

**DATA ANALYSIS ON AIR SHUTTLE DATA
USING SQL & PYTHON
DISSERTATION PROJECT**

Submitted by

Hamza

2020-350-020

In partial fulfillment for award of the degree of

**B.TECH COMPUTER SCIENCE AND ENGINEERING ARTIFICIAL
INTELLIGENCE**

under supervision of

DR.IHTIRAM RAZA KHAN



Department Of Computer Science & Engineering

School Of Engineering Science & Technology

JAMIA HAMDARD

(Deemed To Be University)

New Delhi-110062

(2024)



JAMIA HAMDARD

Phone: 011-2605 9588(12

Lines)

Fax: 01-11-25059663

Website: www.jamiamard.edu

Email :inquiry@jamiahamdard.edu

*(Declared as Deemed-to-be University under Section 3 of
the UGC Act, 1956 vide Notification No. F.9-18/85-U.3)*

CERTIFICATE

On the basis of the declaration submitted by **Mr. Hamza (Enrolment No: 2020-350-020)** a student of **Bachelor of Technology(Computer Science & Engineering with Artificial intelligence)**, I hereby certify that the dissertation titled "**Data Analysis On Airlines Data Using SQL & Python**" being submitted to the Department of Computer Science & Engineering, Jamia Hamdard, New Delhi in partial fulfillment of the requirement for the award of the degree of **Bachelor of Technology (Computer Science & Engineering With Artificial Intelligence)**, is carried out by him under my supervision.

Dr.Ihtiram Raza Khan
(Supervisor)

Dr. Farheen Siddiqui
Dean & Head, Department of CSE

Declaration

I, **Hamza** a student of **Bachelor of Technology in Computer Science & Engineering with Artificial Intelligence (B.Tech CSE -AI)**, (Enrolment No: **2020-350-020**) hereby declare that the dissertation entitled “**Data Analysis On Airlines Data Using SQL & Python**” which is being submitted by me to the Department of Computer Science, Jamia Hamdard, New Delhi in partial fulfillment of the requirement for the award of the degree of **Bachelor of Technology (Computer Science & Engineering Artificial Intelligence)**, is my original work and has not been submitted anywhere else for the award of any Degree, Diploma, Associateship, Fellowship or other similar title or recognition.

Signature and Name of the Applicant

Date:

Place:

ACKNOWLEDGEMENT

I express my sincere thanks to **DR.IHTIRAM RAZA KHAN**, my project in charge, who guided me through the project and also gave valuable suggestions and guidance for completing the project. He helped me to understand the **intricate** issues involved in project-making besides effectively presenting it. These intricacies would have been lost otherwise. My project has been a success only because of his guidance.

I would like to express my gratitude towards my parents for their kind cooperation and encouragement, which helped me in the completion of this project.

TABLE OF CONTENTS

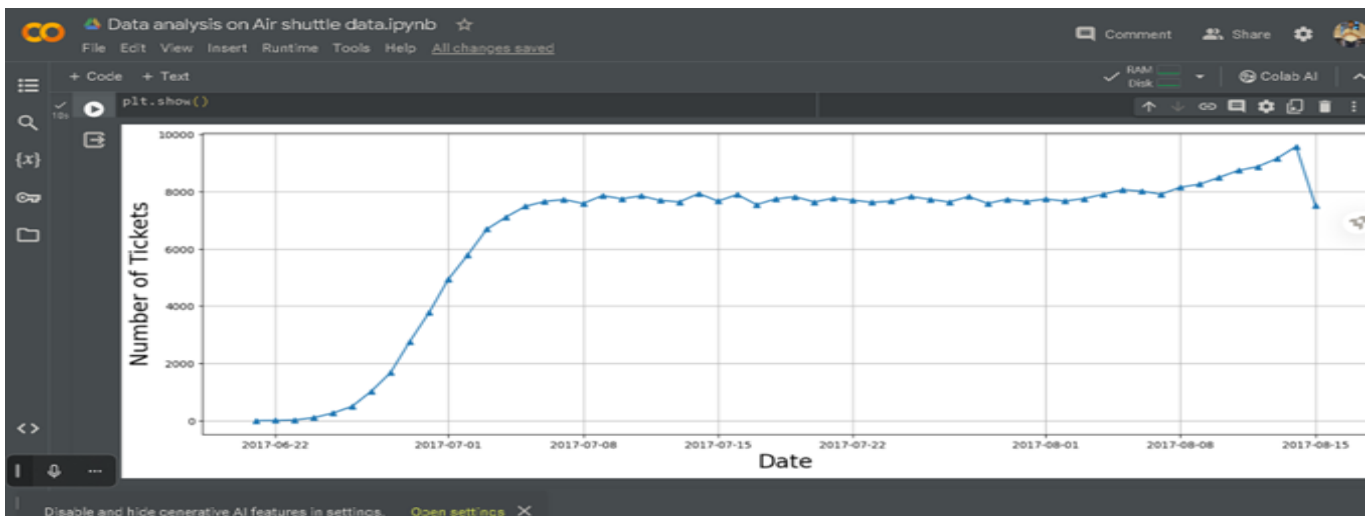
Serial number	TITLE	PAGE NO	Signature
1	Certificate, Declaration & Acknowledgement.	2 to 4	
2	Data Analysis on Airline Data (Complete Overview)	6	
3	Abstract	8	
4	Introduction (Project Overview)	10	
5	Business Problem:	12	
6	Process Flow Diagram	15	
7	Main Challenges	16	
8	Objectives	17	
9	Libraries Used	18	
10	Methodology	23	
	PTO		

