```
-- STEP 1: Filter relevant sessions
WITH sessions_filter AS
(
  SELECT *
  FROM sessions
  WHERE session_start > '2023-01-04'
),
-- STEP 2: Identify users with more than 7 sessions
user_filter AS
(
  SELECT user_id,
  COUNT(*)
  FROM sessions_filter
  GROUP BY user_id
  HAVING COUNT(*) > 7
),
-- STEP 3: Aggregate user data with related entities
session_combined_table AS
(
  SELECT
          sf.session_id,
          sf.user_id,
          sf.trip_id,
          sf.session_start,
          sf.session_end,
          sf.flight_discount,
          sf.hotel_discount,
          sf.flight_discount_amount,
          sf.hotel_discount_amount,
          sf.flight_booked,
          CASE WHEN sf.flight_booked = 'yes' THEN 1 ELSE 0 END AS flights_booking_int,
          sf.hotel_booked,
          CASE WHEN sf.hotel_booked = 'yes' THEN 1 ELSE 0 END AS hotel_booking_int,
          sf.page_clicks,
          sf.cancellation,
          CASE WHEN sf.cancellation = 'yes' THEN 1 ELSE 0 END AS cancellation_int,
          u.birthdate,
          EXTRACT(Year FROM CURRENT_DATE) - EXTRACT(Year FROM u.birthdate)
AS customer_age,
          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,
          COALESCE(haversine_distance(home_airport_lat,home_airport_lon,
                      destination_airport_lat, destination_airport_lon),0) AS
flown_flight_distance,
          SPLIT_PART(hotel_name, '-', 1) AS Hotel_name,
          SPLIT_PART(hotel_name, '-', 2) AS Hotel_location,
          CASE
              WHEN h.nights < 0 THEN ABS (h.nights)
              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
  FROM sessions_filter sf
  LEFT JOIN users u
          ON sf.user_id = u.user_id
  LEFT JOIN flights f
          ON sf.trip_id = f.trip_id
  LEFT JOIN hotels h
          ON sf.trip_id = h.trip_id
  WHERE sf.user_id IN (SELECT user_id FROM user_filter)
),
-- STEP 4: Compute user-level aggregated metrics
user_agg_metric AS
(
          SELECT
              user_id, customer_age, gender,
          married, has_children, home_city,home_country,home_airport,
          COUNT(DISTINCT trip_id) AS num_trips,
          COUNT(session_id) AS num_sessions,
          MIN(session_start::DATE) AS user_start_date,
          MAX(session_end::DATE) AS user_end_date,
     SUM(page_clicks) AS total_page_clicks,
     ROUND(AVG(EXTRACT(EPOCH FROM (session_end - session_start))),3) AS
avg_session_duration_seconds,
          SUM(CASE WHEN cancellation = TRUE THEN 1 ELSE 0 END) AS
num_of_cancellation,
          SUM(CASE WHEN flight_discount = True THEN 1 ELSE 0 END) AS
num_of_flights_discount,
          SUM(CASE WHEN hotel_discount = True THEN 1 ELSE 0 END) AS
num_of_hotel_discount,
          ROUND(SUM(COALESCE(base_fare_usd,0) * COALESCE(1-
flight_discount_amount,1) * COALESCE(seats,0))::NUMERIC,2) AS total_flight_charges,
          ROUND(SUM(COALESCE(hotel_per_room_usd,0) * COALESCE(1-
hotel_discount_amount,1) * COALESCE(rooms,0) * COALESCE(nights,0))::NUMERIC,2)
AS total_hotel_charges,
          ROUND(SUM(COALESCE(base_fare_usd,0) * COALESCE(1-
flight_discount_amount,1) * COALESCE(seats,0)) +
          SUM(COALESCE(hotel_per_room_usd,0) * COALESCE(1-
hotel_discount_amount,1) * COALESCE(rooms,0) * COALESCE(nights,0)),3) AS
total_sales
          FROM session_combined_table
          GROUP BY
          user_id, customer_age, gender, married, has_children, home_city,
home_country,home_airport
),
-- STEP 5: Calculate value scores per user
