codecademy

Codeflix: Churn Rates

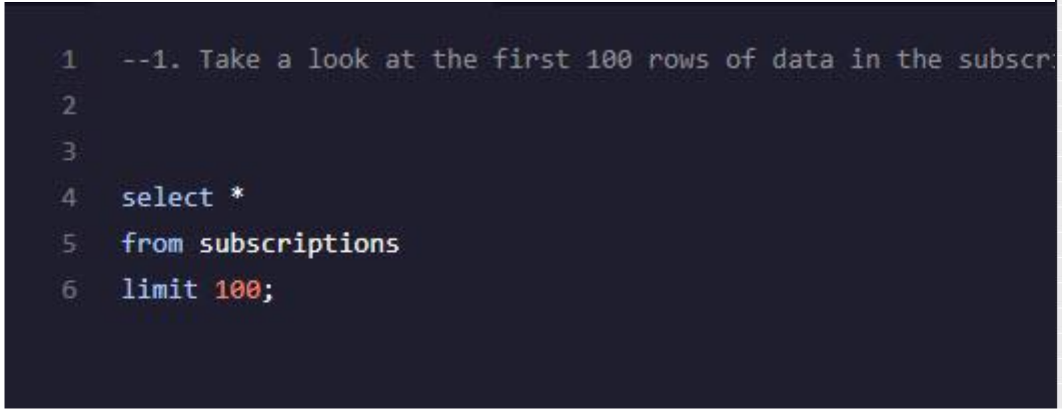
Learn SQL from Scratch

By Evaldas Girenka

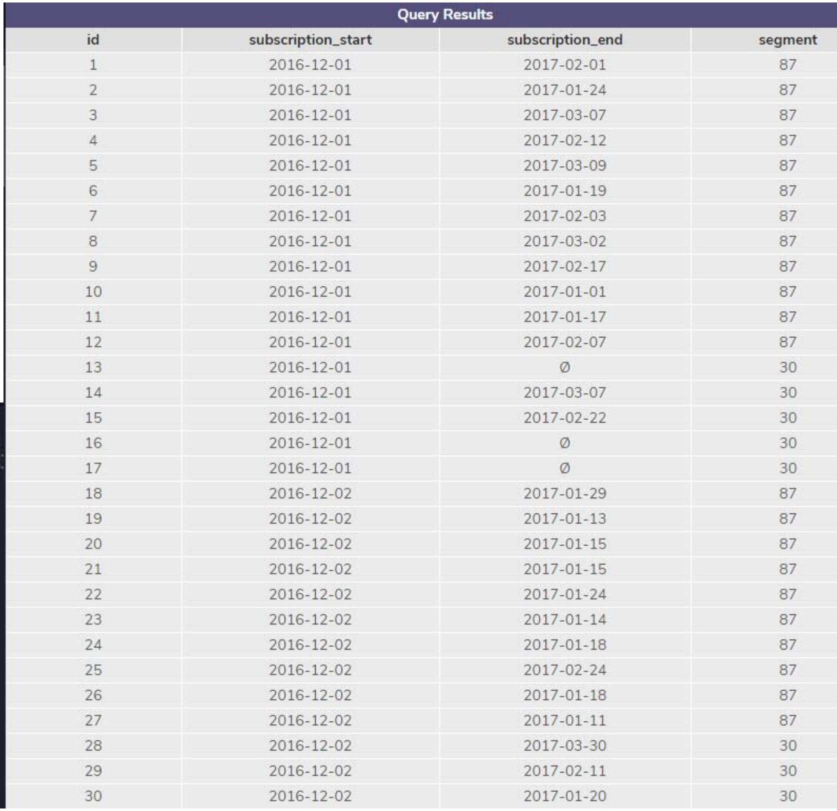
1.1 Get familiar with the company data

Take a look at the first 100 rows of data in the subscriptions table. How many different segments do you see?

- At first lets look at the whole data in the table.



```
--1. Take a look at the first 100 rows of data in the subscr


select *
from subscriptions
limit 100;
```



