codecademy

Calculating Churn Rates: CodeFlix

Learn SQL from Scratch
Opeyemi Basiru Amodu
28.11.2018

Table of Contents

Question 1

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

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

Question 2

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

There are 3 months to calculate churn rates for, CodeFlix has a one-month cancellation policy.

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

Question 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.

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

```
--
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 firt_day,
      '2017-03-31' as last_day
   )
SELECT *
FROM months;
```

Question 4.

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

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

```
--
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 firt_day,
      '2017-03-31' as last_day
   ),
cross_join AS
(SELECT *
    FROM subscriptions
      CROSS JOIN months
 )
SELECT *
FROM cross_join;
```

Question 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 0otherwise.
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 0otherwise.

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

```
--
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 firt_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)
           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,
   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
FROM cross_join
  )
SELECT *
FROM status;
```

Question 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.

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

```
--
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 firt_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)
           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,
   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,
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
FROM cross_join
  )
SELECT *
FROM status;
```

Question 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

month	sum_active_87	sum_active_30	sum_canceled_87	sum_canceled_30
2017-01-01	278	291	70	22
2017-02-01	462	518	148	38
2017-03-01	531	716	258	84

```
--
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 firt_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)
           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,
   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
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 *
FROM status_aggregate;
```

Question 8

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

month	churn_87	churn_30
2017-01-01	0.25(25%)	0.08(8%)
2017-02-01	0.32(32%)	0.07(7%)
2017-03-01	0.49(49%)	0.12(12%)

```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 firt_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)
           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,
   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
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
ROUND(1.0 * sum_canceled_87 / sum_active_87, 2),
ROUND(1.0 * sum_canceled_30 / sum_active_30, 2)
FROM status_aggregate;
```