Database Management Systems

MySQL - Professional Coding

Malay Bhattacharyya

Associate Professor

Machine Intelligence Unit
and
Centre for Artificial Intelligence and Machine Learning
Indian Statistical Institute, Kolkata

April, 2026

Standard ranking

This function assigns rank to each row within a partition with gaps. In case of a tie, values are assigned the same rank, and next rank will be the summation of previous rank and number of duplicates.

```
RANK()
```

Note: The ranks are assigned in a non-consecutive manner.

Dense ranking

This function assigns rank to each row within a partition without gaps. In case of a tie, values are assigned the same rank, and next rank will be one greater than the previous rank.

```
DENSE_RANK()
```

Note: The ranks are assigned in a consecutive manner.

Percentile ranking

This function assigns percentile rank to each row within a partition. It finds the percentage of partition values less than the value in the current row, excluding the highest value.

```
PERCENT_RANK()
```

Note: The returned value ranges between 0 and 1.

Ranking functions – Practical applications

Consider the following table.

Table: BANK1

ID	CUSTOMER	ACCOUNT	YEAR
12340001	Sai Pallav	Savings	2018
12340002	Aarav Roy	Savings	2018
12340003	Vihaan Sen	Current	2018
12340004	Reyanshi Razdan	Current	2018
12340005	Balbindar Singh	Savings	2019
12340006	Aarohi Tendulkar	Current	2019
12340007	Karthikeyan Murugan	Current	2019
12340008	David Barik	Current	2019
12340009	Jagannath Mishra	Current	2019
12340010	Jahanara Begum	Savings	2020

Ranking functions – Practical applications

The following SQL query.

```
select ACCOUNT, CUSTOMER, YEAR, RANK() over
(partition by ACCOUNT order by YEAR) as SRANK from
BANK1;
```

will yield the following result.

ACCOUNT	CUSTOMER	YEAR	SRANK
Current	Vihaan Sen	2018	1
Current	Reyanshi Razdan	2018	1
Current	Aarohi Tendulkar	2019	3
Current	Karthikeyan Murugan	2019	3
Current	David Barik	2019	3
Current	Jagannath Mishra	2019	3
Savings	Sai Pallav	2018	1
Savings	Aarav Roy	2018	1
Savings	Balbindar Singh	2019	3
Savings	Jahanara Begum	2020	4

Ranking functions – Practical applications

The following SQL query.

```
select ACCOUNT, CUSTOMER, YEAR, DENSE_RANK() over
(partition by ACCOUNT order by YEAR) as DRANK from
BANK1;
```

will yield the following result.

ACCOUNT	CUSTOMER	YEAR	DRANK
Current	Vihaan Sen	2018	1
Current	Reyanshi Razdan	2018	1
Current	Aarohi Tendulkar	2019	2
Current	Karthikeyan Murugan	2019	2
Current	David Barik	2019	2
Current	Jagannath Mishra	2019	2
Savings	Sai Pallav	2018	1
Savings	Aarav Roy	2018	1
Savings	Balbindar Singh	2019	2
Savings	Jahanara Begum	2020	3

Ranking functions – Practical applications

The following SQL query.

```
select ACCOUNT, CUSTOMER, YEAR, PERCENT_RANK() over
(partition by ACCOUNT order by YEAR) as PRANK from
BANK1;
```

will yield the following result.

ACCOUNT	CUSTOMER	YEAR	PRANK
Current	Vihaan Sen	2018	0
Current	Reyanshi Razdan	2018	0
Current	Aarohi Tendulkar	2019	0.4
Current	Karthikeyan Murugan	2019	0.4
Current	David Barik	2019	0.4
Current	Jagannath Mishra	2019	0.4
Savings	Sai Pallav	2018	0
Savings	Aarav Roy	2018	0
Savings	Balbindar Singh	2019	0.5
Savings	Jahanara Begum	2020	0.75

Value First value

This function returns the value of argument from first row of window frame.

```
FIRST_VALUE(expr)
```

Value at a lagging position

This function returns the value of argument from row lagging current row within partition.

```
LAG(expr, N)
```

Note: The value of N ranges between 0 and 2^{63}.

Value at the end

This function returns the value of argument from last row of window frame.

```
LAST_VALUE(expr)
```

Value at a leading position

This function returns the value of argument from row leading current row within partition.

```
LEAD(expr, N)
```

Note: The value of N ranges between 0 and 2^{63}.

Value at a particular position

This function returns the value of argument from N-th row of window frame.

```
NTH_VALUE(expr, N)
```

Note: The value of N ranges between 0 and 2^{63}.

Row numbering

This function returns the number of the current row within its partition. The row numbers range from 1 to the number of partition rows.

```
ROW_NUMBER()
```

Note: `order by` affects the order in which rows are numbered.

Bucket numbering

Divides a partition into N groups (buckets), assigns each row in the partition its bucket number, and returns the bucket number of the current row within its partition.

`NTILE(N)`

Note: The value of N ranges between 0 and 2^{63}.

Cumulative distribution

This function returns the cumulative distribution of a value within a group of values (i.e., the percentage of partition values less than or equal to the value in the current row). This represents the number of rows preceding or peer with the current row in the window ordering of the window partition divided by the total number of rows in the window partition.

```
CUME_DIST()
```

Note: The returned value ranges between 0 and 1.

Additional features of MySQL

Feature	Description
`last_insert_id`	Returns the auto increment id of the last row that has been inserted or updated in a table
`coalesce`	Returns the first non-null value in a list
`ifnull`	Returns a specified value if the expression is NULL, otherwise returns the expression itself
`nullif`	Compares two expressions and returns NULL if they are equal, otherwise, returns the first expression
`case`	Passes through conditions and returns a value when the first condition is met

Problems

1. Show the use of case in MySQL.