Business Traval Analytics

Table of Contents

Introduction

The travel industry generates an enormous amount of data each day, especially from flight bookings, hotel reservations, and customer interactions across different platforms. Understanding this data is essential for identifying travel patterns, customer preferences, pricing trends, and overall booking behaviour. Such insights help travel businesses improve their services, optimise pricing strategies, and better understand the needs of different customer groups.

This project explores a comprehensive travel dataset that includes flight details, hotel information, and user demographics. By examining this dataset, the analysis aims to uncover meaningful patterns such as popular travel routes, hotel booking behaviour, spending differences, and demographic trends. The goal is to transform raw travel data into clear, valuable insights that can support decision-making within the travel and hospitality industry.

Problem Statement

The travel industry produces large volumes of data from flight bookings, hotel reservations, and customer demographics. However, these datasets are often stored separately, making it difficult to gain a complete understanding of traveller behaviour. Without combining and analysing these different sources of information, important insights such as spending patterns, booking preferences, and customer segments.

The challenge addressed in this project is to identify meaningful patterns within a combined travel dataset and understand how different factors, such as flight choices, hotel bookings, and user demographics, influence overall travel behaviour. By analysing these patterns, the project aims to provide clearer insights that can support decision-making in areas such as pricing strategy, customer targeting, and travel service planning.

Objectives

1. To prepare and manage the dataset using Python-based ETL
2. To perform data analysis using PostgreSQL and SQL queries
3. To visualise key findings through an interactive Power BI dashboard

Methodology

Step1: Data Collection

The data understanding phase was conducted using Python to comprehensively assess the structure and characteristics of the three datasets.

Step2: Data Understanding

The flight booking dataset 'flights.csv' contains 271,888 complete entries across 10 columns, occupying 20.7 MB of memory. It features integer identifiers 'travelCode', 'userCode', textual route details 'from', 'to', categorical flight classifications 'flightType', and numerical metrics for pricing, duration, and distance 'price', 'time', 'distance', alongside agency names and date information stored as text.

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 271888 entries, 0 to 271887
Data columns (total 10 columns):
 #   Column      Non-Null Count   Dtype
---  ------      --------------   -----
 0   travelCode  271888 non-null  int64
 1   userCode    271888 non-null  int64
 2   from        271888 non-null  object
 3   to          271888 non-null  object
 4   flightType  271888 non-null  object
 5   price       271888 non-null  float64
 6   time        271888 non-null  float64
 7   distance    271888 non-null  float64
 8   agency      271888 non-null  object
 9   date        271888 non-null  object
dtypes: float64(3), int64(2), object(5)
memory usage: 20.7+ MB
None
```

Fig. 1.flight.csv

The hotel reservation dataset ‘hotels.csv’ comprises 40,552 fully populated entries with 8 columns, utilizing 2.5 MB of memory. It shares common identifiers with the flight data ‘travelCode’, ‘userCode’ and includes textual hotel details ‘name’, ‘place’, integer stay duration ‘days’, float-based cost metrics ‘price’, ‘total’, and ‘date’ fields.

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 40552 entries, 0 to 40551
Data columns (total 8 columns):
 #   Column      Non-Null Count  Dtype
---  ------      --------------  -----
 0   travelCode  40552 non-null  int64
 1   userCode    40552 non-null  int64
 2   name        40552 non-null  object
 3   place       40552 non-null  object
 4   days        40552 non-null  int64
 5   price       40552 non-null  float64
 6   total       40552 non-null  float64
 7   date        40552 non-null  object
dtypes: float64(2), int64(3), object(3)
memory usage: 2.5+ MB
None
```

Fig. 2.hotels.csv

The user dataset ‘users.csv’ contains 1,340 entries across 5 columns with 52.5 KB memory, featuring integer user codes 'code’, textual company affiliations 'company', and personal attributes 'name', 'gender', 'age'. All datasets demonstrated complete data integrity with zero missing values, confirming their readiness for integrated analysis. The consistent presence of 'travelCode' and 'userCode' across files enabled reliable cross-referencing of flight bookings, hotel stays, and user profiles during subsequent processing stages.

