**Travel Tide Project**

**Introduction**

I’ll walk you through a comprehensive analysis we’ve conducted to better understand our users' behaviours and preferences in 2023. Our goal was to gather actionable insights from user sessions, flight bookings, hotel stays, and browsing patterns. This analysis will help us tailor our marketing strategies and enhance our customer engagement.

**1. Session Data Filtering**

We start by isolating sessions from January 4, 2023, onward. This gives us a focused dataset of recent user activity.

《

WITH sessions\_2023 AS (

SELECT \*

FROM sessions s

WHERE s.session\_start >= '2023-01-04'

)

》

**2. Identifying Active Users**

Next, we filter out users who have been highly active, specifically those who have more than 7 sessions. This helps us concentrate on engaged users whose behavior is more relevant for deeper analysis.

《sql

filtered\_users AS (

SELECT

user\_id, COUNT(\*) AS session\_count

FROM sessions\_2023

GROUP BY user\_id

HAVING COUNT(\*) > 7

)

》

**3. Analyzing Flight Metrics**

For these active users, we compute various flight-related metrics:

- Discount Flight Proportion: Measures the ratio of flights booked with a discount.

- Average Flight Discount: Shows the average discount amount received.

- Average Dollars Saved per Kilometre: Indicates how much users save per kilometre travelled.

- Average Checked Bags: Reflects the average number of checked bags per flight.

- Total Free Checked Bags: Counts the flights with free checked bags.

- Total Distance Traveled: Sum of distances traveled.

- Total Discounted Trips: Number of discounted flights.

- Total Flights Booked: Total flights booked.

《

flight\_metrics AS (

SELECT

s.user\_id,

-- Ratio of flights booked with discount

SUM(CASE WHEN s.flight\_discount = TRUE THEN 1 ELSE 0 END)::FLOAT / NULLIF(COUNT(s.trip\_id), 0) AS discount\_flight\_proportion,

-- Average flight discount amount

AVG(s.flight\_discount\_amount) AS average\_flight\_discount,

-- Average dollars saved per kilometer

SUM(s.flight\_discount\_amount \* f.base\_fare\_usd) / NULLIF(SUM(haversine\_distance(u.home\_airport\_lat, u.home\_airport\_lon, f.destination\_airport\_lat, f.destination\_airport\_lon)), 0) AS ADS\_per\_km,

-- Average number of checked bags

AVG(f.checked\_bags) AS avg\_checked\_bags,

-- Total free checked bags

SUM(CASE WHEN f.checked\_bags > 0 THEN 1 ELSE 0 END) AS total\_free\_checked\_bag,

-- Total distance travelled

SUM(haversine\_distance(u.home\_airport\_lat, u.home\_airport\_lon, f.destination\_airport\_lat, f.destination\_airport\_lon)) AS total\_distance\_traveled,

-- Total discounted trips

SUM(CASE WHEN s.flight\_discount = TRUE THEN 1 ELSE 0 END) AS total\_discounted\_trips,

-- Total flights booked

SUM(CASE WHEN s.flight\_booked = TRUE THEN 1 ELSE 0 END) AS total\_flight\_booked,

AVG(f.seats) AS avg\_seats\_booked

FROM sessions\_2023 s

LEFT JOIN flights f ON s.trip\_id = f.trip\_id

JOIN users u ON s.user\_id = u.user\_id

WHERE s.user\_id IN (SELECT user\_id FROM filtered\_users)

GROUP BY s.user\_id

),

》

**4. Browsing Behaviour Metrics**

We also evaluate how users interact with our platform:

- Average Session Time: Duration of user sessions.

- Total Page Clicks: Total clicks made during sessions.

- Conversion Rate: Rate at which sessions lead to flight bookings.

《

browsing\_metrics AS (

SELECT

s.user\_id,

AVG(s.session\_end - s.session\_start) AS avg\_session\_time,

SUM(s.page\_clicks) AS total\_page\_clicks,

AVG(CASE WHEN s.flight\_booked = TRUE THEN 1.0 ELSE 0.0 END) AS conversion\_rate

FROM sessions\_2023 s

WHERE s.user\_id IN (SELECT user\_id FROM filtered\_users)

GROUP BY s.user\_id

),

》

---

**5. Hotel Booking Metrics**

For hotel bookings, we analyse:

- Total Hotel Spending: Amount spent on hotels.

- Average Cost per Night: Average price per night.

- Average Rooms Booked: Average number of rooms booked.

- Total Nights: Total nights booked.

