```
-- Final SQL for TravelTide project

WITH sessions_2023 AS (
  SELECT *
  FROM sessions
  WHERE session_start > '2023-01-04'
),
filtered_users AS (
  SELECT user_id, COUNT(*)
  FROM sessions_2023 s
  GROUP BY user_id
  HAVING COUNT(*) > 7
),

 session_base AS (
  SELECT
     s.session_id,
     s.user_id, s.trip_id, s.session_start, s.session_end,
                CASE WHEN s.page_clicks > 100 THEN 18 ELSE page_clicks END AS
Page_clicks_fixed, -- To remove outliners by using the ave 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_fixed, --
change -2, -1, 0 to 1 nights at hotel
     CASE WHEN h.rooms = 0 THEN 1 ELSE h.rooms END AS rooms_fixed, -- to
remove the zero rooms without deleting any rows
     h.check_in_time, h.check_out_time, h.hotel_per_room_usd,
        EXTRACT(EPOCH FROM session_end-session_start) as session_duration,
    CASE WHEN s.flight_booked = TRUE AND s.flight_discount_amount > 0
THEN 1 END AS flight_discount_used,-- flag to dermine if discount is effective
at sessions level
        CASE WHEN s.hotel_booked = TRUE AND s.hotel_discount_amount > 0
THEN 1 END AS hotel_discount_used     -- flag to dermine if discount is
effective at sessions level
  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) AND s.cancellation
```

```
IS FALSE -- removes any events that was cancelled
  ),

user_base_session AS ( -- This CTE aims to create additional features for users
SELECT
  user_id,
  SUM(Page_clicks_fixed) AS num_clicks, -- This feature determines the total
number of click per user
  COUNT(DISTINCT session_id) AS user_visit_frequency, -- This feature counts
how often the client visit teh site
  ROUND(AVG(session_duration),2) AS avg_session_duration,
  ROUND(SUM(session_duration),0) AS session_duration_total,
  CURRENT_DATE - MAX(session_start) AS days_since_last_session,
  CURRENT_DATE - MAX(sign_up_date) AS membership_days,
  ROUND(COUNT(CASE WHEN Page_clicks_fixed < 7 THEN 1 END)* 1.0 /
COUNT(DISTINCT session_id),2) AS bounce_rate, -- at what rate does users
leave the site witjout booking
  ROUND(AVG(
    CASE
      WHEN EXTRACT(EPOCH FROM (session_end - session_start)) > 0
      THEN Page_clicks_fixed / NULLIF(EXTRACT(EPOCH FROM (session_end -
session_start)) / 60, 0)
      ELSE 0
    END
  ),2) AS avg_click_rate
FROM session_base
GROUP BY user_id
),
--SELECT * FROM user_base_session -- TEST

user_base_trips AS ( -- This CTE provides metrics on booking behavour
  SELECT
        user_id,
              COUNT(DISTINCT trip_id) AS repeat_booking_count, -- How much
does the user book a trip.
              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, -- count the trips
        ROUND(AVG(checked_bags),2) AS AVG_num_bags, -- Avg number of
bags booked
        ROUND(AVG(seats),2) AS num_seats_booked, -- this can tell us if he is
travlleingwith family or solo
        CAST(AVG(haversine_distance(home_airport_lat, home_airport_lon,
destination_airport_lat, destination_airport_lon)) AS NUMERIC(10,2)) AS
avg_km_flown,
              ROUND(AVG(EXTRACT(DAY FROM departure_time-session_end)),2)
AS time_after_booking_days,
```

```
   ROUND(COUNT(CASE WHEN flight_booked = TRUE THEN 1 END) * 1.0 /
COUNT(DISTINCT session_id),2) AS flight_booking_rate, -- how ofter did uer
book a flight when visiting the site
       ROUND(COUNT(CASE WHEN hotel_booked = TRUE THEN 1 END) * 1.0 /
COUNT(DISTINCT session_id),2) AS hotel_booking_rate, -- how ofter does
user book hotel when visiting the site
       COUNT(flight_discount_used) AS flight_discount_effectiveness, -- how
effective was the discount
   COUNT(hotel_discount_used) AS hotel_discount_effectiveness -- how
effective was the discount

FROM session_base
GROUP BY user_id
),

user_base_revenue AS (
  SELECT
       sb.user_id,
