Four months into launching Codeflix, management asks you to look into subscription churn rates. It’s early on in the business and people are excited to know how the company is doing.

The marketing department is particularly interested in how the churn compares between two segments of users. They provide you with a dataset containing subscription data for users who were acquired through two distinct channels.

The dataset provided to you contains one SQL table, subscriptions. Within the table, there are 4 columns:

* id - the subscription id
* subscription\_start - the start date of the subscription
* subscription\_end - the end date of the subscription
* segment - this identifies which segment the subscription owner belongs to

Codeflix requires a minimum subscription length of 31 days, so a user can never start and end their subscription in the same month.

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

2 segments: 30 & 87

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

 SELECT  MIN(subscription\_start),

  MAX(subscription\_start)

FROM subscriptions;

|  |  |
| --- | --- |
| **MIN(subscription\_start)** | **MAX(subscription\_start)** |
| 2016-12-01 | 2017-03-30 |

**Calculate churn rate for each segment**

1. 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.

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

),

1. Create a temporary table, cross\_join, from subscriptions and your months. Be sure to SELECT every column.

cross\_join AS (SELECT subscriptions.\*, months.\*

FROM subscriptions

  CROSS JOIN months

),

1. 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.

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,

1. Add an is\_canceled\_87 and an is\_canceled\_30 column to the status temporary table. This should be 1 if the subscription is canceled during the month and 0 otherwise.

  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

),

1. 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

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

)

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

SELECT

month,

1.0 \* sum\_canceled\_87/sum\_active\_87 AS churn\_87,

1.0 \* sum\_canceled\_30/sum\_active\_30 AS churn\_30

FROM status\_aggregate;

|  |  |  |
| --- | --- | --- |
| **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 |

1. How would you modify this code to support a large number of segments?