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1340 entries, 0 to 1339
Data columns (total 5 columns):
 #   Column   Non-Null Count  Dtype
---  ------   --------------  -----
 0   code     1340 non-null   int64
 1   company  1340 non-null   object
 2   name     1340 non-null   object
 3   gender   1340 non-null   object
 4   age      1340 non-null   int64
dtypes: int64(2), object(3)
memory usage: 52.5+ KB
None
```

Fig. 3.users.csv

Step3: Data preprocessing

Before merge the dataset, for dataset ‘users.csv’ column’s name ‘code’ will be renamed as ‘userCode’.

The data preprocessing phase involved integrating user, flight, and hotel records into a unified dataset using Python's pandas library. The three source files—'users.csv', 'flights.csv', and 'hotels.csv'—were loaded into separate DataFrames. These were then merged sequentially using left joins on two key identifiers:

i. Primary Merge: Flight and hotel data were combined via 'travelCode' and 'userCode', preserving all flight records with 271,888 entries while linking hotel bookings where available.

ii. Secondary Merge: The resulting DataFrame was merged with user demographic data using 'userCode', maintaining full flight record integrity.

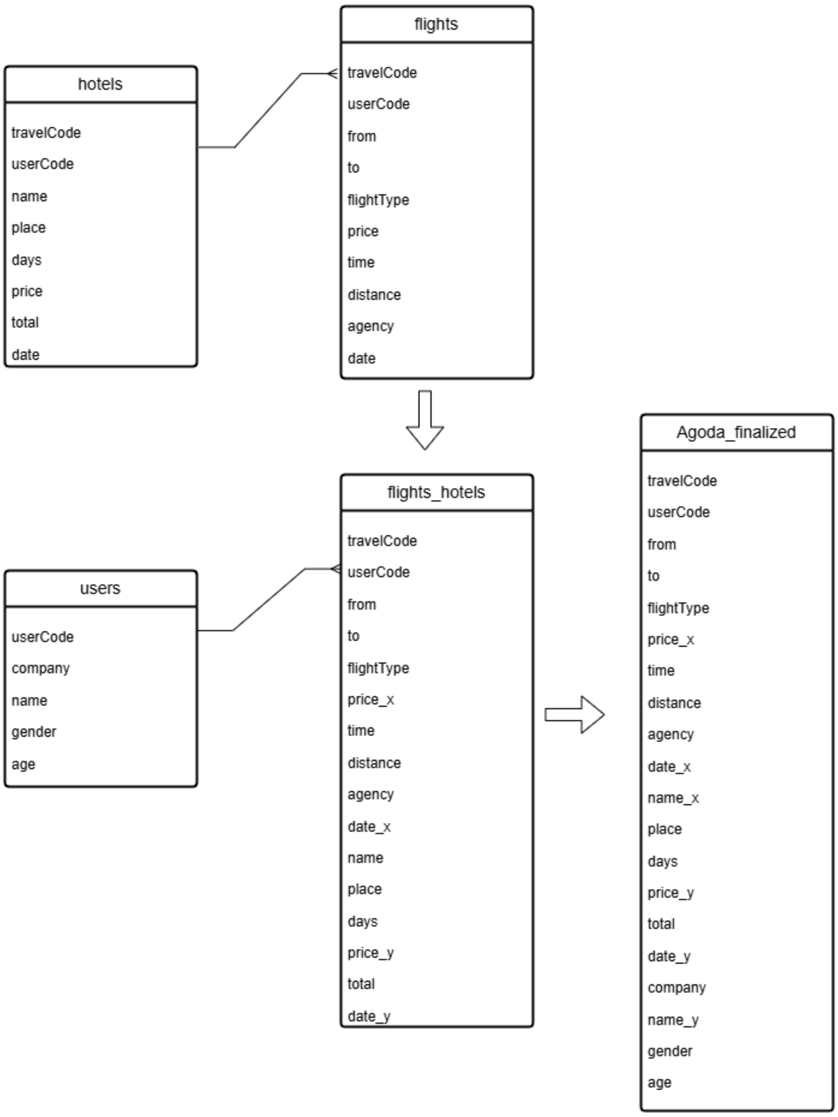


Fig. 4. Two-stage join strategy from flights and hotels data

This two-stage join strategy ensured no flight data loss while enriching records with optional hotel details such as only 81,104 of 271,888 flights had linked hotels and user profiles. The final consolidated dataset was exported to 'Agoda_merged.csv' for downstream analysis.

```
df_u = df_u.rename(columns={"code":"userCode"})
df_merge = pd.merge(df_f, df_h, on=["travelCode", "userCode"], how='left')
df_final = pd.merge(df_merge, df_u, on="userCode", how='left')
df_final.head()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 271888 entries, 0 to 271887
Data columns (total 20 columns):
 #   Column      Non-Null Count   Dtype
---  ------      --------------   -----
 0   travelCode  271888 non-null  int64
 1   userCode    271888 non-null  int64
 2   from        271888 non-null  object
 3   to          271888 non-null  object
 4   flightType  271888 non-null  object
 5   price_x     271888 non-null  float64
 6   time        271888 non-null  float64
 7   distance    271888 non-null  float64
 8   agency      271888 non-null  object
 9   date_x      271888 non-null  object
 10  name_x      81104 non-null   object
 11  place       81104 non-null   object
 12  days        81104 non-null   float64
 13  price_y     81104 non-null   float64
 14  total       81104 non-null   float64
 15  date_y      81104 non-null   object
 16  company     271888 non-null  object
 17  name_y      271888 non-null  object
 18  gender      271888 non-null  object
 19  age         271888 non-null  int64
dtypes: float64(6), int64(3), object(11)
memory usage: 41.5+ MB
Final information:
 None
```

Fig. 5. The variables and its data type after merged

Rename the specified variables and make it be more clearly

```
df_final.rename(columns={'price_x':'price_flight', 'date_x':'date_flight', 'name_x':'name_hotel', 'price_y':'price_hotel',
                         'total':'total_price', 'date_y':'date_hotel', 'name_y':'name_user'}, inplace=True)
print("Data Information after renaming columns:", df_final.info())
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 271888 entries, 0 to 271887
Data columns (total 20 columns):
 #   Column        Non-Null Count   Dtype
---  ------        --------------   -----
 0   travelCode    271888 non-null  int64
 1   userCode      271888 non-null  int64
 2   from          271888 non-null  object
 3   to            271888 non-null  object
 4   flightType    271888 non-null  object
 5   price_flight  271888 non-null  float64
 6   time          271888 non-null  float64
 7   distance      271888 non-null  float64
 8   agency        271888 non-null  object
 9   date_flight   271888 non-null  object
 10  name_hotel    81104 non-null   object
 11  place         81104 non-null   object
 12  days          81104 non-null   float64
 13  price_hotel   81104 non-null   float64
 14  total_price   81104 non-null   float64
 15  date_hotel    81104 non-null   object
 16  company       271888 non-null  object
 17  name_user     271888 non-null  object
 18  gender        271888 non-null  object
 19  age           271888 non-null  int64
dtypes: float64(6), int64(3), object(11)
memory usage: 41.5+ MB
Data Information after renaming columns: None
```

Fig. 6. Finalized data type of each variable

“Agoda_data.csv” will be the final version of the dataset.

