codecademy

Capstone: Churn Rates with Codeflix

Learn SQL from Scratch
By: Brandon Hart
9-24-2018

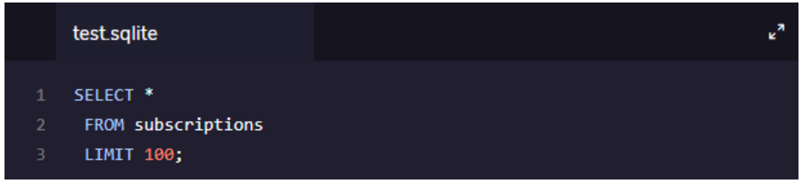
Table of Contents

1. Get Familiar with Codeflix

1.1

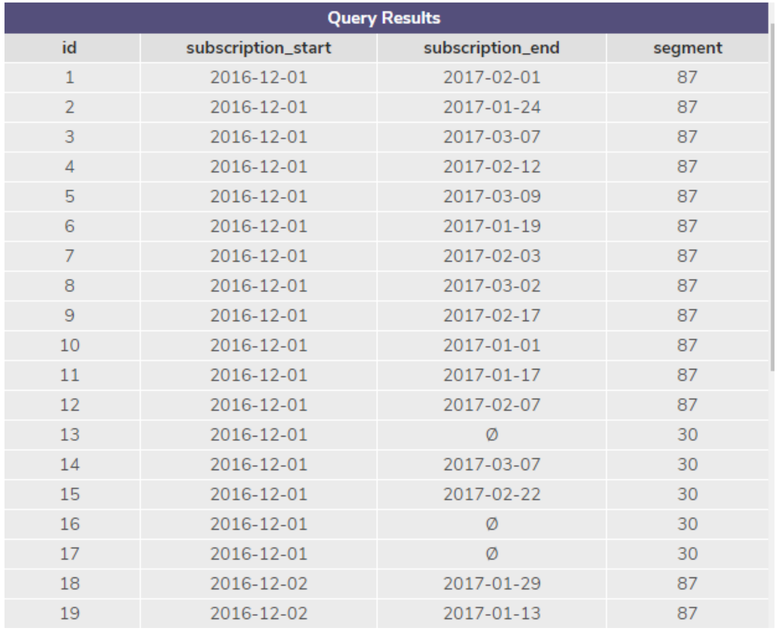
Take a look at the first 100 rows of data in the subscriptions table. How many different segments do you see?

- From the results in the query we have identified 2 segments.
- Segment #87, and Segment #30
- Additionally we have identified that Segment #87 and Segment #30 have 1000 fields of data each.



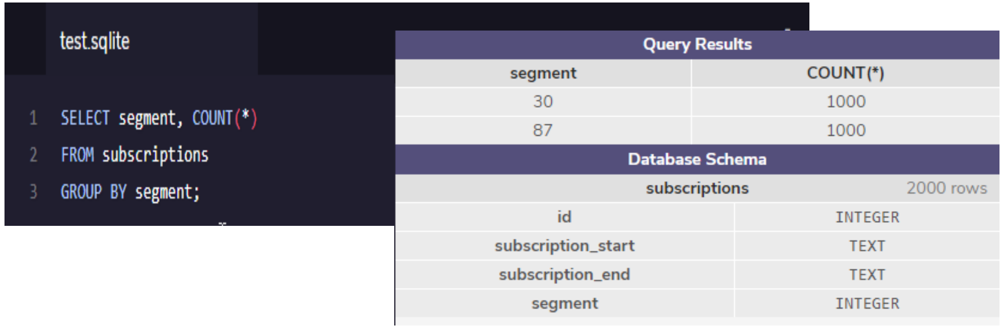
test.sqlite

```
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



test.sqlite

