codecademy

Calculating Churn Rates

Analyze Data with SQL
Àngels Albiol Masip
15/12/2021

Table of Contents

1. GET FAMILIAR WITH THE DATA
2. CALCULATE CHURN RATE FOR EACH SEGMENT
3. BONUS

1. GET FAMILIAR WITH THE DATA

1. Get familiar with the data

1. “How many different segments do you see”
We can see two different segments, 30 and 87

2. “Which months will you be able to calculate churn for”
We will be able to calculate churn for January, February and March

```
SELECT DISTINCT segment
FROM subscriptions;
```

```
SELECT MIN(subscription_end),
  MAX(subscription_end)
FROM subscriptions;
```

segment
87
30

MIN(subscription_end)	MAX(subscription_end)
2017-01-01	2017-03-31

2. CALCULATE CHURN RATE FOR EACH SEGMENT

2. Calculte churn rate for each segment

- **Which segment has a lower churn rate?**
  Segment 30 has a lower churn rate

month	churn rate 87	churn rate 30
2017-01-01	25.18	7.56
2017-02-01	32.03	7.34
2017-03-01	48.59	11.73

```
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
),
status AS (
  SELECT id,
  first_day AS month,
  CASE
    WHEN (segment = 87)
      AND (subscription_start < first_day)
      AND (subscription_end > first_day
        OR subscription_end IS NULL) THEN 1
    ELSE 0
  END AS is_active_87,
  CASE
    WHEN (segment = 30)
      AND (subscription_start < first_day)
      AND (subscription_end > first_day
        OR subscription_end IS NULL) THEN 1
    ELSE 0
  END AS is_active_30,
  CASE
    WHEN (subscription_end BETWEEN first_day AND last_day)

      AND (segment = 87) THEN 1
    ELSE 0
  END AS is_canceled_87,
```

```
CASE
    WHEN (subscription_end BETWEEN
first_day AND last_day)
      AND (segment = 30) THEN 1
    ELSE 0
  END AS is_canceled_30
FROM cross_join
),
status_aggregate AS (
  SELECT month,

         SUM(is_active_87) AS
active_87,
    SUM(is_active_30) AS active_30,
    SUM(is_canceled_87) AS
canceled_87,
    SUM(is_canceled_30) AS
canceled_30
  FROM status
  GROUP BY month
)
SELECT month,
  ROUND(100.0* (canceled_87) /
(active_87), 2) AS 'churn rate 87',

     ROUND(100.0* (canceled_30) /
(active_30), 2) AS 'churn rate 30'
FROM status_aggregate;
```

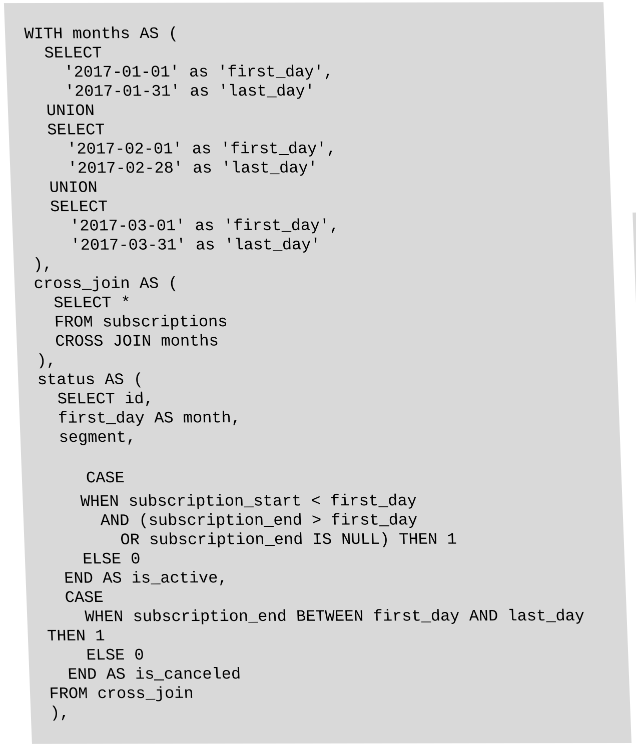
3. BONUS

3. BONUS

- **How would you modify this code to support a large number of segments?**

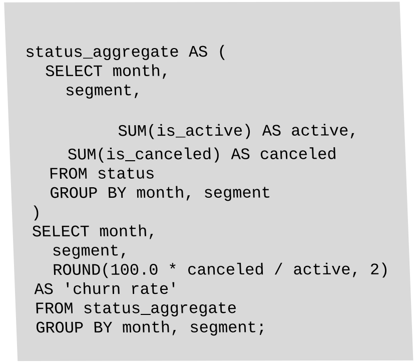
  I would group by segment and erase the case where we check the segment one by one

month	segment	churn rate
2017-01-01	30	7.56
2017-01-01	87	25.18
2017-02-01	30	7.34
2017-02-01	87	32.03
2017-03-01	30	11.73
2017-03-01	87	48.59



```
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
),
status AS (
  SELECT id,
  first_day AS month,
  segment,

     CASE
    WHEN subscription_start < first_day
      AND (subscription_end > first_day
        OR subscription_end IS NULL) THEN 1
    ELSE 0
  END AS is_active,
  CASE
    WHEN subscription_end BETWEEN first_day AND last_day
THEN 1
    ELSE 0
  END AS is_canceled
FROM cross_join
),
```



```
status_aggregate AS (
  SELECT month,
    segment,

         SUM(is_active) AS active,
    SUM(is_canceled) AS canceled
  FROM status
  GROUP BY month, segment
)
SELECT month,
  segment,
  ROUND(100.0 * canceled / active, 2)
AS 'churn rate'
FROM status_aggregate
GROUP BY month, segment;
```