```
df_final.to_csv('./Datasets/Agoda_data.csv', index=False)
print("Agoda_Data.csv has been updated successfully!")
```

Step4: Create Database connection

```
db_parameters = {
    "host": "xxxx",
    "dbname": "xxxx",
    "user": "xxxx",
    "password": "xxxx",
    "port":  xxxx
```

```
def get_connection():
    """Establishes and returns a connection to the PostgreSQL database."""
    try:
        conn = psycopg2.connect(**db_parameters)
        return conn
    except Exception as error:
        print("Error:", error)
        return None
```

```python
def create_sqlalchemy_engine():
    """Creates and returns a SQLAlchemy engine for PostgreSQL"""
    try:
        connection = f"postgresql+psycopg2://{db_parameters['user']}:{db_parameters['password']}@{db_parameters['host']}:{db_parameters['port']}/{db_parameters['dbname']}"
        engine = create_engine(connection)
        return engine
    except Exception as error:
        print("Error:", error)
        return None
```

Step5: Create Table in PostgreSQL

```python
from db_connection import get_connection, create_sqlalchemy_engine

conn = get_connection()
engine = create_sqlalchemy_engine()

if conn is None:
    print("Failed to connect to the Database")

try:
    cur = conn.cursor()
    create_table_travel_data = """
    CREATE TABLE travel_data (
        "travelCode"    BIGINT,
        "userCode"      BIGINT,
        "from"          TEXT,
        "to"            TEXT,
        "flightType"    TEXT,
        "price_flight"  NUMERIC(18,4),
        "time"          NUMERIC(18,4),
        "distance"      NUMERIC(18,4),
        "agency"        TEXT,
        "date_flight"   DATE,
        "name_hotel"    TEXT,
        "place"         TEXT,
        "days"          INTEGER,
        "price_hotel"   NUMERIC(18,4),
        "total_price"   NUMERIC(18,4),
        "date_hotel"    DATE,
        "company"       TEXT,
        "name_user"     TEXT,
        "gender"        TEXT,
        "age"           INTEGER
    );
    """

    cur.execute(create_table_travel_data)
    conn.commit()
    print("Table 'travel_data' created successfully!")

    cur.close()
    conn.close()

except Exception as error:
    print("Error for creating table 'travel_data' :", error)
```

Step6: Load Final Data into PostgreSQL

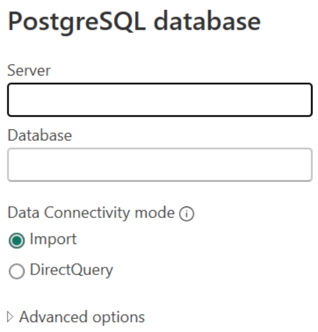
```python
import pandas as pd
from db_connection import get_connection, create_sqlalchemy_engine

conn = get_connection()
engine = create_sqlalchemy_engine()

if conn is None:
    print("Failed to connect to the Database")

try:
    cur = conn.cursor()

    Truncate_table = """
    TRUNCATE TABLE travel_data;
    """

    cur.execute(Truncate_table)
    print("Truncate table data successfully!")
    conn.commit()

    cur.close()
    conn.close()

except Exception as error:
    print("Error for Truncate the table:", error)

