Calculating churn rates

Learn SQL from Scratch

By Kasper Lindhorst

Table of contents

1. Get familiar with the company; Codeflix

    1.1 How many months has the company been operating?

    1.2 Which months do you have enough information to calculate a churn rate?

    1.3 What segments of users exist

2. What is the overall churn rate by month?
3. Compare the churn rates between segments.

    3.1 Which segment of users should be the company focus on expanding?

Codeflix, who are they?

1. Get familiar with the company; Codeflix

1.1 How many months has the company been operating?

They company has been operating for 4 months. Query 1 shows the code how to get this information from the table. It gets the first month when a subscription got active and the last month when a subscription ended. In total there were 4 unique months (12, 01, 02, and 03).

1.2 Which months do you have enough information to calculate a churn rate?

Only the first three months of 2017 have enough information to calculate a churn rate. December can't be used, because the *subscription_end* column contains NULL values for the December month of 2016.

1.3 What segments of users exist?

Used a one time query for this (query 2). User segment 30 and 87 exist.

id	subscription_start	subscription_end	segment
Integer	Text	Text	Integer

```
--Query 1
SELECT MIN(subscription_start), MAX(subscription_end)
FROM subscriptions;

--Query 2
SELECT DISTINCT(segment)
FROM subscriptions;
```

The queries, calculations and other

2. What is the overall churn rate by month? [1/3]

To calculate the churn rate by month (and by segment) I needed first to know what the first and last day per month would be. Therefore, I wrote a query that gets the first day and last day per month. This is then cross joined with the subscriptions table. After that I wrote a query that selects the ID, the first day (alias ‘month) and the segment, followed by case statements.

The first case statement checks if a subscription is active by checking if the subscription start date is before the first day of the month and if the planned subscription end date is after the first day of the month. The subscription is also active when the subscription end date has a value of NULL, which means the subscription has no planned end date. When these conditions are true *is_active* get the value 1 meaning it’s active. Otherwise it’s 0.

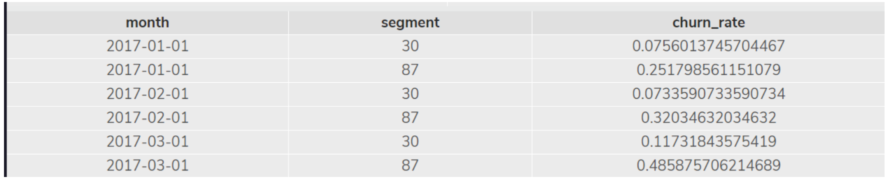
The second case statements checks if a subscription got canceled. The case statement checks if the subscpription end date is between the first date and last date of the month. When this is true *is_canceled* gets the value 1, meaning that the subscription got canceled by the user. Otherwise it’s 0.

```
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 subscriptions.*, months.*
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 sum_active,
    SUM(is_canceled) AS sum_canceled
    FROM status
    GROUP BY month, segment
  )
  SELECT month, segment,
  1.0 * sum_canceled / sum_active AS churn_rate
  FROM status_aggregate;
```

2. What is the overall churn rate by month? [2/3]

These selections and case statements are retrieved from the cross join table. After this a temporary table is created (status_aggregate) which selects the month, segement and the sum of the *is_active* and *is_canceled* columns. These are then grouped by month and segment, so that later on we can calculate the churn rate by these groups.

Finally, the churn rate is calculated by selecting the month, the segment and the calculation for the churn rate. The churn rate is calculated by dividing the sum of the total cancelations with the sum of the total active subscriptions. This is then multiplied by 1.0 to make a float. This gives the following result.



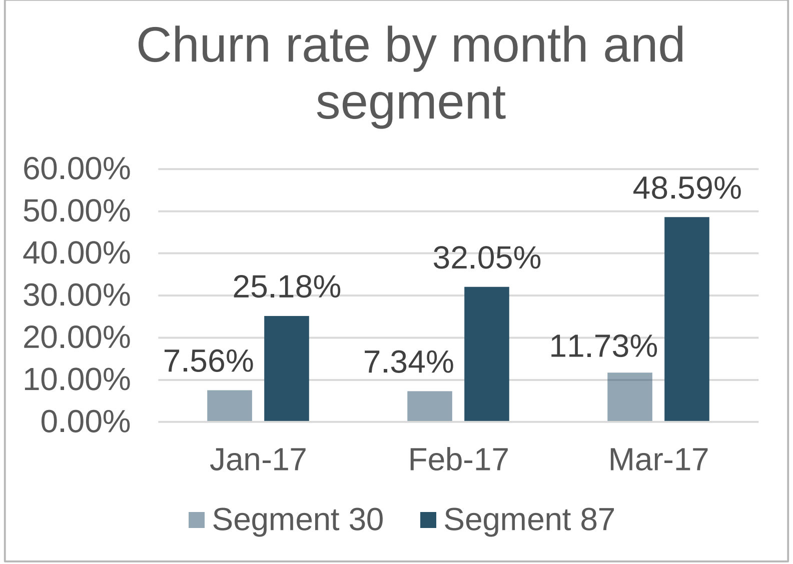
month	segment	churn_rate
2017-01-01	30	0.0756013745704467
2017-01-01	87	0.251798561151079
2017-02-01	30	0.0733590733590734
2017-02-01	87	0.32034632034632
2017-03-01	30	0.11731843575419
2017-03-01	87	0.485875706214689

