DBMS SQL

Lesson 15: Analytical Functions

Lesson Objectives

- To understand the following topics:
  - Analytical Functions
    - RANK & Dense Rank
    - Max, Min, Sum & Avg
    - Row Number
    - Lead & Lag
    - FIRST_VALUE & LAST_VALUE
    - LISTAGG
    - WITH CLAUSE

Analytical Functions

➢ RANK:

- RANK calculates the rank of a value in a group of values. The return type is NUMBER.

- Rows with equal values for the ranking criteria receive the same rank.

- Oracle Database then adds the number of tied rows to the tied rank to calculate the next rank. Therefore, the ranks may not be consecutive numbers

- The function is useful for top to bottom reporting.

- Syntax:

```
RANK() OVER ([ query_partition_clause ] order_by_clause)
```

Analytical Functions

- RANK
- Example:

```
SELECT empno,
      deptno,
       sal,
       RANK() OVER (PARTITION BY deptno ORDER BY sal) AS myrank
FROM emp ORDER BY deptno;
```

- DENSE RANK:
  - The DENSE_RANK function acts like the RANK function except that it assigns consecutive ranks.
  - DENSE_RANK computes the rank of a row in an ordered group of rows and returns the rank as a NUMBER. The ranks are consecutive integers beginning with 1

Analytical Functions

- DENSE RANK:
- Example:
  - SELECT empno,
    deptno,
    sal,
    DENSE_RANK() OVER (PARTITION BY deptno ORDER BY sal) AS myrank
    FROM emp;

- ROW_NUMBER:
  - ROW_NUMBER is an analytic function. It assigns a unique number to each row to which it is applied (either each row in the partition or each row returned by the query), in the ordered sequence of rows specified in the order_by_clause, beginning with 1.

Analytical Functions

- The ROW_NUMBER analytic function is order-sensitive and produces an error if you attempt to use it without an ORDER BY in the analytic clause.Omitting a partitioning clause from the OVER clause means the whole result set is treated as a single partition.

- In the following example we assign a unique row number to each employee based on their salary (lowest to highest). The example also includes RANK and DENSE_RANK to show the difference in how ties are handled.

- Example: ROW_NUMBER

```
SELECT empno, ename, deptno,sal,
       ROW_NUMBER() OVER (ORDER BY sal) AS row_num,
       RANK() OVER (ORDER BY sal) AS row_rank,
       DENSE_RANK() OVER (ORDER BY sal) AS row_dense_rank
FROM   emp;
```

Analytical Functions

- **Lead Function:**
  - LEAD is an analytic function. It provides access to more than one row of a table at the same time without a self join.
  - Given a series of rows returned from a query and a position of the cursor, LEAD provides access to a row at a given physical offset beyond that position.
  - If you do not specify offset, then its default is 1. The optional default value is returned if the offset goes beyond the scope of the table. If you do not specify default, then its default value is null.
  - You cannot use LEAD or any other analytic function for value_expr. That is, you cannot nest analytic functions, but you can use other built-in function expressions for value_expr.

Analytical Functions

- Lead Example:

  -
    ```
    SELECT empno,
      ename,
      job,
      sal,
      LEAD(sal, 1) OVER (ORDER BY sal) AS sal_next
    FROM   emp;
    ```

  -
    ```
    SELECT empno,
        ename,
        job,
        sal,
        LEAD(sal, 1) OVER (ORDER BY sal) AS sal_next,
        LEAD(sal, 1) OVER (ORDER BY sal) - sal AS sal_diff
    FROM   emp;
    ```

Analytical Functions

➢ Lag Function:

- LAG is an analytic function. It provides access to more than one row of a table at the same time without a self join.

- Given a series of rows returned from a query and a position of the cursor, LAG provides access to a row at a given physical offset prior to that position.

- If you do not specify offset, then its default is 1. The optional default value is returned if the offset goes beyond the scope of the window. If you do not specify default, then its default is null.

- You cannot use LAG or any other analytic function for value_expr. That is, you cannot nest analytic functions, but you can use other built-in function expressions for value_expr

Analytical Functions

- Lag Example:
  - SELECT empno,

    ```
    SELECT empno,
      ename,
      job,
      sal,
      LAG(sal, 1) OVER (ORDER BY sal) AS sal_prev
    FROM   emp;
    ```

  - SELECT empno,

    ```
    SELECT empno,
        ename,
         job,
        sal,
        LAG(sal, 1) OVER (ORDER BY sal) AS sal_prev,
         sal - LAG(sal, 1) OVER (ORDER BY sal) AS sal_diff
    FROM   emp;
    ```

Analytical Functions

➢ FIRST_VALUE Function:

- FIRST_VALUE is an analytic function. It returns the first value in an ordered set of values.
- If the first value in the set is null, then the function returns NULL unless you specify IGNORE NULLS.
- If you specify IGNORE NULLS, then FIRST_VALUE returns the fist non-null value in the set, or NULL if all values are null.
- Example:

```
SELECT   empno,
          deptno,
          sal,
          FIRST_VALUE(sal) IGNORE NULLS
         OVER (PARTITION BY deptno ORDER BY sal) AS lowest_in_dept
FROM   emp;
```

Analytical Functions

➢ LAST_VALUE Function:

- LAST_VALUE is an analytic function. It returns the last value in an ordered set of values.
- If the last value in the set is null, then the function returns NULL unless you specify IGNORE NULLS.
- If you specify IGNORE NULLS, then LAST_VALUE returns the fist non-null value in the set, or NULL if all values are null.
- Example:

```
SELECT empno, deptno,
            sal,
             LAST_VALUE(sal) IGNORE NULLS
             OVER (PARTITION BY deptno ORDER BY sal ROWS BETWEEN
            UNBOUNDED PRECEDING AND UNBOUNDED FOLLOWING) AS
            highest_in_dept
FROM   emp;
```

Analytical Functions

- **LISTAGG Function:**
  - For a specified measure, LISTAGG orders data within each group specified in the ORDER BY clause and then concatenates the values of the measure column.
  - As a single-set aggregate function, LISTAGG operates on all rows and returns a single output row.
  - Example:
  - COLUMN employees FORMAT A50
  - SELECT deptno, LISTAGG(ename, ',’)
    WITHIN GROUP (ORDER BY ename) AS employees
    FROM   emp
    GROUP BY deptno;

Analytical Functions

WITH Clause:

- The SQL WITH clause allows you to give a sub-query block a name (a process also called sub-query refactoring), which can be referenced in several places within the main SQL query..

- The clause is used for defining a temporary relation such that the output of this temporary relation is available and is used by the query that is associated with the WITH clause.

- Example:

```
WITH temporaryTable(averageValue) as
  (SELECT avg(Sal)
  from Emp) SELECT Empno,Ename, Sal
            FROM Emp, temporaryTable
            WHERE Emp.Sal > temporaryTable.averageValue
```

Summary

- In this lesson you have learnt,
  - Use of RANK & Dense Rank
  - Use of Row Number, Lead & Lag Function.
  - Use of FIRST_VALUE & LAST_VALUE
  - Use of LISTAGG and WITH clause