df_final = pd.read_csv("./Datasets/Agoda_data.csv")
df_final.to_sql("travel_data", engine, if_exists='append', index=False)
print("Data import into Database successfully!")
```

Step7: Import Data from PostgreSQL into Power BI

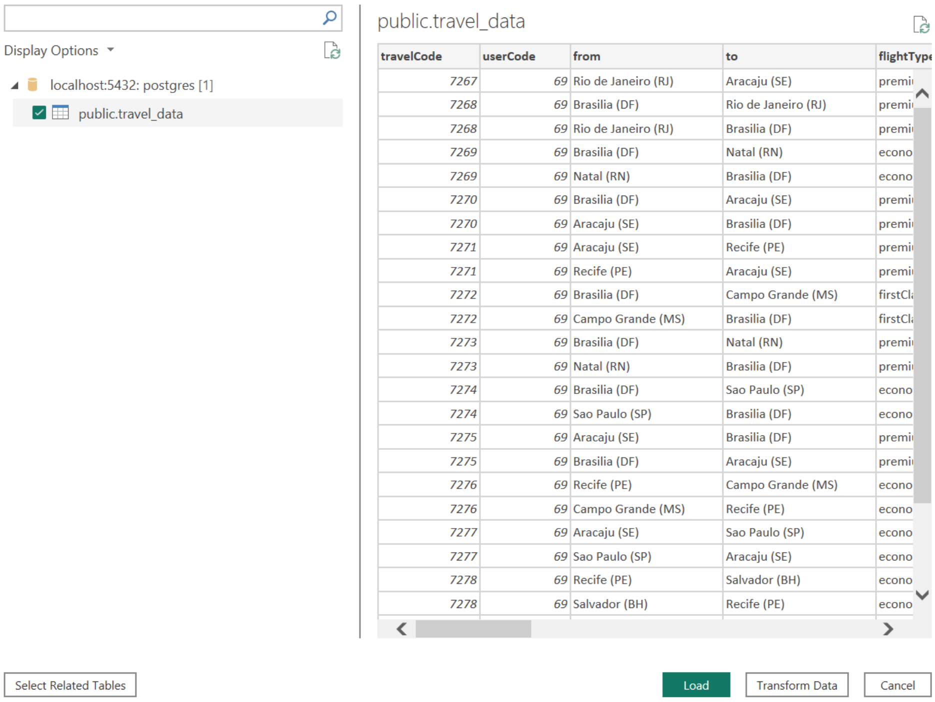
In Power BI,

Home > More > PostgreSQL database

Then fill in the server and Database



Tick the Dataset and then load



travelCode	userCode	from	to	flightType
7267	69	Rio de Janeiro (RJ)	Aracaju (SE)	premi
7268	69	Brasilia (DF)	Rio de Janeiro (RJ)	premi
7268	69	Rio de Janeiro (RJ)	Brasilia (DF)	premi
7269	69	Brasilia (DF)	Natal (RN)	econo
7269	69	Natal (RN)	Brasilia (DF)	econo
7270	69	Brasilia (DF)	Aracaju (SE)	premi
7270	69	Aracaju (SE)	Brasilia (DF)	premi
7271	69	Aracaju (SE)	Recife (PE)	premi
7271	69	Recife (PE)	Aracaju (SE)	premi
7272	69	Brasilia (DF)	Campo Grande (MS)	firstCl
7272	69	Campo Grande (MS)	Brasilia (DF)	firstCl
7273	69	Brasilia (DF)	Natal (RN)	premi
7273	69	Natal (RN)	Brasilia (DF)	premi
7274	69	Brasilia (DF)	Sao Paulo (SP)	econo
7274	69	Sao Paulo (SP)	Brasilia (DF)	econo
7275	69	Aracaju (SE)	Brasilia (DF)	premi
7275	69	Brasilia (DF)	Aracaju (SE)	premi
7276	69	Recife (PE)	Campo Grande (MS)	econo
7276	69	Campo Grande (MS)	Recife (PE)	econo
7277	69	Aracaju (SE)	Sao Paulo (SP)	econo
7277	69	Sao Paulo (SP)	Aracaju (SE)	econo
7278	69	Recife (PE)	Salvador (BH)	econo
7278	69	Salvador (BH)	Recife (PE)	econo

Tools and Technologies Used

1. Python

Python is used to perform the Extract, Transform, and Load (ETL) process. It enables efficient data cleaning, preprocessing, joining of multiple datasets, and formatting the data into a structured form suitable for analysis. Python also provides flexibility in handling missing values, feature engineering, and exporting the processed dataset into PostgreSQL.

2. PostgreSQL

PostgreSQL serves as the primary database system for storing and managing the cleaned data. SQL queries are used to explore travel patterns, analyse customer spending behaviour, and uncover relationships within the dataset. PostgreSQL allows efficient querying, indexing, and organising of large datasets, making it suitable for analytical workloads.

3. Power BI

Power BI is used to visualise the analytical results obtained from PostgreSQL. It provides interactive dashboards and visual representations such as charts, tables, and filters, which help translate raw data insights into meaningful business information. Power BI enhances the presentation of findings and supports clearer interpretation for decision-making.

Data Analysis

Flight Analysis

- Total Flight Price, Average Flight Price and Total Number of Flights Booked.
- Most Popular Travel Routes: Top5 Routes.
- Distribution of flights booked: By flightType and agency.
- Flight Cost Analysis: Total and Average price_flight by flightType, agency, and distance.
- Price Elasticity by Flight Type and Distance: Analyze average price per km by flightType. Detect pricing inefficiencies or premium threshold. Formula: price_per_km = price_flight / distance

Hotel Analysis

- Total Hotel Price, Average Hotel Price and Total number of hotels booked.
- Hotel Booking Rate: How many trips included hotels. Formula = (Total number of travelCode – total number count missing values in name_hotel) / Total number of travelCode * 100.
- Most Popular Hotels & Places: Distribution of name_hotel and place.
- Hotel Stay Duration & Price: Analyze average of days, average of price_hotel and total_price for each of name_hotel.
- Flight-Hotel Bundling: Compare total price with and without hotel bookings. With_hotel mean that total price included the flight_price and total_price of the hotel under the condition of hotel booked, while without_hotel mean that total price of the flight_price under the condition of non-hotel is booked.

Demographic Analysis

- Demographic Pricing Bias: Group by gender or age groups (<25, 25–45, 45+) and compare spending behavior.
- Booking Frequency by Company: This identifies companies with the most flight bookings, hinting at corporate travel partners or repeat customers.

Result and Discussion

Flight Analysis:

KPI: Total Flight Price, Average Flight Price and Total Number of Flights Booked.

```sql
select
round(sum(price_flight),2) as total_flight_price,
round(avg(price_flight),2) as avg_flight_price,
count(*) as total_flight_bookings
from travel_data
where price_flight is not null;
```

	total_flight_price numeric	avg_flight_price numeric	total_flight_bookings bigint
1	260298782.14	957.38	271888

Most Popular Travel Routes: Top5 Routes.

```sql
select "from", "to", count(*) as route_count
from travel_data
group by "from", "to"
order by route_count desc
limit 5;
```

	from text	to text	route_count bigint
1	Florianopolis (SC)	Aracaju (SE)	8643
2	Aracaju (SE)	Florianopolis (SC)	8643
3	Campo Grande (...	Florianopolis (SC)	8253
4	Florianopolis (SC)	Campo Grande (...	8253
5	Brasilia (DF)	Florianopolis (SC)	7779

Distribution of flights booked: flightType and agency.

```sql
SELECT
    "flightType",
    COUNT(*) AS count,
    ROUND(COUNT(*) * 100.0 / t.total_rows, 2) AS percentage
FROM travel_data
JOIN (
    SELECT COUNT(*) AS total_rows
    FROM travel_data
) t ON TRUE
WHERE "flightType" IS NOT NULL
  AND TRIM("flightType") != ''
GROUP BY "flightType", t.total_rows
ORDER BY count DESC;
```

	flightType text	count bigint	percentage numeric
1	firstClass	116418	42.82
2	premium	78004	28.69
3	economic	77466	28.49

```sql
SELECT
    agency,
    COUNT(*) AS count,
    ROUND(COUNT(*) * 100.0 / SUM(COUNT(*)) OVER (), 2) AS percentage
FROM travel_data
WHERE agency IS NOT NULL
  AND TRIM(agency) != ''
GROUP BY agency
ORDER BY count DESC;
```

	agency text	count bigint	percentage numeric
1	Rainbow	116752	42.94
2	CloudFy	116378	42.80
3	FlyingDrops	38758	14.26

Flight Cost Analysis: Total and Average price_flight by flightType, agency, and distance.

