codecademy

Codeflix Churn

Learn SQL from Scratch

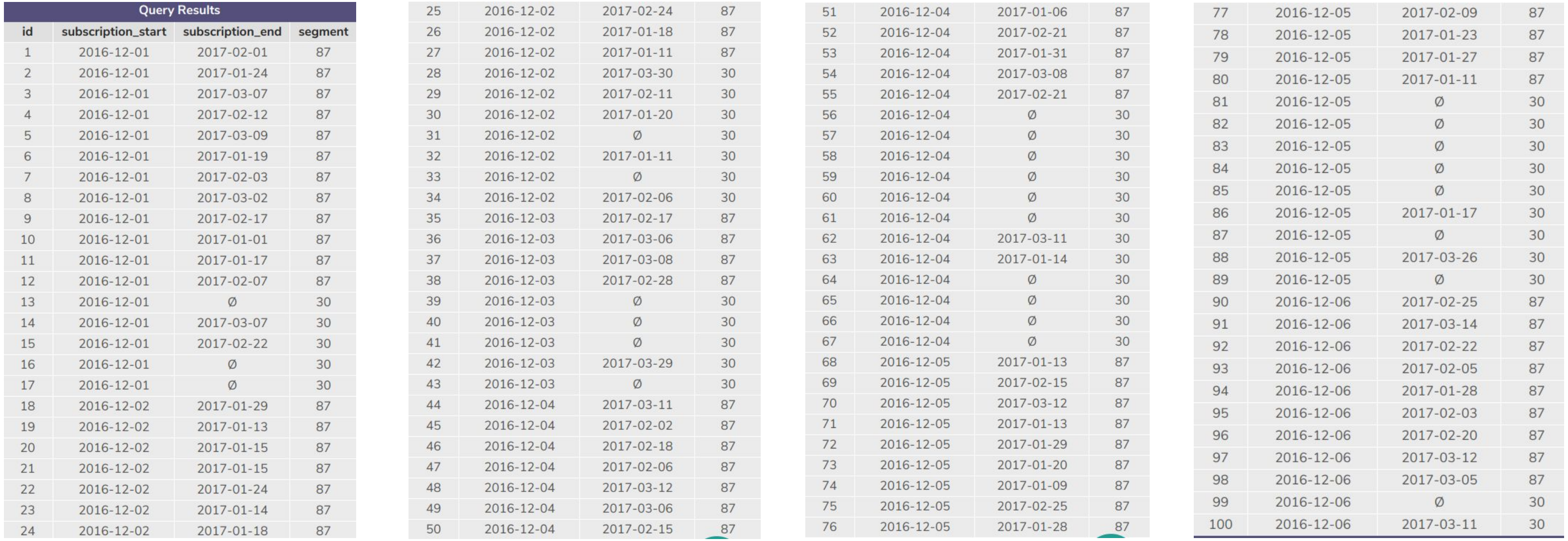
Example Table of Contents

1. Codeflix Introduction
2. A brief history of Codeflix
3. Beginning to organize the data
4. Codeflix churn trend
5. Visual churn

Getting Familiar with Codefilx ;)

1. Codeflix Introduction:

- Below is the first 100 rows of data from Codeflix Subscriptions table.
- There are two different segments.