《

hotel\_metrics AS (

SELECT

s.user\_id,

SUM((CASE WHEN h.nights <= 0 THEN 1 ELSE h.nights END) \* h.rooms \* h.hotel\_per\_room\_usd) AS total\_hotel\_spending,

AVG((CASE WHEN h.nights <= 0 THEN 1 ELSE h.nights END) \* h.hotel\_per\_room\_usd) AS avg\_cost\_per\_night,

AVG(h.rooms) AS avg\_rooms\_booked,

SUM(CASE WHEN h.nights <= 0 THEN 1 ELSE h.nights END) AS total\_nights

FROM sessions\_2023 s

LEFT JOIN hotels h ON s.trip\_id = h.trip\_id

WHERE s.user\_id IN (SELECT user\_id FROM filtered\_users)

GROUP BY s.user\_id

),

》

---

**6. Perk Preferences Summary**

We summarise user preferences and behaviour related to perks:

- Cancellation Preferences: Number of cancellations.

- Flight and Hotel Discount Preferences: Counts of discounted flights and hotels.

- Frequent Traveller and Discount Lover Flag Flags: indicating frequent travel and discount preferences.

《

perks\_summary AS (

SELECT

s.user\_id,

COUNT(s.trip\_id) AS num\_trip,

SUM(CASE WHEN s.cancellation = 'true' THEN 1 ELSE 0 END) AS cancellation\_pref,

SUM(CASE WHEN s.flight\_discount = 'true' AND s.cancellation = 'false' AND s.flight\_booked = 'true' THEN 1 ELSE 0 END) > 0.5 AS is\_flight\_discount\_lover,

SUM(CASE WHEN s.hotel\_discount = 'true' AND s.cancellation = 'false' AND s.hotel\_booked = 'true' THEN 1 ELSE 0 END) > 0.5 AS is\_hotel\_discount\_lover,

AVG(hm.avg\_cost\_per\_night) AS avg\_cost\_per\_night,

AVG(hm.avg\_rooms\_booked) AS avg\_rooms\_booked,

SUM(fm.total\_free\_checked\_bag) AS total\_free\_checked\_bag,

SUM(fm.total\_distance\_traveled) AS total\_distance\_traveled,

SUM(fm.total\_discounted\_trips) AS total\_discounted\_trips,

SUM(fm.total\_distance\_traveled) / NULLIF(SUM(fm.total\_discounted\_trips), 0) AS avg\_distance\_per\_trip,

COALESCE(SUM(fm.total\_distance\_traveled) / NULLIF(SUM(fm.total\_discounted\_trips), 0) > 4000, FALSE) AS is\_frequent\_traveler,

AVG(fm.avg\_seats\_booked) > 1 AS is\_family\_traveler

FROM sessions\_2023 s

LEFT JOIN flight\_metrics fm ON s.user\_id = fm.user\_id

LEFT JOIN hotel\_metrics hm ON s.user\_id = hm.user\_id

WHERE s.user\_id IN (SELECT user\_id FROM filtered\_users)

GROUP BY s.user\_id

),

》

**7. Combining Metrics**

We merge flight, browsing, and hotel metrics into a single dataset. We use COALESCE to handle any missing values and ensure that our analysis is comprehensive.

combined\_metrics AS (

SELECT

COALESCE(fm.user\_id, bm.user\_id) AS user\_id,

COALESCE(fm.discount\_flight\_proportion, 0) AS discount\_flight\_proportion,

COALESCE(fm.average\_flight\_discount, 0) AS avg\_flight\_discount,

COALESCE(fm.ADS\_per\_km, 0) AS ADS\_flight\_per\_km,

COALESCE(fm.avg\_checked\_bags, 0) AS avg\_checked\_bags,

COALESCE(fm.total\_free\_checked\_bag, 0) AS total\_free\_checked\_bag,

COALESCE(bm.avg\_session\_time, INTERVAL '0 seconds') AS avg\_session\_time,

COALESCE(bm.total\_page\_clicks, 0) AS total\_page\_clicks,

COALESCE(bm.conversion\_rate, 0) AS conversion\_rate,

COALESCE(hm.total\_hotel\_spending, 0) AS total\_hotel\_spending,

COALESCE(hm.avg\_cost\_per\_night, 0) AS avg\_cost\_per\_night,

COALESCE(hm.avg\_rooms\_booked, 0) AS avg\_rooms\_booked,

COALESCE(ps.total\_distance\_traveled, 0) AS total\_distance\_traveled,

COALESCE(ps.total\_discounted\_trips, 0) AS total\_discounted\_trips

FROM flight\_metrics fm

FULL OUTER JOIN browsing\_metrics bm ON fm.user\_id = bm.user\_id

FULL OUTER JOIN hotel\_metrics hm ON fm.user\_id = hm.user\_id

LEFT JOIN perks\_summary ps ON fm.user\_id = ps.user\_id

),

**8. Normalising Metrics**

To ensure comparability, we scale the metrics to a 0-1 range. This allows us to compare different metrics on a similar scale and helps in accurate indexing.

