Metrocar Project

Funnel Analysis

1 Objectives

According to this metrocar project, here are the objectives to achieve:

1. To answer all business questions by data-driven methods
2. To perform funnel analysis
3. To visualize insights from funnel data

2 Data

There are five tables from the given database: app_downloads, reviews, ride_requests, sign-ups, and transactions.

- app_downloads: the app downloading data contains the app downloading key, downloading time, and platform.
- reviews: the review data contains the reviews and ratings from users related to ride and driver id.
- ride_requests: the ride data contains the corresponding user id and driver id, the time of ride request, ride accept, pick-up and drop-off, and the location of pick-up and drop-off.
- sign-ups: the sign-up data contains the age range of users, signing-up time, and the session id.
- transactions: the transaction data contains the purchase amount of the ride, transaction time, and charge status.

3 Methodology

3.1 Access the database using BeeKeeper Studio

To acquire the data, BeeKeeper was installed to access the database. After that, SQL language was implemented to query for required data.

Here is the list of SQL commands that are used to filter datasets:

- SELECT: using for showing the data from the database column
- CASE: using for filtering groups into control and treatment groups
- SUM: using for add values
- MAX: using to obtain the max value
- LEFT JOIN: using for joining tables
- GROUP BY: using for grouping data based on some columns
- WITH…AS: using to create a preview table

- UNION: using to combine the results
- LAG: using to access data from a previous row without joining
- VALUE: using to specify the datatype of math operation

All SQL code is shown in Appendix A

3.2 Funnel Analysis using Tableau

To analyze the funnel data, Tableau is used to visualize and plot the funnel analysis.

1. Insert the funnel data querying from SQL.
2. Present the data in a bar graph using funnel metric as a row and funnel value as a column.
3. Calculate the percentage from the top and the percentage from the previous
4. Show two bar graphs in the same plot to compare

4 Results

In general funnel analysis, Figure 1 shows the percent of the top on the left-hand side and the percent of the previous on the right-hand side.

In percent of Top view, 74.65% of users sign up, 52.55% request the rides, 26.40% complete the rides, and only 18.42% review the rides. On the other hand, the percentage of previous shows the percentage of users from the previous step accomplishes the next step. It shows that 74.65% of downloading users sign up for the app, 70.40% of sign-up users request the rides, 50.24% of requesting ride users complete the rides, and 69.76% of completing ride users review the rides.

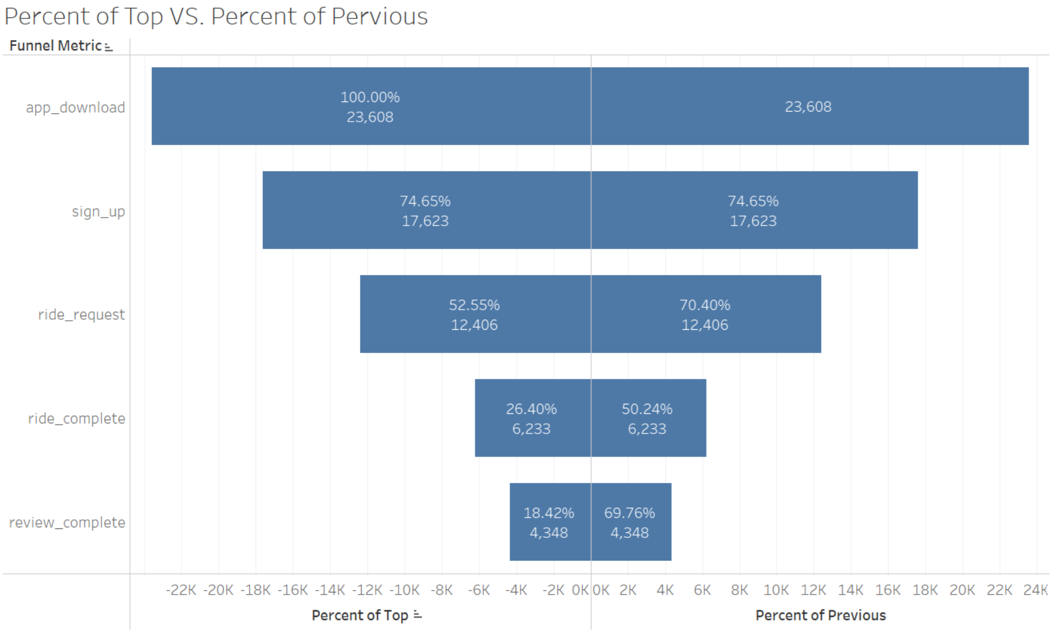


Figure 1 shows the percent of top VS. the percent of previous of funnel analysis.

