-----------------------

--Задание 1.1

--Соберите таблицу, в которой для каждого студента будет храниться его первая дата успешной (со статусом success) транзакции (из таблицы payments).В результирующей таблице должно быть два поля: user\_id и first\_payment\_date (дата первой успешной транзакции).

-----------------------

-- select user\_id

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

-- from skyeng\_db.payments

-- where status\_name='success'

-- group by 1

-- order by 1

-----------------------

--Задание 1.2

--Соберите таблицу, которая будет хранить в себе все дни 2016 года. Для этого необходимо выбрать все даты из таблицы classes.В результирующей таблице должно быть одно поле dt (уникальные даты уроков, без времени).

-----------------------

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

-- from skyeng\_db.classes

-- where class\_start\_datetime between '2016-01-01' and '2016-12-31'

-----------------------

--Задание 1.3

--Соберите таблицу со списком успешных транзакций из таблицы payments. Сделайте агрегацию, в которой для каждого клиента на каждую дату необходимо вывести сумму поля classes (только для успешных транзакций).В результирующей таблице должно быть три поля: user\_id, payment\_date, transaction\_balance\_change (сколько уроков было начислено или списано в этот день)

-----------------------

-- select user\_id

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

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

-- from skyeng\_db.payments

-- where lower(status\_name) ~ 'success' or lower(status\_name)~'failed\_by\_student'

-- group by user\_id, payment\_date

-- order by user\_id

-----------------------

--Задание 2.1

--Создайте СТЕ для всех запросов из первой части:

--Первый запрос назовите first\_paymentsВторой запрос назовите all\_dates

--Третий запрос назовите payments\_by\_dates

-----------------------

with first\_payments as (

select user\_id

,date\_trunc('day',min(transaction\_datetime)) as first\_payment\_date

from skyeng\_db.payments

where status\_name='success'

group by user\_id, status\_name

order by 1

),

all\_dates as (

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

from skyeng\_db.classes

where class\_start\_datetime between '2016-01-01' and '2016-12-31 23:59:59'

),

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 1, 2

order by user\_id

)

,

-----------------------

--Задание 2.2

--Создайте СТЕ под названием all\_dates\_by\_user, в котором объедините all\_dates и first\_payments, где будут храниться все даты жизни студента после того, как произошла его первая транзакция.В результирующем СТЕ должны быть поля: user\_id, dt.

-----------------------

all\_dates\_by\_user as (

select f.user\_id

,dt

from first\_payments f

join all\_dates a on a.dt>=f.first\_payment\_date

--group by 1,2

),

-----------------------

--Задание 2.3

--Найдите изменения балансов из-за прохождения уроков. Создайте CTE classes\_by\_dates, посчитав в таблице classes количество уроков за каждый день для каждого ученика. Вас не должны интересовать вводные уроки и уроки со статусом, отличным от success и failed\_by\_student. В результирующем СТЕ должны быть поля: user\_id, class\_date, classes (количество пройденных в этот день уроков). Причем classes нужно умножить на -1, чтобы отразить, что - — это списания с баланса.

-----------------------

classes\_by\_dates as (

select user\_id

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

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

from skyeng\_db.classes

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

group by 1, 2

)

,

-----------------------

--Задание 3.1

--Найдите баланс студентов, который сформирован только транзакциями. Для этого объедините all\_dates\_by\_userи payments\_by\_datesтак, чтобы совпадали даты и user\_id . Используйте оконные функции (в частности, функцию sum), чтобы найти кумулятивную сумму по полю \*\*\*\*transaction\_balance\_change для всех строк до текущей включительно с разбивкой по user\_id и сортировкой по dt.В результате получим CTE payments\_by\_dates\_cumsum с полями: user\_id, dt, transaction\_balance\_change , transaction\_balance\_change\_cs(кумулятивная сумма по transaction\_balance\_change). При подсчете кумулятивной суммы можно заменить пустые значения нулями.

-----------------------

payments\_by\_dates\_cumsum as(

select a.user\_id

,dt

--,case when transaction\_balance\_change is null then 0 else transaction\_balance\_change end as transaction\_balance\_change

,coalesce(transaction\_balance\_change,0) as transaction\_balance\_change

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

from all\_dates\_by\_user a

left join payments\_by\_dates p on a.user\_id=p.user\_id and a.dt= p.payment\_date

),

----------------------------------

--Задание 3.2

--По аналогии с уже проделанным шагом для оплат создайте CTE classes\_by\_dates\_dates\_cumsum для хранения кумулятивной суммы количества пройденных уроков. Для этого объедините таблицы all\_dates\_by\_user и classes\_by\_dates так, чтобы совпадали даты и user\_id. Используйте оконные выражения (функцию sum), чтобы найти кумулятивную сумму по полю classes для всех строк до текущей включительно с разбивкой по user\_id и сортировкой по dt.В результате получим CTE classes\_by\_dates\_dates\_cumsumс полями: user\_id, dt, classes , classes\_cs(кумулятивная сумма по classes). При подсчете кумулятивной суммы обязательно нужно заменить пустые значения нулями.