```sql
select
"flightType",
round(sum("price_flight"),2) as total_price,
round(avg("price_flight"),2) as average_price
from travel_data
group by "flightType"
order by total_price desc;
```

	flightType text	total_price numeric	average_price numeric
1	firstClass	137497542.67	1181.07
2	premium	71794286.80	920.39
3	economic	51006952.67	658.44

```sql
select
"agency",
round(sum("price_flight"),2) as total_price,
round(avg("price_flight"),2) as average_price
from travel_data
group by "agency"
order by total_price desc;
```

	agency text	total_price numeric	average_price numeric
1	Rainbow	107386243.32	919.78
2	CloudFy	106939334.89	918.90
3	FlyingDrops	45973203.93	1186.16

```
select
"distance",
round(sum("price_flight"),2) as total_price,
round(avg("price_flight"),2) as average_price
from travel_data
group by "distance"
order by total_price desc
limit 5;
```

	distance numeric (18,4)	total_price numeric	average_price numeric
1	808.8500	21424393.20	1239.41
2	637.5600	16699941.72	1073.40
3	676.5300	16207360.32	1065.01
4	709.3700	15925499.04	1186.88
5	937.7700	15639687.90	1348.25

Price Elasticity by Flight Type and Distance: Analyze average price per km by flightType. Detect pricing inefficiencies or premium threshold. Formula: price_per_km = price_flight / distance

```
select
"flightType",
round(avg("price_flight" / "distance"), 2) as avg_price_per_km
from travel_data
group by "flightType"
order by avg_price_per_km desc;
```

	flightType text	avg_price_per_km numeric
1	firstClass	2.34
2	premium	1.82
3	economic	1.32

Key Insights:

The flight data from the Agoda dataset offers some fascinating insights into how customers book their flights and the pricing trends at play. Overall, flight revenue hit around $2.60 billion, coming from 271,888 bookings, with an average ticket price of $957.38. This suggests a pretty lucrative market, likely driven by a good number of premium and first-class travelers.

Looking at the most popular travel routes, it’s clear that many of them are round trips, like the ones between Aracaju (SE) and Florianopolis (SC), as well as Florianopolis (SC) and Campo Grande (MS), each boasting over 8,000 bookings. This trend points to a lot of back-and-forth travel between these regional hubs.

When it comes to flight classes, first-class tickets take the lead, making up 42.82% of all bookings, followed closely by premium at 28.69% and economy at 28.49%. A similar trend is seen among booking agencies, with Rainbow and CloudFly each handling about 43% of the bookings, while FlyingDrops has a smaller slice at 14.26%.

Analyzing costs by flight type reveals that first-class tickets have the highest average price at $1,181.07, while economy class averages $658.44, highlighting a significant price difference likely due to extra services and exclusivity. Among the agencies, FlyingDrops charges the most for an average ticket ($1,186.16), surpassing both Rainbow and CloudFly, which hover just below $920. This indicates that FlyingDrops may be targeting a more upscale market.

When we look at pricing based on distance, there's a clear trend: longer flights generally come with higher total and average prices, with the highest average price ($1,348.25) linked to a distance of 937.77 km. However, if we break it down by price per kilometer, first-class passengers pay about $2.34/km, while premium travelers pay $1.82/km and economy passengers pay $1.32/km. This shows a tiered pricing system that caters to both luxury seekers and budget-conscious travelers.

Hotel Analysis:

KPI: Total Hotel Price, Average Hotel Price and Total number of hotels booked.

```
select
round(sum("total_price")/2,2) as total_hotel_price,
round(sum("total_price") / (count("name_hotel")), 2) as avg_hotel_price,
count("name_hotel") / 2 as total_hotel_bookings
from travel_data;
```

	total_hotel_price numeric	avg_hotel_price numeric	total_hotel_bookings bigint
1	21745179.21	536.23	40552

Hotel Booking Rate: How many trips included hotels. Formula = (Total number of travelCode – total number count missing values in name_hotel) / Total number of travelCode * 100.

```
SELECT
    ROUND(
        COUNT(CASE
                  WHEN "name_hotel" IS NOT NULL
                       AND TRIM("name_hotel") != ''
                  THEN 1
             END) * 100.0 / COUNT(*),
          2
    ) AS hotel_booking_rate
FROM travel_data;
```

	hotel_booking_rate numeric
1	29.83

Most Popular Hotels & Places: Distribution of name_hotel and place.