Figure 2 shows the users' activity on the different platforms. Most of the users come from the iOS platform. The total number of ios users is 14,290, and 18.55% accomplish all ride steps. Android is the 2nd largest segment of users at 6,935, and 18.36% of them complete until the last step, ride review. The least platform users use is the web browser at 2,383 users, and only 17.79% of them review the rides.

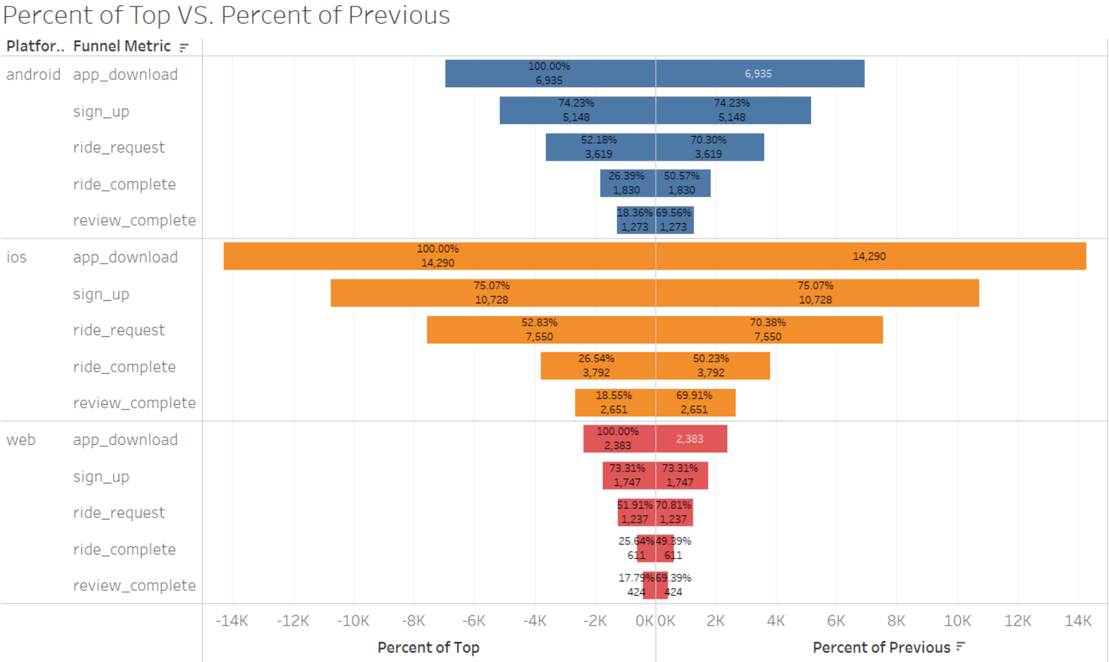


Figure 2 shows the percent of top VS. the percent of previous of funnel analysis for each platform.

Speaking about the age range of users, the 35-44 group is the majority of the users. Even though the unknown age group is a large segment as well, it barely provides any constructive information for data analysis. The rank of users' age is 35-44, 25-34, and 18-24 as well as 45-54 at almost the same amount.

