codecademy

Codeflix Churn Analysis

Tyler Whitlow

Codeflix Table of Contents

1.1 Codeflix Company Information

- Codeflix started their services in December of 2016
- The company has been in business for approximately 4 months (December 2016-March 2017)
- Min/Max query was used to determine months of data available and which months could be used for calculating churn

MIN (subscription_start)	Max(Subsciption_start)
2016-12-01	2017-03-30

Task 2 Query:

```
Select MIN(subscription_start)
From subscriptions;

Select Max(subscription_start)
From subscriptions;
```

1.2 Codeflix Company Information cont'd.

- Codeflix requires a minimum subscription length of 31 days, therefore we could only calculate churn from January-March 2017

MIN (subscription_start)	Max(Subsciption_start)
2016-12-01	2017-03-30

Task 2 Query:

```
Select MIN(subscription_start)
From subscriptions;

Select Max(subscription_start)
From subscriptions;
```

2.1 Codeflix Customer Segments

There are two distinct segments that we evaluated for our churn analysis

- Segment 30
- Segment 87

Id	Subscription_start	Subscription_end	segment
11	2016-12-01	2017-01-17	87
12	2016-12-01	2017-02-07	87
13	2016-12-01	n/a	30
14	2016-12-01	2017-03-07	30

Task 1 Query:

```
Select *
From subscriptions
Limit 100;
```

2.2 Codeflix Customer Segment 30

- Customer segment 30 has had some churn, but the churn rate has remained fairly consistent over the past 3 months.
- There was a slight spike in March, but the overall churn rate over the past 3 months is 8.9%.

month	churn_rate_30 (%)
2017-01-01	.076
2017-02-01	.073
2017-03-01	.117
Average	.089

2.3 Codeflix Customer Segment 87

- Customer segment 87 has seen a steady increase in churn month over month, with March's churn rate climbing to 48.6%

- Customer segment 87's churn rate average over the past 3 months is 35.3%

month	churn_rate_87 (%)
2017-01-01	.252
2017-02-01	.320
2017-03-01	.486
Average	.353

3.1 Codeflix Customer Segment Comparison/Recommendations

month	churn_rate_30 (%)	churn_rate_87 (%)
2017-01-01	.076	.252
2017-02-01	.073	.320
2017-03-01	.117	.486

***Customer segment 30 clearly has a much lower churn rate than customer segment 87:**

- Codeflix needs to spend more time on customer segment 30 in order to continue to grow market share and optimize the customer experience in an effort to keep churn at a minimum

.

- Our analysis cannot determine why segment 87's churn rate is so high. If warranted, the company may want to take a closer look at this segment to refocus marketing and/or services offered to drive down this segments churn rate.

Additional Queries Used

4.1 Task 3

- Created temporary ‘months’ table for January-March 2017 to be used for calculating churn.

first_day	last_day
2017-01-01	2017-01-31
2017-01-01	2017-02-28
2017-01-01	2017-03-31

Query:

```
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

) Select * From months;
```

4.2 Task 4

- Created 2nd temporary table ‘cross_join’ to cross join subscriptions and months table to get desired columns. New query in **bold** text.

Query:

```
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)
 Select *
 From cross_join;
```

4.3 Task 5

- Created 3rd temporary table ‘status’ which outlines user ids and aliases first_day as month.
- Case When used to determine if users from each segment were active in the prior month, if so the active column indicates 1, if not active column displays 0.
- Last line used to run query to determine if working correctly. This does not carry over to next task.

Query: **query below added to previous query outlined in slide above**

```
),
status AS
(Select id,
 first_day AS month,
 CASE
     WHEN (subscription_start < first_day) AND
(subscription_end >   first_day OR subscription_end
IS NULL) AND (segment = 87) THEN 1
     ELSE 0
 END AS is_active_87,
 CASE
     WHEN (subscription_start < first_day) AND
(subscription_end > first_day OR subscription_end IS
NULL) AND (segment = 30) Then 1
     ELSE 0
 END AS is_active_30
From cross_join
)
Select * From status Limit 5;
```

4.4 Task 6

- Case When created to determine if users canceled subscription during the dates outlined in months table for both segment 30 and 87.

- Last line used to run query to determine if working correctly. This does not carry over to next task.

Query: **query below added to previous query outlined in slide above**

```
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
)

Select * From status Limit 10;
```

4.5 Task 7

- Created temporary table ‘status_aggregate’ which displays summed totals for each segment and provides insight into how many users were active and how many users canceled their subscriptions in the months being evaluated.

- Last line used to run query to determine if working correctly. This does not carry over to next task.

Query: **query below added to previous query outlined in slide above**

```
 ), 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;
```

4.5 Task 8

- Calculated churn rates for each segment to determine which segment had a lower churn rate. And provided findings and additional recommendations in the presentation above.

Query: **query below added to previous query outlined in slide above**

```
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;
```