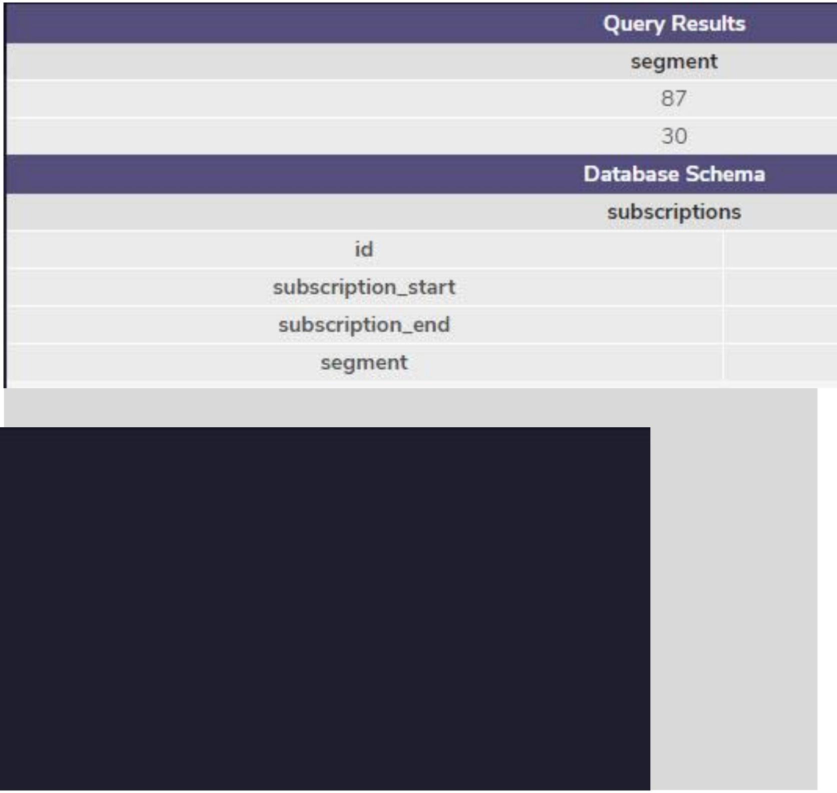
Query Results

id	subscription_start	subscription_end	segment
1	2016-12-01	2017-02-01	87
2	2016-12-01	2017-01-24	87
3	2016-12-01	2017-03-07	87
4	2016-12-01	2017-02-12	87
5	2016-12-01	2017-03-09	87
6	2016-12-01	2017-01-19	87
7	2016-12-01	2017-02-03	87
8	2016-12-01	2017-03-02	87
9	2016-12-01	2017-02-17	87
10	2016-12-01	2017-01-01	87
11	2016-12-01	2017-01-17	87
12	2016-12-01	2017-02-07	87
13	2016-12-01	Ø	30
14	2016-12-01	2017-03-07	30
15	2016-12-01	2017-02-22	30
16	2016-12-01	Ø	30
17	2016-12-01	Ø	30
18	2016-12-02	2017-01-29	87
19	2016-12-02	2017-01-13	87
20	2016-12-02	2017-01-15	87
21	2016-12-02	2017-01-15	87
22	2016-12-02	2017-01-24	87
23	2016-12-02	2017-01-14	87
24	2016-12-02	2017-01-18	87
25	2016-12-02	2017-02-24	87
26	2016-12-02	2017-01-18	87
27	2016-12-02	2017-01-11	87
28	2016-12-02	2017-03-30	30
29	2016-12-02	2017-02-11	30
30	2016-12-02	2017-01-20	30

1.2 Get familiar with the company data

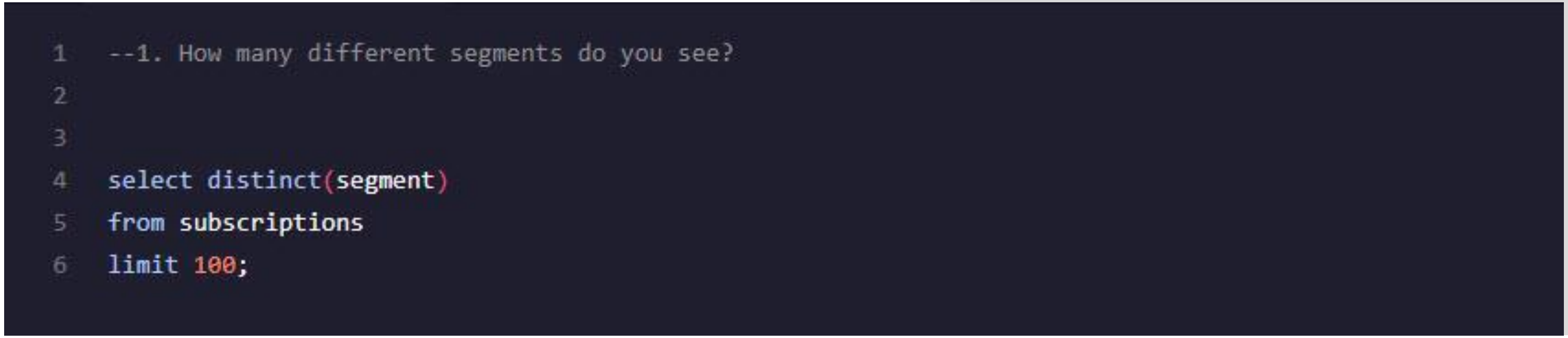
Take a look at the first 100 rows of data in the subscriptions table. How many different segments do you see?

