Script to generate funnel data for Tableau upload.

```
--0-downloads step
WITH
downloads_count AS (
  select platform,
                        age_range,
                        date(download_ts) AS download_dt,
                        count(*) AS user_count,
                        null AS ride_count
          from app_downloads d
  left join signups s ON
           d.app_download_key = s.session_id
          group by 1,2,3
),
--1-signups step
signups_count AS (
  select platform,
                        age_range,
                        date(signup_ts) AS download_dt,
                        count(distinct s.user_id) AS user_count,
                        null AS ride_count
```

```
            from app_downloads d
  left join signups s ON
             d.app_download_key = s.session_id
            group by 1,2,3
),
--2-ride-requesteds step
ride_requested_status AS (
    SELECT
        user_id,
        MAX(
            CASE
                WHEN dropoff_ts IS NOT NULL
                THEN 1
                ELSE 0
            END
        ) AS ride_completed
    FROM ride_requests
    GROUP BY user_id
),
ride_requested_count as (
SELECT
```

```
                platform,
    age_range,
    date(download_ts) as download_dt,
    COUNT(*) AS total_users_ride_requested,
    SUM(ride_completed)::text AS total_users_ride_completed
FROM ride_requested_status u
left join signups s on
        s.user_id = u.user_id
inner join app_downloads a on
        a.app_download_key = s.session_id
group by 1,2,3
),
--3-ride-accept-status step
ride_accept_status AS (
    SELECT
        user_id,
        MAX(
            CASE
                WHEN accept_ts IS NOT NULL
                THEN 1
                ELSE 0
```

```
            END
        ) AS ride_accepted
    FROM ride_requests
    GROUP BY user_id
),
ride_accepted_count AS (
SELECT
                    platform,
    age_range,
    date(download_ts) as download_dt,
    SUM(ride_accepted) AS total_users_ride_accepted,
    COUNT(*)::text AS total_users_ride_requested
FROM ride_accept_status r
left join signups s on
            s.user_id = r.user_id
inner join app_downloads a on
            a.app_download_key = s.session_id
group by 1,2,3
  ),
--4-ride-completed step
ride_complete_status AS (
```

```
    SELECT
        user_id,
        MAX(
            CASE
                WHEN dropoff_ts IS NOT NULL
                THEN 1
                ELSE 0
            END
        ) AS ride_completed
    FROM ride_requests
    GROUP BY user_id
),
ride_completed_count as (
SELECT
                platform,
    age_range,
    date(download_ts) as download_dt,
            sum(ride_completed) AS total_users_ride_completed,
    COUNT(*)::text AS total_users_ride_requested
FROM ride_complete_status r
left join signups s on
```

```
        s.user_id = r.user_id
inner join app_downloads a on
        a.app_download_key = s.session_id
group by 1,2,3
),
--5-payment step
payment_status as (
SELECT
    user_id,
    MAX(
      CASE
        WHEN ride_id IN (
          select ride_id from transactions
          where charge_status = 'Approved')
        THEN 1
        ELSE 0
      END
    ) AS ride_payment
  FROM ride_requests r
  GROUP BY user_id
),
```

```
payments_count as (


SELECT
                platform,
    age_range,
    date(download_ts) as download_dt,
        sum(ride_payment) AS total_users_ride_payment,
    COUNT(*)::text AS total_users_ride_requested
FROM payment_status p
left join signups s on
        s.user_id = p.user_id
inner join app_downloads a on
        a.app_download_key = s.session_id
group by 1,2,3
),
--6-review step
review_status as (
SELECT
    user_id,
    MAX(
        CASE
```

```
            WHEN user_id IN (
              select user_id from reviews)
            THEN 1
            ELSE 0
          END
        ) AS ride_review
    FROM ride_requests r
    GROUP BY user_id
),
reviews_count as (
SELECT
                platform,
    age_range,
    date(download_ts) as download_dt,
    sum(ride_review) AS total_users_ride_review,
    COUNT(*)::text AS total_users_ride_requested
FROM review_status r
left join signups s on
        s.user_id = r.user_id
inner join app_downloads a on
        a.app_download_key = s.session_id
```

```
group by 1,2,3
)

--Funnel
SELECT
        0 AS funnel_step,
        'downloads' AS funnel_name, *
FROM downloads_count

        UNION

SELECT
        1 AS funnel_step,
        'signups' AS funnel_name, *
FROM signups_count

        UNION

SELECT
        2 AS funnel_step,
        'ride_requested' AS funnel_name, *
```

```
FROM ride_requested_count

        UNION

SELECT
        3 AS funnel_step,
        'ride_accepted' AS funnel_name, *
FROM ride_accepted_count

        UNION

SELECT
        4 AS funnel_step,
        'ride_completed' AS funnel_name, *
FROM ride_completed_count

        UNION

SELECT
        5 AS funnel_step,
        'payment' AS funnel_name, *
```

```
FROM payments_count

        UNION

SELECT
        6 AS funnel_step,
        'review' AS funnel_name, *
FROM reviews_count
ORDER BY 1,3
```