customer_value AS(
SELECT user_id,
  CASE
          WHEN customer_age < 18 THEN '< 18'
                  WHEN customer_age BETWEEN 18 AND 24 THEN 'Student'
          WHEN customer_age BETWEEN 25 AND 34 THEN 'Young Aged'
          WHEN customer_age BETWEEN 35 AND 60 THEN 'Middle Aged'
          WHEN customer_age > 60 THEN 'Senior Citizen'
          ELSE 'Unknown'
          END AS age_group,
  ROUND(AVG(total_sales)::NUMERIC,2) AS avg_total_sales,
  ROUND(SUM(total_sales)::NUMERIC/NULLIF(SUM(num_trips),0),2) AS
customer_value_per_trip,
  ROUND(SUM(total_sales)::NUMERIC / NULLIF(SUM(num_sessions), 0),2) AS
customer_value_per_session,
  ROUND(AVG('2023-07-29' - user_start_date)/180,2) AS avg_cust_lifespan
FROM user_agg_metric
GROUP BY user_id,customer_age
),
-- Merging Above 2 CTEs
combined_metric AS
(
  SELECT *
          FROM user_agg_metric
          JOIN customer_value
  USING(user_id)
),
-- STEP 6: Compute Discount Proportion metrics
discount_propn AS(
  SELECT user_id,
          ROUND((SUM(CASE WHEN hotel_discount THEN 1 ELSE 0 END)::FLOAT /
COUNT(*))::NUMERIC, 2) AS hotel_discount_proportion,
   ROUND((SUM(CASE WHEN flight_discount THEN 1 ELSE 0 END)::FLOAT /
COUNT(*))::NUMERIC, 2) AS flight_discount_proportion,
   COALESCE(ROUND(AVG(flight_discount_amount)::NUMERIC,2),0) AS
avg_flight_discount_charges,
        COALESCE(ROUND(AVG(hotel_discount_amount)::NUMERIC,2),0) AS
avg_hotel_discount_charges,
        ROUND(((SUM(CASE WHEN flight_discount THEN 1 ELSE 0 END)+
        SUM(CASE WHEN hotel_discount THEN 1 ELSE 0 END)) :: FLOAT
       / COUNT(*))::NUMERIC,2) AS total_discount_proportion,

COALESCE(ROUND(AVG(flight_discount_amount*base_fare_usd)::NUMERIC,2),0) AS
avg_dollar_saved,
   COALESCE(ROUND(SUM(flight_discount_amount*base_fare_usd)/
SUM(flown_flight_distance)::NUMERIC,3),0) AS avg_dollar_saved_per_km
        FROM session_combined_table
        GROUP BY user_id
),
-- STEP 7: Session level final aggregated table
session_level_final_table AS(
  SELECT user_id,
  COUNT(DISTINCT trip_id) AS booking_count,
  ROUND((COUNT(DISTINCT (trip_id)) *1.0) / COUNT(session_id),2) AS booking_rate,
  EXTRACT(DAY FROM (MAX(session_end)- MIN(session_start))) AS active_days,
  ROUND(AVG(page_clicks),2) as avg_page_clicks,
  COALESCE(ROUND(SUM(checked_bags) * 1.0 / COUNT(DISTINCT (trip_id)),2),0) AS
avg_bags,
  CASE
                WHEN COUNT(DISTINCT (trip_id)) > 0 THEN
                ROUND(SUM(cancellation_int)*1.0 / COUNT(DISTINCT (trip_id)),2)
ELSE 0
                END AS cancellation_rate,
        SUM(flights_booking_int) - SUM(cancellation_int) AS num_flight_booked,
  SUM(hotel_booking_int) - SUM(cancellation_int) AS num_hotel_booked,
  COALESCE(ROUND(SUM(seats)*1.0 / COUNT(DISTINCT trip_id)),0) AS
avg_num_seats,
  COALESCE(SUM(checked_bags),0) AS total_checked_bags,
  ROUND(AVG(departure_time::DATE - session_end::DATE),2) AS travel_lead_time,
  ROUND(AVG(CASE WHEN cancellation = TRUE THEN (departure_time::DATE -
session_end::DATE)
                        ELSE NULL END),2) AS avg_cancel_lead_time,
        ROUND(COALESCE(SUM(flown_flight_distance), 0)::NUMERIC, 2) AS
total_dist_flown,
  ROUND(COALESCE(AVG(flown_flight_distance),0)::NUMERIC,2) AS
avg_dist_flown_incl, --cancelled trip is also included
  COALESCE(ROUND(SUM(nights)*1.0 / COUNT(DISTINCT trip_id)),0) AS
avg_hotel_stay,
  COALESCE(ROUND(SUM(rooms)*1.0 / COUNT(DISTINCT trip_id)),0) AS
avg_hotel_rooms,
