Назад к списку тем

Основы оконных функций

01 Как устроены оконные функции. Выражение OVER

Практические задания отсутствуют

---

02 Определение окна

Задача 1

```
SELECT * , SUM(total_amt) OVER() AS sum
FROM tools_shop.orders
```

Задача 2

```
SELECT *, COUNT(user_id) OVER()
FROM tools_shop.users
```

---

03 Операторы окна: PARTITION BY

Задача 1

```
SELECT *, SUM(total_amt) OVER(PARTITION BY user_id)
FROM tools_shop.orders
```

Задача 2

```
SELECT *, SUM(total_amt) OVER (PARTITION BY CAST(DATE_TRUNC('month', created_at) AS date))
FROM tools_shop.orders
```

---

04 Функции ранжирования: ROW_NUMBER()

Задача 1

```
SELECT *, ROW_NUMBER() OVER ()
FROM tools_shop.items
```

Задача 2

```
SELECT *, ROW_NUMBER() OVER ()
FROM tools_shop.orders
```

---

05 Операторы окна: ORDER BY

Задача 1

```
WITH dr AS
  (SELECT *,
          ROW_NUMBER() OVER (ORDER BY created_at) AS dr
   FROM tools_shop.users)
SELECT user_id
FROM dr
WHERE dr = 2021;
```

Задача 2

```
WITH dt AS
  (SELECT *,
          ROW_NUMBER() OVER (ORDER BY paid_at DESC) AS dt
   FROM tools_shop.orders)
SELECT 1431
FROM dt
WHERE dt = 50;
```

---

06 Функции ранжирования: RANK() и DENSE_RANK()

Задача 1

```
SELECT *, RANK() OVER (ORDER BY item_id)
FROM tools_shop.order_x_item
```

Задача 2

```
SELECT *, DENSE_RANK() OVER (ORDER BY created_at DESC) FROM tools_shop.users
```

---

07 Функция ранжирования NTILE()

Задача 1

```
SELECT order_id, total_amt, NTILE(10) OVER (ORDER BY total_amt)
FROM tools_shop.orders
```

Задача 2

```
SELECT user_id, created_at, NTILE(5) OVER (ORDER BY created_at DESC)
FROM tools_shop.users
```

---

08 Операторы окна: продолжение

Задача 1

```
SELECT *, RANK() OVER (PARTITION BY user_id ORDER BY paid_at)
FROM tools_shop.orders
```

Задача 2

```
SELECT *,
ROW_NUMBER() OVER (PARTITION BY user_id ORDER BY CAST(event_time AS timestamp) DESC)
FROM tools_shop.events
```

---

09 Агрегирующие оконные функции

Задача 1

```
SELECT *, COUNT(created_at) OVER (PARTITION BY created_at::date)
FROM tools_shop.orders
```

Задача 2

```
SELECT *,
SUM(total_amt) OVER (PARTITION BY CAST(DATE_TRUNC('month', created_at) AS date))
```

```
FROM tools_shop.orders
```

---

10 Расчёт кумулятивных значений

Задача 1

```
SELECT created_at, total_amt,
SUM(total_amt) OVER (ORDER BY created_at)
FROM tools_shop.orders
```

Задача 2

```
SELECT user_id, created_at, total_amt,
SUM(total_amt) OVER (PARTITION BY user_id ORDER BY created_at)
FROM tools_shop.orders
```

Задача 3

```
SELECT CAST(DATE_TRUNC('month', created_at) AS date), total_amt,
SUM(total_amt) OVER (ORDER BY CAST(DATE_TRUNC('month', created_at) AS date))
FROM tools_shop.orders
```

---

11 Функции смещения: LEAD(), LAG()

Задача 1

```
SELECT order_id, user_id, paid_at,
LAG(paid_at,1,'1980-01-01') OVER (PARTITION BY user_id ORDER BY paid_at )
FROM tools_shop.orders
```

