Churn Rate for Codeflix

Leanne Harney
Learn SQL from Scratch
Group of May-8th

Table of Contents

Subscriptions Database

The database is comprised of 4 information sources:

- id = the unique identifier of the user in INT format
- subscription_start = the date in which the user subscribed in TEXT format
- subscription_end = the date in which the user unsubscribed in TEXT format
- segment = in INT format

```
SELECT *
FROM subscriptions
LIMIT 10;
```

Database Schema

subscriptions	2000 rows
id	INTEGER
subscription_start	TEXT
subscription_end	TEXT
segment	INTEGER

(n.b. the total number of clients is 2,000 as stated next to the chart name)

First and Latest Subscriptions

Using the formula:

```
SELECT
  MIN(subscription_start) AS 'first_subscription_date',
  MAX(subscription_start) AS 'latest_subscription_date'
FROM subscriptions;
```

we are able to see the date of the first subscriber, and the date of the latest subscriber:

Query Results	
first_subscription_date	latest_subscription_date
2016-12-01	2017-03-30

In the above query results we can see that Codeflix has been in operation since December 2016.

We will be able to calculate Churn as of January 1st 2017. We cannot calculate Churn for the first month of operations as you must subscribe for 1 month at a time.

Churn Rate by Month

We can calculate Churn for month where:

- Month 01 = January 2017, Churn Rate = 16%
- Month 02 = February 2017, Churn Rate = 19%
- Month 03 = March 2017, Churn Rate = 27%

Query Results	
month_number	churn
01	0.161687170474517
02	0.189795918367347
03	0.274258219727346

The Churn Rate is increasing with every month in business.

Segments

Using the following query we can identify that there are 2 unique segments in our database of 2,000 users.

```
SELECT DISTINCT(segment) AS 'unique_segments'
FROM subscriptions;
```

Query Results
unique_segments
87
30

and that both segment 87 and 30 have had 1,000 unique users

```
SELECT COUNT(DISTINCT(id)) AS 'unique_users_87'
FROM subscriptions
WHERE segment = 87;
```

Query Results
unique_users_87
1000

```
SELECT COUNT(DISTINCT(id)) AS 'unique_users_30'
FROM subscriptions
WHERE segment = 30;
```

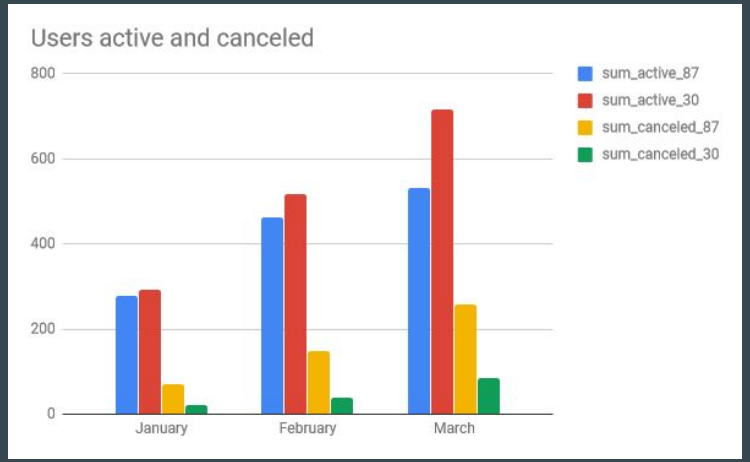
Query Results
unique_users_30
1000

Active and Canceled by Month per Segment (1 of 2)

By creating temporary tables we are able to see the number of users who are active and have canceled per month, per segment:

Results =

Query Results				
month	sum_active_87	sum_active_30	sum_canceled_87	sum_canceled_30
01	278	291	70	22
02	462	518	148	38
03	531	716	258	84



Code on next page...

