# **Using ROW\_NUMBER, RANK, and DENSE\_RANK**

## **Overview**

ROW\_NUMBER, RANK, and DENSE\_RANK are powerful SQL window functions used to assign rankings to rows within a result set. They are commonly applied in analytical queries, such as creating sequential row numbers, ranking data, and handling ties in ordered datasets. These functions provide essential tools for organizing, analyzing, and filtering data without aggregating it.

## **Learning Objectives**

By the end of this topic, learners will be able to:

* Understand the purpose and differences between ROW\_NUMBER, RANK, and DENSE\_RANK.
* Write SQL queries that implement each function effectively.
* Use PARTITION BY and ORDER BY clauses to control the behavior of these functions.
* Solve real-world data problems such as pagination, ranking, and duplicate handling with these functions.

## **Prerequisites**

Learners should have the following foundational knowledge:

* Basic SQL queries (SELECT, FROM, WHERE, GROUP BY).
* Understanding of sorting with ORDER BY.
* Familiarity with SQL syntax for window functions (OVER clause).

## **Key Concepts**

### **For Intermediate Learners:**

#### **1. ROW\_NUMBER**

Assigns a unique, sequential integer to rows within a partition of a result set.

* **Key Behavior**: No ties—each row gets a unique number regardless of duplicate values.

**Syntax**:  
sql  
Copy code  
SELECT column\_name,

ROW\_NUMBER() OVER (PARTITION BY partition\_column ORDER BY order\_column) AS row\_num

FROM table\_name;

**Use Case**: Pagination in reports or identifying duplicate rows for cleanup.  
Example:  
sql  
Copy code  
SELECT customer\_id, purchase\_date,

ROW\_NUMBER() OVER (PARTITION BY customer\_id ORDER BY purchase\_date DESC) AS row\_num

FROM purchases;

* This query ranks each customer’s purchases, assigning 1 to the most recent purchase.

#### **2. RANK**

Assigns a rank to rows within a partition, with the same rank for rows with identical values (ties). Gaps occur in the ranking sequence after ties.

* **Key Behavior**: Ties exist, causing gaps in the rank.

**Syntax**:  
sql  
Copy code  
SELECT column\_name,

RANK() OVER (PARTITION BY partition\_column ORDER BY order\_column) AS rank

FROM table\_name;

**Use Case**: Ranking products based on sales or performance metrics.  
Example:  
sql  
Copy code  
SELECT product\_id, total\_sales,

RANK() OVER (PARTITION BY category\_id ORDER BY total\_sales DESC) AS product\_rank

FROM sales\_data;

* This query assigns a rank to products within each category based on total sales.

#### **3. DENSE\_RANK**

Similar to RANK, but without gaps in the ranking sequence after ties.

* **Key Behavior**: Ties exist, but the ranking sequence remains dense (no gaps).

**Syntax**:  
sql  
Copy code  
SELECT column\_name,

DENSE\_RANK() OVER (PARTITION BY partition\_column ORDER BY order\_column) AS dense\_rank

FROM table\_name;

**Use Case**: Creating rank-based filters or classifications where gaps in ranks are not desirable.  
Example:  
sql  
Copy code  
SELECT department, employee\_name, salary,

DENSE\_RANK() OVER (PARTITION BY department ORDER BY salary DESC) AS salary\_rank

FROM employees;

* This query ranks employees within each department by salary, with no gaps between ranks.

### **Key Differences**

| **Function** | **Gaps in Ranks** | **Handles Ties** | **Unique Sequential IDs** |
| --- | --- | --- | --- |
| ROW\_NUMBER | No | No | Yes |
| RANK | Yes | Yes | No |
| DENSE\_RANK | No | Yes | No |

### **For Advanced Learners:**

* **Complex Partitioning**: Using multiple columns in the PARTITION BY clause to create multi-dimensional partitions.
* **Performance Optimization**: Techniques for handling large datasets when ranking, such as creating indexes on columns used in ORDER BY.
* **Real-World Applications**:
  + Multi-level ranking (e.g., ranking products by sales within regions and then by categories).
  + Fraud detection by identifying duplicates with ROW\_NUMBER.

## **Graphs/Diagrams**

1. **Comparison Table**: A visual comparison of ROW\_NUMBER, RANK, and DENSE\_RANK with use-case examples.
2. **Partition and Order Workflow**: A diagram showing how PARTITION BY and ORDER BY interact to produce rankings.
3. **Ranking Sequence Examples**: Visualizing output for a dataset ranked by RANK and DENSE\_RANK to highlight the differences.

## **Hands-On Practice**

1. **Basic Exercises**:
   * Use ROW\_NUMBER to assign sequential numbers to customer transactions.
   * Use RANK to rank employees by performance scores.
2. **Intermediate Exercises**:
   * Write a query using DENSE\_RANK to rank sales representatives by revenue, without gaps.
   * Identify and delete duplicate rows in a table using ROW\_NUMBER.
3. **Advanced Exercises**:
   * Combine ROW\_NUMBER and RANK to implement pagination in a large dataset.
   * Analyze trends by ranking products sold within different time windows.

## **Additional Notes**

* **Common Misconceptions**:
  + Learners often confuse RANK with DENSE\_RANK. Remember, RANK leaves gaps in the numbering after ties, while DENSE\_RANK does not.
  + ROW\_NUMBER is not affected by ties and always assigns a unique number.
* **Performance Tips**: Use indexing on columns referenced in PARTITION BY and ORDER BY to improve query execution times.

## **Additional Learning Paths**

* Explore **SQL Window Functions** to deepen understanding of other functions like NTILE, LEAD, and LAG.
* Learn **Query Optimization** techniques to enhance the performance of ranking functions.
* Study advanced analytics use cases, such as cohort analysis or churn prediction, where ranking plays a key role.

## **Resources**

* [Official PostgreSQL Documentation on Window Functions](https://www.postgresql.org/docs/current/tutorial-window.html)
* Mode Analytics: Advanced SQL with Window Functions
* [Microsoft SQL Server Window Functions Documentation](https://learn.microsoft.com/en-us/sql/t-sql/queries/select-over-clause)

**Suggested Search Queries**:

* "Difference between ROW\_NUMBER, RANK, and DENSE\_RANK in SQL"
* "SQL ranking functions tutorial"
* "Using PARTITION BY in SQL ranking functions"
* "SQL ROW\_NUMBER to remove duplicates"
* "SQL pagination with ranking functions"

## **Community and Support**

* **Stack Overflow**: [SQL Ranking Functions](https://stackoverflow.com/questions/tagged/sql-ranking-functions)
* **Reddit**: [r/SQL](https://www.reddit.com/r/SQL/)
* **LinkedIn**: Join SQL-focused professional groups for discussions and learning.

## **Citations/References**

* Celko, J. (2014). *SQL for Smarties: Advanced SQL Programming*. Morgan Kaufmann.
* Pratt, P., & Adamski, M. (2018). *Concepts of Database Management*. Cengage Learning.
* SQL Documentation: Window Functions Overview. Available at: https://docs.sql