Задача 2

```
SELECT event_id, event_time, user_id,
LEAD(event_time) OVER (PARTITION BY user_id ORDER BY event_time)
FROM tools_shop.events
```

Задача 3

```
SELECT event_id, event_time, user_id,
((LEAD(event_time) OVER (PARTITION BY user_id ORDER BY event_time)) - event_time) AS delta
```

```
FROM tools_shop.events
```

---

12 Особенности оконных функций

Практические задания отсутствуют

---

13 Оконные функции: практика

Задача 1

```
SELECT CAST(created_at AS date), costs,
ROW_NUMBER() OVER(ORDER BY costs DESC)
FROM tools_shop.costs
```

Задача 2

```
SELECT CAST(created_at AS date), costs,
DENSE_RANK() OVER(ORDER BY costs DESC)
FROM tools_shop.costs
```

Задача 3

```
WITH rang_users AS
(SELECT *, RANK() OVER (PARTITION BY user_id ORDER BY created_at) AS rn
FROM tools_shop.orders
)
SELECT DISTINCT(user_id)
FROM rang_users
WHERE rn = 3
```

Задача 4

```
WITH items AS
(SELECT *, ROW_NUMBER() OVER (PARTITION BY order_id) AS rn
FROM tools_shop.order_x_item
)

SELECT COUNT(*)
FROM items
WHERE rn = 4
```

Задача 5

```
SELECT
DISTINCT(CAST(DATE_TRUNC('month', created_at) AS date)),
COUNT(user_id) OVER (ORDER BY CAST(DATE_TRUNC('month', created_at) AS date))
FROM tools_shop.users
```

Задача 6

```
SELECT
DISTINCT(CAST(DATE_TRUNC('month', created_at) AS date)),
SUM(costs) OVER (ORDER BY CAST(DATE_TRUNC('month', created_at) AS date))
FROM tools_shop.costs
WHERE CAST(created_at AS date) BETWEEN '2017-01-01' AND '2018-12-31'
```

Задача 7

```
WITH
buyers AS
(
SELECT DISTINCT(user_id)
FROM tools_shop.orders
),
views AS
(
SELECT
DISTINCT(CAST(DATE_TRUNC('month', event_time) AS date)) AS month,
COUNT(events) OVER (PARTITION BY CAST(DATE_TRUNC('month', event_time) AS date)) AS count
FROM tools_shop.events
WHERE event_name LIKE 'view_item' AND user_id IN (SELECT * FROM buyers)
)

SELECT *, SUM(count) OVER (ORDER BY CAST(DATE_TRUNC('month', month) AS date))
FROM views
```

Задача 8

```
SELECT order_id, CAST(DATE_TRUNC('month', created_at) AS date), total_amt,
       COUNT(total_amt) OVER my_window,
       SUM(total_amt) OVER my_window
FROM tools_shop.orders
WINDOW my_window AS (ORDER BY CAST(DATE_TRUNC('month', created_at) AS date))
```

Задача 9

```
WITH
agg AS (SELECT CAST(DATE_TRUNC('month', created_at) AS date) AS month, SUM(costs) AS costs
```

```
FROM tools_shop.costs
GROUP BY CAST(DATE_TRUNC('month', created_at) AS date)
ORDER BY CAST(DATE_TRUNC('month', created_at) AS date)
)
SELECT *,
costs - LAG(costs,1, costs) OVER () AS delta
FROM agg
```

Задача 10

```
WITH
agg AS (SELECT CAST(DATE_TRUNC('year', paid_at) AS date) AS year, SUM(total_amt) AS total
FROM tools_shop.orders
GROUP BY CAST(DATE_TRUNC('year', paid_at) AS date)
ORDER BY CAST(DATE_TRUNC('year', paid_at) AS date)
)
SELECT *,
(LEAD(total,1, total) OVER ()) - total AS delta
FROM agg
```

---

14 Заключение

Практические задания отсутствуют

Назад к списку тем

Искренне ваш
A1ex Ma