```sql
-- CREATING THE Cohort Table based on Elena's criteria of considering
sessions starting after 2023/01/04 and users with > 7 sessions
WITH cohort
     AS (SELECT user_id
         FROM   sessions
         WHERE  session_start >= '2023/01/04'
         GROUP  BY user_id
         HAVING Count(session_id) > 7),
     --- Identified Metrics
     ---  The no of times they got a flight discount when compared
across bookings.Final column name is % of flight discount
     ads_flight_percentage
     AS (SELECT Coalesce(SUM(CASE
                               WHEN flight_discount THEN 1
                               ELSE 0
                             END) :: FLOAT / Count(*), 0) AS
                discount_flight_proportion,
                user_id
         FROM   sessions
         WHERE  session_start >= '2023/01/04'
         GROUP  BY user_id
         HAVING Count(session_id) > 7),
     --- How much discount they got off the flight base fare value.
The name of final column is Average flight discount
     avg_f_dis_amt
     AS (SELECT Coalesce(Avg(flight_discount_amount), 0) AS
                average_flight_discount,
                user_id
         FROM   sessions
         WHERE  session_start >= '2023/01/04'
         GROUP  BY user_id
         HAVING Count(session_id) > 7),
     --- This gives the value of avg dollars saved per km in the final
column name Average_dollassaved_per_km
     avg_dollars_saved
     AS (SELECT user_id,
                Coalesce(SUM(s.flight_discount_amount *
f.base_fare_usd) / SUM(
Haversine_distance(u.home_airport_lat, u.home_airport_lon,
f.destination_airport_lat, f.destination_airport_lon)), 0) AS ADS
 FROM   sessions s
        left join flights f
               ON s.trip_id = f.trip_id
        left join users u USING (user_id)
 WHERE  session_start >= '2023/01/04'
```

```
 GROUP  BY s.user_id
 HAVING Count(session_id) > 7),
     ---How many times they got a hotel discount across all bookings .
final column name is Average hotel discount
     ads_hotel_percentage
     AS (SELECT Coalesce(SUM(CASE
                               WHEN hotel_discount THEN 1
                               ELSE 0
                             END) :: FLOAT / Count(*), 0) AS
                discount_hotel_proportion,
                user_id
         FROM   sessions
         GROUP  BY user_id),
     ---The amount of hotel discount . Final column name is % of Hotel
Discount
     avg_h_dis_amt
     AS (SELECT Coalesce(Avg(hotel_discount_amount), 0) AS
                average_hotel_discount,
                user_id
         FROM   sessions
         WHERE  session_start >= '2023/01/04'
         GROUP  BY user_id
         HAVING Count(session_id) > 7),
     --- Average Number of clicks per user
     average_num_clicks
     AS (SELECT Coalesce(Avg(page_clicks), 0) AS average_clicks,
                user_id
         FROM   sessions
         WHERE  session_start >= '2023/01/04'
         GROUP  BY user_id
         HAVING Count(session_id) > 7),
     --- How many trips in percent were cancelled by a user. Final
column is % of trips cancelled
     cancellation_percentage
     AS (SELECT user_id,
                Coalesce(SUM(CASE
                               WHEN cancellation THEN 1
                               ELSE 0
                             END) :: FLOAT / Count(*), 0) AS
                percentage_of_cancellations_across_sessions,
                Count(trip_id)                            AS
count_of_trip,
                SUM(CASE
                      WHEN cancellation = TRUE THEN 1
                      ELSE 0
```

```
                    END)                                          AS
cancellation,
                CASE
                  WHEN Count(trip_id) > 0
                       AND SUM(CASE
                                 WHEN cancellation = TRUE THEN 1
                                 ELSE 0
                               END) > 0 THEN Coalesce(SUM(CASE
                                                            WHEN
cancellation
                                                          THEN 1
                                                            ELSE 0
                                                          END) ::
FLOAT / Count(

trip_id)
                                                   , 0)
                  ELSE 0
                END                                           AS
                   percentage_of_trips_cancelled
         FROM   sessions
         WHERE  session_start >= '2023/01/04'
                AND user_id IN (SELECT user_id
                                FROM   cohort)
         GROUP  BY user_id),
     --- Its the percentage of flights booked as comapared to all
browsing sessions by a user .