```
SELECT segment, COUNT(*)
FROM subscriptions
GROUP BY segment;
```

Query Results

segment	COUNT(*)
30	1000
87	1000

Database Schema

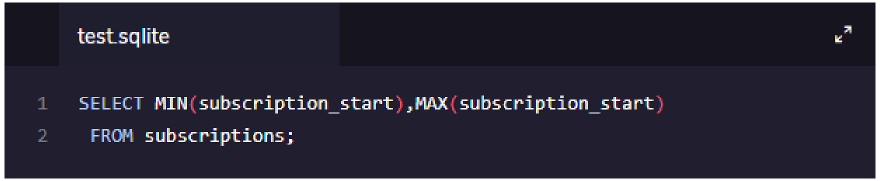
subscriptions 2000 rows

id	INTEGER
subscription_start	TEXT
subscription_end	TEXT
segment	INTEGER

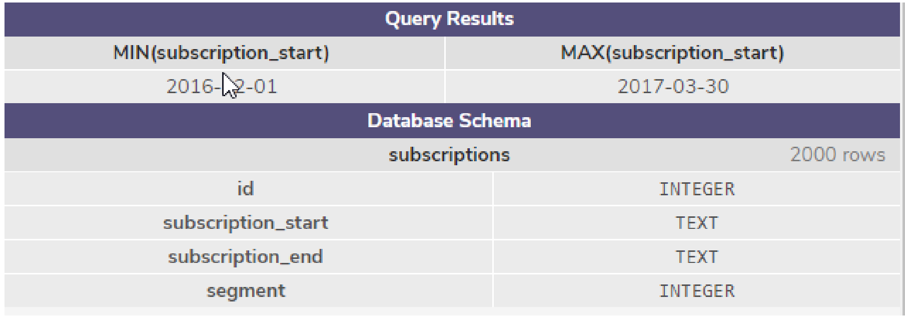
1.2

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

- The range starts from (2016-12-01 through 2017-03-30)
- Note* we can't calculate for December, since there are not subscription_end values yet.


test.sqlite

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


Query Results

MIN(subscription_start)	MAX(subscription_start)
2016-12-01	2017-03-30

Database Schema

subscriptions 2000 rows

id	INTEGER
subscription_start	TEXT
subscription_end	TEXT
segment	INTEGER

2. Calculate Churn Rates by Month

2.1

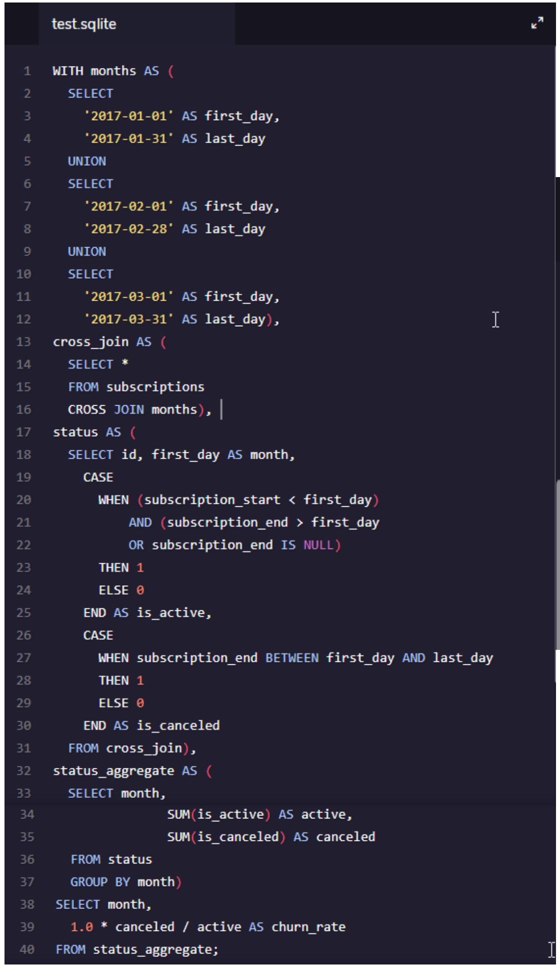
Churn Rates by month?

“**Churn rate** is the percent of subscribers that have canceled within a certain period, usually a month. For a user base to grow, the churn rate must be less than the new subscriber rate for the same period.”

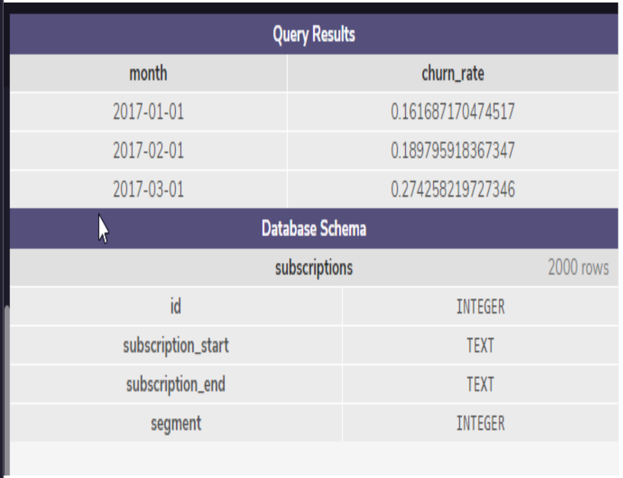
-Codecacademy

- January 2017 Churn Rate = 16.17%
- February 2017 Churn Rate = 18.99%
- March 2017 Churn Rate = 27.43%

The Churn rate is increase as time goes on which is a negative sign for Codeflix.


test.sqlite