《scaled\_metrics AS (

SELECT

user\_id,

discount\_flight\_proportion, -- Ensure this column is included

CASE WHEN (MAX(discount\_flight\_proportion) OVER() - MIN(discount\_flight\_proportion) OVER()) != 0

THEN (discount\_flight\_proportion - MIN(discount\_flight\_proportion) OVER()) / (MAX(discount\_flight\_proportion) OVER() - MIN(discount\_flight\_proportion) OVER())

ELSE 0 END AS scaled\_discount\_flight\_proportion,

CASE WHEN (MAX(avg\_flight\_discount) OVER() - MIN(avg\_flight\_discount) OVER()) != 0

THEN (avg\_flight\_discount - MIN(avg\_flight\_discount) OVER()) / (MAX(avg\_flight\_discount) OVER() - MIN(avg\_flight\_discount) OVER())

ELSE 0 END AS scaled\_avg\_flight\_discount,

CASE WHEN (MAX(ADS\_flight\_per\_km) OVER() - MIN(ADS\_flight\_per\_km) OVER()) != 0

THEN (ADS\_flight\_per\_km - MIN(ADS\_flight\_per\_km) OVER()) / (MAX(ADS\_flight\_per\_km) OVER() - MIN(ADS\_flight\_per\_km) OVER())

ELSE 0 END AS scaled\_ADS\_flight\_per\_km,

CASE WHEN (MAX(avg\_checked\_bags) OVER() - MIN(avg\_checked\_bags) OVER()) != 0

THEN (avg\_checked\_bags - MIN(avg\_checked\_bags) OVER()) / (MAX(avg\_checked\_bags) OVER() - MIN(avg\_checked\_bags) OVER())

ELSE 0 END AS scaled\_avg\_checked\_bags,

CASE WHEN (MAX(total\_free\_checked\_bag) OVER() - MIN(total\_free\_checked\_bag) OVER()) != 0

THEN (total\_free\_checked\_bag - MIN(total\_free\_checked\_bag) OVER()) / (MAX(total\_free\_checked\_bag) OVER() - MIN(total\_free\_checked\_bag) OVER())

ELSE 0 END AS scaled\_total\_free\_checked\_bag,

CASE WHEN (MAX(avg\_cost\_per\_night) OVER() - MIN(avg\_cost\_per\_night) OVER()) != 0

THEN (avg\_cost\_per\_night - MIN(avg\_cost\_per\_night) OVER()) / (MAX(avg\_cost\_per\_night) OVER() - MIN(avg\_cost\_per\_night) OVER())

ELSE 0 END AS scaled\_avg\_cost\_per\_night,

CASE WHEN (MAX(avg\_rooms\_booked) OVER() - MIN(avg\_rooms\_booked) OVER()) != 0

THEN (avg\_rooms\_booked - MIN(avg\_rooms\_booked) OVER()) / (MAX(avg\_rooms\_booked) OVER() - MIN(avg\_rooms\_booked) OVER())

ELSE 0 END AS scaled\_avg\_rooms\_booked,

CASE WHEN EXTRACT(EPOCH FROM (MAX(avg\_session\_time) OVER() - MIN(avg\_session\_time) OVER())) != 0

THEN (EXTRACT(EPOCH FROM avg\_session\_time) - EXTRACT(EPOCH FROM MIN(avg\_session\_time) OVER())) / EXTRACT(EPOCH FROM (MAX(avg\_session\_time) OVER() - MIN(avg\_session\_time) OVER()))

ELSE 0 END AS scaled\_avg\_session\_time,

CASE WHEN (MAX(total\_page\_clicks) OVER() - MIN(total\_page\_clicks) OVER()) != 0

THEN (total\_page\_clicks - MIN(total\_page\_clicks) OVER()) / (MAX(total\_page\_clicks) OVER() - MIN(total\_page\_clicks) OVER())

ELSE 0 END AS scaled\_total\_page\_clicks,

CASE WHEN (MAX(conversion\_rate) OVER() - MIN(conversion\_rate) OVER()) != 0

THEN (conversion\_rate - MIN(conversion\_rate) OVER()) / (MAX(conversion\_rate) OVER() - MIN(conversion\_rate) OVER())

ELSE 0 END AS scaled\_conversion\_rate

FROM combined\_metrics

),

》

**9. Calculating Composite Indices**

We create several indices to provide a clearer picture of user behaviour:

- Bargain Hunter Index: Reflects how much users hunt for bargains.