Script to generate the data to answer “What steps of the funnel should we research and improve?”

```
WITH all_downloads AS (
  select count(*) AS total_downloads
  from app_downloads
),
user_ride_status AS (
    SELECT
        user_id
    FROM ride_requests
    GROUP BY user_id
),
user_ride_accepted AS (
                    SELECT
                    COUNT(DISTINCT user_id) AS total_users_ride_accepted
              FROM ride_requests
                    WHERE accept_ts IS NOT null
),
totals AS (
    SELECT
        COUNT(*) AS total_users_signed_up,
        COUNT(DISTINCT urs.user_id) AS total_users_ride_requested
    FROM signups s
    LEFT JOIN user_ride_status urs ON
```

```
        s.user_id = urs.user_id
),
ride_completed AS (
  SELECT
        user_id,
        MAX(
            CASE
                WHEN dropoff_ts IS NOT NULL
                THEN 1
                ELSE 0
            END
        ) AS ride_completed
    FROM ride_requests
    GROUP BY user_id
),
ride_payment AS (
SELECT
        user_id,
        MAX(
            CASE
                WHEN ride_id IN (
                  select ride_id from transactions
                  where charge_status = 'Approved')
                THEN 1
                ELSE 0
            END
        ) AS ride_paid
    FROM ride_requests r
    GROUP BY user_id
),
review_counts AS (
  SELECT user_id, count(review_id) AS tot_reviews
          FROM reviews
  GROUP BY 1
),
funnel_stages AS (
          SELECT
        0 AS funnel_step,
        'downloads' AS funnel_name,
        total_downloads AS value
    FROM all_downloads

    UNION
```

```
  SELECT
    1 AS funnel_step,
    'signups' AS funnel_name,
    total_users_signed_up AS value
  FROM totals

  UNION

  SELECT
    2 AS funnel_step,
    'ride_requested' AS funnel_name,
    total_users_ride_requested AS value
  FROM totals

  UNION

  SELECT
    3 AS funnel_step,
    'ride_accepted' AS funnel_name,
    sum(total_users_ride_accepted) AS value
  FROM user_ride_accepted

  UNION

  SELECT
    4 AS funnel_step,
    'ride_completed' AS funnel_name,
    sum(ride_completed) AS value
  FROM ride_completed

UNION

  SELECT
    5 AS funnel_step,
    'ride_payment' AS funnel_name,
    sum(ride_paid) AS value
  FROM ride_payment

  UNION

  SELECT
    6 AS funnel_step,
    'review_count' AS funnel_name,
    count(*) AS value
```

```
    FROM review_counts

)
SELECT *,
    value::float / FIRST_VALUE(value) OVER (
        ORDER BY funnel_step
    ) AS pct_top_conversion_rate,
   round((1.0 - value::numeric/ FIRST_VALUE(value) over (
                    ORDER BY funnel_step)),2) as pct_top_drop_off,
     value::float / LAG(value) OVER (ORDER BY funnel_step
    ) AS pct_prev_conversion_rate,
    round((1.0 - value::numeric/lag(value, 1) over ()),2) as pct_prev_drop_off
FROM funnel_stages
ORDER BY funnel_step;
```

