--1. Узнаем, когда была первая транзакция для каждого студента. Начиная с этой даты, мы будем собирать его баланс уроков.

with first\_payments as

(

select user\_id

, min(transaction\_datetime) as first\_payment\_date

from skyeng\_db.payments

where status\_name = 'success'

group by user\_id

),

--2. Соберем таблицу с датами за каждый календарный день 2016 года

all\_dates as

(

select distinct date\_trunc('day', class\_start\_datetime) as dt

from skyeng\_db.classes

where date\_trunc('year', class\_start\_datetime) = '2016-01-01'

),

--3. Узнаем, за какие даты имеет смысл собирать баланс для каждого студента

all\_dates\_by\_user as

(

select user\_id

, dt

from first\_payments

cross join all\_dates

where all\_dates.dt >= date\_trunc('day', first\_payments.first\_payment\_date)

),

--4. Найдем все изменения балансов, связанные с успешными транзакциями.

payments\_by\_dates as

(

select user\_id

, date\_trunc('day', transaction\_datetime) as payment\_date

, sum (classes) as transaction\_balance\_change

from skyeng\_db.payments

where status\_name = 'success'

group by user\_id, payment\_date

),

--5. Найдем баланс студентов, который сформирован только транзакциями

payments\_by\_dates\_cumsum as

(

select a.user\_id

, dt

, transaction\_balance\_change

, coalesce(sum(transaction\_balance\_change) over(partition by a.user\_id order by dt), 0) as transaction\_balance\_change\_cs

from all\_dates\_by\_user a

left join payments\_by\_dates b

on a.user\_id = b.user\_id and a.dt = b.payment\_date

),

--6. Найдем изменения балансов из-за прохождения уроков

classes\_by\_dates as

(

select user\_id

, date\_trunc('day', class\_start\_datetime) as class\_date

, count(distinct id\_class) \* (-1) as classes

from skyeng\_db.classes

where class\_status in ('success', 'failed\_by\_student') and class\_type <> 'trial'

group by user\_id, class\_date

),

--7. По аналогии с уже проделанным шагом для оплат создадим CTE для хранения кумулятивной суммы количества пройденных уроков

classes\_by\_dates\_dates\_cumsum as

(

select a.user\_id

, dt

, classes

, coalesce(sum(classes) over(partition by a.user\_id order by dt), 0) as classes\_cs

from all\_dates\_by\_user a

left join classes\_by\_dates b

on a.user\_id = b.user\_id and a.dt = b.class\_date

),

--8. Создадим CTE balances с вычисленными балансами каждого студента.

balances as

(

select a.user\_id

, a.dt

, transaction\_balance\_change

, transaction\_balance\_change\_cs

, classes

, classes\_cs

, (classes\_cs + transaction\_balance\_change\_cs) as balance

from payments\_by\_dates\_cumsum a

full join classes\_by\_dates\_dates\_cumsum b

on a.user\_id = b.user\_id and a.dt = b.dt

)

--9. Посмотрим, как менялось общее количество уроков на балансах студентов

select dt

, sum(transaction\_balance\_change) sum\_transaction\_balance\_change

, sum(transaction\_balance\_change\_cs) sum\_transaction\_balance\_change\_cs

, sum(classes) sum\_classes

, sum(classes\_cs) sum\_classes\_cs

, sum(balance) sum\_balance

from balances

group by dt

order by dt