```
SELECT
    "name_hotel",
    COUNT(*) AS count,
    ROUND(COUNT(*) * 100.0 / SUM(COUNT(*)) OVER(), 2) AS percentage
FROM travel_data
WHERE "name_hotel" IS NOT NULL
  AND TRIM("name_hotel") != ''
GROUP BY "name_hotel"
ORDER BY count DESC;
```

	name_hotel text	count bigint	percentage numeric
1	Hotel K	10188	12.56
2	Hotel CB	10058	12.40
3	Hotel BD	9658	11.91
4	Hotel AF	9656	11.91
5	Hotel AU	8934	11.02
6	Hotel BP	8874	10.94
7	Hotel BW	8666	10.69
8	Hotel Z	8410	10.37
9	Hotel A	6660	8.21

```
SELECT
    "place",
    COUNT(*) AS count,
    ROUND(COUNT(*) * 100.0 / SUM(COUNT(*)) OVER(), 2) AS percentage
FROM travel_data
WHERE "place" IS NOT NULL
  AND TRIM("place") != ''
GROUP BY "place"
ORDER BY count DESC;
```

	place text	count bigint	percentage numeric
1	Salvador (BH)	10188	12.56
2	Rio de Janeiro (RJ)	10058	12.40
3	Natal (RN)	9658	11.91
4	Sao Paulo (SP)	9656	11.91
5	Recife (PE)	8934	11.02
6	Brasilia (DF)	8874	10.94
7	Campo Grande (...	8666	10.69
8	Aracaju (SE)	8410	10.37
9	Florianopolis (SC)	6660	8.21

Hotel Stay Duration & Price: Analyze average of days, average of price_hotel and total_price for each of name_hotel.

```
select
"name_hotel",
round(avg("days"), 2) as average_day,
round(avg("price_hotel"), 2) as average_price_hotel,
round(sum("total_price") / 2, 2) as total_price_hotel
from travel_data
where "name_hotel" is not null and trim("name_hotel") != ''
group by "name_hotel"
order by total_price_hotel desc;
```

	name_hotel text	average_day numeric	average_price_hotel numeric	total_price_hotel numeric
1	Hotel AU	2.52	312.83	3522778.63
2	Hotel K	2.52	263.41	3376652.79
3	Hotel BD	2.50	242.88	2931075.84
4	Hotel BP	2.49	247.62	2731743.84
5	Hotel A	2.48	313.02	2588362.38
6	Hotel Z	2.49	208.04	2179635.08
7	Hotel CB	2.49	165.99	2075206.98
8	Hotel AF	2.51	139.10	1686726.60
9	Hotel BW	2.50	60.39	652997.07

Flight-Hotel Bundling: Compare total price with and without hotel bookings. With_hotel mean that total price included the flight_price and total_price of the hotel under the condition of hotel booked, while without_hotel mean that total price of the flight_price under the condition of non-hotel is booked.

```sql
SELECT
    -- Total spending for trips WITH hotel
    Round(SUM(
        CASE
            WHEN "name_hotel" IS NOT NULL AND TRIM("name_hotel") <> ''
            THEN "price_flight"
            ELSE 0
        END
    )
    +
    (
        SUM(
            CASE
                WHEN "name_hotel" IS NOT NULL AND TRIM("name_hotel") <> ''
                THEN "total_price"
                ELSE 0
            END
        ) / 2
    ), 2) AS total_price_with_hotel,

    -- Total spending for trips WITHOUT hotel
    Round(SUM(
        CASE
            WHEN "name_hotel" IS NULL OR TRIM("name_hotel") = ''
            THEN "price_flight"
            ELSE 0
        END
    ), 2) AS total_price_without_hotel
FROM travel_data;
```

	total_price_with_hotel numeric	total_price_without_hotel numeric
1	99420530.44	182623430.91

Key Insights:

The analysis of hotel data from Agoda provides some interesting insights into how people choose their accommodations. With a whopping 40,552 hotel bookings logged, the total estimated spending on hotels reached around $2.17 billion, which breaks down to an average of $536.23 per booking. These numbers highlight that while accommodations are important, they still take a backseat to flights when it comes to overall travel expenses.

When we look at the hotel booking rate, it comes in at 29.83%. This means that less than a third of the trips in this dataset included hotel stays, hinting that many travelers might have opted to arrange their own places to stay or perhaps traveled without needing overnight accommodations.

In terms of popularity, Hotel K is at the top with 12.56% of all hotel bookings, closely followed by Hotel CB at 12.4% and Hotel BD at 11.91%. This trend is also reflected in where people are staying, with Salvador (BH) and Rio de Janeiro (RJ) being the most popular spots, each accounting for over 12% of hotel stays. This concentration suggests these cities are favored travel destinations, likely attracting more tourists and business visitors.

Looking at how long people stay and what they pay, the average hotel stays hovers around 2.5 days. However, Hotel AU and Hotel A stand out with average daily rates that exceed $300, indicating they offer a premium experience or extra services. On the flip side, Hotel BW caters to budget travelers with the lowest average price at $60.39. Despite the price variations, all the top hotels are raking in significant revenue, with both Hotel K and Hotel AU generating over $3.3 million in total bookings.

Finally, comparing travel costs for trips with and without hotel bookings sheds light on the benefits of bundling. The total cost for trips that included hotel stays is about $99.42 million, while trips without hotel accommodations also represent a substantial amount.

Demographic Analysis:

Demographic Pricing Bias: Group by gender or age groups (<25, 25–45, 45+) and Compare spending behavior.

```
select
"gender",
round(avg("price_flight"), 2) as average_flight_price,
round(avg("total_price"), 2) as average_total_price
from travel_data
group by gender;
```

	gender text	average_flight_price numeric	average_total_price numeric
1	female	956.58	535.57
2	male	960.96	537.34
3	none	954.51	535.77

```
select
    "age_group",
    count(*) as count,
round(avg("total_price"), 2) as average_total_price
from(
    select
        case
            when age < 25 then 'Young'
            when age <= 45 then 'Adult'
            else 'Senior'
        end as age_group,
        "total_price"
    from travel_data
) as t
group by age_group
order by count desc;
```

	age_group text	count bigint	average_total_price numeric
1	Adult	130770	536.61
2	Senior	118046	535.91
3	Young	23072	535.75

Booking Frequency by Company: This identifies companies with the most flight bookings, hinting at corporate travel partners or repeat customers.