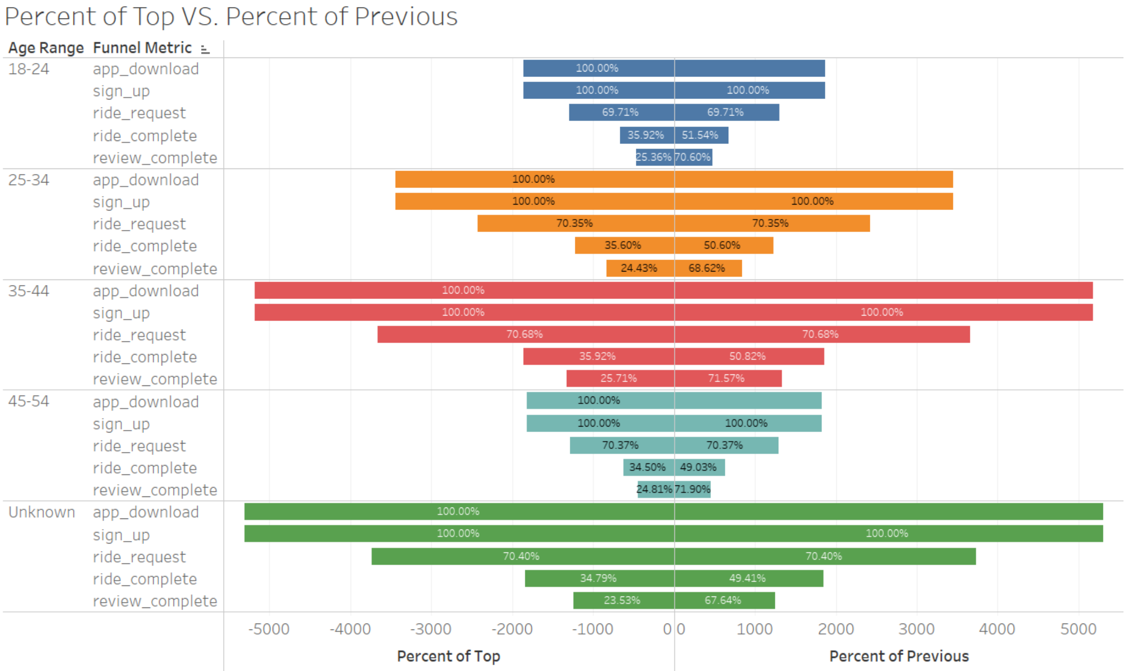


Figure 3 shows the percent of top VS. the percent of previous of funnel analysis for each age range.

To better understand the relationship between day hours and rides, Figure 4 is plotted. Even though the trend of ride request and ride complete seems similar, the ride request has a higher number, which is reasonable because there might be some cases which users request the ride and cancel it afterward. Speaking about the hour, the prime time that users usually request the rides is at 8.00 – 9.00 and 16.00 – 17.00.

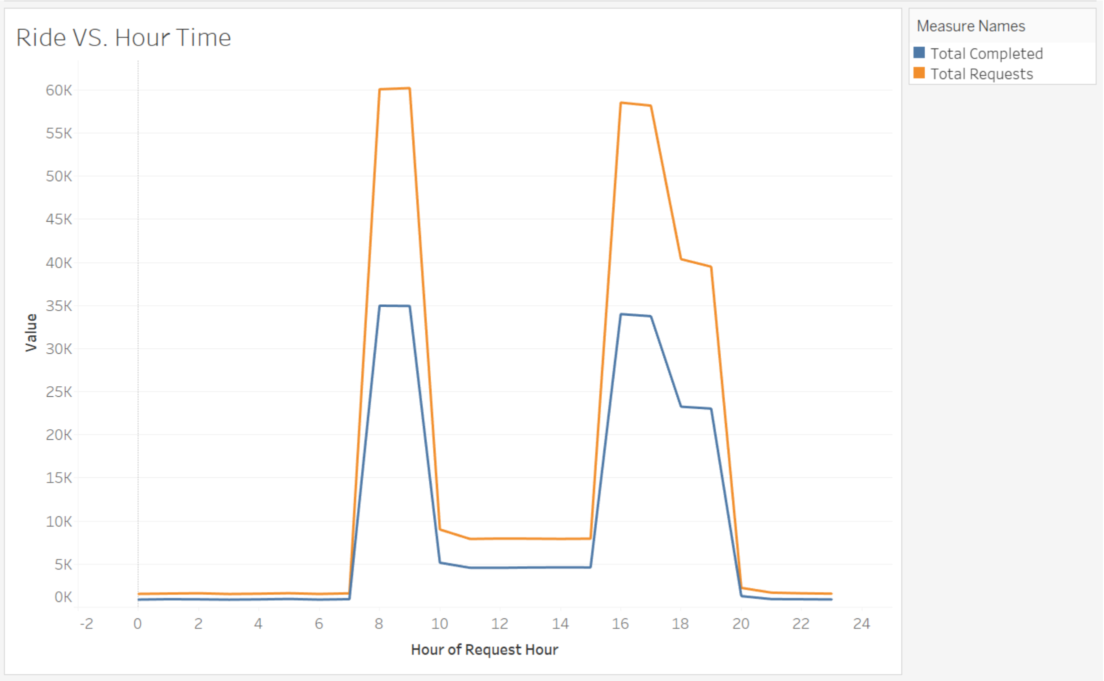


Figure 4 shows the plot of ride requests, ride completes, and hours.