     flight_conversion_rate
     AS (SELECT user_id,
                SUM(CASE
                      WHEN flight_booked THEN 1
                      ELSE 0
                    END) * 1.0 / ( Count(session_id) * 1.0 ) AS
flight_conv_rate
         FROM   sessions
         WHERE  session_start >= '2023/01/04'
         GROUP  BY user_id
         HAVING Count(session_id) > 7),
     --- This is the conversion rate for no of bookings done(hotel or
flight) as compared to browsing sessions
     conversion_rate
     AS (SELECT user_id,
                Count(trip_id) * 1.0 / ( Count(session_id) * 1.0 ) AS
conv_rate
         FROM   sessions
```

```
        WHERE  session_start >= '2023/01/04'
        GROUP  BY user_id
        HAVING Count(session_id) > 7),
    ---Age group (Finding Seniors)
    above55
    AS (SELECT u.user_id,
               CASE
                 WHEN Extract(year FROM birthdate) < 1968 THEN 1
                 ELSE 0
               END AS above_55
        FROM   sessions s
               left join users u USING (user_id)
        WHERE  session_start >= '2023/01/04'
        GROUP  BY u.user_id
        HAVING Count(session_id) > 7),
  age AS(
        SELECT u.user_id,
        EXTRACT(YEAR FROM AGE(birthdate)) AS age
        FROM sessions s
        LEFT JOIN users u USING (user_id)
        WHERE  session_start >= '2023/01/04'
        GROUP  BY u.user_id
        HAVING Count(session_id) > 7),

    --Has kids(Identifying Families)
    kids
    AS (SELECT u.user_id,
               CASE
                 WHEN has_children THEN 1
                 ELSE 0
               END AS children
        FROM   sessions s
               left join users u USING (user_id)
        WHERE  session_start >= '2023/01/04'
        GROUP  BY 1,
                  2
        HAVING Count(session_id) > 7),
    --- Frequent travellers . Gives the count of total trips per
user. Final column name Frequent traveller
    trip_count
    AS (SELECT u.user_id,
               Count(s.trip_id) AS trip_id_count
        FROM   sessions s
               left join users u USING (user_id)
        WHERE  session_start >= '2023/01/04'
```

```
        GROUP  BY 1
        HAVING Count(session_id) > 7),
    --- Calculating avg distance travelled.
    longdistancetraveller
    AS (SELECT s.user_id,
               Coalesce(Avg(Haversine_distance(home_airport_lat,
                            home_airport_lon,
                                    destination_airport_lat,
                                        destination_airport_lon)),
0) distance
        FROM   sessions s
               left join users u USING (user_id)
               left join flights f USING(trip_id)
        WHERE  session_start >= '2023/01/04'
        GROUP  BY 1
        HAVING Count(session_id) > 7),
    ---Count of checked bags and seats.
    sumofbagsnseats
    AS (SELECT u.user_id,
               Coalesce(SUM(f.checked_bags), 0) AS
total_checked_bags,
               Coalesce(SUM(f.seats), 0)        AS num_of_seats
        FROM   users u
               left join sessions s
                      ON u.user_id = s.user_id
               left join flights f
                      ON s.trip_id = f.trip_id
        WHERE  s.session_start >= '2023-01-04'
        GROUP  BY u.user_id
        HAVING Count(s.session_id) > 7),
    --- Comparing if no of checked bags were greater than no of seats
    hasmorebagsthanseats
    AS (SELECT user_id,
               CASE
                 WHEN total_checked_bags > num_of_seats THEN 1
                 ELSE 0
               END AS morebags
        FROM   sumofbagsnseats
        GROUP  BY 1,
                  2),
NumOfBags AS
(
         SELECT    u.user_id,
                   COALESCE(sum(f.checked_bags), 0) AS totalbags
         FROM      users u
```

```
            LEFT JOIN sessions s
            ON        u.user_id = s.user_id
            LEFT JOIN flights f
            ON        s.trip_id = f.trip_id
            WHERE     s.session_start >= '2023-01-04'
            GROUP BY  u.user_id
            HAVING    count(s.session_id) > 7),

      ---Finding the max days out of all the trips taken by a user in
Days
      longtrip
      AS (SELECT user_id,
                 Max(Coalesce(Extract(epoch FROM ( return_time -
departure_time )
                                      ), 0)
                 ) /
                 86400
                 AS timedifference
          FROM   sessions s
                 left join flights f USING(trip_id)
          WHERE  session_start >= '2023/01/04'
          GROUP  BY 1
          HAVING Count(session_id) > 7),
      -- Table that converts the binary values from cancellation to 0
and 1
      cancellation
      AS (SELECT c.user_id,
                 -- Sum gets the count of the amount of trips they have
canceled from the 1/0 values
                 SUM(CASE
                       WHEN cancellation = TRUE THEN 1
                       ELSE 0
                     END)       AS cancellation,
                 Count(trip_id) AS count_of_trip
          FROM   cohort c
                 inner join sessions USING (user_id)
          WHERE  session_start >= '2023/01/04'
          GROUP  BY user_id),
      -- Advanced booking group (flights only). Calculating how far in
advance the flight is booked
      advanced_flight_booking
      AS (SELECT c.user_id,
                 Avg(Coalesce(Extract(day FROM ( departure_time -
session_start )
                                      ), 0)
```

```
                )
                   AS
                day_difference
         FROM   cohort c
                left join sessions USING (user_id)
                left join flights USING (trip_id)
         GROUP  BY 1),
     -- We use the table below to figure out wether they have traveled
on a Sunday(0),Friday(5)
     -- or Saturday (6). This gives use the values of each trip they
did.