```sql
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 (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,
               SUM(is_active) AS active,
               SUM(is_canceled) AS canceled
   FROM status
   GROUP BY month)
 SELECT month,
   1.0 * canceled / active AS churn_rate
 FROM status_aggregate;
```


Query Results

month	churn_rate
2017-01-01	0.161687170474517
2017-02-01	0.189795918367347
2017-03-01	0.274258219727346

Database Schema

subscriptions 2000 rows

id	INTEGER
subscription_start	TEXT
subscription_end	TEXT
segment	INTEGER

3. Compare Churn Rates Between Segments

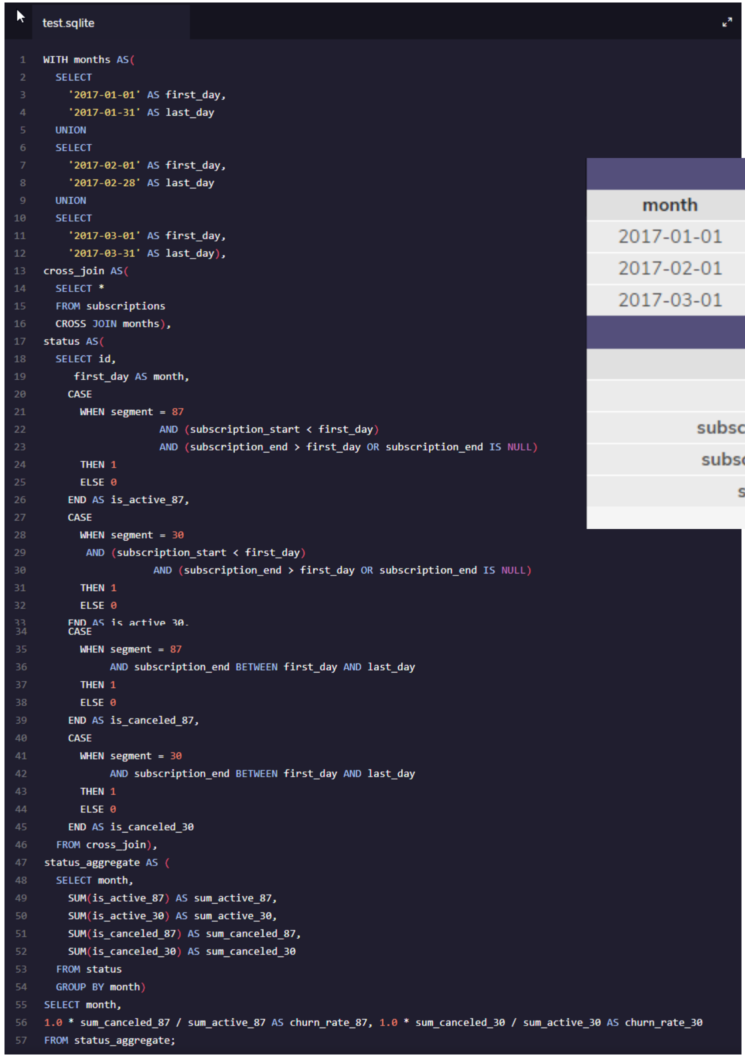
3.1

Compare Churn Rates between user segments

<u>January 2017 Churn Rates</u>
- Segment #87 = 25.18%
- Segment #30 =7.56%

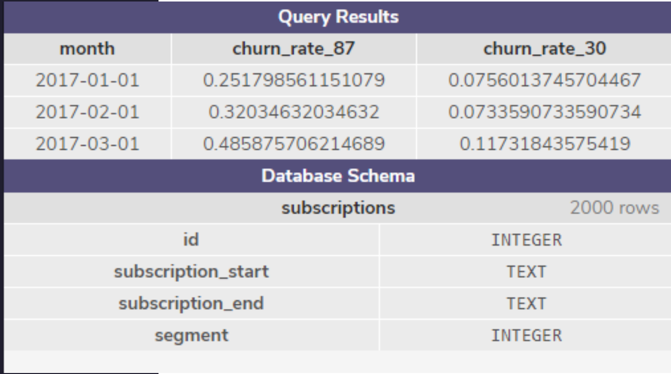
<u>February 2017 Churn Rates</u>
- Segment #87 = 32.03%
- Segment #30 = 7.34%

<u>March 2017 Churn Rates</u>
- Segment #87 = 48.59%
- Segment #30 = 11.73%



test.sqlite