  COALESCE(EXTRACT (DAY FROM AVG(return_time - departure_time)),0) AS
avg_trip_duration,
  AVG(session_end - session_start) AS avg_session_duration,
  SUM(CASE WHEN flight_discount = True AND trip_id IS NOT NULL THEN 1 ELSE 0
END) AS num_flights_discount_applied,
          SUM(CASE WHEN hotel_discount = True AND trip_id IS NOT NULL THEN 1
ELSE 0 END) AS num_hotel_discount_applied,
          SUM(CASE WHEN flight_discount = True AND trip_id IS NULL THEN 1 ELSE 0
END) AS num_flights_discount_offered,
  SUM(CASE WHEN hotel_discount = True AND trip_id IS NULL THEN 1 ELSE 0 END)
AS num_hotel_discount_offered,

SUM(COALESCE(hotel_per_room_usd,0)*COALESCE(hotel_discount_amount,0)*COAL
ESCE(rooms,0)*COALESCE(nights,0)) total_hotel_discount_charges,

ROUND(SUM(COALESCE(base_fare_usd,0)*COALESCE(flight_discount_amount,0)*CO
ALESCE(seats,0)),2) total_flight_discount_charges
  FROM session_combined_table
          GROUP BY user_id

),
-- STEP 6: Compute trip ratio metrics
trip_ratios AS
(
  SELECT
                              user_id,
                              COALESCE((num_flights_discount_applied +
num_hotel_discount_applied) / NULLIF(booking_count, 0),0) AS discounted_booking_rate,
                              COALESCE(total_dist_flown / NULLIF(num_flight_booked,0),0) AS
avg_dist_flown,
                              COALESCE(NULLIF(num_flight_booked,0) /
NULLIF(num_hotel_booked,0), 0) AS flight_to_hotel_booking_ratio,
                              COALESCE(total_checked_bags / NULLIF(num_flight_booked,0),0)
AS checked_bags_ratio
  FROM session_level_final_table
  ),
--Combining above 2 CTEs
final_single_user_table AS
(
  SELECT *
  FROM trip_ratios
  JOIN session_level_final_table
  USING(user_id)
),
--STEP 7: final metrics calculation and combining all the above combined CTEs
final_metrics AS
(
  SELECT *,
                              ROUND((avg_cust_lifespan * customer_value_per_trip),2) AS
customer_lifetime_value,
                              ROUND((avg_dollar_saved_per_km -
MIN(avg_dollar_saved_per_km) OVER ()) /
```

```
                            NULLIF((MAX(avg_dollar_saved_per_km) OVER () -
MIN(avg_dollar_saved_per_km) OVER ()), 0),3) AS scaled_ADS
         FROM final_single_user_table
  LEFT JOIN combined_metric
  USING (user_id)
         JOIN discount_propn
  USING(user_id)
),
--STEP 8: Assigning perks
perks_assignment AS
(
  SELECT
                            *,
    ROUND(scaled_ADS * flight_discount_proportion *
avg_flight_discount_charges::NUMERIC,4) AS bargain_hunter_index,
  CASE
                  WHEN num_trips >= 4 AND avg_dist_flown >=2000 THEN 'Priority Check-
in and Boarding Privilege'
     WHEN num_trips <= 3 AND discounted_booking_rate >= 1 THEN 'Access to Special
Booking Discounts'
     WHEN avg_total_sales >= 1500 AND travel_lead_time >= 9 THEN 'Extended
Cancellation Window at No Additional Cost'
                  WHEN num_flight_booked >= 3 AND total_checked_bags >1 THEN
'Additional Free Checked Baggage Allowance'
     WHEN avg_cust_lifespan >0.60 AND num_trips >= 3 THEN 'Complimentary Room or
Seat Upgrade on Next Booking'
     WHEN active_days >=150 AND cancellation_rate <= 0.10 THEN 'Extended Booking
Flexibility Options'
     WHEN num_hotel_booked >= 1  AND avg_hotel_stay >= 2 THEN 'Free Meal Included
With Hotel Booking'
     WHEN avg_page_clicks >= 4 AND booking_rate < 3 THEN '10% Discount on Next
Confirmed Booking'
     END AS perks_offered
FROM final_metrics
)
SELECT *
FROM perks_assignment;
```