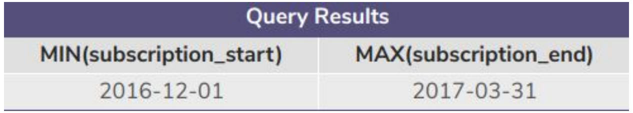


Query Results

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
31	2016-12-02	Ø	30
32	2016-12-02	2017-01-11	30
33	2016-12-02	Ø	30
34	2016-12-02	2017-02-06	30
35	2016-12-03	2017-02-17	87
36	2016-12-03	2017-03-06	87
37	2016-12-03	2017-03-08	87
38	2016-12-03	2017-02-28	87
39	2016-12-03	Ø	30
40	2016-12-03	Ø	30
41	2016-12-03	Ø	30
42	2016-12-03	2017-03-29	30
43	2016-12-03	Ø	30
44	2016-12-04	2017-03-11	87
45	2016-12-04	2017-02-02	87
46	2016-12-04	2017-02-18	87
47	2016-12-04	2017-02-06	87
48	2016-12-04	2017-03-12	87
49	2016-12-04	2017-03-06	87
50	2016-12-04	2017-02-15	87
51	2016-12-04	2017-01-06	87
52	2016-12-04	2017-02-21	87
53	2016-12-04	2017-01-31	87
54	2016-12-04	2017-03-08	87
55	2016-12-04	2017-02-21	87
56	2016-12-04	Ø	30
57	2016-12-04	Ø	30
58	2016-12-04	Ø	30
59	2016-12-04	Ø	30
60	2016-12-04	Ø	30
61	2016-12-04	Ø	30
62	2016-12-04	2017-03-11	30
63	2016-12-04	2017-01-14	30
64	2016-12-04	Ø	30
65	2016-12-04	Ø	30
66	2016-12-04	Ø	30
67	2016-12-04	Ø	30
68	2016-12-05	2017-01-13	87
69	2016-12-05	2017-02-15	87
70	2016-12-05	2017-03-12	87
71	2016-12-05	2017-01-13	87
72	2016-12-05	2017-01-29	87
73	2016-12-05	2017-01-20	87
74	2016-12-05	2017-01-09	87
75	2016-12-05	2017-02-25	87
76	2016-12-05	2017-01-28	87
77	2016-12-05	2017-02-09	87
78	2016-12-05	2017-01-23	87
79	2016-12-05	2017-01-27	87
80	2016-12-05	2017-01-11	87
81	2016-12-05	Ø	30
82	2016-12-05	Ø	30
83	2016-12-05	Ø	30
84	2016-12-05	Ø	30
85	2016-12-05	Ø	30
86	2016-12-05	2017-01-17	30
87	2016-12-05	Ø	30
88	2016-12-05	2017-03-26	30
89	2016-12-05	Ø	30
90	2016-12-06	2017-02-25	87
91	2016-12-06	2017-03-14	87
92	2016-12-06	2017-02-22	87
93	2016-12-06	2017-02-05	87
94	2016-12-06	2017-01-28	87
95	2016-12-06	2017-02-03	87
96	2016-12-06	2017-02-20	87
97	2016-12-06	2017-03-12	87
98	2016-12-06	2017-03-05	87
99	2016-12-06	Ø	30
100	2016-12-06	2017-03-11	30

2. A brief history of Codeflix

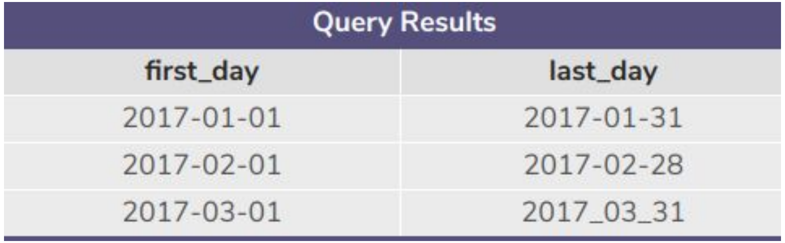
- Codeflix has been operating for 4 months:



Query Results

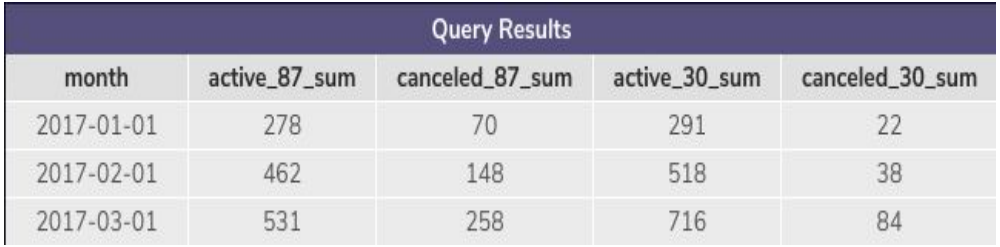
MIN(subscription_start)	MAX(subscription_end)
2016-12-01	2017-03-31