- Once we know what columns table has, we can answer 2nd part of the question



Query Results
segment
87
30

Database Schema	
subscriptions	
id	
subscription_start	
subscription_end	
segment	

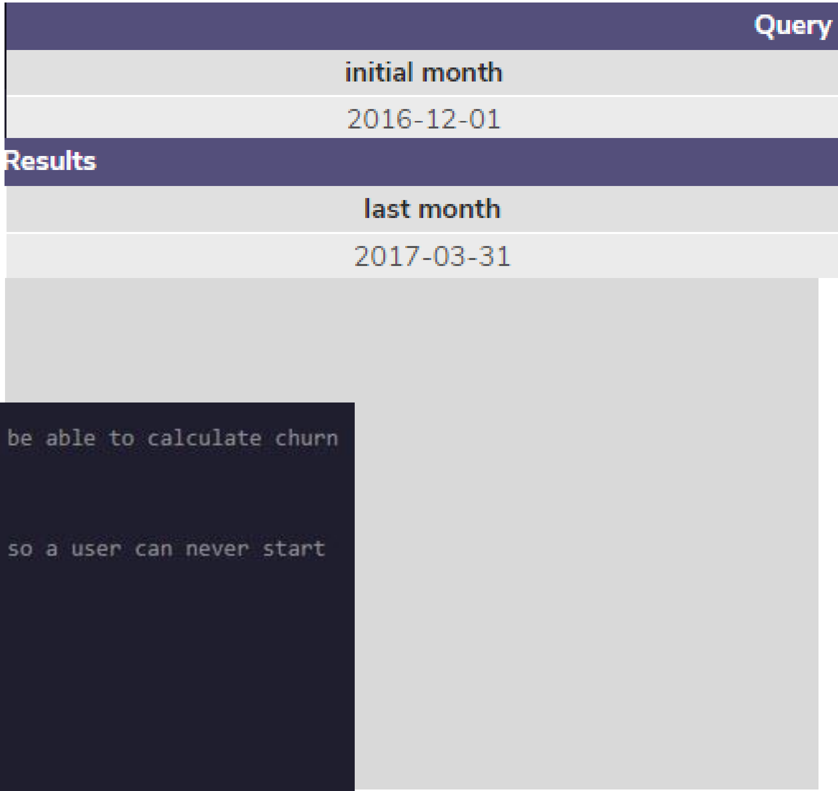


```
--1. How many different segments do you see?


select distinct(segment)
from subscriptions
limit 100;
```

1.3 Get familiar with the company data

Determine the range of months of data provided. Which months will you be able to calculate churn for?

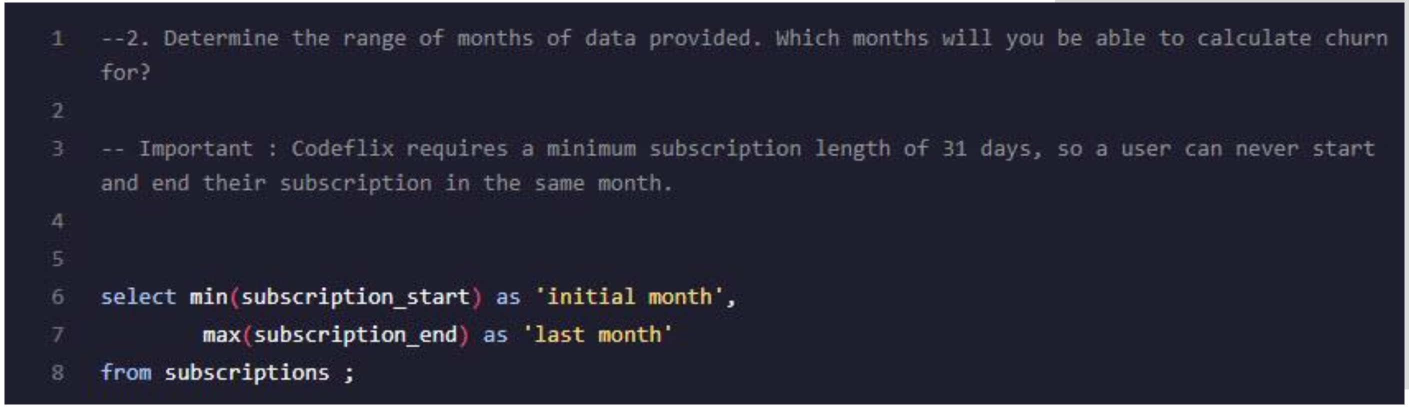
- If subscriptions started on DEC, but we can only calculate churn for JAN-MAR.



Query Results

initial month
2016-12-01

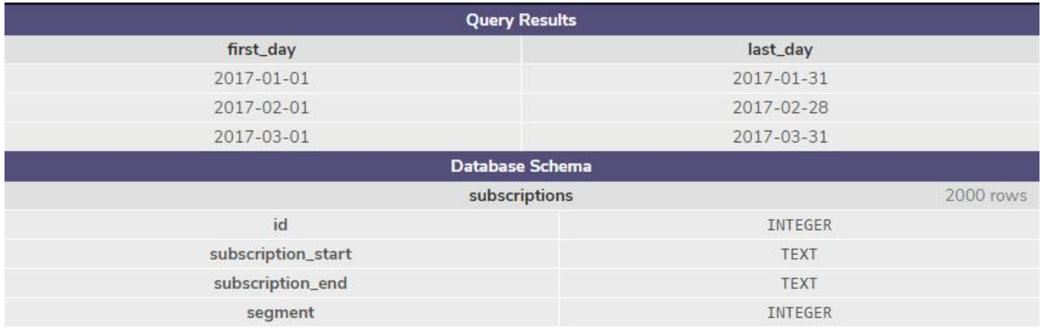
last month
2017-03-31



```sql
--2. Determine the range of months of data provided. Which months will you be able to calculate churn for?

-- Important : Codeflix requires a minimum subscription length of 31 days, so a user can never start and end their subscription in the same month.


select min(subscription_start) as 'initial month',
        max(subscription_end) as 'last month'
from subscriptions ;
```