11	Exploratory Data Analysis or Data Exploration	25	
12	SQL	32	
13	Data Cleaning	34	
14	Limitations	35	
15	Let Us Start with Data Analysis Now	36	
16	Conclusion/ Result/ Findings	53-54	
17	Future Score	55	
18	Bibliography	56	
19	Useful Links	57	
20	Thank You Note	58	

DATA ANALYSIS ON AIRLINE DATA:

Some highlights which we got during the analysis are the following:

- Upon analysis of the chart, we observe that the number of tickets booked exhibits a **gradual increase from June 22nd to July 7th, followed by a relatively stable pattern from July 8th until August**, with a noticeable peak in ticket bookings where the highest number of tickets were booked on a single day.



So if we **observe the plot** it starts with zero so we can see an **average approx 8000 per day** and we can also see a spike in the month of August approximately 10,000 per day.

→ A SOLUTION TO THE PROBLEM\ RESULT

Seasonal Fluctuations:

Airlines often experience seasonal fluctuations in demand. For example, during off-peak seasons or specific months with lower travel demand, airlines may struggle to fill seats. Factors such as weather conditions, holidays, and school schedules can impact travel patterns, contributing to periods of low occupancy.

Solution: Implement dynamic pricing models that adjust ticket prices based on demand and seasonality. Offer attractive promotions and packages during off-peak seasons to incentivize travel. Explore new routes or destinations that may have higher demand during specific seasons.

So we can get insight from the following chart that there are two aircraft out of nine which does not support business class. And there is one aircraft which has all three fare conditions like business economy and comfort.