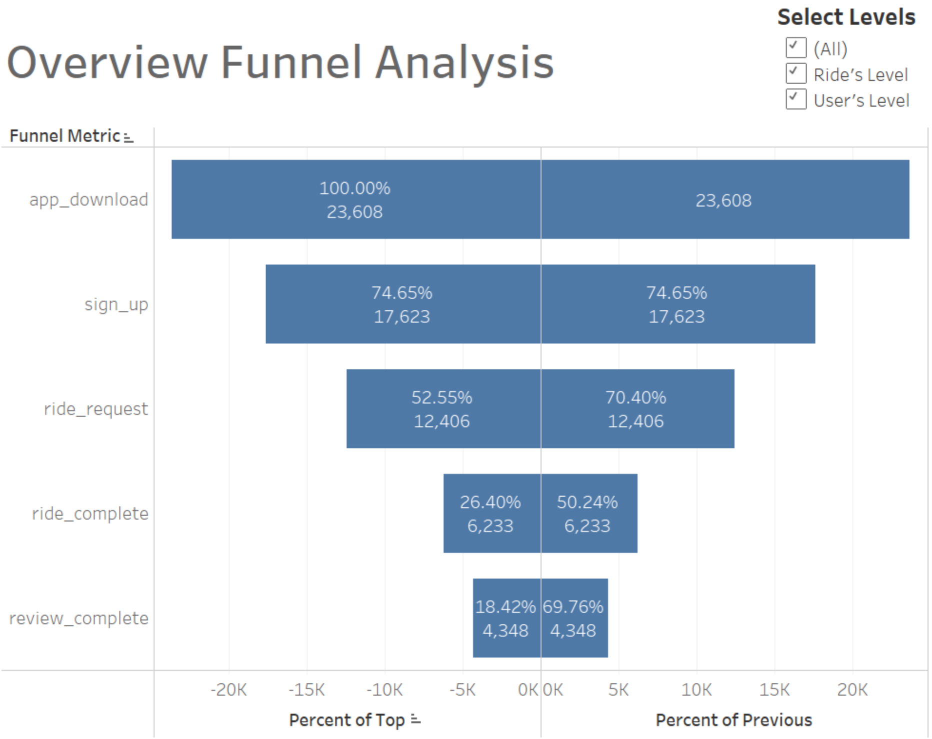
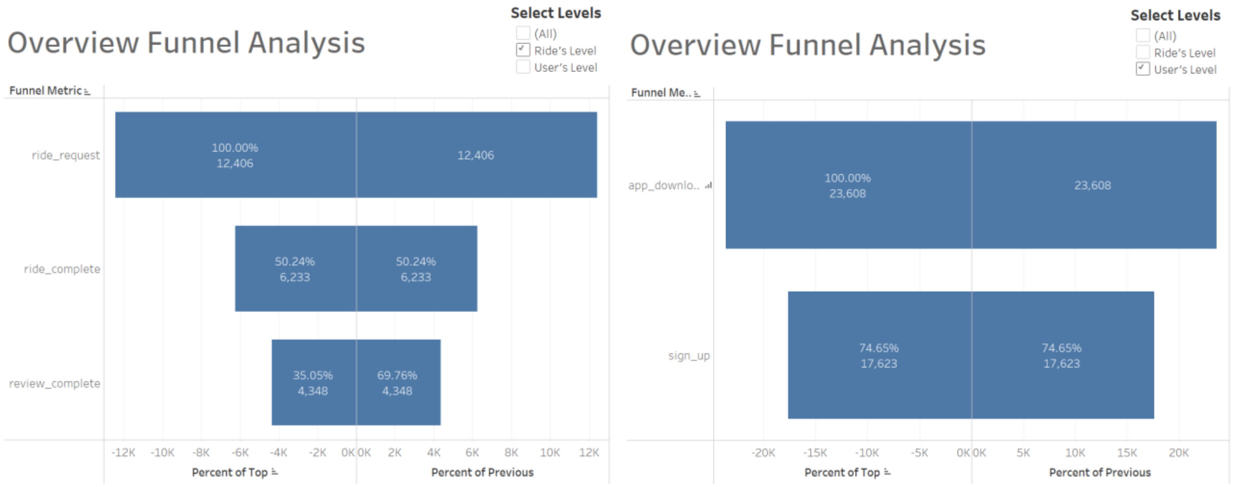