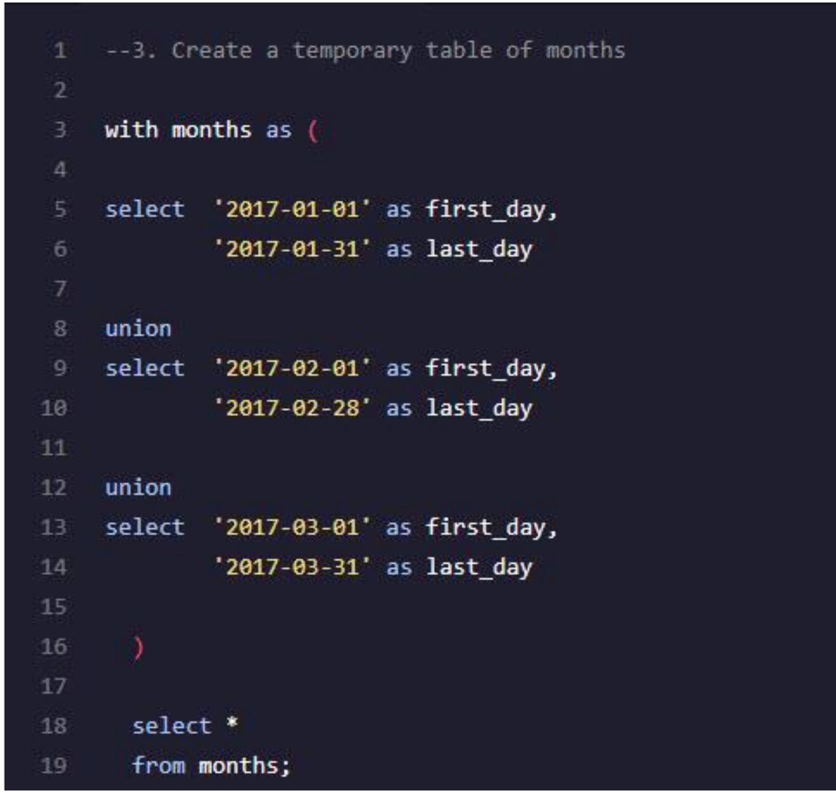
2.1 Getting to Codeflix Churn rate

Firstly create a temporary table of months



Query Results	
first_day	last_day
2017-01-01	2017-01-31
2017-02-01	2017-02-28
2017-03-01	2017-03-31

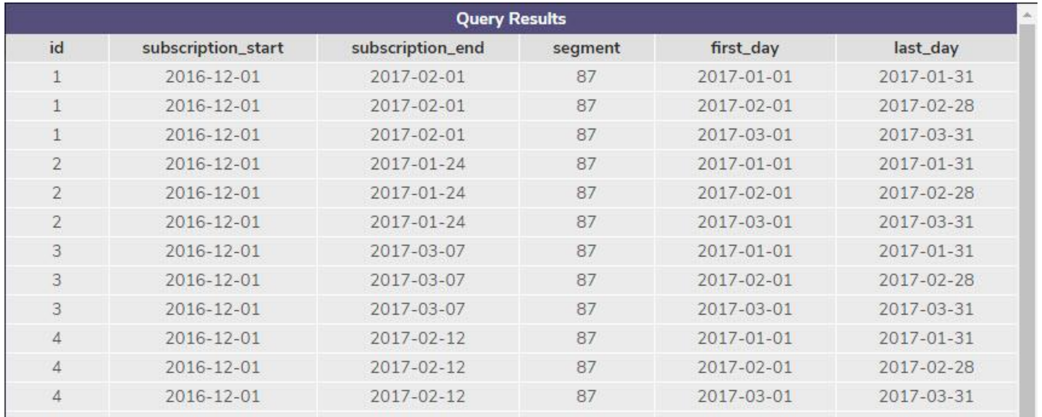
Database Schema	
subscriptions	2000 rows
id	INTEGER
subscription_start	TEXT
subscription_end	TEXT
segment	INTEGER



```
--3. Create a temporary table of months

with months as (

select  '2017-01-01' as first_day,
        '2017-01-31' as last_day

union
select  '2017-02-01' as first_day,
        '2017-02-28' as last_day

union
select  '2017-03-01' as first_day,
        '2017-03-31' as last_day


  )

  select *
  from months;
```

