

Урок 10.1. Основные оконные функции ранжирования

Основные ранжирующие функции	2
Применение на практике	3
Общий синтаксис	4
Примеры оконных функций ранжирования	5
Практическая работа	9

Основные ранжирующие функции



Важно!

База данных с доступом на чтение:

hostname: ich-db.edu.itcareerhub.de

username: ich1

password: password



ROW_NUMBER() — это функция, которая присваивает уникальный номер каждой строке в рамках окна.



RANK() — это функция, которая присваивает ранги строкам, пропуская ранги при одинаковых значениях.



DENSE_RANK() — это функция, которая присваивает ранги строкам, без пропуска рангов при одинаковых значениях.



NTILE() — это функция, которая делит строки на равные части (квантили) и присваивает номер группе.

Применение на практике

1. Анализ Позиций и Рангов

Оконные функции ранжирования, такие как `ROW_NUMBER()`, `RANK()`, и `DENSE_RANK()`, позволяют легко определить позиции и ранги строк в пределах данных.

Это полезно в ряде сценариев:

- Рейтинг товаров или сотрудников: Например, вы можете использовать `RANK()` для создания списка лучших продавцов или топ-продуктов на основе их продаж.
- Оценка результатов конкурсов: Вы можете ранжировать участников на основе их результатов.

2. Упрощение Сложных Запросов

Оконные функции упрощают написание сложных запросов, которые включают в себя многократные агрегирования или расчеты:

- Замена подзапросов: Оконные функции могут заменить сложные подзапросы или объединения, что делает запросы более читабельными и поддерживаемыми.
- Избежание дублирования данных: Функции, такие как `ROW_NUMBER()`, позволяют избежать необходимости повторного использования тех же данных в нескольких частях запроса.

3. Упорядочение и Группировка Данных

- Группировка данных: Оконные функции помогают упорядочить данные в рамках группы, определенной `PARTITION BY`, и затем выполнять операции, такие как подсчет или ранжирование, в рамках каждой группы.
- Создание отчетов: Вы можете использовать `NTILE()` для распределения данных по квартилям или другим группам, что полезно для создания отчетов о распределении данных.

Общий синтаксис

Unset

```
SELECT column1,  
       ranking_function(column2) OVER (  
         PARTITION BY column3  
         ORDER BY column4  
       ) AS rank_column  
FROM table_name;
```

- `ranking_function(column2)`: Оконная функция ранжирования, которая применяется.
- `PARTITION BY column3`: Разделяет данные на группы, если требуется.
- `ORDER BY column4`: Устанавливает порядок строк в рамках окна для ранжирования.

Примеры оконных функций ранжирования



Пример 1: ROW_NUMBER()

Присвоить уникальный номер каждой строке в таблице продаж, упорядоченной по дате.

Unset

```
SELECT SaleID, SaleDate, SaleAmount,
       ROW_NUMBER() OVER (ORDER BY SaleDate) AS RowNum
FROM Sales;
```

Входные данные

SaleID	SaleDate	SaleAmount
1	2024-01-01	100
2	2024-01-02	150
3	2024-01-03	200

Результат

SaleID	SaleDate	SaleAmount	RowNum
1	2024-01-01	100	1
2	2024-01-02	150	2
3	2024-01-03	200	3

ROW_NUMBER() присваивает уникальные номера строкам в порядке возрастания даты продажи.



Пример 2: RANK()

Unset

```
SELECT ProductID, DepartmentID, SaleAmount,  
       RANK() OVER (PARTITION BY DepartmentID ORDER BY SaleAmount DESC) AS Rank  
FROM ProductSales;
```

Входные данные

ProductID	DepartmentID	SaleAmount
1	10	300
2	10	300
3	10	200
4	20	500
5	20	400

Результат

ProductID	DepartmentID	SaleAmount	Rank
1	10	300	1
2	10	300	1
3	10	200	3
4	20	500	1
5	20	400	2

RANK() присваивает ранги по убыванию продаж в пределах каждого отдела, пропуская ранги при равенстве значений.



Пример 3: DENSE_RANK()

Unset

```
SELECT ProductID, SaleAmount,
       DENSE_RANK() OVER (ORDER BY SaleAmount DESC) AS DenseRank
FROM ProductSales;
```

Входные данные

ProductID	SaleAmount
1	300
2	300
3	200
4	500
5	400

Результат

ProductID	SaleAmount	DenseRank
1	300	1
2	300	1
3	200	3
4	500	1
5	400	2

DENSE_RANK() присваивает ранги по убыванию продаж без пропуска значений при равенстве.



Пример 4: NTILE()

Unset

```
SELECT SaleID, SaleAmount,  
       NTILE(3) OVER (ORDER BY SaleAmount) AS Quartile  
FROM Sales;
```

Входные данные

SaleID	SaleAmount
1	100
2	200
3	300
4	400
5	500

Результат

SaleID	SaleAmount	Quartile
1	100	1
2	200	1
3	300	2
4	400	2
5	500	3

NTILE(3) делит данные на 3 равные части по SaleAmount, присваивая номер каждой группе.

Практическая работа

Таблица products

1. Присвоить ранг продукту без пропусков значений в ранге от больше себестоимости к меньшей. Вывести ТОП 10 продуктов `product_name`.
2. Пронумеровать строки в таблице в зависимости от названия товара от А до Z.
3. Разделить все продукты на 4 равных группы в зависимости от `list_price`. Вывести имя продукта, `list_price` и номер группы.

Решение

1. Работа с таблицей `products`. Присвоить ранг продукту без пропусков значений в ранге от больше себестоимости к меньшей. Вывести ТОП 10 продуктов `product_name`.

Unset

```
SELECT product_name, DENSE_RANK() OVER (ORDER BY standard_cost DESC)
FROM products
LIMIT 10
```

2. Работа с таблицей `products`. Пронумеровать строки в таблице в зависимости от названия товара от А до Z.

Unset

```
SELECT product_name, ROW_NUMBER() OVER (ORDER BY product_name)
FROM products
```

3. Работа с таблицей `products`. Разделить все продукты на 4 равных группы в зависимости от `list_price`. Вывести имя продукта, `list_price` и номер группы.

Unset

```
SELECT product_name, list_price, NTILE(4) OVER (ORDER BY list_price)
FROM products
```