```
select
"company",
count("company") as total_bookings,
count(Distinct "userCode") as unique_users
from travel_data
where "company" is not null and trim("company") != ''
group by "company"
order by total_bookings desc;
```

	company text	total_bookings bigint	unique_users bigint
1	4You	92986	452
2	Acme Factory	50944	259
3	Wonka Com...	45882	235
4	Umbrella LTDA	41596	194
5	Monsters CYA	40480	195

Key Insights:

When we dive into the topic of demographic pricing bias, the dataset sheds light on how consumer behavior varies by gender and age. It turns out that, on average, male travelers tend to spend a bit more on flights, with an average ticket costing $960.96, while females spend around $956.58, and those who don't specify their gender spend about $954.51. Interestingly, even with these differences in flight prices, hotel costs remain pretty steady across all gender groups, sitting just above $535. This indicates that while there might be slight variations in flight choices or classes based on gender, overall travel budgets seem to stay pretty balanced.

Looking at age demographics, adults aged 25 to 45 make up the largest share of travelers, with 130,770 bookings. Seniors (45+) follow closely with 118,046 bookings, and young travelers under 25 come in at 23,072. Despite the differences in numbers, the average spending per traveler stays quite consistent across age groups: adults spend about $536.61, seniors $535.91, and young travelers $535.75. This stability suggests that age doesn't play a huge role in travel spending habits when we look at flights and hotels together, hinting at a common approach to travel packages or pricing norms set by companies.

A closer look at company booking patterns reveals some key players that likely represent corporate travel partners or frequent flyer accounts. The company "4You" stands out with a whopping 92,986 total bookings from 452 unique users, indicating that these users travel frequently, possibly for sales or logistics purposes. Other significant companies include Acme Factory with 50,944 bookings and Wonka Company with 45,882 bookings, each having a smaller yet notable user base. These numbers underscore the importance of corporate clients in the overall booking landscape and point to potential opportunities for loyalty programs, tailored packages, or bulk pricing strategies aimed at high-volume business customers.

Recommendation

Recommendations based on Flight Analysis:

- Leverage Popular Routes for Promotions
  Focus marketing efforts on highly trafficked routes such as *Aracaju - Florianopolis* and *Florianopolis - Campo Grande*. Introduce loyalty programs, discounted bundles, or frequent flyer perks on these routes to further encourage bookings and retain regular travelers.
- Optimize Pricing Across Flight Classes
  Since first-class tickets command the highest price per km ($2.34) but still account for the largest share of bookings (42.82%), consider segmenting first-class travelers further to offer tiered luxury experiences. At the same time, enhance visibility and appeal of economic class to attract price-sensitive customers.
- Target High-Value Agencies