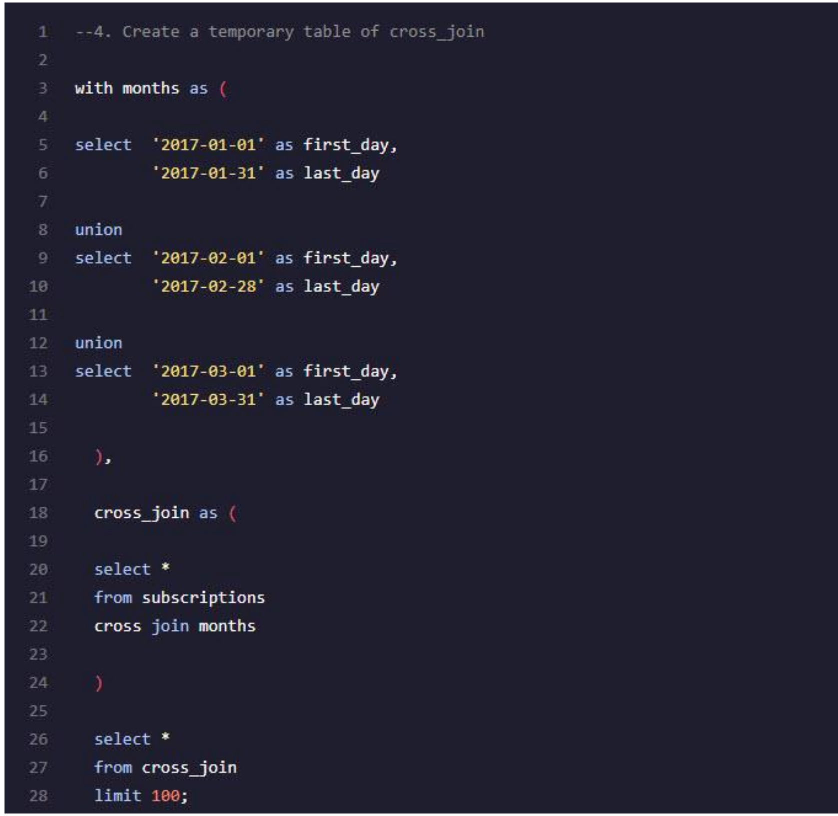
2.2 Getting to Codeflix Churn rate

Secondly create a temporary table of Cross_join



Query Results

id	subscription_start	subscription_end	segment	first_day	last_day
1	2016-12-01	2017-02-01	87	2017-01-01	2017-01-31
1	2016-12-01	2017-02-01	87	2017-02-01	2017-02-28
1	2016-12-01	2017-02-01	87	2017-03-01	2017-03-31
2	2016-12-01	2017-01-24	87	2017-01-01	2017-01-31
2	2016-12-01	2017-01-24	87	2017-02-01	2017-02-28
2	2016-12-01	2017-01-24	87	2017-03-01	2017-03-31
3	2016-12-01	2017-03-07	87	2017-01-01	2017-01-31
3	2016-12-01	2017-03-07	87	2017-02-01	2017-02-28
3	2016-12-01	2017-03-07	87	2017-03-01	2017-03-31
4	2016-12-01	2017-02-12	87	2017-01-01	2017-01-31
4	2016-12-01	2017-02-12	87	2017-02-01	2017-02-28
4	2016-12-01	2017-02-12	87	2017-03-01	2017-03-31

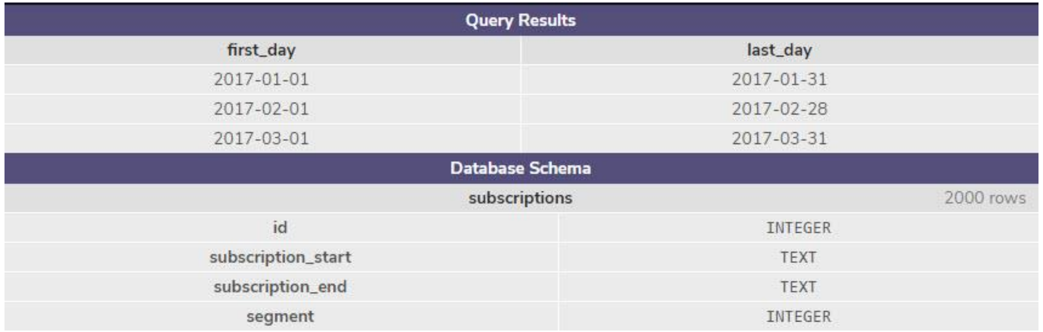


```sql
--4. Create a temporary table of cross_join

with months as (

select  '2017-01-01' as first_day,
        '2017-01-31' as last_day

union
select  '2017-02-01' as first_day,
        '2017-02-28' as last_day

union
select  '2017-03-01' as first_day,
        '2017-03-31' as last_day

  ),

  cross_join as (

  select *
  from subscriptions
  cross join months

  )

  select *
  from cross_join
  limit 100;
```