Figure 5 Overview Funnel Analysis with all level selection



(A) (B)

Figure 6 Overview Funnel Analysis with (A) Ride's Level and (B) Users' Level

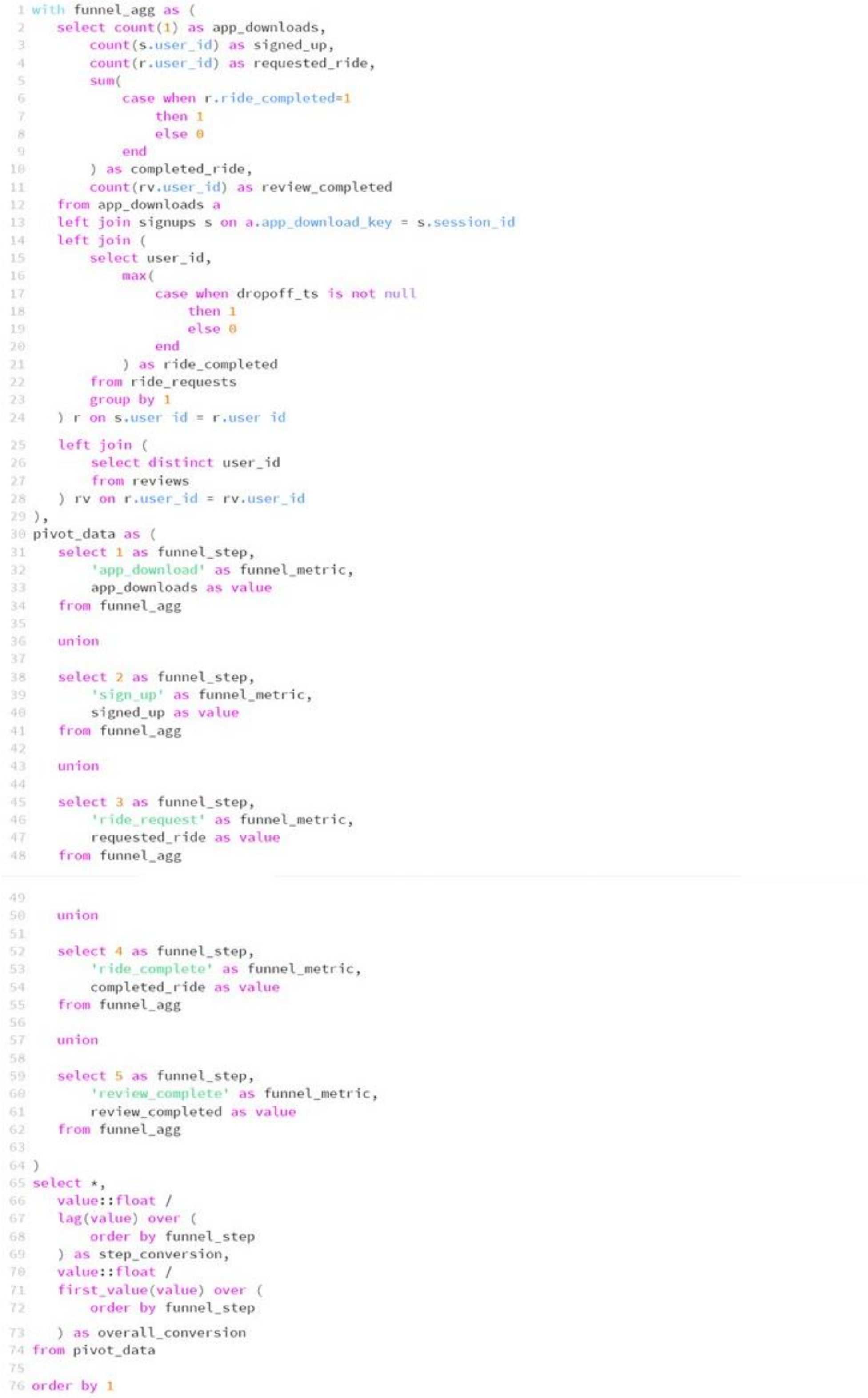
5 Conclusion and Discussion

To answer the business questions, funnel analysis is performed and shown in Figure 1 to Figure 4. The general funnel analysis tells the overview of users' behavior through each step. The platform and age range funnel analysis reveals users' behavior regarding specific conditions. The ride and hour time plot demonstrates the ride request and complete prime time.

