

***SELECT \****  
***FROM RankedProducts***  
***WHERE RowNum <= 3;***