2.3 Getting to Codeflix Churn rate

Then create a temporary table, status, from the cross_join table

Here we need to add argument for distinct segment value



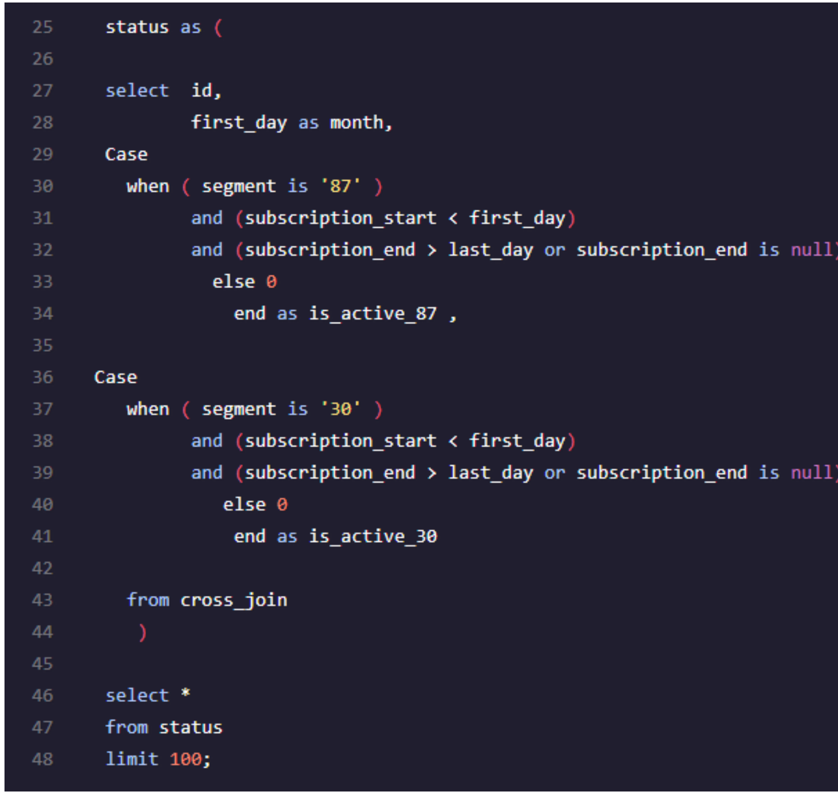
Query Results

first_day	last_day
2017-01-01	2017-01-31
2017-02-01	2017-02-28
2017-03-01	2017-03-31

Database Schema

subscriptions 2000 rows

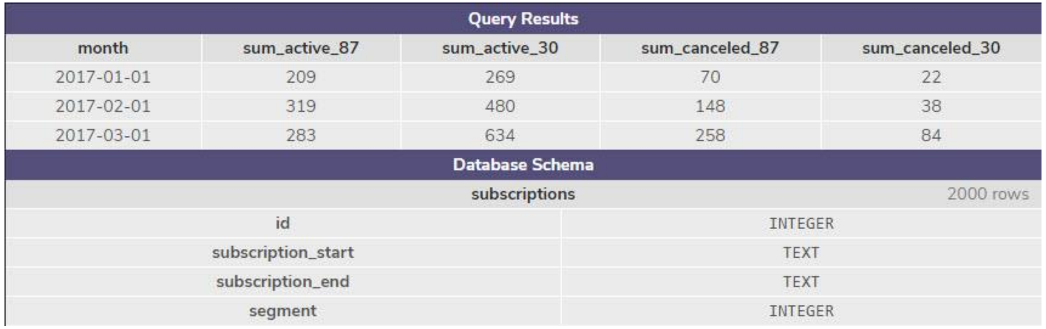
id	INTEGER
subscription_start	TEXT
subscription_end	TEXT
segment	INTEGER



```
   status as (

   select  id,
           first_day as month,
   Case
     when ( segment is '87' )
           and (subscription_start < first_day)
           and (subscription_end > last_day or subscription_end is null
             else 0
               end as is_active_87 ,

  Case
     when ( segment is '30' )
           and (subscription_start < first_day)
           and (subscription_end > last_day or subscription_end is null
              else 0
               end as is_active_30


     from cross_join
      )

   select *
   from status
   limit 100;
```

2.4 Getting to Codeflix Churn rate

Create a status_aggregate temporary table that is a SUM of the active and canceled subscriptions for each segment, for each month.