The lowest conversion rate is the riding completion stage, which requires further improvement. Speaking about specific condition analysis, there are platforms, age ranges, and hour time for riding activities. IOS is the platform users participate in most, and it is worth putting marketing effort into it. For age range analysis, all age groups sign up for the app after downloading. The group of 35 – 44 is the highest percentage requesting rides after the sign-up stage at 70.68%. However, the group of 18 - 24 is the group that most completed the rides at 51.54%. The group with the most reviews after completing the rides is the 45 – 54 group at 71.90%. The ride and hour time plot shows the distribution of riding activities throughout the day, and the prime time of ride request and completion is 8.00 – 9.00 and 16.00 – 17.00.

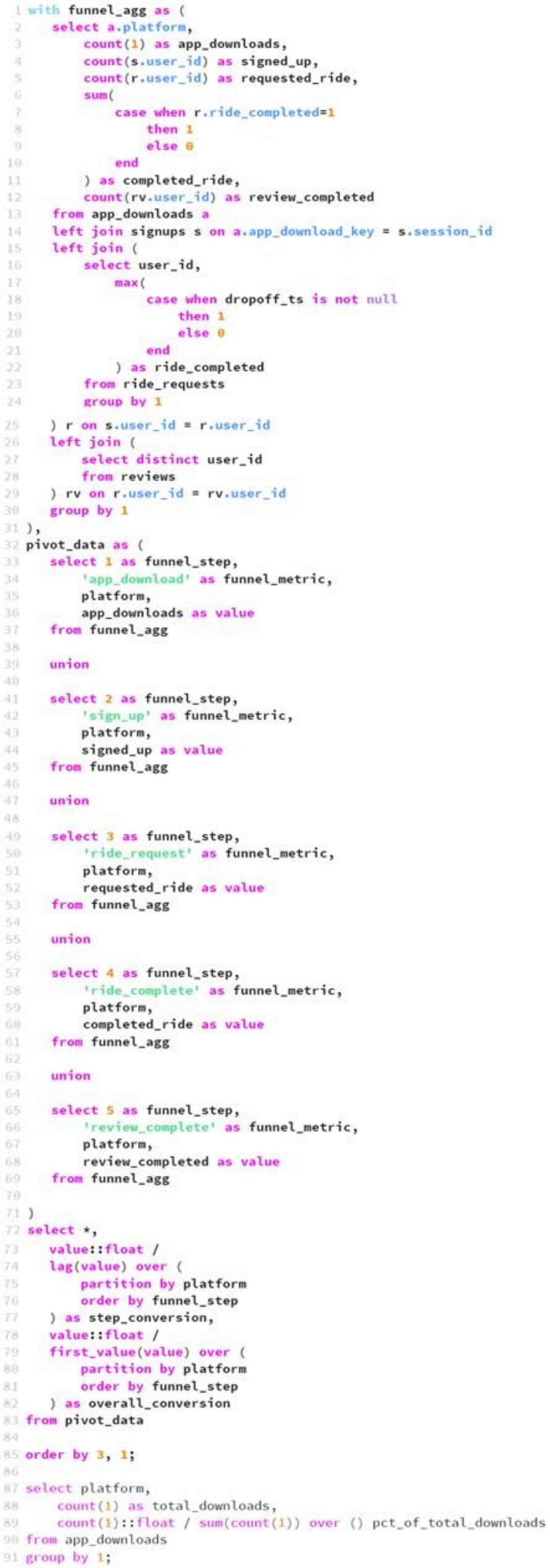
In conclusion, the step that requires improvement is riding complete, and the suggestion is to study the ride management system further because it might have some issues that cause the users to cancel requests. Regarding marketing investment, the iOS platform is the most users' contribution platform. The app's target group will be between 35 and 44 since it shows the biggest user segment. To adopt a surge pricing strategy, 8.00 to 9.00 and 16.00 to 17.00 is the best time for this because it shows the highest participation from users in both ride requests and ride completion.