- But we can only calculate churn for 3 months. Below to the left is the Temporary Table I created for months. It is a list of our 3 churnable months.
- This is because Codeflix has a 31 day policy, so they don't count customers until they have finished their first month.
- Since they started the company in December, there are no customers who had been a customer for more than 31 days until January.
- Below and to the right the is a snapshot of how many customers were active and canceled, by month and segment. Codeflix has 2 segments. As you can see, I have only included months January, February and March, since these are the churnable 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

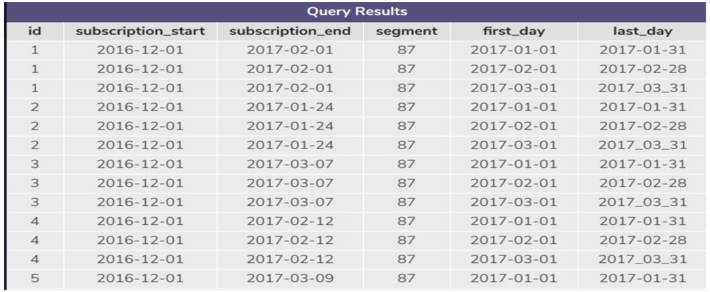


Query Results

month	active_87_sum	canceled_87_sum	active_30_sum	canceled_30_sum
2017-01-01	278	70	291	22
2017-02-01	462	148	518	38
2017-03-01	531	258	716	84

3. Beginning to organize the data:

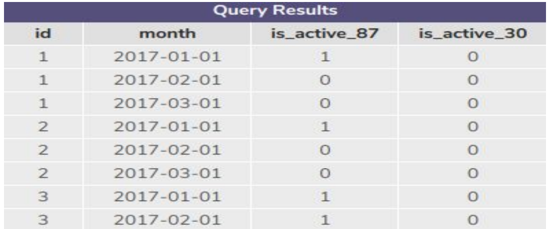
- To the right is every column from the temporary table I created, Cross_join. The problem with this table is it shows duplicate data, with the subscription_start, subscription_end and the first_day and last_day. We don't need both of these.



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
5	2016-12-01	2017-03-09	87	2017-01-01	2017-01-31

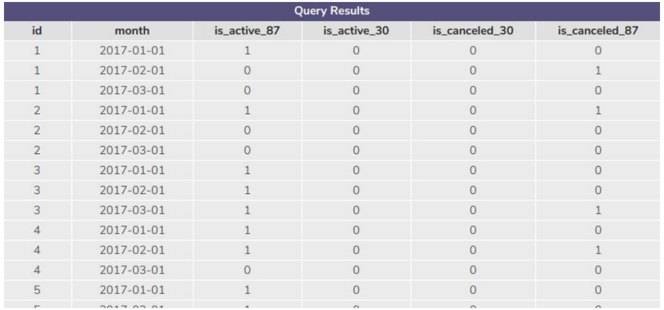
- I then created another temporary table, Status. As you can see this one does not have the duplicate data, and instead just has is_active_87, and is_active_30. Showing us each month each id was active.



Query Results

id	month	is_active_87	is_active_30
1	2017-01-01	1	0
1	2017-02-01	0	0
1	2017-03-01	0	0
2	2017-01-01	1	0
2	2017-02-01	0	0
2	2017-03-01	0	0
3	2017-01-01	1	0
3	2017-02-01	1	0

- I then added is_canceled for each segment. This lets us know each month our segments were active or canceled for:

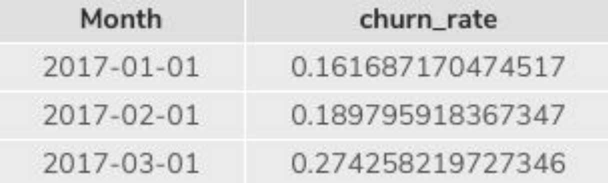


Query Results

id	month	is_active_87	is_active_30	is_canceled_30	is_canceled_87
1	2017-01-01	1	0	0	0
1	2017-02-01	0	0	0	1
1	2017-03-01	0	0	0	0
2	2017-01-01	1	0	0	1
2	2017-02-01	0	0	0	0
2	2017-03-01	0	0	0	0
3	2017-01-01	1	0	0	0
3	2017-02-01	1	0	0	0
3	2017-03-01	1	0	0	1
4	2017-01-01	1	0	0	0
4	2017-02-01	1	0	0	1
4	2017-03-01	0	0	0	0
5	2017-01-01	1	0	0	0

