Hàm cửa sổ

Ngày học	@November 6, 2023

1. Khái niệm

Trong SQL, hàm cửa sổ (window function) là một hàm được sử dụng để thực hiện tính toán trên một tập hợp dữ liệu được phân vùng bởi “cửa sổ” (window) trong kết quả của một truy vấn. Các hàm cửa sổ thực hiện việc tính toán trên một tập hợp các dòng và trả về một giá trị tổng hợp cho mỗi dòng. Hiểu một cách đơn giản, “cửa sổ” là một tập con của dữ liệu trong kết quả truy vấn và được xác định bằng cách sử dụng mệnh đề OVER().

2. Cú pháp

💡 **window_function([ALL] expression)**
OVER
(
[PARTITION BY partition_columns]
[ORDER BY sort_columns [ASC|DESC]]
[ROWS|RANGE BETWEEN start_position AND end_position]
)

> Note:
>
> - **window_function**
> Là tên của hàm cửa sổ như AVG, SUM, RANK, ROW_NUMBER,...
> - **ALL**
> Từ khóa ALL trong hàm cửa sổ được sử dụng để xác định rằng

hàm cửa sổ sẽ thực hiện tính toán trên tất cả các hàng trong cửa sổ xác định, bất kể giá trị trong cửa sổ có trùng nhau hay không. Nếu không sử dụng từ khóa ALL, hàm cửa sổ mặc định sẽ thực hiện tính toán trên các giá trị duy nhất trong cửa sổ.

- **expression**
  Là tên cột hoặc biểu thức được sử dụng để hàm cửa sổ tính toán trên đó.

- **PARTITION BY partition_columns**
  Danh sách các cột được sử dụng để phân vùng dữ liệu, mỗi phân vùng được gọi là một “cửa sổ”(window). Hàm cửa sổ sẽ thực hiện việc tính toán giá trị tổng hợp trên riêng từng phân vùng và khởi động lại quá trình tính toán khi qua phân vùng khác.
  Nếu PARTITION BY không được chỉ định thì toàn bộ dữ liệu sẽ được hiểu là nằm trong cùng một phân vùng.

- **ORDER BY sort_columns**
  Chỉ định danh sách các cột dùng để xác định thứ tự sắp xếp logic của các dòng khi thực hiện tính toán hàm cửa sổ.
  CÓ ORDER BY THÌ SẼ ÁP DỤNG PHẠM VI MẶC ĐỊNH, CẦN ĐIỀU CHỈNH LẠI KHI CODE

- Nếu mệnh đề ORDER BY không được chỉ định, thứ tự sắp xếp mặc định sẽ là tăng dần (ASC) và hàm cửa sổ sẽ sử dụng tất cả các dòng trong phân vùng để tính toán.

- Trong trường hợp có chỉ định mệnh đề ORDER BY mà không chỉ định phạm vi tính toán (tức là không sử dụng ROWS hoặc RANGE cho hàm cửa sổ) thì phạm vi tính toán mặc định sẽ là

UNBOUNDED PRECEDING AND CURRENT ROW (Nếu hàm cửa sổ có cho phép sử
dụng tùy chọn ROWS/RANGE trong cú pháp của hàm).
Lưu ý rằng, mệnh đề ORDER BY trong hàm cửa sổ có mục đích sử dụng khác với mệnh đề ORDER BY trong câu lệnh SELECT, vì nó chỉ ảnh hưởng đến cách tính toán của hàm cửa sổ, chứ không ảnh hưởng đến cách sắp xếp khi hiển thị kết quả truy vấn.

- **ROWS|RANGE BETWEEN start_position AND end_position**
  Xác định phạm vi tính toán của hàm cửa sổ, tức là giới hạn các dòng được sử dụng để tính toán trong phân vùng bằng cách chỉ định điểm bắt đầu và điểm kết thúc trong phân vùng. Mệnh đề ORDER BY bắt buộc phải được sử dụng trong trường hợp có sử dụng đến ROWS hoặc RANGE trong hàm cửa sổ.
  start_position và end_position có thể sử dụng một trong số các giá trị sau:

Giá trị	Ý nghĩa
UNBOUNDED PRECEDING	Dòng đầu tiên của phân vùng
N PRECEDING	N Dòng trước dòng hiện tại
CURRENT ROW	Dòng hiện tại
N FOLLOWING	N Dòng sau dòng hiện tại
UNBOUNDED FOLLOWING	Dòng cuối cùng của phân vùng

💡 **PARTITION BY: tạo ra các Window**

> 💡 **ORDER BY: không mang ý nghĩa sắp xếp, dùng để tạo Window Frame, mặc định quá trình tính toán theo kiểu lũy kế ([ROWS|RANGE BETWEEN start_position AND end_position]: Row hay Range mặc định tùy từng hàm)**

Muốn tính tổng so sánh trong phạm vi range

RANGE BETWEEN **CURRENT ROW and CURRENT ROW**

LAST VALUES: Nếu muốn lấy giá trị last values trong cả window là theo Window

RANGE BETWEEN **UNBOUNDED PRECEDING AND UNBOUNDED FOLLOWING**

3. Ví dụ

```
SELECT MatHang, Nam, DoanhThu,
SUM(DoanhThu) OVER() AS TongTatCa,

SUM(DoanhThu) OVER
(
PARTITION BY MatHang
) AS TongTheoMatHang,

SUM(DoanhThu) OVER
(
PARTITION BY MatHang
ORDER BY Nam
ROWS BETWEEN UNBOUNDED PRECEDING AND CURRENT ROW
) AS TongLuyKe,

SUM(DoanhThu) OVER
(
PARTITION BY MatHang
ORDER BY Nam
ROWS BETWEEN 1 PRECEDING AND 1 FOLLOWING
) AS TongLanCan
FROM ThongKeDoanhThu
```

	MatHang	Nam	DoanhThu	TongTatCa	TongTheoMatHang	TongLuyKe	TongLanCan
1	Beer	2010	10.00	156.00	47.00	10.00	14.00
2	Beer	2011	4.00	156.00	47.00	14.00	28.00
3	Beer	2013	14.00	156.00	47.00	28.00	23.00
4	Beer	2014	5.00	156.00	47.00	33.00	33.00
5	Beer	2015	14.00	156.00	47.00	47.00	19.00
6	Cafe	2010	2.00	156.00	38.00	2.00	13.00
7	Cafe	2011	11.00	156.00	38.00	13.00	28.00
8	Cafe	2012	15.00	156.00	38.00	28.00	31.00
9	Cafe	2013	5.00	156.00	38.00	33.00	25.00
10	Cafe	2015	5.00	156.00	38.00	38.00	10.00
11	Đường	2010	7.00	156.00	42.00	7.00	10.00
12	Đường	2011	3.00	156.00	42.00	10.00	20.00
13	Đường	2012	10.00	156.00	42.00	20.00	25.00
14	Đường	2013	12.00	156.00	42.00	32.00	32.00
15	Đường	2014	10.00	156.00	42.00	42.00	22.00
16	Gạo	2010	5.00	156.00	29.00	5.00	13.00
17	Gạo	2011	8.00	156.00	29.00	13.00	19.00
18	Gạo	2012	6.00	156.00	29.00	19.00	19.00
19	Gạo	2013	5.00	156.00	29.00	24.00	16.00
20	Gạo	2014	5.00	156.00	29.00	29.00	10.00

- Cột **TongTatCa** là kết quả của hàm: **SUM(DoanhThu) OVER()**

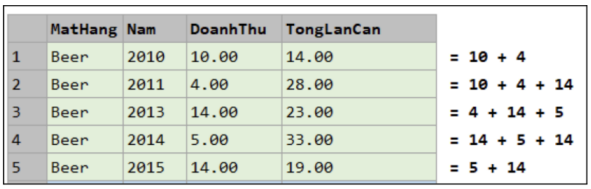
Hàm này **không chỉ định mệnh đề PARTITION BY** nên **tất cả dữ liệu trong bảng đều nằm trong cùng một phân vùng**. Hàm cửa sổ sẽ thực hiện tính toán tổng của cột DoanhThu trên tất cả các dòng dữ liệu hiện có trong bảng. Như chúng ta có thể thấy, giá trị của hàm này giống nhau ở tất cả các dòng.

