1. Considering data after Jan 4 2023
Aggregated User Level

```
WITH joined_tables AS (
   SELECT s.user_id as u_id,
      s.trip_id as trip_id_final,
      *,
      CASE
         WHEN check_out_time::date - check_in_time::date < 1 THEN 1
         ELSE check_out_time::date - check_in_time::date
      END as nights_new,
      CASE
         when rooms = 0 THEN 1
         ELSE rooms
      END as new_rooms
   FROM sessions s
      LEFT JOIN users u ON s.user_id = u.user_id
      LEFT JOIN hotels h ON s.trip_id = h.trip_id
      LEFT JOIN flights f ON s.trip_id = f.trip_id
   WHERE session_start >= '2023-01-04'
),
```

2. user_level - demographics

```
user_level AS (
   SELECT u_id as ul_id,
      EXTRACT(
         year
         from age('2023-07-28', birthdate)
      ) AS user_age,
      ---- Calculates the numerical age
      CASE
         WHEN EXTRACT(
            year
            from age('2023-07-28', birthdate)
         ) < 18 THEN 'Under 18'
         WHEN EXTRACT(
            year
            from age('2023-07-28', birthdate)
         ) < 25 THEN '18-25'
         WHEN EXTRACT(
            year
            from age('2023-07-28', birthdate)
         ) < 35 THEN '25-35'
```

```
            WHEN EXTRACT(
                year
                from age('2023-07-28', birthdate)
            ) < 45 THEN '35-45'
            WHEN EXTRACT(
                year
                from age('2023-07-28', birthdate)
            ) < 55 THEN '45-55'
            WHEN EXTRACT(
                year
                from age('2023-07-28', birthdate)
            ) < 65 THEN '55-65'
            ELSE '65+'
        END as age_group
    FROM joined_tables
    GROUP BY u_id,
        birthdate
    HAVING COUNT(session_id) > 7
),
```

3. session_level_base filtering Users with more than 7 Sessions in the selected time frame(2023-01-04)

```
session_level_base AS (
    SELECT s.session_id,
        AVG(s.flight_discount_amount) AS avg_flight_discount,
        AVG(s.hotel_discount_amount) AS avg_hotel_discount
    FROM joined_tables jt
        JOIN user_level ul ON jt.u_id = ul.ul_id --joining users and sessions table to joined_table
        JOIN sessions s ON ul_id = s.user_id
    GROUP BY s.session_id
),
```

4. trip_level_metrics

```
trip_metrics AS (
    SELECT u_id,
        AVG(nights_new) as avg_nights,
        AVG(checked_bags) as avg_bags,
        AVG(seats) as avg_seats,
        AVG(rooms) as avg_rooms,
        AVG(base_fare_usd) as avg_spend,
        COUNT(DISTINCT trip_id_final) as num_trips,
        SUM(
```

```
        CASE
          WHEN flight_booked = TRUE
          and hotel_booked = FALSE
          and cancellation = FALSE THEN 1
        END
      ) AS flights_booked_only, --
      SUM(
        CASE
          WHEN hotel_booked = TRUE
          and flight_booked = FALSE
          and cancellation = FALSE THEN 1
        END
      ) AS hotel_booked_only
    FROM joined_tables
    GROUP BY u_id
),
```

5. session_level_metrics

```
session_metrics AS (
    SELECT u_id,
      COUNT(DISTINCT session_id),
      AVG(page_clicks) AS avg_page_clicks,
      AVG(flight_discount_amount) AS avg_flight_discount,
      AVG(hotel_discount_amount) AS avg_hotel_discount
    FROM joined_tables
    GROUP BY u_id
),
```

6. user metrics

```
user_metrics AS (
    SELECT u_id,
      SUM(
        CASE
          WHEN flight_booked = TRUE
          and hotel_booked = FALSE and cancellation = FALSE THEN jt.base_fare_usd
        END --calculating flight booked only without cancellations, hotel booked
      ) AS money_spent_flight,
      SUM(
        CASE
          WHEN hotel_booked = TRUE
          and flight_booked = FALSE and cancellation = FALSE THEN jt.hotel_per_room_usd
        END --calculating hotel booked only without cancellations, flight booked
```

```
    ) AS money_spent_hotel,
    SUM(
      CASE
        WHEN hotel_booked = TRUE
        and flight_booked = FALSE
        and cancellation = FALSE THEN jt.hotel_per_room_usd
      END
    ) + SUM(
      CASE
        WHEN flight_booked = TRUE
        and hotel_booked = FALSE
        and cancellation = FALSE THEN jt.base_fare_usd
      END --calculating hotel + flight booked without cancellations
    ) AS money_spent_total,
    AVG(
      haversine_distance(
        u.home_airport_lat,
        u.home_airport_lon,
        f.destination_airport_lat,
        f.destination_airport_lon
      )
    ) AS avg_distance_kms --calculating average distance in kms
  FROM joined_tables jt
    LEFT JOIN users u ON u.user_id = jt.u_id --joining user table and joined tables
    LEFT JOIN flights f ON f.trip_id = jt.trip_id_final --joining flight
  GROUP BY u_id
),
```

7. user_summary

```
user_summary AS (
  SELECT u_id,
    SUM(jt.hotel_per_room_usd) AS total_money_spent_hotel,
                    SUM(jt.base_fare_usd) AS total_money_spent_flight,
    COUNT(DISTINCT trip_id_final) as num_trips
  FROM joined_tables jt
  GROUP BY u_id
),
user_features AS (
  SELECT ul.ul_id,
    jt.sign_up_date,
                    ul.user_age,
                    jt.has_children,
    jt.home_country,
```

```
    jt.home_city,
    COUNT(DISTINCT session_id) AS sessions,
    COUNT(jt.cancellation) as cancellations,
    sm.avg_page_clicks,
    tm.avg_spend,
    tm.avg_nights,
    tm.avg_rooms,
    tm.avg_seats,
    sm.avg_hotel_discount,
    sm.avg_flight_discount,
    um.avg_distance_kms,
    um.money_spent_total,
    um.money_spent_flight,
    um.money_spent_hotel,
                        us.total_money_spent_hotel,
                        us.total_money_spent_flight,
    ul.age_group,
    us.num_trips
FROM joined_tables jt
    JOIN user_level ul ON jt.u_id = ul.ul_id
    LEFT JOIN trip_metrics tm ON tm.u_id = jt.u_id
    LEFT JOIN session_metrics sm ON sm.u_id = jt.u_id
    LEFT JOIN user_metrics um ON um.u_id = jt.u_id
    LEFT JOIN user_summary us ON us.u_id = jt.u_id
GROUP BY ul.ul_id,
    jt.sign_up_date,
    jt.home_country,
    jt.home_city,
    sm.avg_page_clicks,
    tm.avg_spend,
    tm.avg_nights,
    tm.avg_rooms,
    tm.avg_seats,
    sm.avg_hotel_discount,
    sm.avg_flight_discount,
    um.avg_distance_kms,
    um.money_spent_total,
    um.money_spent_flight,
    um.money_spent_hotel,
                        us.total_money_spent_hotel,
                        us.total_money_spent_flight,
    ul.user_age,
    jt.has_children,
    ul.age_group,
```

```
        us.num_trips
)
SELECT *
           FROM user_features
  ORDER BY ul_id;
```