



Calculating Churn Rates With Codeflix

Learn SQL from Scratch

Table of Contents

1. Get familiar with Codeflix
2. What is the overall churn rate by month?
3. Compare the churn rates between segments

1. Get familiar with Codeflix

Codeflix it's a startup company that offers video streamins services and after 4 months of it's launch the managments wants to know the subscription churn rates.

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

```
SELECT *  
FROM subscriptions  
LIMIT 100;
```

```
SELECT DISTINCT segment  
FROM subscriptions;
```

id	subscription_start	subscription_end	segment
1	2016-12-01	2017-02-01	87
2	2016-12-01	2017-01-24	87
...

segment
87
30

1. Get familiar with Codeflix

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

```
-- Getting MAX and MIN values of subscriptions  
SELECT MIN(subscription_start), MAX(subscription_end)  
FROM subscriptions;
```

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

2. Calculate Churn Rate For Each Segment

3. You'll be calculating the churn rate for both segments (87 and 30) over the first 3 months of 2017 (you can't calculate it for December, since there are no subscription_end values yet). To get started, create a temporary table of months.

```
-- Create a temporary table for each months
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
),
```

2. Calculate Churn Rate For Each Segment

4. Create a temporary table, `cross_join`, from `subscriptions` and your months. Be sure to `SELECT` every column.

```
-- Cross Join the Months table with the subscriptions table  
cross_join AS  
(SELECT *  
FROM subscriptions  
CROSS JOIN months),
```

2. Calculate Churn Rate For Each Segment

5. Create a temporary table, *status*, from the *cross_join* table you created. This table should contain:

- *id* selected from *cross_join*
- *month* as an alias of *first_day*
- *is_active_87* created using a *CASE WHEN* to find any users from segment 87 who existed prior to the beginning of the month. This is 1 if true and 0 otherwise.
- *is_active_30* created using a *CASE WHEN* to find any users from segment 30 who existed prior to the beginning of the month. This is 1 if true and 0 otherwise.

```
-- Create the temporary status to get active and canceled subscribers
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),
```

2. Calculate Churn Rate For Each Segment

6. Add an `is_canceled_87` and an `is_canceled_30` column to the `statustemporary` table. This should be 1 if the subscription is canceled during the month and 0 otherwise.

Churn rate for both segments (87 and 30) over the first 3 months of 2017 is:

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

```
-- Create Aggregate numbers
status_aggregate AS
(SELECT
month,
SUM(is_active) as sum_active,
SUM(is_canceled) as sum_canceled
FROM status
GROUP BY month)
-- calculating churn rate of each month and show results
SELECT month,
1.0*sum_canceled/sum_active as 'overall
churn_rate'
FROM status_aggregate;
```


3. Compare the Churn Rates Between Segments

7. Create a *status_aggregate* temporary table that is a SUM of the active and canceled subscriptions for each segment, for each month. The resulting columns should be:

- *sum_active_87*
- *sum_active_30*
- *sum_canceled_87*
- *sum_canceled_30*

```
-- This code only create status_aggregate temporary table
status_aggregate AS
(SELECT
month,
SUM(is_active) as sum_active,
SUM(is_active_30) as sum_active_30,
SUM(is_active_87) as sum_active_87,
SUM(is_canceled) as sum_canceled,
SUM(is_canceled_30) as sum_canceled_30,
SUM(is_canceled_87) as sum_canceled_87
FROM status
GROUP BY month)
```

3. Compare the Churn Rates Between Segments

8. Calculate the churn rates for the two segments over the three month period. Which segment has a lower churn rate?

month	total_churn_rate	churn_rate_30	churn_rate_87
2017-01-01	0.161687170474517	0.0756013745704467	0.251798561151079
2017-02-01	0.189795918367347	0.0733590733590734	0.32034632034632
2017-03-01	0.274258219727346	0.11731843575419	0.485875706214689

* For complete code see file `code.sql`

The segment that has a lower churn rate on the first 3 months of the year is the segment 30, with a average of 0.088759628 churn rate.

As result, Codeflix should focuses on expanding in segment 30 that is more stable, otherwise, segment 87 also needs attention.