DATA AND AI

SQL Window Functions

Shwetank Singh
GritSetGrow - GSGLearn.com

gsglearn.com

Data and AI

NTH_VALUE

Retrieves the n-th value in an ordered partition of a result set. Useful for accessing specific rows within a window frame.

```sql
SELECT
    employee_id,
    salary,
    NTH_VALUE(salary, 3)
        OVER (ORDER BY salary DESC) AS third_highest_salary
FROM
    employees;
```

Retrieves the third highest salary for each employee in the dataset.

Shwetank Singh
GritSetGrow - GSGLearn.com

Data and AI

CUME_DIST

Calculates the cumulative distribution of a value within a group of values. It returns the relative position of a value in a group.

```sql
SELECT
    employee_id,
    salary,
    CUME_DIST() OVER (ORDER BY salary) AS cumulative_distribution
FROM
    employees;
```

Shows the proportion of employees earning less than or equal to each employee's salary.

Shwetank Singh
GritSetGrow - GSGLearn.com

Data and AI

PERCENT_RANK

Computes the relative rank of a row within a partition as a percentage, ranging from 0 to 1.

```sql
SELECT
    employee_id,
    salary,
    PERCENT_RANK() OVER (ORDER BY salary) AS percent_rank
FROM
    employees;
```

Assigns a percentile rank to each employee based on their salary compared to others.

Shwetank Singh
GritSetGrow - GSGLearn.com

Data and AI

PERCENTILE_CONT

A continuous percentile function that calculates a specific percentile value within a group, interpolating between values if necessary.

```sql
SELECT
    department_id,
    PERCENTILE_CONT(0.5)
        WITHIN GROUP (ORDER BY salary)
        OVER (PARTITION BY department_id) AS median_salary
FROM
    employees;
```

Computes the median salary for each department by interpolating between salaries if needed.

Shwetank Singh
GritSetGrow - GSGLearn.com

Data and AI

PERCENTILE_DISC

A discrete percentile function that returns the first value in the ordered set of values that is greater than or equal to the specified percentile.

```sql
SELECT
    department_id,
    PERCENTILE_DISC(0.9)
        WITHIN GROUP (ORDER BY salary)
        OVER (PARTITION BY department_id) AS 90th_percentile_salary
FROM
    employees;
```

Finds the 90th percentile salary within each department, returning the smallest salary that is at least the 90th percentile.

Shwetank Singh
GritSetGrow - GSGLearn.com

Data and AI

FIRST_VALUE

Returns the first value in an ordered partition of a result set.

```sql
SELECT
    employee_id,
    salary,
    FIRST_VALUE(salary)
        OVER (PARTITION BY department_id
ORDER BY hire_date) AS first_hired_salary
FROM
    employees;
```

Retrieves the salary of the first hired employee in each department.

Shwetank Singh
GritSetGrow - GSGLearn.com

Data and AI

LAST_VALUE

Returns the last value in an ordered partition of a result set.

```sql
SELECT
    employee_id,
    salary,
    LAST_VALUE(salary)
        OVER (
            PARTITION BY department_id
            ORDER BY hire_date
            ROWS BETWEEN UNBOUNDED PRECEDING
             AND UNBOUNDED FOLLOWING
        ) AS latest_salary
FROM  employees;
```

Retrieves the salary of the most recently hired employee in each department.
Note: Requires specifying the frame to access the last value correctly.

Shwetank Singh
GritSetGrow - GSGLearn.com

Data and AI

NTILE

Distributes the rows in an ordered partition into a specified number of approximately equal groups, assigning a bucket number to each row.

```sql
SELECT
     employee_id,
     salary,
     NTILE(4) OVER
     (ORDER BY salary DESC) AS quartile
FROM  employees;
```

Divides employees into four quartiles based on their salaries, assigning each employee to a quartile from 1 to 4.

Shwetank Singh
GritSetGrow - GSGLearn.com

Data and AI

LAG

Accesses data from a previous row in the same result set without the use of a self-join.

```sql
SELECT
    employee_id,
    salary,
    LAG(salary, 1) OVER (ORDER BY hire_date)
      AS previous_salary
FROM employees;
```

Shows the salary of the employee who was hired immediately before each employee.

Shwetank Singh
GritSetGrow - GSGLearn.com

Data and AI

LEAD

Accesses data from a subsequent row in the same result set without the use of a self-join.

```sql
SELECT
    employee_id,
    salary,
    LEAD(salary, 1) OVER (ORDER BY hire_date)
       AS next_salary
FROM employees;
```

Shows the salary of the employee who will be hired immediately after each employee.

Shwetank Singh
GritSetGrow - GSGLearn.com

Data and AI

ROW_NUMBER

Assigns a unique sequential integer to rows within a partition of a result set, starting at 1 for the first row in each partition.

```sql
SELECT
    employee_id,
    department_id,
    ROW_NUMBER()
        OVER (PARTITION BY department_id
        ORDER BY salary DESC) AS row_num
FROM employees;
```

Numbers employees within each department based on their salary, with the highest-paid employee being row number 1.

Shwetank Singh
GritSetGrow - GSGLearn.com

Data and AI

RANK

Assigns a rank to each row within a partition of a result set, with gaps in rank values when there are ties.

```sql
SELECT
    employee_id,
    department_id,
    salary,
    RANK()
        OVER (PARTITION BY department_id
        ORDER BY salary DESC) AS salary_rank
FROM employees;
```

Ranks employees within each department by salary. Employees with the same salary receive the same rank, and subsequent ranks are skipped accordingly.

Shwetank Singh
GritSetGrow - GSGLearn.com

Data and AI

DENSE_RANK

Similar to RANK(), but without gaps in rank values when there are ties.

```sql
SELECT
    employee_id,
    department_id,
    salary,
    DENSE_RANK()
        OVER (PARTITION BY department_id
        ORDER BY salary DESC) AS dense_salary_rank
FROM employees;
```

Ranks employees within each department by salary without skipping ranks for tied salaries.

Shwetank Singh
GritSetGrow - GSGLearn.com

Data and AI

SUM

Calculates the cumulative sum of a numeric column over a specified window frame.

```sql
SELECT
    employee_id,
    salary,
    SUM(salary)
        OVER (
            ORDER BY hire_date
            ROWS BETWEEN UNBOUNDED PRECEDING
                AND CURRENT ROW
        ) AS cumulative_salary
FROM employees;
```

Computes the running total of salaries ordered by hire date.

Shwetank Singh
GritSetGrow - GSGLearn.com

Data and AI

AVG

Calculates the average of a numeric column over a specified window frame.

```sql
SELECT
    employee_id,
    salary,
    AVG(salary)
        OVER (PARTITION BY department_id)
        AS average_department_salary
FROM employees;
```

Determines the average salary within each department for every employee.

Shwetank Singh
GritSetGrow - GSGLearn.com

Thank you!