```
-- This CTE prelimits our sessions on the suggested timeframe (After Jan 4 2023)
WITH sessions_2023 AS (
  SELECT *
  FROM sessions s
  where s.session_start > '2023-01-04'
),

-- This CTE returns the ids of all users with more than 7 sessions in 2023
filtered_users AS (
  SELECT user_id, COUNT(*) FROM sessions_2023 s
  GROUP BY user_id
  HAVING COUNT(*) > 7
),

-- This is our main session base table
-- It joins Sessions with all available user and trip information
-- We made sure to limit the session according to Head of Marketing recommendations
-- Each row is a browsing session on the TravelTide App
-- Sessions have trips connected to them when they were booked or canceled
-- We have already cleaned the column of nights in a hotel, because it contained negative
values

session_base AS
(select s.session_id, s.user_id, s.trip_id, s.session_start, s.session_end, EXTRACT(EPOCH
FROM s.session_end-s.session_start) as session_duration, s.page_clicks,
 s.flight_discount, s.flight_discount_amount, s.hotel_discount, s.hotel_discount_amount,
s.flight_booked, s.hotel_booked, s.cancellation,
 u.birthdate, u.gender, u.married, u.has_children, u.home_country, u.home_city,
u.home_airport, u.home_airport_lat, u.home_airport_lon,u.sign_up_date,
 f.origin_airport, f.destination, f.destination_airport, f.seats, f.return_flight_booked,
f.departure_time, f.return_time, f.checked_bags, f.trip_airline, f.destination_airport_lat,
f.destination_airport_lon,f.base_fare_usd,
 h.hotel_name, CASE WHEN h.nights < 0 THEN 1 ELSE h.nights END AS nights, h.rooms,
h.check_in_time, h.check_out_time, h.hotel_per_room_usd AS
hotel_price_per_room_night_usd
from sessions_2023 s
left join users u
on s.user_id = u.user_id
left join flights f
on s.trip_id = f.trip_id
left join hotels h
on s.trip_id = h.trip_id
WHERE s.user_id IN (SELECT user_id FROM filtered_users)),

-- This CTE returns the ids of all trips that have been canceled through a session
-- We use this list to filter all canceled sessions in the next CTE

canceled_trips AS (
  SELECT DISTINCT trip_id
  FROM session_base
```

```
  WHERE cancellation = TRUE
),

-- This is our second base table to aggregate later
-- It is derived from our session_base table, but we focus on valid trips
-- All sessions without trips, all canceled trips have been removed
-- Each row represents a trip that a user did

not_canceled_trips AS(
  SELECT *
  FROM session_base
        WHERE trip_id IS NOT NULL
        AND trip_id NOT IN (SELECT trip_id FROM canceled_trips)),



-- We want to aggregate user behaviour into metrics (a row per user)
-- This CTE contains metrics that have to do with the browsing behaviour
-- ALL SESSION within our cohort get aggregated

user_base_session AS
(
SELECT user_id,
  SUM(page_clicks) AS num_clicks,
  COUNT(DISTINCT session_id) AS num_sessions,
  AVG(session_duration) AS avg_session_duration
FROM session_base
GROUP BY user_id),

-- We want to aggregate user behaviour into metrics (a row per user)
-- This CTE contains metrics that have to do with the travel behavious
-- Only rows with VALID trips within our cohort get aggregated

user_base_trip AS
(SELECT user_id,
 COUNT(DISTINCT trip_id) AS num_trips,
 SUM(CASE WHEN (flight_booked = TRUE) AND (return_flight_booked = TRUE) THEN 2
WHEN flight_booked = TRUE THEN 1 ELSE 0 END) AS num_flights,
COALESCE((SUM((hotel_price_per_room_night_usd * nights * rooms) * (1 - (CASE WHEN
hotel_discount_amount IS NULL THEN 0 ELSE hotel_discount_amount END)))),0) AS
money_spend_hotel,
AVG(EXTRACT(DAY FROM departure_time-session_end)) AS time_after_booking,
AVG(haversine_distance(home_airport_lat, home_airport_lon, destination_airport_lat,
destination_airport_lon)) AS avg_km_flown,
AVG(checked_bags) as avg_bags
FROM not_canceled_trips
GROUP BY user_id
),
```

```
-- For our final user table, we join the session metric, trip metrics and general user information
-- Using a left join, we will get a row for each user from our original cohort codition (7+ browsing sessions in 2023)
-- If we used an inner join, we could get rid of users that have not actually travelled

user_metrics AS
(SELECT b.*,
EXTRACT(YEAR FROM AGE(u.birthdate)) AS age, u.gender, u.married, u.has_children, u.home_country, u.home_city, u.home_airport,
COALESCE(t.num_trips,0) AS num_trips, COALESCE(t.num_flights,0) AS num_flights, COALESCE(t.money_spend_hotel,0) AS money_spend_hotel,
COALESCE(t.time_after_booking,0) AS time_after_booking, COALESCE(t.avg_km_flown,0) AS avg_km_flown, COALESCE(t.avg_bags,0) AS avg_bags
FROM user_base_session b
LEFT JOIN users u
ON b.user_id = u.user_id
LEFT JOIN user_base_trip t
ON b.user_id = t.user_id)

SELECT *,
  CASE
    WHEN has_children=True THEN 'Family Traveller'
    WHEN age>60 THEN 'Senior travellers'
    WHEN age<60 THEN
        CASE
          WHEN num_trips<=2 THEN 'Dreamers'
          WHEN age<30 AND num_trips>2 THEN 'Young frequent traveller'
          WHEN age>30 THEN
            CASE
              WHEN num_trips>5 THEN 'Business Travellers'
              WHEN num_trips BETWEEN 2 AND 5 THEN 'Young adult frequent travellers'
              ELSE 'Others'
            END
          ELSE 'Others'
      END
    ELSE 'Others'
  END AS Customer_group
FROM user_metrics
```