       ROUND(COALESCE((SUM((hotel_per_room_usd * nights_fixed *
rooms_fixed) * (1 - (CASE WHEN hotel_discount_amount IS NULL THEN 0 ELSE
hotel_discount_amount END)))),0),2) AS money_spend_hotel,
            ROUND(SUM(base_fare_usd * num_flights) +
SUM(hotel_per_room_usd * nights_fixed * rooms_fixed * (1 -
COALESCE(hotel_discount_amount, 0))),2) AS total_spent
       FROM user_base_trips ubt
       LEFT JOIN session_base sb ON ubt.user_id = sb.user_id
       GROUP BY sb.user_id
),

user_travel_type AS ( -- GROUPING OF USERS
 SELECT
  sb.user_id,
  CASE --Solo, Family, Business, Couple, Weekenders, Groups
   WHEN COALESCE(AVG(seats), 0) = 1  AND AVG(nights_fixed) > 2 THEN
'Solo'
   WHEN COALESCE(AVG(seats), 0) = 1 AND AVG(nights_fixed) <= 2 THEN
'Business'
   WHEN COALESCE(AVG(seats), 0) = 2 THEN 'Couple'
   WHEN COALESCE(AVG(seats), 0) > 1 AND BOOL_OR(has_children) THEN
'Family'
   WHEN COALESCE(AVG(seats), 0) > 2 AND NOT BOOL_OR(has_children)
THEN 'Groups'
     ELSE 'Local'
            END AS travel_type,
   CASE
        WHEN COALESCE(SUM(ubr.total_spent), 0) /
NULLIF(SUM(ubt.repeat_booking_count), 0) > 5001 THEN
```

```
'high_value_Customer'
          WHEN COALESCE(SUM(ubr.total_spent), 0) /
NULLIF(SUM(ubt.repeat_booking_count), 0) < 5000 THEN
'low_value_Customer'
          ELSE 'No_trips_booked' -- gather information on the amount pent by a
client per trip.
    END AS customer_value,
              AVG(repeat_booking_count) AS avg_trips
FROM session_base sb
JOIN  user_base_revenue ubr ON sb.user_id = ubr.user_id
JOIN        user_base_trips ubt ON sb.user_id = ubt.user_id
GROUP BY sb.user_id

),
user_perks AS ( -- assigning perks to each group created
  SELECT
  user_id,
  CASE
        WHEN customer_value = 'No_trips_booked' THEN '30% off first travel'
        WHEN travel_type = 'Solo' AND avg_trips  > 5 THEN 'Free upgade to
premium class ticket'
        WHEN travel_type = 'Solo' AND avg_trips <= 5 THEN '10% Discount on
next trip'
        WHEN travel_type = 'Business' AND avg_trips  > 4 THEN 'Free ugrade to
business class ticket'
        WHEN travel_type = 'Business' AND avg_trips <= 4 THEN 'Free Priority
Boarding'
        WHEN travel_type = 'Couple' THEN 'Free hotel night'
        WHEN travel_type = 'Family' THEN 'Free Cancelation' -- and high value
client
        WHEN travel_type = 'Groups' AND customer_value =
'low_value_Customer' THEN 'Discount at special events'
        WHEN travel_type = 'Groups' AND customer_value =
'high_value_Customer' THEN 'Free chacked bag'
  ELSE 'Free Meal' END AS user_perks
  FROM user_travel_type
 -- GROUP BY user_id
),

final_table As (
SELECT
        EXTRACT(YEAR FROM AGE(u.birthdate)) AS age, u.gender, u.married,
u.has_children, u.home_country, u.home_city, u.home_airport,
  u.sign_up_date,
        ubs.user_id,
  ubs.num_clicks,
  ubs.user_visit_frequency,
```

```
  ubs.avg_session_duration,
  ubs.session_duration_total,
  ubs.days_since_last_session,
  ubs.membership_days,
  ubs.bounce_rate,
  ubs.avg_click_rate,
  ubt.repeat_booking_count,
  ubt.num_flights,
  ubt.AVG_num_bags,
  ubt.num_seats_booked,
  ubt.avg_km_flown,
  ubt.time_after_booking_days,
  ubt.flight_booking_rate,
  ubt.hotel_booking_rate,
  ubt.flight_discount_effectiveness,
  ubt.hotel_discount_effectiveness,
  ubr.money_spend_hotel,
  ubr.total_spent,
  utt.customer_value, -- value spend per transaction
  utt.Travel_type,
  up.user_perks
FROM user_base_session ubs
LEFT JOIN users u ON ubs.user_id = u.user_id
LEFT JOIN user_base_trips ubt ON ubs.user_id = ubt.user_id
LEFT JOIN user_base_revenue ubr ON ubs.user_id = ubr.user_id
LEFT JOIN user_travel_type utt ON ubs.user_id = utt.user_id
LEFT JOIN user_perks up ON ubs.user_id = up.user_id
)


SELECT * FROM final_table
;
```