Appendix A



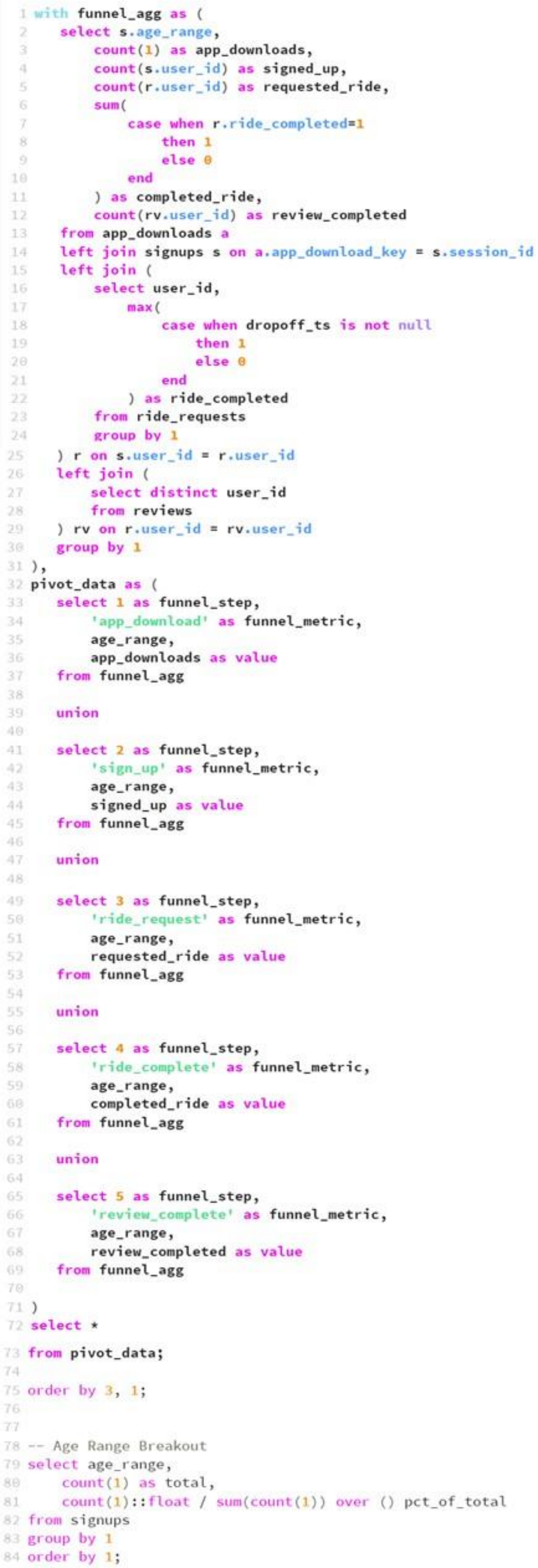
```sql
with funnel_agg as (
    select count(1) as app_downloads,
        count(s.user_id) as signed_up,
        count(r.user_id) as requested_ride,
        sum(
            case when r.ride_completed=1
                then 1
                else 0
            end
        ) as completed_ride,
        count(rv.user_id) as review_completed
    from app_downloads a
    left join signups s on a.app_download_key = s.session_id
    left join (
        select user_id,
            max(
                case when dropoff_ts is not null
                    then 1
                    else 0
                end
            ) as ride_completed
        from ride_requests
        group by 1
    ) r on s.user_id = r.user_id
    left join (
        select distinct user_id
        from reviews
    ) rv on r.user_id = rv.user_id
),
pivot_data as (
    select 1 as funnel_step,
        'app_download' as funnel_metric,
        app_downloads as value
    from funnel_agg

    union

    select 2 as funnel_step,
        'sign_up' as funnel_metric,
        signed_up as value
    from funnel_agg

    union

    select 3 as funnel_step,
        'ride_request' as funnel_metric,
        requested_ride as value
    from funnel_agg

    union

    select 4 as funnel_step,
        'ride_complete' as funnel_metric,
        completed_ride as value
    from funnel_agg

    union

    select 5 as funnel_step,
        'review_complete' as funnel_metric,
        review_completed as value
    from funnel_agg

)
select *,
    value::float /
    lag(value) over (
        order by funnel_step
    ) as step_conversion,
    value::float /
    first_value(value) over (
        order by funnel_step
    ) as overall_conversion
from pivot_data

order by 1
```

