## Calculating Churn Rates

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?

```
select *
from subscriptions
limit 10;
```

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

```
select (select min(date(subscription_start))
from subscriptions) as start, (select max(date(subscription_start))
from subscriptions) as end;

select (select min(date(subscription_end))
from subscriptions) as start, (select max(date(subscription_end))
from subscriptions) as end;
```

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`.

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

```
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
LIMIT 10;
```

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.

```
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 (segment = 87
      and subscription_start < 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)
      or subscription_end is null
THEN 1
ELSE 0
END as is_active_30
FROM cross_join)
SELECT *
FROM status
group by 1
LIMIT 10;
```

6. 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.

```
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 (segment = 87
      and subscription_start < 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)
      or subscription_end is null
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
LIMIT 100;
```

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`

```
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 (segment = 87
      and subscription_start < 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)
      or subscription_end is null
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
LIMIT 5;
```

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

```
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 (segment = 87
      and subscription_start < 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)
      or subscription_end is null
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
month,
round(1.0 * sum_canceled_87/sum_active_87, 3) AS churn_rate_87,
round(1.0 * sum_canceled_30/sum_active_30, 3) AS churn_rate_30
FROM status_aggregate;
```