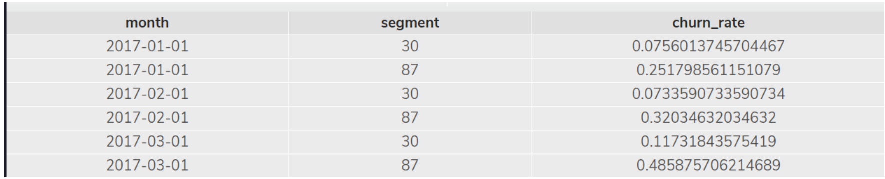
```
WITH months AS
(SELECT
  '2017-01-01' AS first_day,
  '2017-01-31' AS last_day
UNION
SELECT
  '2017-02-01' AS first_day,
  '2017-02-28' AS last_day
UNION
SELECT
  '2017-03-01' AS first_day,
  '2017-03-31' AS last_day
),
cross_join AS
(SELECT subscriptions.*, months.*
FROM subscriptions
CROSS JOIN 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 sum_active,
    SUM(is_canceled) AS sum_canceled
    FROM status
    GROUP BY month, segment
  )
  SELECT month, segment,
  1.0 * sum_canceled / sum_active AS churn_rate
  FROM status_aggregate;
```

2. What is the overall churn rate by month? [3/3]

These selections and case statements are retrieved from the cross join table. After this a temporary table is created (status_aggregate) which selects the month, segement and the sum of the *is_active* and *is_canceled* columns. These are then grouped by month and segment, so that later on we can calculate the churn rate by these groups.

Finally, the churn rate is calculated by selecting the month, the segment and the calculation for the churn rate. The churn rate is calculated by dividing the sum of the total cancelations with the sum of the total active subscriptions. This is then multiplied by 1.0 to make a float. This gives the following result.





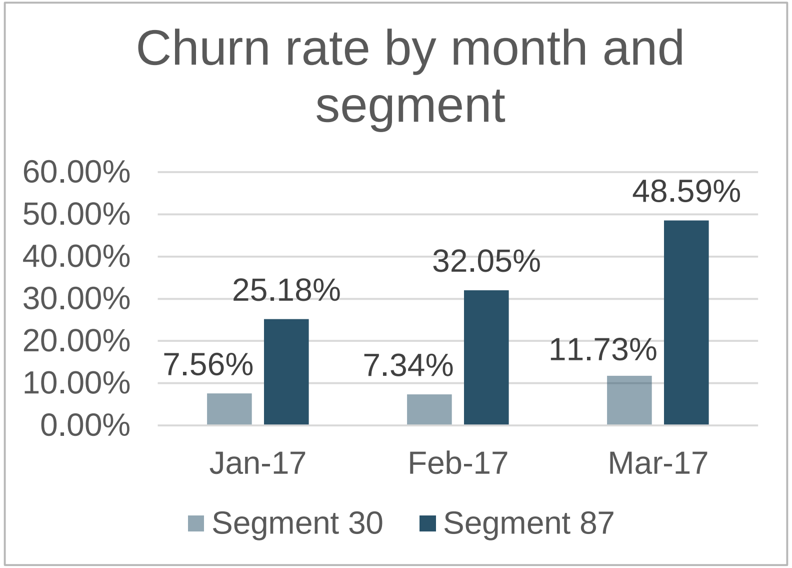
month	segment	churn_rate
2017-01-01	30	0.0756013745704467
2017-01-01	87	0.251798561151079
2017-02-01	30	0.0733590733590734
2017-02-01	87	0.32034632034632
2017-03-01	30	0.11731843575419
2017-03-01	87	0.485875706214689

The churn rate analysis

3. Compare the churn rates between segments.

Comparing the two segments and their churn rates in the first three months of 2017, we can conclude that segment 30 has a steady churn rate that almost stays the same with a slight increase in March 2017. This is in trend with the general assumption for a subscription based model such as Codeflix uses. Every month subscribers will come and go, and thus keeping an eye if this trend line stays steady and doesn't keep increasing, Codeflix has no need to worry about segment 30.

Segment 87 is something Codeflix should worry about. First of all, these churn rates are higher than those of segment 30. Furthermore there is a strong increase month by month in the churn rate of segment 87 going up to 48.59% in March 2017, meaning that almost half of the subscribers of segment 87 canceled their subscription. That's a lot and something Codeflix should focus on. Not just trying to keep the subscribers, but also understanding why they canceled.


Churn rate by month and segment
60.00%
50.00%
40.00%
30.00%
20.00%
10.00%
0.00%
7.56%
25.18%
7.34%
32.05%
11.73%
48.59%
Jan-17
Feb-17
Mar-17
Segment 30
Segment 87