Script to answer the question “Are there any specific drop-off points preventing users from completing their first ride?”

```
WITH all_downloads AS (

  select count(*) AS total_downloads

  from app_downloads

),

user_ride_status AS (

    SELECT

        user_id

    FROM ride_requests

    GROUP BY user_id

),

user_ride_accepted AS (

                    SELECT
```

```
        COUNT(DISTINCT user_id) AS total_users_ride_accepted
    FROM ride_requests
        WHERE accept_ts IS NOT null
),
user_ride_cancelled AS (
        SELECT
        COUNT(DISTINCT user_id) AS total_users_ride_cancelled
    FROM ride_requests
        WHERE accept_ts IS NOT null
        AND cancel_ts IS NOT null
),
totals AS (
  SELECT
    COUNT(*) AS total_users_signed_up,
    COUNT(DISTINCT urs.user_id) AS total_users_ride_requested
  FROM signups s
  LEFT JOIN user_ride_status urs ON
    s.user_id = urs.user_id
),
ride_completed AS (
 SELECT
```

```
        user_id,
        MAX(
          CASE
            WHEN dropoff_ts IS NOT NULL
            THEN 1
            ELSE 0
          END
        ) AS ride_completed
    FROM ride_requests
    GROUP BY user_id
),
review_counts AS (
  SELECT user_id, count(review_id) AS tot_reviews
          FROM reviews
  GROUP BY 1
),
funnel_stages AS (
          SELECT
        0 AS funnel_step,
        'downloads' AS funnel_name,
        total_downloads AS value
```

```
FROM all_downloads
UNION
  SELECT
    1 AS funnel_step,
    'signups' AS funnel_name,
    total_users_signed_up AS value
FROM totals
UNION
SELECT
    2 AS funnel_step,
    'ride_requested' AS funnel_name,
    total_users_ride_requested AS value
FROM totals
        UNION
SELECT
    3 AS funnel_step,
    'ride_accepted' AS funnel_name,
    sum(total_users_ride_accepted) AS value
FROM user_ride_accepted
UNION
SELECT
```

```
        4 AS funnel_step,
        'ride_cancelled' AS funnel_name,
        total_users_ride_cancelled AS value
    FROM user_ride_cancelled
            UNION
     SELECT
        5 AS funnel_step,
        'ride_completed' AS funnel_name,
        sum(ride_completed) AS value
    FROM ride_completed
            UNION
    SELECT
        5 AS funnel_step,
        'review_count' AS funnel_name,
        count(*) AS value
    FROM review_counts)
SELECT *,
    value::float / FIRST_VALUE(value) OVER (
        ORDER BY funnel_step
    ) AS pct_top_conversion_rate,
   round((1.0 - value::numeric/ FIRST_VALUE(value) over (
```

```
                    ORDER BY funnel_step)),2) as pct_top_drop_off,
    value::float / LAG(value) OVER (ORDER BY funnel_step
  ) AS pct_prev_conversion_rate,
  round((1.0 - value::numeric/lag(value, 1) over ()),2) as pct_prev_drop_off
FROM funnel_stages
ORDER BY funnel_step;
```

Below script to generate the ride requests daily distribution for the answer of this question “Surge pricing is the practice of increasing the price of goods or services when there is the greatest demand for them. If we want to adopt a price-surging strategy, what does the distribution of ride requests look like throughout the day?”

```
WITH ride_hour AS (
  SELECT
    user_id,
    extract(hour from request_ts) AS requested_hour,
    count(distinct ride_id) as total_ride_requests
  FROM ride_requests
  where request_ts is not null
  GROUP BY user_id,2
)
select requested_hour,
                    sum(total_ride_requests) AS ride_counts
from ride_hour
```

group by 1

order by 1