Active and Canceled by Month per Segment (2 of 2)

```
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 cross_join.id, strftime('%m', 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 = 87
  AND (subscription_end BETWEEN first_day AND last_day)
THEN 1
ELSE 0
END AS 'is_canceled_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 segment = 30
  AND (subscription_end BETWEEN first_day AND last_day)
THEN 1
ELSE 0
END AS 'is_canceled_30'
FROM cross_join),

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;
```

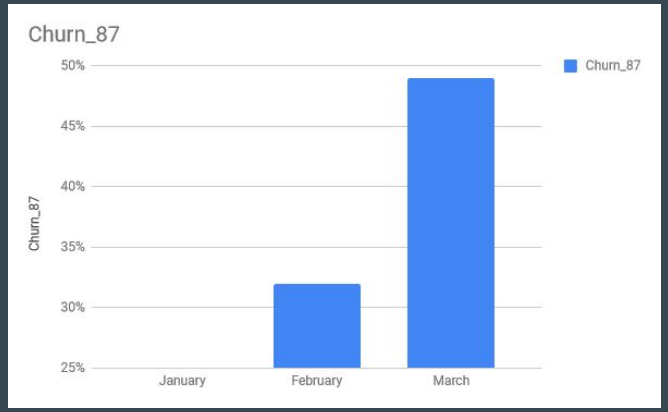
Churn Rate per Segment by Month

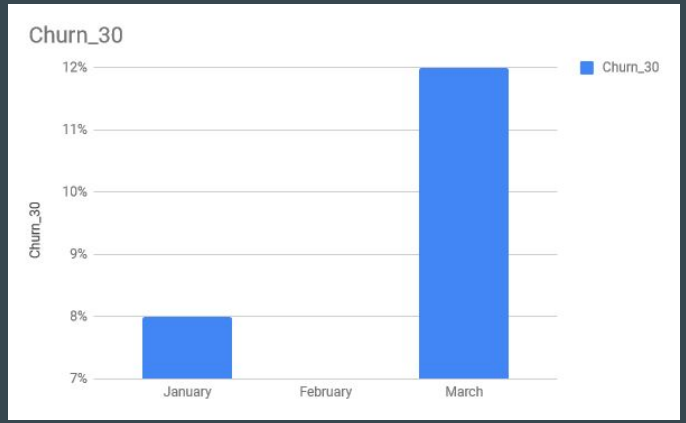
By breaking down the 2 different segments we can see the Churn Rate per month by segment: (add the below code at the end of the previous query)

```
SELECT month,
1.0* sum_canceled_87 / sum_active_87 AS 'chrun_87',
1.0* sum_canceled_30 / sum_active_30 AS 'churn_30'
FROM status_aggregate;
```

Query Results

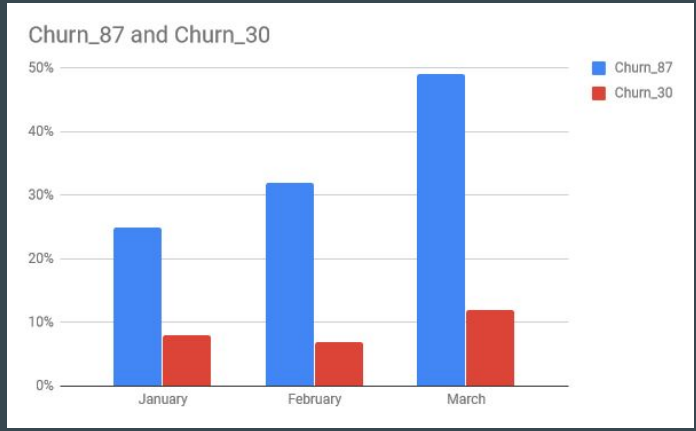
month	chrun_87	churn_30
01	0.251798561151079	0.0756013745704467
02	0.32034632034632	0.0733590733590734
03	0.485875706214689	0.11731843575419





Findings

Query Results		
month	churn_87	churn_30
01	0.251798561151079	0.0756013745704467
02	0.32034632034632	0.0733590733590734
03	0.485875706214689	0.11731843575419



Codeflix should put their focus into expanding segment 30 based on the Churn Rate as they are more loyal over time.

Segment 30 has an average Churn Rate of only 8% where segment 87 has an average Churn Rate of 35%.

It might be helpful to try to determine why segment 87 has such a high Churn Rate in hopes of formulating a strategy to reduce the amount of users lost, and to avoid similar reasons for loss in segment 30.