**Вопросы к дата-инженерам:**

У нас есть много отрицательных значений баланса. Как будто ученики проходили уроки, не оплатив их предварительно. Возможно, в таблице payments есть ошибки? Может ли быть такое, что не вся информация об оплатах хранится в БД?

Есть аномалия 8 декабря 2017, когда за один день было оплачено 2868 уроков. Также подозреваем ошибку в таблице payments.

Возможно ли, что оплаты какое-то время не подгружались в БД, и уроки списывались как будто в минус, а оплаты подгрузились позже с датой 8 декабря?

**Общий вывод:**

По графику видно, что кумулятивная сумма балансов и списаний увеличивается, что значит, что ученики оплачивают и расходуют уроки в течение всего периода наблюдений. Делаем вывод, что динамика развития сервиса положительная.

**Код:**

with first\_payments as (

select user\_id, min(date(transaction\_datetime)) as first\_payment\_date

from skyeng\_db.payments

where status\_name = 'success'

group by user\_id),

all\_dates as (

select distinct date(class\_start\_datetime) as dt

from skyeng\_db.classes),

payments\_by\_dates as (

select user\_id, date(transaction\_datetime) as date, sum(classes) as transaction\_balance\_change

from skyeng\_db.payments

where status\_name = 'success'

group by user\_id, date),

all\_dates\_by\_user as (

select user\_id, dt

from first\_payments

join all\_dates on first\_payments.first\_payment\_date <= all\_dates.dt

order by user\_id),

balance as (

select \*

, sum(transaction\_balance\_change) over (partition by payments\_by\_dates.user\_id order by date) as balance

from payments\_by\_dates

join all\_dates\_by\_user on payments\_by\_dates.user\_id = all\_dates\_by\_user.user\_id and payments\_by\_dates.date = all\_dates\_by\_user.dt

order by payments\_by\_dates.user\_id),

classes\_by\_dates as (

select user\_id, date(class\_start\_datetime) as class\_date

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

from skyeng\_db.classes

where class\_type != 'trial' and (class\_status = 'success' or class\_status = 'failed\_by\_student')

group by user\_id, class\_date),

payments\_by\_dates\_cumsum as (

select a.user\_id, dt, coalesce(b.transaction\_balance\_change, 0) as transaction\_balance\_change,

sum(transaction\_balance\_change) over (partition by a.user\_id order by dt rows between unbounded preceding and current row) 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.date),

classes\_by\_dates\_dates\_cumsum as (

select a.user\_id, a.dt, coalesce(b.classes, 0) as classes,

sum(classes) over (partition by a.user\_id order by dt rows between unbounded preceding and current row) 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),

balances as (

select a.user\_id, a.dt, transaction\_balance\_change, transaction\_balance\_change\_cs, classes, coalesce(classes\_cs,0) as classes\_cs,

case when classes\_cs is null then transaction\_balance\_change\_cs else transaction\_balance\_change\_cs + classes\_cs end as balance

from payments\_by\_dates\_cumsum a

left join classes\_by\_dates\_dates\_cumsum b on a.user\_id = b.user\_id and a.dt = b.dt)

-- select \*

-- from balances

-- order by balance, user\_id, dt

-- limit 1000

select dt,

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

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

sum(classes) as classes,

sum (classes\_cs) as classes\_cs,

sum(balance) as balance

from balances

group by dt

order by dt