Figure 7 General Funnel SQL



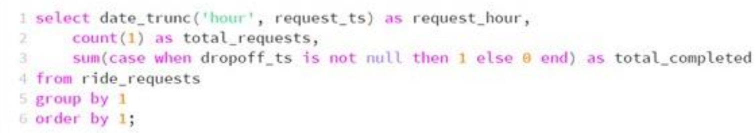
```
with funnel_agg as (
   select a.platform,
       count(1) as app_downloads,
       count(s.user_id) as signed_up,
       count(r.user_id) as requested_ride,
       sum(
           case when r.ride_completed=1
               then 1
               else 0
           end
       ) as completed_ride,
       count(rv.user_id) as review_completed
   from app_downloads a
   left join signups s on a.app_download_key = s.session_id
   left join (
       select user_id,
           max(
               case when dropoff_ts is not null
                   then 1
                   else 0
               end
           ) as ride_completed
       from ride_requests
       group by 1
   ) r on s.user_id = r.user_id
   left join (
       select distinct user_id
       from reviews
   ) rv on r.user_id = rv.user_id
   group by 1
),
pivot_data as (
   select 1 as funnel_step,
       'app_download' as funnel_metric,
       platform,
       app_downloads as value
   from funnel_agg

   union

   select 2 as funnel_step,
       'sign_up' as funnel_metric,
       platform,
       signed_up as value
   from funnel_agg

   union

   select 3 as funnel_step,
       'ride_request' as funnel_metric,
       platform,
       requested_ride as value
   from funnel_agg

   union

   select 4 as funnel_step,
       'ride_complete' as funnel_metric,
       platform,
       completed_ride as value
   from funnel_agg

   union

   select 5 as funnel_step,
       'review_complete' as funnel_metric,
       platform,
       review_completed as value
   from funnel_agg

)
select *,
   value::float /
   lag(value) over (
       partition by platform
       order by funnel_step
   ) as step_conversion,
   value::float /
   first_value(value) over (
       partition by platform
       order by funnel_step
   ) as overall_conversion
from pivot_data

order by 3, 1;

select platform,
    count(1) as total_downloads,
    count(1)::float / sum(count(1)) over () pct_of_total_downloads
from app_downloads
group by 1;
```

Figure 8 Funnel SQL for platform



```sql
with funnel_agg as (
   select s.age_range,
       count(1) as app_downloads,
       count(s.user_id) as signed_up,
       count(r.user_id) as requested_ride,
       sum(
           case when r.ride_completed=1
               then 1
               else 0
           end
       ) as completed_ride,
       count(rv.user_id) as review_completed
   from app_downloads a
   left join signups s on a.app_download_key = s.session_id
   left join (
       select user_id,
           max(
               case when dropoff_ts is not null
                   then 1
                   else 0
               end
           ) as ride_completed
       from ride_requests
       group by 1
   ) r on s.user_id = r.user_id
   left join (
       select distinct user_id
       from reviews
   ) rv on r.user_id = rv.user_id
   group by 1
),
pivot_data as (
   select 1 as funnel_step,
       'app_download' as funnel_metric,
       age_range,
       app_downloads as value
   from funnel_agg

   union

   select 2 as funnel_step,
       'sign_up' as funnel_metric,
       age_range,
       signed_up as value
   from funnel_agg

   union

   select 3 as funnel_step,
       'ride_request' as funnel_metric,
       age_range,
       requested_ride as value
   from funnel_agg

   union

   select 4 as funnel_step,
       'ride_complete' as funnel_metric,
       age_range,
       completed_ride as value
   from funnel_agg

   union

   select 5 as funnel_step,
       'review_complete' as funnel_metric,
       age_range,
       review_completed as value
   from funnel_agg

)
select *
from pivot_data;

order by 3, 1;


-- Age Range Breakout
select age_range,
    count(1) as total,
    count(1)::float / sum(count(1)) over () pct_of_total
from signups
group by 1
order by 1;
```

Figure 9 Funnel SQL for age range



```sql
select date_trunc('hour', request_ts) as request_hour,
    count(1) as total_requests,
    sum(case when dropoff_ts is not null then 1 else 0 end) as total_completed
from ride_requests
group by 1
order by 1;
```

Figure 10 Ride and hour time SQL