select user\_id

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

from skyeng\_db.payments

where status\_name = 'success'

group by user\_id

order by user\_id

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

from skyeng\_db.classes

where class\_start\_datetime between '201-01-01' and '2016-12-31'

select user\_id

, transaction\_datetime as payment\_date

, sum(classes) as transaction\_balance\_change

from skyeng\_db.payments

where status\_name = 'success'

group by user\_id

, transaction\_datetime

, classes

order by user\_id

with first\_payments as (

select user\_id,

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

from SKYENG\_DB.payments

where status\_name = 'success'

group by user\_id -- готово, это Шаг 1 из плана курсовой

) ,

all\_dates as (

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

-- , user\_id -- это поле не нужно мы его удаляем

from skyeng\_db.classes

where class\_start\_datetime between '2015-12-31' and '2017-01-01'

order by dt -- готово, это Шаг 2 из плана курсовой

),

all\_dates\_by\_user as (

select p.user\_id,

d.dt

from first\_payments p

join all\_dates d

on dt >= first\_payment\_date -- готово, это Шаг 3 из плана курсовой

order by user\_id, dt

),

payments\_by\_dates as (

select user\_id,

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

sum(classes) as transaction\_balance\_change

from SKYENG\_DB.payments

where status\_name = 'success'

group by user\_id, payment\_date

order by user\_id -- готово, это Шаг 4 из плана курсовой

) ,

classes\_by\_dates as ( -- Найдите изменения балансов из-за прохождения уроков.

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

user\_id,

count (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 -- это Шаг 6 из плана

),

payments\_by\_dates\_cumsum as (

select a.user\_id,

a.dt,

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

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

order by a.user\_id -- это шаг 5 из плана

),

classes\_by\_dates\_dates\_cumsum as (

select a.user\_id,

a.dt,

coalesce(b.classes,0) as classes,

sum(coalesce(b.classes,0)) over (partition by a.user\_id order by a.dt) 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) -- шаг 7 кумулятивной суммы количества пройденных уроков.

),

balances as (

select a.user\_id,

a.dt,

a.transaction\_balance\_change,

a.transaction\_balance\_change\_cs,

b.classes,

b.classes\_cs,

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

from payments\_by\_dates\_cumsum a

join classes\_by\_dates\_dates\_cumsum b using (user\_id, dt)

)

-- select \*

-- from balances

-- order by balance

-- limit 1000

select dt

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

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

, sum(classes) as sum\_classes

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

, sum(balance) as sum\_balance

from balances

group by dt

order by dt

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 user\_id),

all\_dates as (select distinct date\_trunc('day', class\_start\_datetime) as dates\_of\_classes

from SKYENG\_DB.classes

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

order by dates\_of\_classes),

payments\_by\_dates as (select user\_id, date\_trunc('day',transaction\_datetime) as class\_date, sum(classes) as transaction\_balance\_change

from SKYENG\_DB.payments

where status\_name = 'success'

group by user\_id, class\_date

order by user\_id),

all\_dates\_by\_user as (select f.user\_id, d.dates\_of\_classes

from first\_payments f

join all\_dates d on d.dates\_of\_classes>=f.first\_payment\_date),

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\_status in ('success', 'failed\_by\_student')

and class\_type != 'trial'

group by user\_id, class\_date),

classes\_by\_dates\_dates\_cumsum as (select m.user\_id, coalesce(k.classes,0) as classes,

sum(coalesce(k.classes,0)) over (partition by m.user\_id order by m.dates\_of\_classes rows between unbounded preceding and current row) classes\_cs,

m.dates\_of\_classes

from all\_dates\_by\_user m

left join classes\_by\_dates k on (m.user\_id=k.user\_id and m.dates\_of\_classes=k.class\_date) ),

payments\_by\_dates\_cumsum as (select u.user\_id,

u.dates\_of\_classes,

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

sum(coalesce(s.transaction\_balance\_change, 0)) over (partition by u.user\_id order by u.dates\_of\_classes) as transaction\_balance\_change\_cs

from all\_dates\_by\_user u

left join payments\_by\_dates s on (u.user\_id=s.user\_id and u.dates\_of\_classes=s.class\_date)

order by s.class\_date),

balances as (select a.user\_id,

v.dates\_of\_classes,

a.transaction\_balance\_change,

a.transaction\_balance\_change\_cs,

v.classes,

v.classes\_cs,

v.classes\_cs + a.transaction\_balance\_change\_cs as balance

from payments\_by\_dates\_cumsum a

join classes\_by\_dates\_dates\_cumsum v on (a.user\_id=v.user\_id and a.dates\_of\_classes=v.dates\_of\_classes))

-- select \*

-- from balances

-- order by user\_id, dates\_of\_classes

-- limit 1000

select dates\_of\_classes,

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 dates\_of\_classes

order by dates\_of\_classes

**По анализу графика можно сделать следующие выводы:**

* стабильный рост количества покупок в течении года
* предпочтение студентов проходить уроки в будние дни, а не в выходные
* транзакции, в начале или конце месяца, вероятно связанные с получением зарплаты или аванса
* накопление на конец года около 4,5 тысяч не пройденных уроков, что может сигнализировать о проблеме - возможно, уроки стали не интересными или слишком сложными

**Вопрос ы к Датаинженерам на основе анализа графика:**

Баланс идет достаточно ровно, оплаты с уроками увеличиваются пропорционально, но в разных плоскостях. Поэтому у меня вопрос - почему уроки уходят в минус? Перенос другому ученику? Система оплаты с овердрафтом?