1. В таблице trips содержатся поездки таксопарка ООО 'КЕХ Ромашка'. Для каждой поездки указаны client\_id и driver\_id,

которые являются внешними ключами на таблицу users (поле user\_id).

В поле status могут содержаться значения ('completed', 'cancelled\_by\_driver', 'cancelled\_by\_client').

Таблица users содержит всех пользователей таксопарка (и клиентов и водителей).

В поле role указана их роль в таксопарке, а в поле banned их статус блокировки.

Напишите запрос, который найдет коэффициент отмены в промежутке между 2020-02-01 и 2020-02-03.

Коэффициент отмены - отношение отмененных поездок к общему количеству поездок.

Учитывать нужно только незаблокированных пользователей (незаблокированы должны быть и клиент и водитель).

**Ограничения**: запрещено использовать HAVING, JOIN, связанные подзапросы (обращение в подзапросе к таблицам из внешнего запроса), LIMIT, UNION, UNION ALL, VALUES.

SELECT trips.request\_at,

TRUNCATE(

SUM(CASE WHEN trips.status != 'completed' THEN 1 ELSE 0 END) / COUNT(\*), 2

) as cancel\_rate

FROM

trips, users

WHERE trips.client\_id NOT IN (SELECT user\_id FROM users WHERE banned = '1')

AND trips.driver\_id NOT IN (SELECT user\_id FROM users WHERE banned = '1')

AND (trips.request\_at BETWEEN '2020-02-01' AND '2020-02-03')

GROUP BY trips.request\_at

2. Дана таблички с транзакциями пользователей.

CREATE TABLE transactions (

id int,

user\_id int4,

amount int4,

dtime timestamp

);

Нужно найти наибольшее число транзакций, которые сделал юзер за 30 суток (max\_count\_30\_day).

Можно считать, что у одного юзера нет 2 транзакция на один timestamp (timestamp = дата+время).

**Ограничения**: запрещено использовать LIMIT, UNION, UNION ALL, VALUES, IN.

WITH t3 AS(

SELECT user\_id, COUNT(\*) AS cnt

FROM transactions t1

JOIN transactions t2 USING (user\_id)

WHERE t2.dtime >= t1.dtime AND t2.dtime < (t1.dtime + INTERVAL 30 day)

GROUP BY t1.id,user\_id

)

SELECT user\_id, MAX(cnt) AS max\_count\_30\_day

FROM t3

GROUP BY user\_id

ORDER BY user\_id

3. -- d7\_buyer(id int, name text, surname text, last\_action\_date date);-- d7\_seller(id int, name text, surname text, last\_action\_date date);-- d7\_manager(id int, name text, last\_action\_date date);-- d7\_user(id serial, role text, registration\_date date)

Посчитайте количество пользователей в разрезе квартала регистрации и месяца.   
Подведите подытог по каждому разрезу  и общий подытог, отсортируйте по первой и второй колонке.  
Формат ответа 'q', 'm', 'cntd\_users'

SELECT QUARTER(registration\_date) as q, MONTH(registration\_date) as m, COUNT(DISTINCT id) as cntd\_users

FROM d7\_user

GROUP BY QUARTER(registration\_date), MONTH(registration\_date) WITH ROLLUP

ORDER BY 1,2

4. -- d8\_scores (event\_date, category, subcategory, value)

Для дат с 2021-01-19 по 2021-01-21 выведите среднее значение value за предыдущие два дня для каждой подкатегории категории A

Например, для даты 2021-01-19 вычисление среднего должно происходить по датам 2021-01-17, 2021-01-18.

Ожидаемый формат ответа

WITH cte AS

(SELECT \*, AVG(value) OVER

(

PARTITION BY subcategory ORDER BY event\_date RANGE BETWEEN INTERVAL 2 day PRECEDING AND interval 1 day PRECEDING

) AS last\_2d\_avg

FROM d8\_scores

WHERE category = 'A')

SELECT \* FROM cte

WHERE (event\_date BETWEEN '2021-01-19' AND '2021-01-21')

AND (category = 'A')

ORDER BY 1,2,3

5. -- create table d9\_datamarts (dm varchar(64), calc\_time int);  
-- create table d9\_dag (src varchar(64), tgt varchar(4));

Даны расчеты табличек в хранилище данных.

*d9\_datamarts* - хранит название таблички и время в минутах которое необходимо для ее заполнения

*d9\_dag* - хранит последовательность, в которой нужно заполнять таблички в виде пар.

Нужно вывести цепочку табличек которая заполняется дольше всего.

Цепочка состоит из названий таблиц, разделенных символами ' -> '

WITH recursive path(pth,prev,time) AS

(

SELECT d1.src AS pth,

d1.src AS prev,

d2.calc\_time FROM

d9\_dag d1 JOIN d9\_datamarts d2

ON d1.src = d2.dm

UNION ALL

SELECT CONCAT(path.pth,' -> ',dg.tgt),

dg.tgt,

path.time + dm.calc\_time

FROM path

JOIN d9\_dag dg

ON path.prev = dg.src

JOIN d9\_datamarts dm

ON dg.tgt = dm.dm

)

SELECT DISTINCT pth AS calc\_path, time AS calc\_time

FROM path

WHERE time = (SELECT MAX(time) FROM path)