Подразумевается использование postgresql

**Задание 1.1**

with main as

(

select to\_char(show\_date,'YYYY-mm-DD') as date,watch\_id,content\_id from content\_watch

Where to\_timestamp(extract(epoch from current\_timestamp) - 86400\*30) < show\_date and platform in (10,11)

)

select a.date,b.paid\_type,count(a.watch\_id) as watch\_count from main a left join content b on a.content\_id=b.content\_id

group by a.date,b.paid\_type

order by a.date desc,b.paid\_type;

**Задание 1.2**

with main as

(

select to\_char(a.show\_date, 'YYYY-mm') as date,a.content\_id,a.user\_id,compilation\_id from content\_watch a left join content b on a.content\_id=b.content\_id

),

film as

(

select date,content\_id,count(distinct user\_id) as film\_count from

(

select \* from main where compilation\_id is null

) as foo

group by date,content\_id

),

serial as

(

select date,compilation\_id,count(distinct user\_id) as serial\_count from

(

select \* from main where compilation\_id is not null

) as foo

group by date,compilation\_id

),

film\_rank as

(

select \*,row\_number() over(partition by date order by film\_count desc) as film\_rank from film

),

serial\_rank as

(

select \*,row\_number() over(partition by date order by serial\_count desc) as serial\_rank from serial

)

select a.date,a.content\_id as film,a.film\_count,b.compilation\_id as serial,serial\_count from film\_rank a full outer join serial\_rank b on a.date=b.date and a.film\_rank=b.serial\_rank

where serial\_rank <= 5 or film\_rank <=5

order by a.date,a.film\_count;

**Задание 1.3**

with main as

(

select user\_id,utm\_medium as first,lead(utm\_medium) over(partition by user\_id order by show\_date) as second from

(

select \* from content\_watch

where to\_char(show\_date,'YYYY-mm-DD') = to\_char(to\_timestamp(extract(epoch from current\_timestamp)-86400),'YYYY-mm-DD')

) as foo

)

Select user\_id from main Where first='organic' and second='referral';

**Задание 2.1**

Так как мы ориентируемся на 4 серии условимся оценивать в этих рамках, для этого я использовал отношение разности между максимальным и минимальным номером эпизода к 3(максимальная разница в рамках 4 серий) по каждому пользователю и взял среднее. Также можно было бы посмотреть статистику просмотренного хронометража от общего, чтобы можно было сравнивать сериалы с различным количеством серий в рамках всех серий, а не только 4.

with first as

(

select \* from content

where compilation\_id in

(

select distinct compilation\_id from content where episode = 4

)

and episode <= 4

),

second as

(

select distinct b.compilation\_id,a.user\_id,min(b.episode) over(partition by b.compilation\_id,a.user\_id) as min\_episode,max(b.episode) over(partition by b.compilation\_id,a.user\_id) as max\_episode from content\_watch a left join first b on a.content\_id=b.content\_id

)

select compilation\_id,round(avg(max\_episode-min\_episode)/3,2) as serial\_retention from second

group by compilation\_id;

**Задание 2.2**

Сегментировать пользователей можно по многим параметрам, здесь я использовал айди платформы для примера. Что касается рекомендаций по улучшению показателя ретеншн, то

* я бы использовал информацию о возрасте, поле, региона человека и обратил бы внимание на улучшение рекомендательной системы либо поставил бы вопрос о необходимости дополнительного целевого контента для проблемных сегментов
* разделил бы юзеров на платящих и неплатящих и отдельно исследовал их сессии(возможно они встречаются с какой-то проблемой или не доходят до полезных разделов/фич)
* сравнил показатели в зависимости от платформы и обратил бы внимание на преимущества одной перед другой
* также как вариант, можно сегментировать юзеров по предпочтению видов контента в основном сериалы/фильмы или же смешанный тип и предлагать индивидуальные подписки или разделы

with main as

(

select user\_id,platform,to\_char(start,'YYYY-mm') as start,extract(epoch from show\_date) as date,

(extract(epoch from (start))+86400\*30\*1) as month\_1,

(extract(epoch from (start))+86400\*30\*2) as month\_2,

(extract(epoch from (start))+86400\*30\*3) as month\_3,

(extract(epoch from (start))+86400\*30\*4) as month\_4,

(extract(epoch from (start))+86400\*30\*5) as month\_5,

(extract(epoch from (start))+86400\*30\*6) as month\_6,

(extract(epoch from (start))+86400\*30\*7) as month\_7 from

(

select user\_id,platform,show\_date,min(show\_date) over(partition by user\_id) as start

from content\_watch

) as foo

),

a as

(

select distinct user\_id from main where date between month\_1 and month\_2

),

b as

(

select distinct user\_id from main where date between month\_2 and month\_3

),

c as

(

select distinct user\_id from main where date between month\_3 and month\_4

),

d as

(

select distinct user\_id from main where date between month\_4 and month\_5

),

e as

(

select distinct user\_id from main where date between month\_5 and month\_6

),

f as

(

select distinct user\_id from main where date between month\_6 and month\_7

)

select main.platform,count(distinct main.user\_id),(count(distinct a.user\_id)\*100)/count(distinct main.user\_id) as month\_1,(count(distinct b.user\_id)\*100)/count(distinct main.user\_id) as month\_2,

(count(distinct c.user\_id)\*100)/count(distinct main.user\_id) as month\_3,(count(distinct d.user\_id)\*100)/count(distinct main.user\_id) as month\_4,(count(distinct e.user\_id)\*100)/count(distinct main.user\_id) as month\_5,

(count(distinct f.user\_id)\*100)/count(distinct main.user\_id) as month\_6 from main left join

a on main.user\_id=a.user\_id left join

b on main.user\_id=b.user\_id left join

c on main.user\_id=c.user\_id left join

d on main.user\_id=d.user\_id left join

e on main.user\_id=e.user\_id left join

f on main.user\_id=f.user\_id

group by main.platform;

**Задание 3**

[Ноутбук с решением](https://colab.research.google.com/drive/1PXj1R1RCzRyRF165Z2sQTlFaTpIcV1gq?usp=sharing)