     -- For example if one user took 3 trips but only two of them were
on the weekend
     -- they would have one row with 0 and two rows with 1.
     weekend_traveller
     AS (SELECT c.user_id,
                CASE
                  WHEN Extract(dow FROM departure_time) IN ( 0, 5, 6
)
                       OR Extract(dow FROM check_in_time) IN ( 0, 5,
6 ) THEN 1
                  ELSE 0
                END AS "weekend_traveller?"
         FROM   cohort c
                inner join sessions s USING (user_id)
                inner join flights f USING (trip_id)
                inner join hotels h USING (trip_id)),
     -- We use the table below to figure out whether they have appear
more than twice in the weekend_traveller
     -- table. By using SUM we count how many trips they have  taken
during the weekends.
     regular_weekend_traveller
     AS (SELECT user_id,
                CASE
                  WHEN SUM("weekend_traveller?") > 2 THEN 1
                  ELSE 0
                END AS reg_weekend_traveller
         FROM   weekend_traveller
         GROUP  BY user_id)
--Main query
---Called all Metrics to view the results and decided on scaling
method as well as indexes for each perk
SELECT DISTINCT c.user_id,
                avghp.discount_hotel_proportion        AS "% of Hotel
discount",
```

```
                avghd.average_hotel_discount          AS
                "Average Hotel Discount",
                avgfp.discount_flight_proportion      AS "% of
Flight discount"
                ,
                avgfd.average_flight_discount         AS
                "Average Flight Discount",
                cp.percentage_of_trips_cancelled      AS "% of Trips
Cancelled"
                ,
                ---We use Coalesce to replace the null values for the
users that did not have a flight reservation
                Coalesce(rwt.reg_weekend_traveller, 0) AS "Weekend
Traveller",
                a55.above_55                          AS "Above 55
?",
                kid.children                          AS "Has
kids?",
                ldt.distance                          AS
                "Long Distance Traveller",
                cr.conv_rate                          AS "Conversion
rate",
                bns.morebags                          AS
"Bags>Seats",
                lt.timedifference                     AS "Length of
trip",
                tc.trip_id_count                      AS "Frequent
traveller",
                afb.day_difference                    AS "Long Term
Planning",
                ads.ads                               AS
                "Average_dollarsaved_per_km",
                fcr.flight_conv_rate                  AS
                "Flight_Conversion_Rate"
                age.age                               AS "User Age"
FROM   cohort c
       LEFT JOIN users u
              ON c.user_id = u.user_id
       LEFT JOIN sessions s
              ON s.user_id = u.user_id
       LEFT JOIN flights f
              ON s.trip_id = f.trip_id
       LEFT JOIN ads_flight_percentage avgfp
              ON u.user_id = avgfp.user_id
       LEFT JOIN ads_hotel_percentage avghp
```

```
            ON u.user_id = avghp.user_id
       LEFT JOIN avg_f_dis_amt avgfd
            ON u.user_id = avgfd.user_id
       LEFT JOIN avg_h_dis_amt avghd
            ON u.user_id = avghd.user_id
       LEFT JOIN avg_dollars_saved ads
            ON u.user_id = ads.user_id
       LEFT JOIN flight_conversion_rate fcr
            ON u.user_id = fcr.user_id
       LEFT JOIN average_num_clicks anc
            ON u.user_id = anc.user_id
       LEFT JOIN cancellation_percentage cp
            ON u.user_id = cp.user_id
       LEFT JOIN above55 a55
            ON u.user_id = a55.user_id
       LEFT JOIN kids kid
            ON u.user_id = kid.user_id
       LEFT JOIN trip_count tc
            ON u.user_id = tc.user_id
       LEFT JOIN longdistancetraveller ldt
            ON u.user_id = ldt.user_id
       LEFT JOIN hasmorebagsthanseats bns
            ON u.user_id = bns.user_id
       LEFT JOIN longtrip lt
            ON u.user_id = lt.user_id
       LEFT JOIN advanced_flight_booking afb
            ON u.user_id = afb.user_id
       LEFT JOIN regular_weekend_traveller rwt
            ON u.user_id = rwt.user_id
       LEFT JOIN conversion_rate cr
            ON u.user_id = cr.user_id
       LEFT JOIN age age
             ON u.user_id = age.user_id
WHERE  session_start >= '2023/01/04'
GROUP  BY c.user_id,
          avghp.discount_hotel_proportion,
          avgfp.discount_flight_proportion,
          avgfd.average_flight_discount,
          avghd.average_hotel_discount,
          ads.ads,
          anc.average_clicks,
          cp.percentage_of_trips_cancelled,
          rwt.reg_weekend_traveller,
          a55.above_55,
          kid.children,
```

```
            tc.trip_id_count,
            ldt.distance,
            cr.conv_rate,
            bns.morebags,
            lt.timedifference,
            afb.day_difference,
            tc.trip_id_count,
            afb.day_difference,
            ads.ads,
            Fcr.flight_conv_rate,
            age.age
HAVING Count(session_id) > 7
ORDER  BY c.user_id;
```