```sql
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 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_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 month,
1.0 * sum_canceled_87 / sum_active_87 AS churn_rate_87, 1.0 * sum_canceled_30 / sum_active_30 AS churn_rate_30
FROM status_aggregate;
```



Query Results

month	churn_rate_87	churn_rate_30
2017-01-01	0.251798561151079	0.0756013745704467
2017-02-01	0.32034632034632	0.0733590733590734
2017-03-01	0.485875706214689	0.11731843575419

Database Schema

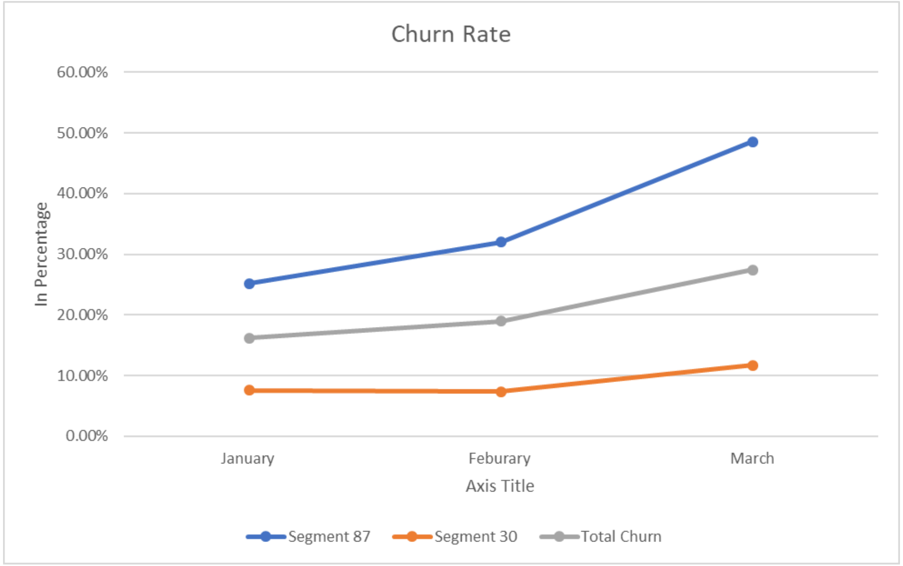
subscriptions	2000 rows
id	INTEGER
subscription_start	TEXT
subscription_end	TEXT
segment	INTEGER

4. Conclusion

4.1

Conclusion

- By graphing out the churn rates by segments over time we can identify that segment **#87** contributes the majority of lost users.
- Although the churn rates of segment #87 and #30 are increasing I would advice Codeflix to ask users in segment #87 why they left the service, and what additional features/services would be required for them to continue use Codeflix.
- According to the current tread, if not action is taken the total churn rate will continue to increase damaging the Codeflix user base and current financial situation.



	January	Feburary	March
Segment 87	25.18%	32.03%	48.59%
Segment 30	7.56%	7.34%	11.73%
Total Churn	16.17%	18.99%	27.43%