- Cột **TongTheoMatHang** là kết quả của hàm:
  SUM(DoanhThu) OVER
  (
  PARTITION BY MatHang
)
  Trong hàm này dữ liệu được **phân thành các phân vùng dựa trên MatHang** (có tất cả 4 phân vùng). Hàm cửa sổ thực hiện tính **tổng các dòng dữ liệu trong mỗi phân vùng**. Chính vì vậy, chúng ta thấy rằng các dòng dữ liệu nằm trong cùng một phân vùng có cùng giá trị như nhau.

- Cột **TongLuyKe** được tính bởi hàm cửa sổ:
  SUM(DoanhThu) OVER
  (
  PARTITION BY MatHang
  ORDER BY Nam
  ROWS BETWEEN UNBOUNDED PRECEDING AND CURRENT ROW
) AS TongLuyKe
  Hàm này cũng sử dụng cột MatHang để phân chia dữ liệu thành các phân vùng. Trong hàm này, mệnh đề:
  ORDER BY Nam
  chỉ định thứ tự logic để tính toán trong mỗi phân vùng là theo thứ tự tăng dần
  của Nam, còn mệnh đề:
  ROWS BETWEEN UNBOUNDED PRECEDING AND CURRENT ROW
  xác định cách tính tổng tại mỗi dòng là từ dòng đầu tiên của phân vùng

	MatHang	Nam	DoanhThu	TongLuyKe	
1	Beer	2010	10.00	10.00	= 10
2	Beer	2011	4.00	14.00	= 10 + 4
3	Beer	2013	14.00	28.00	= 10 + 4 + 14
4	Beer	2014	5.00	33.00	= 10 + 4 + 14 + 5
5	Beer	2015	14.00	47.00	= 10 + 4 + 14 + 5 + 14
6	Cafe	2010	2.00	2.00	= 2
7	Cafe	2011	11.00	13.00	= 2 + 11
8	Cafe	2012	15.00	28.00	= 2 + 11 + 15
9	Cafe	2013	5.00	33.00	= 2 + 11 + 15 + 5
10	Cafe	2015	5.00	38.00	= 2 + 11 + 15 + 5 + 5

(UNBOUNDED PRECEDING) cho đến dòng hiện tại (CURRENT ROW). Có thể hình dung cách tính toán của hàm cửa sổ này như hình minh họa bên



	MatHang	Nam	DoanhThu	TongLanCan	
1	Beer	2010	10.00	14.00	= 10 + 4
2	Beer	2011	4.00	28.00	= 10 + 4 + 14
3	Beer	2013	14.00	23.00	= 4 + 14 + 5
4	Beer	2014	5.00	33.00	= 14 + 5 + 14
5	Beer	2015	14.00	19.00	= 5 + 14

- Kết quả ở cột **TongLanCan** được tính bởi hàm:
  SUM(DoanhThu) OVER
  (
  PARTITION BY MatHang
  ORDER BY Nam
  ROWS BETWEEN 1 PRECEDING AND 1 FOLLOWING
)
  Hàm này cũng sử dụng cách phân vùng dữ liệu và chỉ định thứ tự logic tính toán như ở trường hợp trên. Điểm khác biệt là phạm vi tính toán được xác định bởi mệnh đề:
  ROWS BETWEEN 1 PRECEDING AND 1 FOLLOWING
  Tức là, tại mỗi dòng, giá trị tổng sẽ được tính trên phạm vi từ dòng trước dòng hiện tại 1 dòng (1 PRECEDING) cho đến dòng phía sau dòng hiện tại 1 dòng (1 FOLLOWING). Lưu ý rằng, dòng đầu tiên (và dòng cuối cùng) trong phân vùng không có dòng trước (và dòng sau) nên tổng sẽ được tính dựa trên 2 giá trị thay vì 3 giá trị như ở các dòng còn lại.
  Có thể hình dung cách tính toán của hàm này như hình minh họa sau:

Trong mỗi năm, hãy lấy ra thông tin của những mặt hàng có doanh thu nằm trong top 2 doanh thu cao nhất

```
SELECT *
FROM (
SELECT *,
DENSE_RANK() OVER (PARTITION BY Nam ORDER BY DoanhThu DESC) AS ViThu
FROM ThongKeDoanhThu
) AS t
WHERE t.ViThu <= 2
```