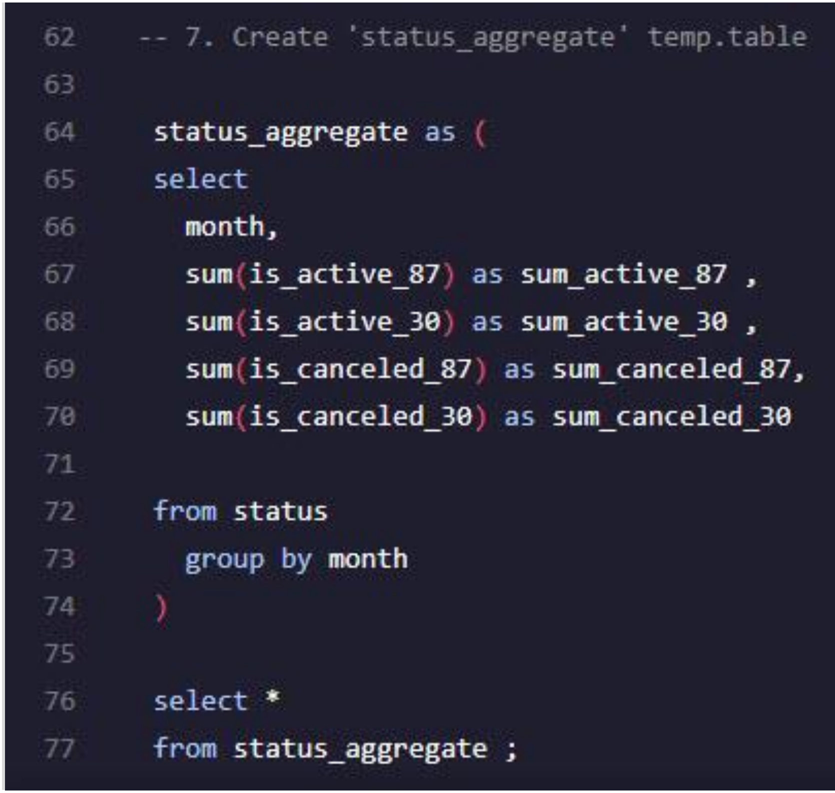
Query Results

month	sum_active_87	sum_active_30	sum_canceled_87	sum_canceled_30
2017-01-01	209	269	70	22
2017-02-01	319	480	148	38
2017-03-01	283	634	258	84

Database Schema

subscriptions 2000 rows

id	INTEGER
subscription_start	TEXT
subscription_end	TEXT
segment	INTEGER

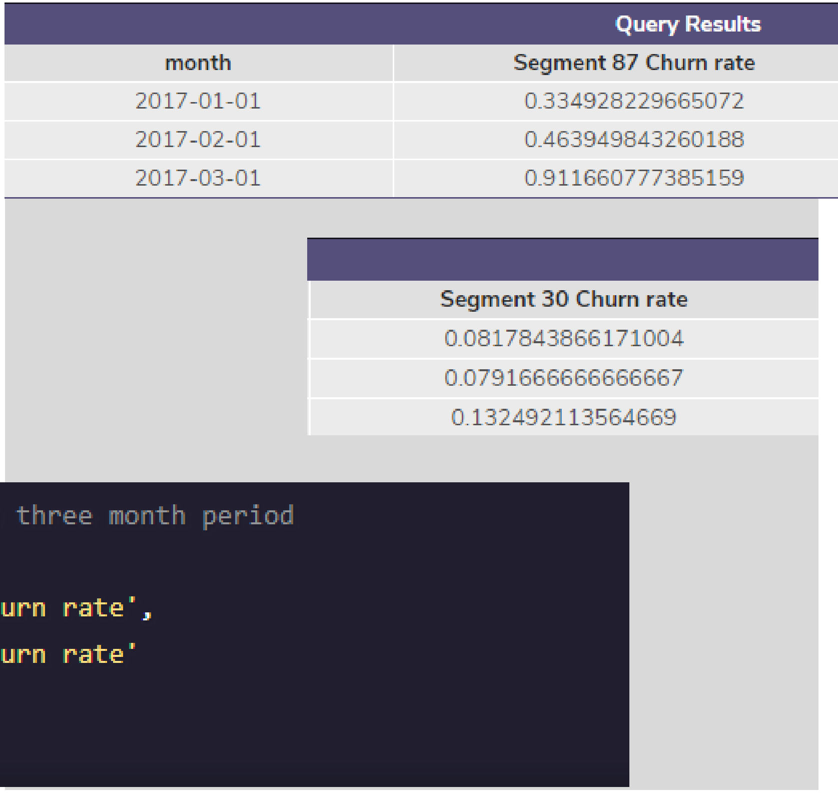


```
 -- 7. Create 'status_aggregate' temp.table

status_aggregate as (
select
  month,
  sum(is_active_87) as sum_active_87 ,
  sum(is_active_30) as sum_active_30 ,
  sum(is_canceled_87) as sum_canceled_87,
  sum(is_canceled_30) as sum_canceled_30

from status
  group by month
)

select *
from status_aggregate ;
```

3.1 Compare the churn rates between user segments

Let’s see the churn rates by Segment !

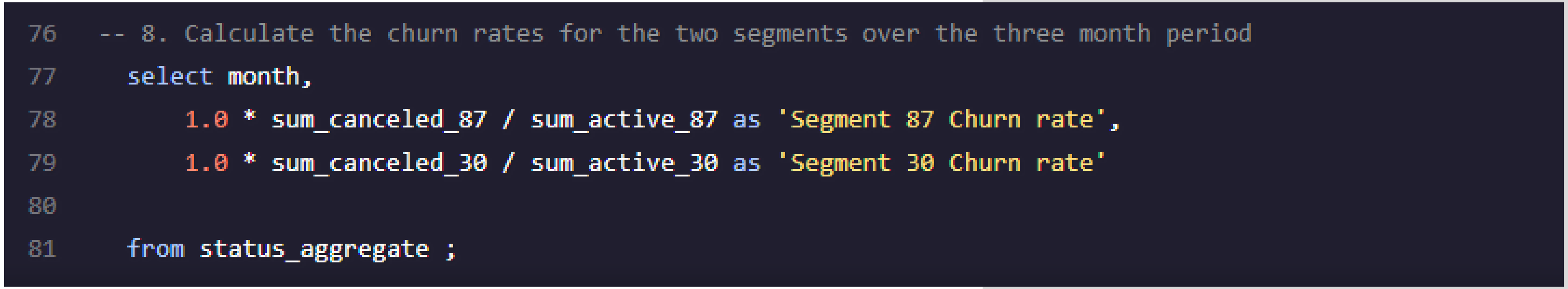
We can distinguish that Segment 87 lost almost all subscribers in 3 months.