- Flight Index: Composite measure of flight-related behaviour.

- Browsing Index: Composite measure of browsing activity.

- Hotel Index: Composite measure of hotel-related behavior.

《final\_metrics AS (

SELECT

sm.user\_id,

-- Bargain Hunter Index

(sm.scaled\_discount\_flight\_proportion \* sm.scaled\_avg\_flight\_discount \* sm.scaled\_ADS\_flight\_per\_km) AS bargain\_hunter\_index,

-- Flight Index

(sm.scaled\_discount\_flight\_proportion + sm.scaled\_avg\_flight\_discount + sm.scaled\_ADS\_flight\_per\_km + sm.scaled\_avg\_checked\_bags) AS flight\_index,

-- Browsing Index

(sm.scaled\_avg\_session\_time + sm.scaled\_total\_page\_clicks + sm.scaled\_conversion\_rate) AS browsing\_index,

-- Hotel Index

(sm.scaled\_avg\_cost\_per\_night + sm.scaled\_avg\_rooms\_booked) AS hotel\_index

FROM scaled\_metrics sm

),

》

**10. Classifying Users**

Finally, we classify users based on their behaviour and preferences into categories such as 'Luxury Traveller', 'Bargain Hunter', and 'Frequent Traveller'. This classification helps us tailor our strategies and offerings.

《

combined\_final AS (

SELECT

fm.user\_id,

fm.bargain\_hunter\_index,

fm.flight\_index,

fm.browsing\_index,

fm.hotel\_index,

cm.discount\_flight\_proportion,

hm.total\_hotel\_spending,

ps.cancellation\_pref,

ps.avg\_cost\_per\_night,

ps.avg\_rooms\_booked,

ps.total\_free\_checked\_bag,

ps.is\_frequent\_traveler,

ps.is\_flight\_discount\_lover,

ps.is\_hotel\_discount\_lover,

CASE

WHEN (ps.cancellation\_pref >= 1) THEN 'Free Cancellation'

WHEN ps.is\_frequent\_traveler THEN 'Free Hotel Night with Flight'

WHEN ps.is\_flight\_discount\_lover > ps.is\_hotel\_discount\_lover THEN 'Exclusive Flight Discount'

WHEN ps.is\_hotel\_discount\_lover THEN 'Exclusive Hotel Discount'

WHEN ps.total\_free\_checked\_bag >= 8 THEN 'Free Checked Bags'

ELSE 'Luxury Hotel'

END AS preferred\_perk,

CASE

WHEN( hm.total\_hotel\_spending > 5000 OR num\_trip > 7) THEN 'Luxury Traveler'

WHEN fm.bargain\_hunter\_index > 1 THEN 'Bargain Hunter'

WHEN ps.is\_frequent\_traveler THEN 'Frequent Traveler'

WHEN ps.is\_family\_traveler THEN 'Family Traveler'

WHEN (ps.is\_flight\_discount\_lover OR ps.is\_hotel\_discount\_lover) THEN 'Exclusive Discount Lover'

ELSE 'Comfort Seeker'

END AS user\_group\_classification

FROM final\_metrics fm

LEFT JOIN hotel\_metrics hm ON fm.user\_id = hm.user\_id

LEFT JOIN perks\_summary ps ON fm.user\_id = ps.user\_id

LEFT JOIN combined\_metrics cm ON fm.user\_id = cm.user\_id

LEFT JOIN browsing\_metrics bm ON ps.user\_id = bm.user\_id

)

SELECT

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,

cf.\*

FROM combined\_final cf

LEFT JOIN users u ON cf.user\_id = u.user\_id;

》

**Recommendations**

1. Tailor Marketing Campaigns: Use the insights from the indices and user classifications to develop targeted marketing campaigns. For instance, offer exclusive deals to ‘Bargain Hunters’ and premium experiences to ‘Luxury Travelers’.

2. Enhance Personalization: Improve user experience by personalising offers based on user preferences, such as providing more flight discounts to users who are identified as ‘Flight Discount Lovers’.

3.Optimize Resource Allocation: Focus on frequent travellers and high spenders in your loyalty programs. Offer incentives to these users to increase their engagement and satisfaction.

4. Monitor and Iterate: Regularly review and update the metrics and classifications to adapt to changing user behaviours and market conditions. This will ensure that your strategies remain relevant and effective.

5.Expand Analysis: Consider integrating additional data sources, such as user feedback or external market trends, to further enrich your analysis and improve the accuracy of user classifications.