  Agencies like Rainbow and CloudFly handle over 85% of bookings combined. Strengthen partnerships with these agencies through exclusive deals, API integrations, or preferred provider programs to maintain and grow booking volume.

Recommendations based on Hotel Analysis:

- Promote Hotel Bundling to Increase Conversion
  With only 29.83% of trips including hotel bookings, there’s significant untapped potential. Introduce “Flight + Hotel” bundled discounts, automated recommendations, or visual cues during the flight checkout process to nudge users into booking hotels alongside flights.
- Personalize Hotel Offers by Destination
  Given the popularity of cities like Salvador, Rio de Janeiro, and Sao Paulo, tailor hotel promotions by location and seasonality. Highlight top-performing hotels like *Hotel K* and *Hotel AU* for these destinations with customer reviews, photos, and time-limited discounts.

- Monetize Hotel Stay Duration
  Since the average stay hovers around 2.5 days, hotels can introduce mid-stay upselling strategies (e.g., late checkout, meal plans, or local tour packages) to drive additional revenue per booking.

Recommendations based on Demographics & Company Insights:

- Segment and Target by Gender & Age
  Although average total spending varies little by gender and age, male users and adults (25–45) have marginally higher flight expenditures. Use this insight to target these groups with premium upgrades, business-class bundles, or travel insurance offers during checkout.
- Develop Corporate Travel Solutions
  Companies like 4You and Acme Factory have high booking volumes per user. Create custom dashboards, invoicing features, and dedicated support channels for corporate clients. Consider introducing tiered corporate accounts with volume-based discounts or early-access features.

Enhance User Retention with Loyalty Programs
With both individual and corporate repeat usage evident, a points-based loyalty program redeemable across flights and hotels could improve retention and increase cross-service uptake.

Data Visualization Interface

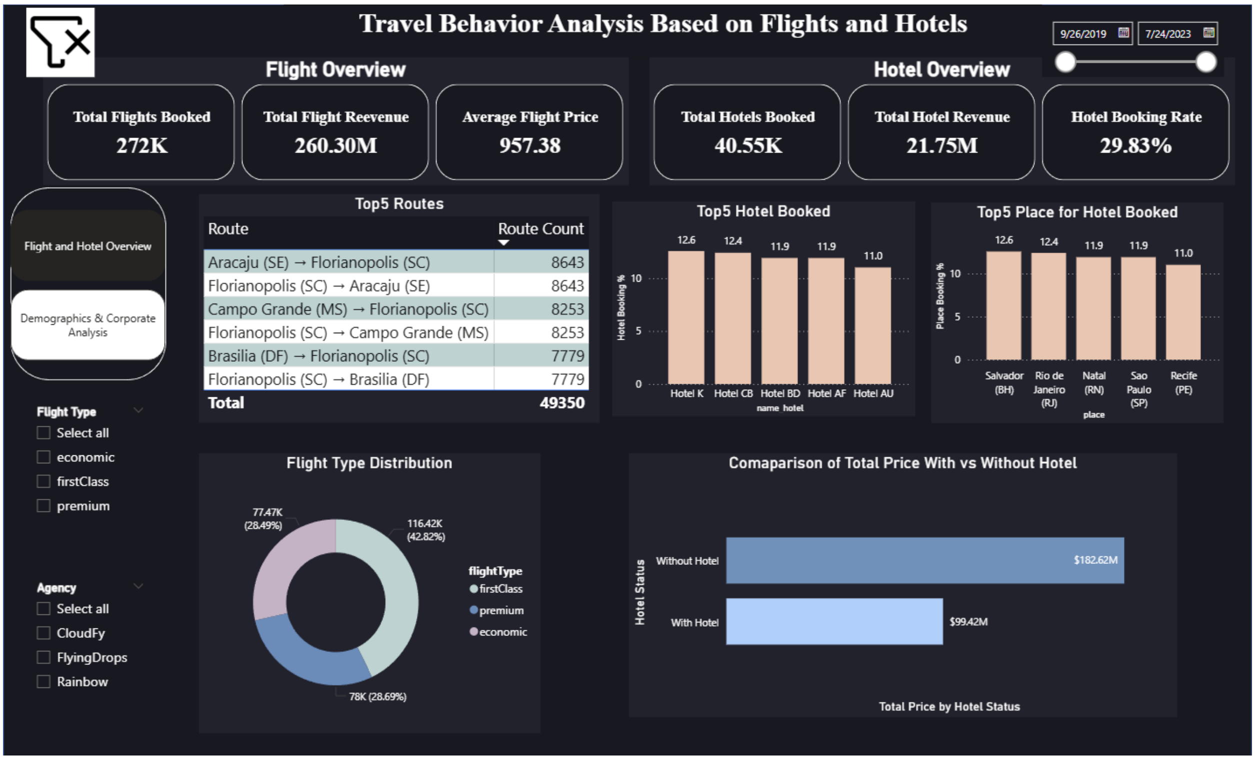


Fig. 7. Flight and Hotel Overview

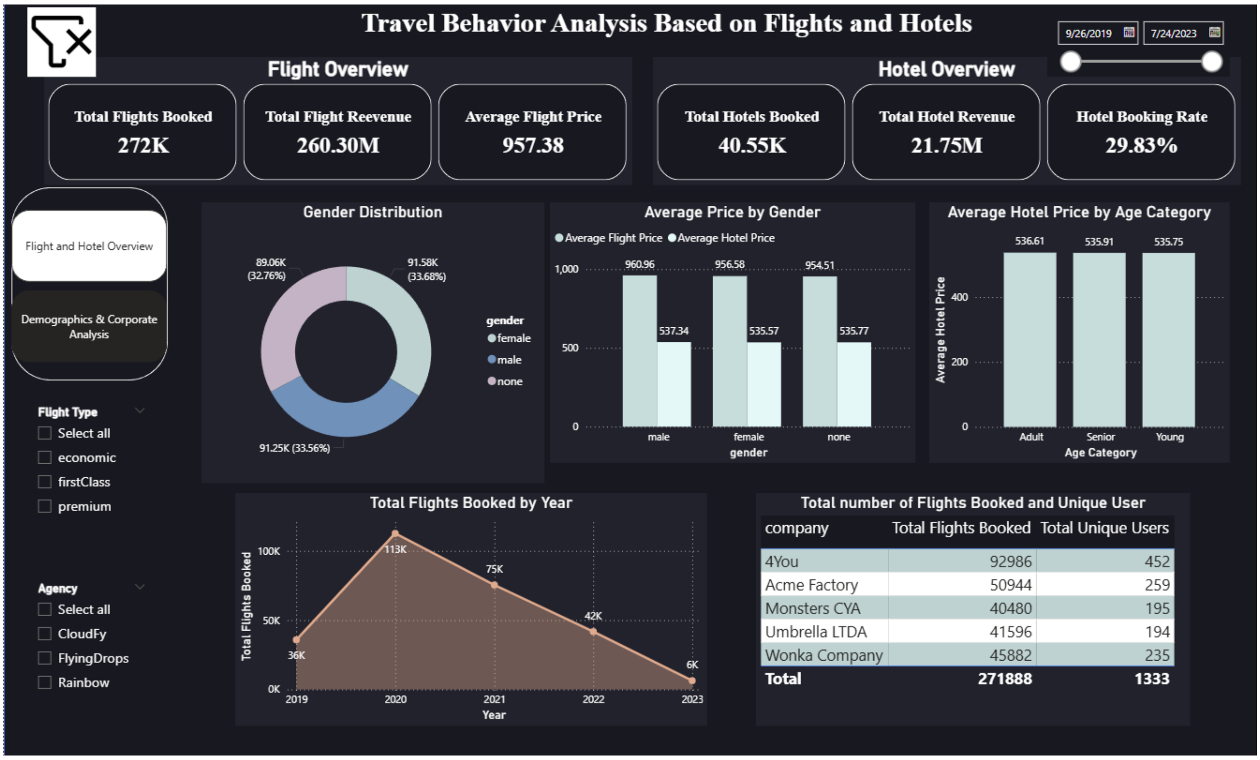


Fig. 8. Demographics and Corporate Analysis

Conclusion

This project brought together flight, hotel, and user data to give a clearer picture of how people travel and where they spend the most. From the analysis, we can see clear booking patterns, popular travel routes, frequently used agencies, and how hotel choices vary among users. The results also show differences between trips with and without hotel bookings, as well as spending behaviours across various demographic groups.

Overall, the project highlights how valuable insights can be uncovered when different travel datasets are combined and analysed together. The use of PostgreSQL for analysis made it easier to explore patterns in a structured way, while Python helped prepare the data for deeper investigation. Once the Power BI dashboard is complete, the insights will be even easier to understand and communicate. This project demonstrates how data can support better decision-making in the travel industry and guide improvements in pricing, marketing, and customer experience.