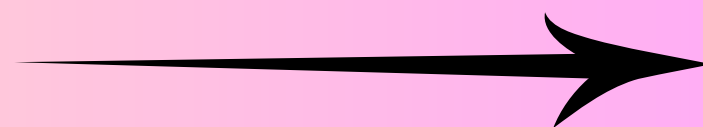


Learning SQL Window Functions

@Vaishnavi Dauale



What is Window Function?

- Window functions in SQL perform calculations across a set of table rows related to the current row.



Why Use Window Functions?

- **Enhanced Data Analysis:** Perform complex calculations while still being able to see each row of data.
- **Efficient Queries:** Avoid multiple joins and subqueries, making your SQL queries cleaner and faster.
- **Advanced Insights:** Gain deeper insights by comparing data across rows within a result set



How Do They Work?

- **Function:** The operation you want to perform (e.g., SUM(), RANK()).
- **OVER clause:** Specifies that the function is a window function and defines the window.
- **Optional PARTITION BY and ORDER BY:** Used to group and sort the data within the window.



Types of Window Functions

- **Aggregate Functions:** Like SUM(), AVG(), COUNT() – used over specific partitions or entire datasets.
- **Ranking Functions:** Such as RANK(), ROW_NUMBER(), DENSE_RANK() – used for assigning ranks to rows.
- **Value Functions:** Including LAG(), LEAD(), FIRST_VALUE() – used for retrieving specific values from the window.



Basic Syntax:

```
function(column) OVER (  
    [PARTITION BY expr_list]  
    [ORDER BY order_list]  
)
```





Window Functions

| Aggregate | Value | Ranking |
|---|---|---|
| <ul style="list-style-type: none">• AVG()• MAX()• MIN()• SUM()• COUNT() | <ul style="list-style-type: none">• ROW_NUMBER()• RANK()• DENSE_RANK()• PERCENT_RANK()• NTILE() | <ul style="list-style-type: none">• LAG()• LEAD()• FIRST_VALUE()• LAST_VALUE()• NTH_VALUE() |



Examples

Suppose you have a sales table with columns salesperson, sale_amount, and sale_date. You want to calculate the total sales amount for each salesperson over a specific period, say a month.

```
SELECT
  student_name,
  dep_name,
  score,
  SUM(score) OVER (PARTITION BY dep_name ORDER BY score DESC) AS cumulative_score
FROM student_score;
```





- You want to rank the salespersons based on their total sales amount in descending order.

```
SELECT
    student_name,
    dep_name,
    score,
    RANK() OVER (PARTITION BY dep_name ORDER BY score DESC) AS rank_within_department
FROM student_score;
```



- You want to find the difference between the current sales amount and the previous sales amount for each salesperson.

```
SELECT salesperson,  
       sale_amount,  
       LAG(sale_amount) OVER (PARTITION BY salesperson ORDER BY sale_amount) AS prev_sale  
FROM sales_table;
```



10

Thank You

@Vaishnavi Dauale