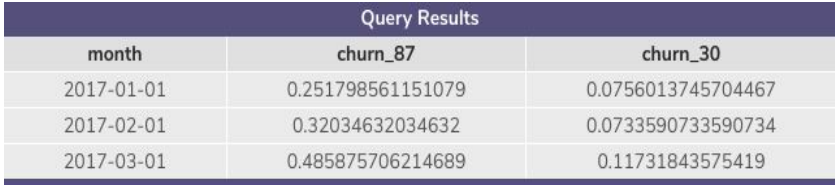
4.Codeflix Churn Trend

- For each month the trend is as follows below:



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

- And now I add the segments. As you can see in the table below the Churn rate is greater for segment 87.



Query Results

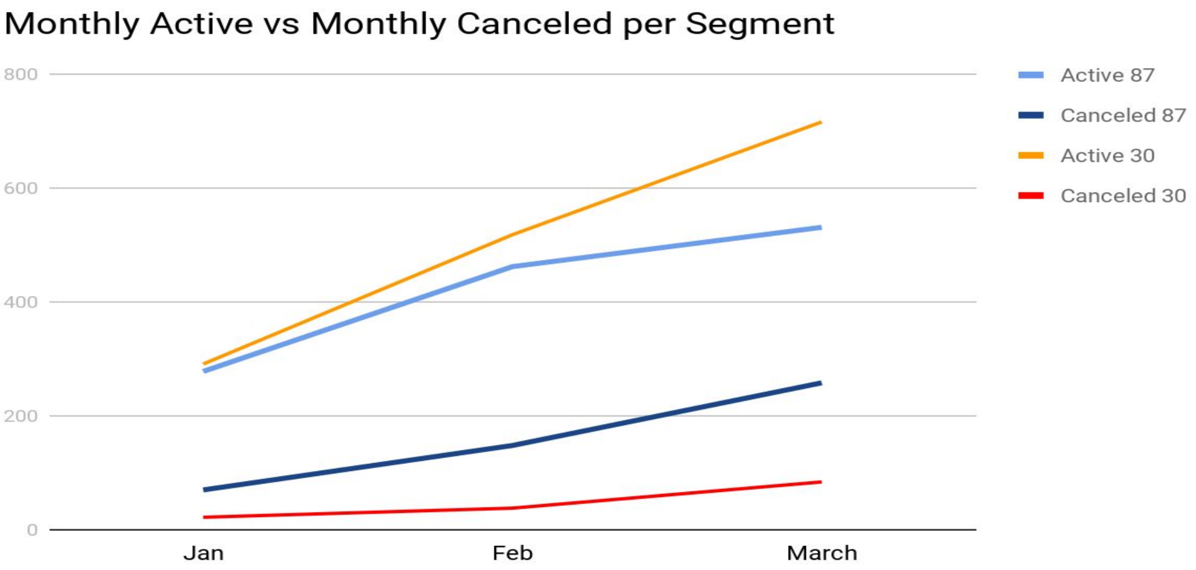
month	churn_87	churn_30
2017-01-01	0.251798561151079	0.0756013745704467
2017-02-01	0.32034632034632	0.0733590733590734
2017-03-01	0.485875706214689	0.11731843575419

This is my query to find the Churn by month:

```
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;
```

1.Visual Chrun

- In the graph below, you can see that each month Segment 30 has more Active Customers and less cancels than Segment 87. Therefore Codeflix should expand Segment 30.


Monthly Active vs Monthly Canceled per Segment
Jan
Feb
March
Active 87
Canceled 87
Active 30
Canceled 30