Window Functions continued

Agenda:

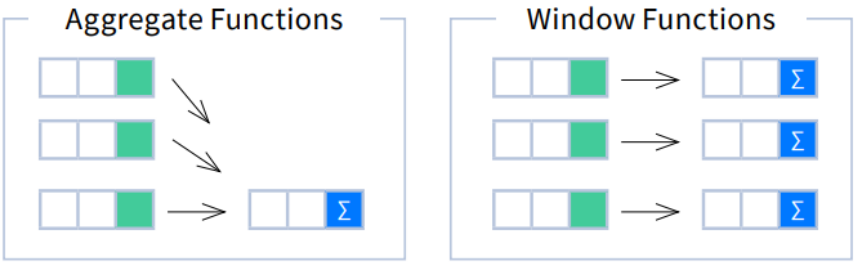
In today's session, we'll cover essential topics, including:-

- Problem Statement
- Window Frames
- Ntile()
- Lag(), Lead()
- First_value(), Nth_value()

Summary of Previous Lecture:

Window Functions

- Window functions are used to perform some operation on a group of rows and provide a resultant value for each row in the table. They compute their result based on a sliding window frame, a set of rows that are somehow related to the current row.
- OVER clause is used with window functions to define the window.
- GROUP BY collapses the individual records into groups. i.e. after using GROUP BY, you cannot refer to any individual field because it is collapsed.
- However, Window functions do not collapse individual records and the **row-level information** is intact in the partitions. Thus, we can create queries showing data from the individual record together with the result of the window function.



- **Syntax**

```
SELECT
      <column_1>,
       <column_2>,
       <window_function>() OVER (
       PARTITION BY < column_to_partition_by >
       ORDER BY < column_to_order_by >
```

```
        <window_frame>) AS <window_column_alias>
FROM <table_name>;
```

PARTITION BY

- It divides the rows into multiple groups, called partitions, to which the window function is applied.
- With no PARTITION BY clause, the entire result set is the partition

Ranking Window Functions

- **ROW_NUMBER**() - This function assigns a unique number to each row in the table.
- **RANK**() - This function is used to assign a value to the records in the table, and it can be the same for two or more records having the same values (lie within the same partition). It also skips ranks if the records are the same.
- **DENSE_RANK**() - This function assigns a unique rank to the records in the table. It is similar to the RANK() function but it doesn't skip ranks if two or more records have the same ranks.
- **Example:** In the given employee table, calculate the row no., rank, and dense rank of each employee according to salary within each department.
- **Query:**

```
SELECT
ROW_NUMBER() OVER (PARTITION BY Department ORDER BY Salary
DESC) AS emp_row_no,
Name, Department, Salary,
RANK() OVER (PARTITION BY Department ORDER BY Salary DESC)
AS emp_rank,
DENSE_RANK() OVER (PARTITION BY Department ORDER BY Salary
DESC) AS emp_dense
FROM employee
```