-----------------------------------

classes\_by\_dates\_dates\_cumsum as (

select u.user\_id

,dt

,case when c.classes is null then 0 else c.classes end as classes

,sum(case when c.classes is null then 0 else c.classes end) over (partition by u.user\_id order by dt) as classes\_cs

from all\_dates\_by\_user u

left join classes\_by\_dates c on (u.user\_id=c.user\_id and u.dt=c.class\_date )

order by user\_id, dt

),

----------------------------------------

--Задание 3.3

Создайте CTE balances \*\*\*\*с вычисленными балансами каждого студента. Для этого объедините таблицы payments\_by\_dates\_cumsum \*\*\*\*и classes\_by\_dates\_dates\_cumsum так, чтобы совпадали даты и user\_id.В результате получим \*CTE \*\*\*\*\*balances \*\*\*\*со следующими полями: user\_id, dt, transaction\_balance\_change, transaction\_balance\_change\_cs, classes, classes\_cs, balance (classes\_cs + transaction\_balance\_change\_cs).

------------------------------------------------------

balances as (

select t.user\_id

,t.dt

,case when transaction\_balance\_change is null then 0 else transaction\_balance\_change end as transaction\_balance\_change

,transaction\_balance\_change\_cs

,case when classes is null then 0 else classes end as classes

,classes\_cs

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

from payments\_by\_dates\_cumsum t

join classes\_by\_dates\_dates\_cumsum r on (t.user\_id=r.user\_id and t.dt=r.dt )

)

----------------------------------------

--Задание 3.4

--Выберите топ-1000 строк из CTE balances с сортировкой по user\_id и dt. Посмотрите на изменения балансов студентов. Какие вопросы стоит задать дата-инженерам и владельцам таблицы payments?

----------------------------------------

-- select \*

-- from balances

-- order by user\_id, dt

-- limit 1000

----------------------------------------

--Задание 3.5

--Посмотрите, как менялось общее количество уроков на балансах студентов. Для этого просуммируйте поля transaction\_balance\_change, transaction\_balance\_change\_cs, classes, classes\_cs, balance из CTE balances с группировкой и сортировкой по dt.

----------------------------------------

select dt

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

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

,sum(classes) as sum\_cl\_cl

,sum(classes\_cs) as sum\_cl\_cs

,sum(balance) as sum\_bal

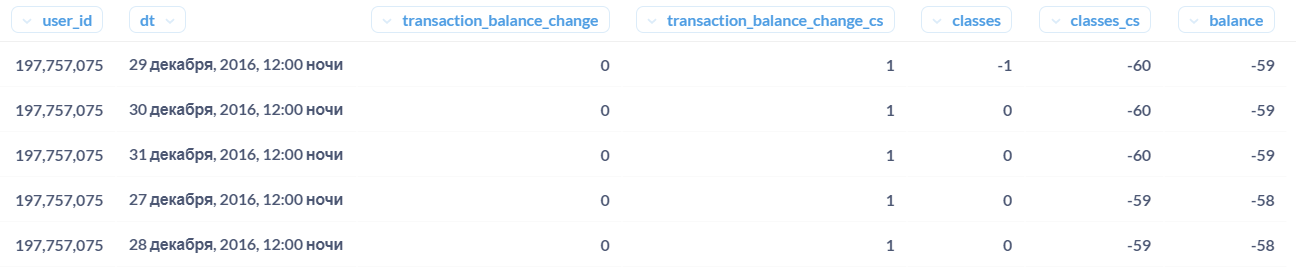
from balances

group by dt`

order by dt

Задание 1

соберите вопросы к дата-инженерам и владельцам таблицы payments



Вывод:Имеется отрицательный баланс, что странно! Возможно это ошибка работы , возможность бесплатно учится?!?!