Query Results

month	Segment 87 Churn rate
2017-01-01	0.334928229665072
2017-02-01	0.463949843260188
2017-03-01	0.911660777385159

Segment 30 Churn rate
0.0817843866171004
0.0791666666666667
0.132492113564669

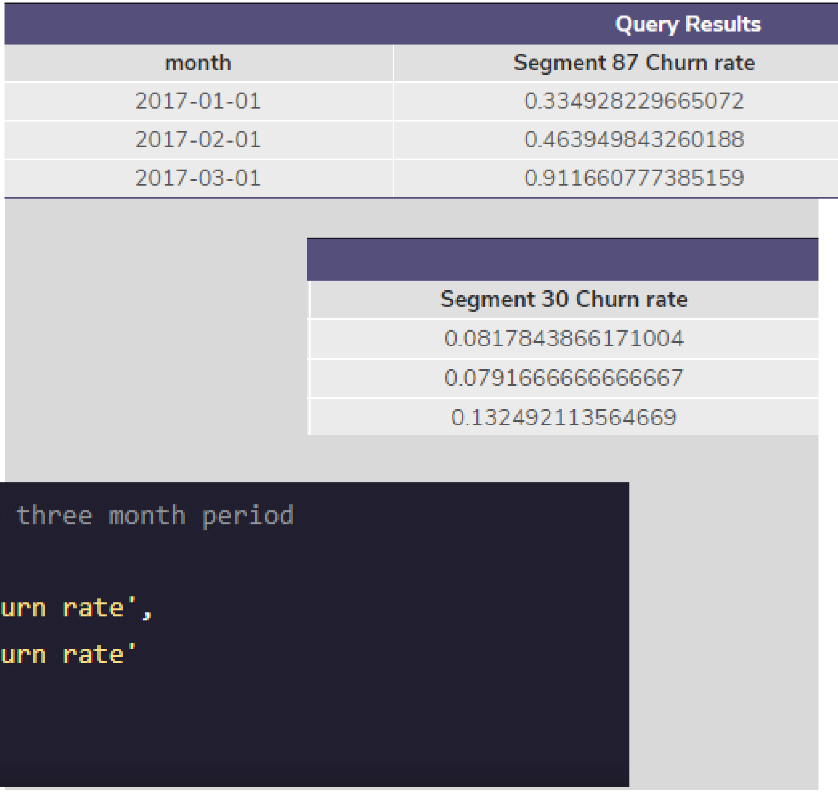


```
-- 8. Calculate the churn rates for the two segments over the three month period
  select month,
      1.0 * sum_canceled_87 / sum_active_87 as 'Segment 87 Churn rate',
      1.0 * sum_canceled_30 / sum_active_30 as 'Segment 30 Churn rate'

  from status_aggregate ;
```

3.2 Compare the churn rates between user segments

Which segment of users should the company focus on expanding?

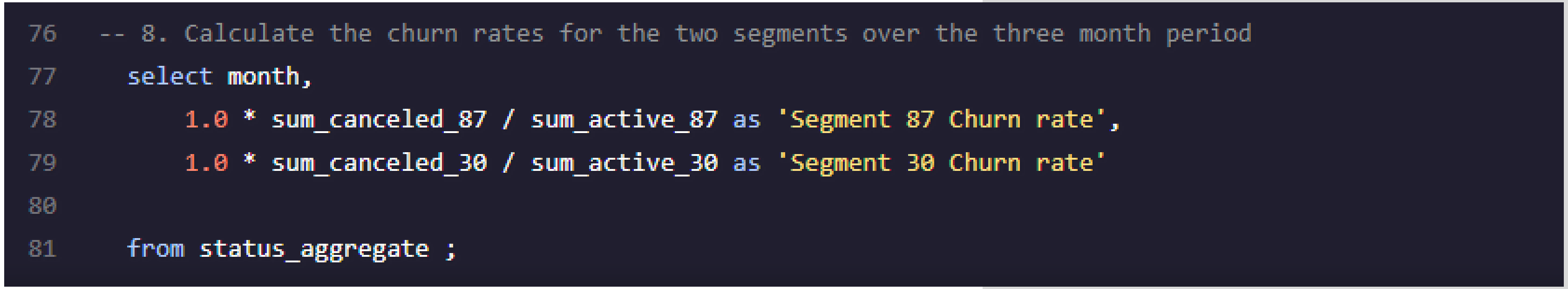
Obviously Segment ’30’ where Churn is just 8-13% compared to 33-92%



Query Results

month	Segment 87 Churn rate
2017-01-01	0.334928229665072
2017-02-01	0.463949843260188
2017-03-01	0.911660777385159

Segment 30 Churn rate
0.0817843866171004
0.0791666666666667
0.132492113564669



```sql
-- 8. Calculate the churn rates for the two segments over the three month period
  select month,
      1.0 * sum_canceled_87 / sum_active_87 as 'Segment 87 Churn rate',
      1.0 * sum_canceled_30 / sum_active_30 as 'Segment 30 Churn rate'

  from status_aggregate ;
```