Задание 2

Cоберите выводы из визуализации (линейной диаграммы) получившегося результата

**График с Matabase**

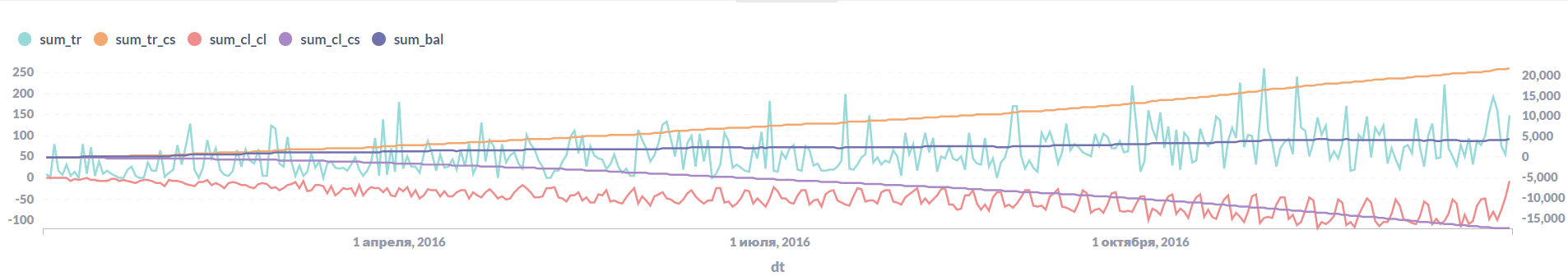
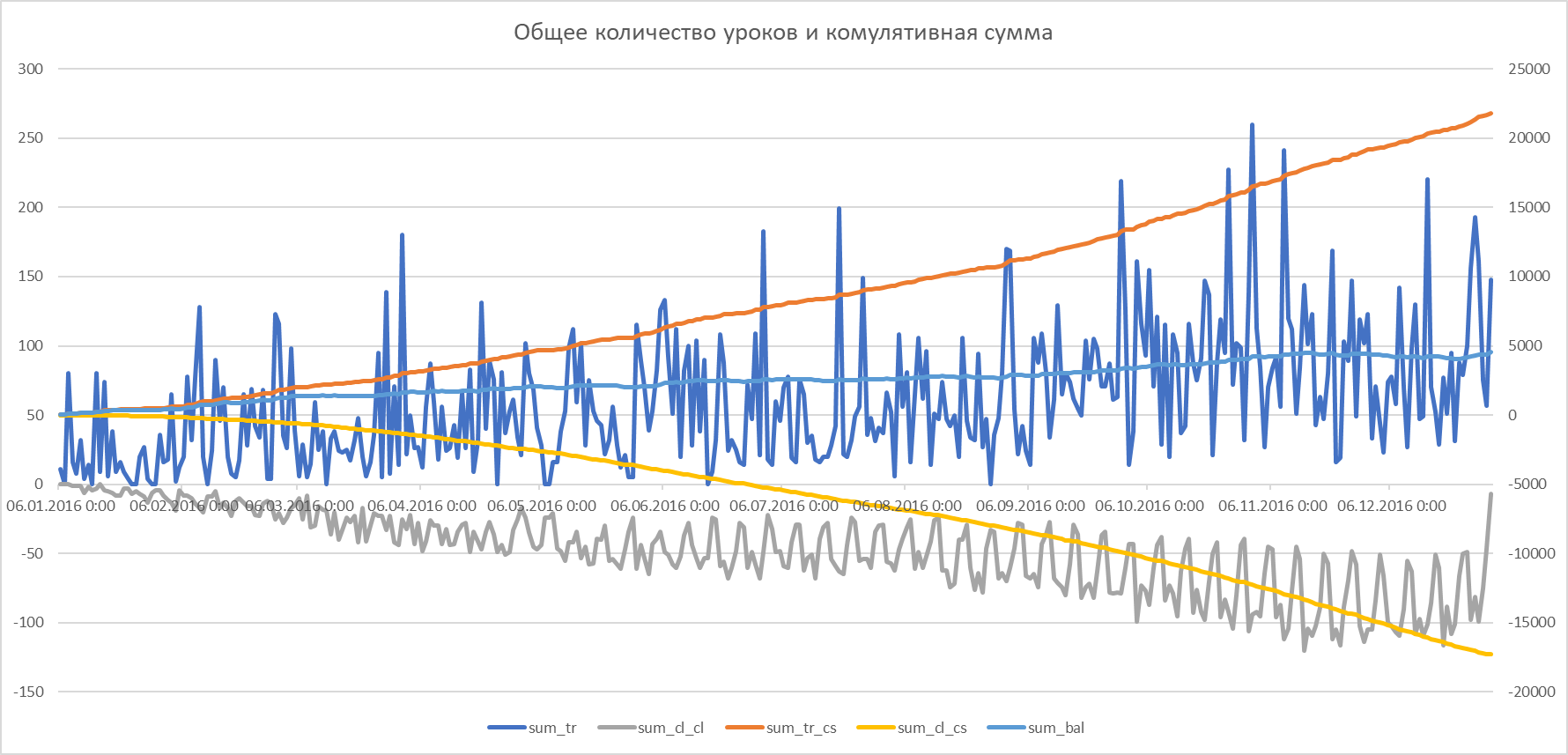


График с Excel



Вывод:Как видно из графика кумулятивной функции баланса у нас большой профицит по проданным урокам! С одной стороны это положительный эффект с точки зрения полученных денег для школы! Однако накопительный эффект количества купленных уроков может отрицательно сказаться - при резком запросе этих уроков сможет ли с этим справиться с этим платформа, хватит ли учителей и выдержат ли они такую нагрузку! нужно рассмотреть данную проблему!