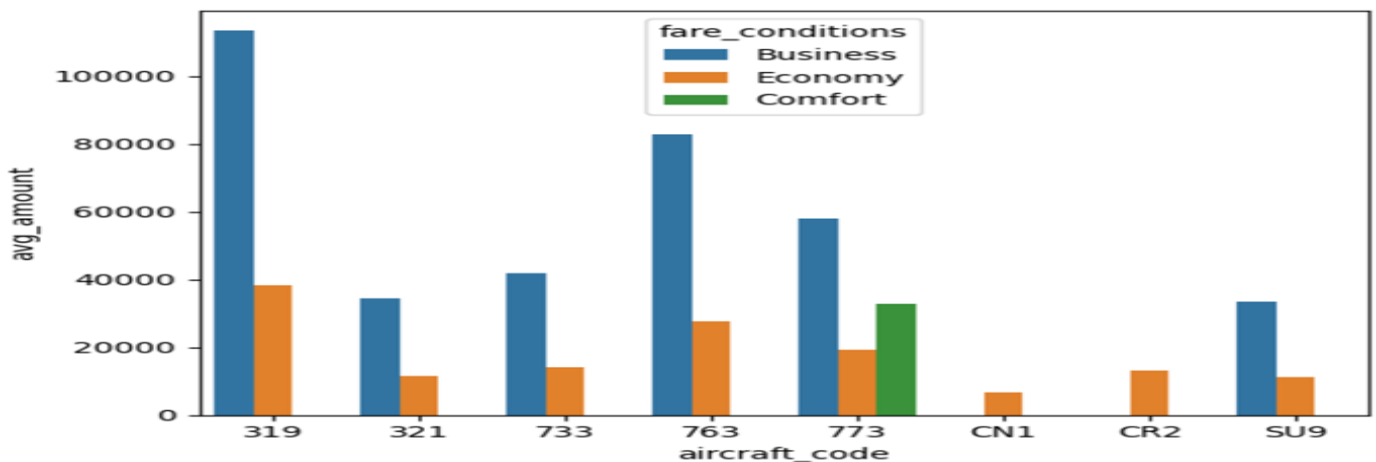


Figure 3

- ✓ It is worth mentioning that the comfort class is available on **only one aircraft, the 773**. The **CN1 and CR2 planes**, on the other hand, only **provide the economy class**.
- ✓ Rest supports only business and economy-class

We can observe there is one craft which has more fare for business and economy class as compared to other

❖Finding/ Insights for airlines company (Output)

Airlines can find areas for improvement and modify their pricing and route plans as a result of assessing these indicators. A greater occupancy rate is one important feature that can enhance profitability since it allows airlines to maximize revenue while minimizing costs associated with vacant seats

ABSTRACT

- In response to the challenges faced by our airline company, operating a diverse fleet of aircraft, this data analysis project aims to address profitability concerns exacerbated by factors such as stricter environmental regulations, higher flight taxes, increased interest rates, rising fuel prices, and a tight labor market leading to elevated labor costs.
- The primary focus is on increasing the company's occupancy rate to boost the average profit earned per seat.
- The main challenges, including environmental regulations, flight taxes, and labor market issues, have prompted the company to set objectives centered around increasing the occupancy rate, improving pricing strategies, and enhancing customer experience.
- The ultimate goal is to identify opportunities for increasing the occupancy rate on underperforming flights, leading to heightened profitability.
- The Starting analysis examines data on planes with more than 100 seats, changes in ticket bookings, total revenue over time, and average fare for each aircraft under different fare conditions.
- Visualization tools such as line charts and bar graphs provide insights into ticket booking trends and average costs associated with different fare conditions.

- The analysis delves into critical metrics, including total revenue, total tickets, average revenue per ticket, and average occupancy per aircraft. These metrics aid in identifying profitable aircraft types and itineraries, guiding decisions on route optimization and resource allocation.
- Examining a 10% increase in occupancy rates demonstrates potential benefits in terms of increased total revenue. The project emphasizes the importance of a balanced approach, considering pricing strategies, operational efficiency, and customer satisfaction.
- So, the data-driven strategy presented in this analysis provides actionable insights for the airline industry. By focusing on increasing occupancy rates, refining pricing strategies, and delivering exceptional customer experiences, airlines can navigate challenges, enhance profitability, and ensure sustainable success in a dynamic and competitive business environment.

Introduction:

→ **Data analysis** is a process that involves inspecting, cleansing, transforming, and modeling data. The goal of data analysis is to discover useful information, draw conclusions, and support decision-making.

→ **Need of data analysis** is a tool that helps people understand data and use it to make informed decisions. Some objectives of data analysis include:

- Identifying trends and patterns
- Making data-driven decisions
- Finding correlations and relationships
- Detecting anomalies
- Improving performance
- Predictive modeling

Project overview:

- ❖ This report **solves the Business problem** and at the end of this file will present analysis in the form of **words** with the help of data analysis.
- ❖ This project is powered by **SQL** and **Python** using them we **fetch** the data.
- ❖ **Different steps** to implement this project are:
 - Understand the business problem
 - Importing dependencies
 - Make a connection with the database to extract the data
 - Then explore and identify the key variables that are present in table
 - Then analyse the key variables like **occupancy rate**
 - Finally **Express the output of analysis in the form of words**
- ❖ So this project is for the airline industry which is facing some **challenges**.
- ❖ Less profit more investment.
- ❖ Some seats are vacant which makes this problem or not sold.
- ❖ So we have to minimize it that is our end goal.
- ❖ The **Solution** is to increase the occupancy rate.

Business Problem:



- Our company operates a diverse fleet of aircraft ranging from small business jets to medium-sized machines.
- We have been providing high-quality air transportation services to our clients for several years, and our primary focus is to ensure a safe, comfortable, and convenient journey for our passengers.
- However, we are currently **facing challenges** due to several factors such as **stricter environmental regulations, higher flight taxes,**

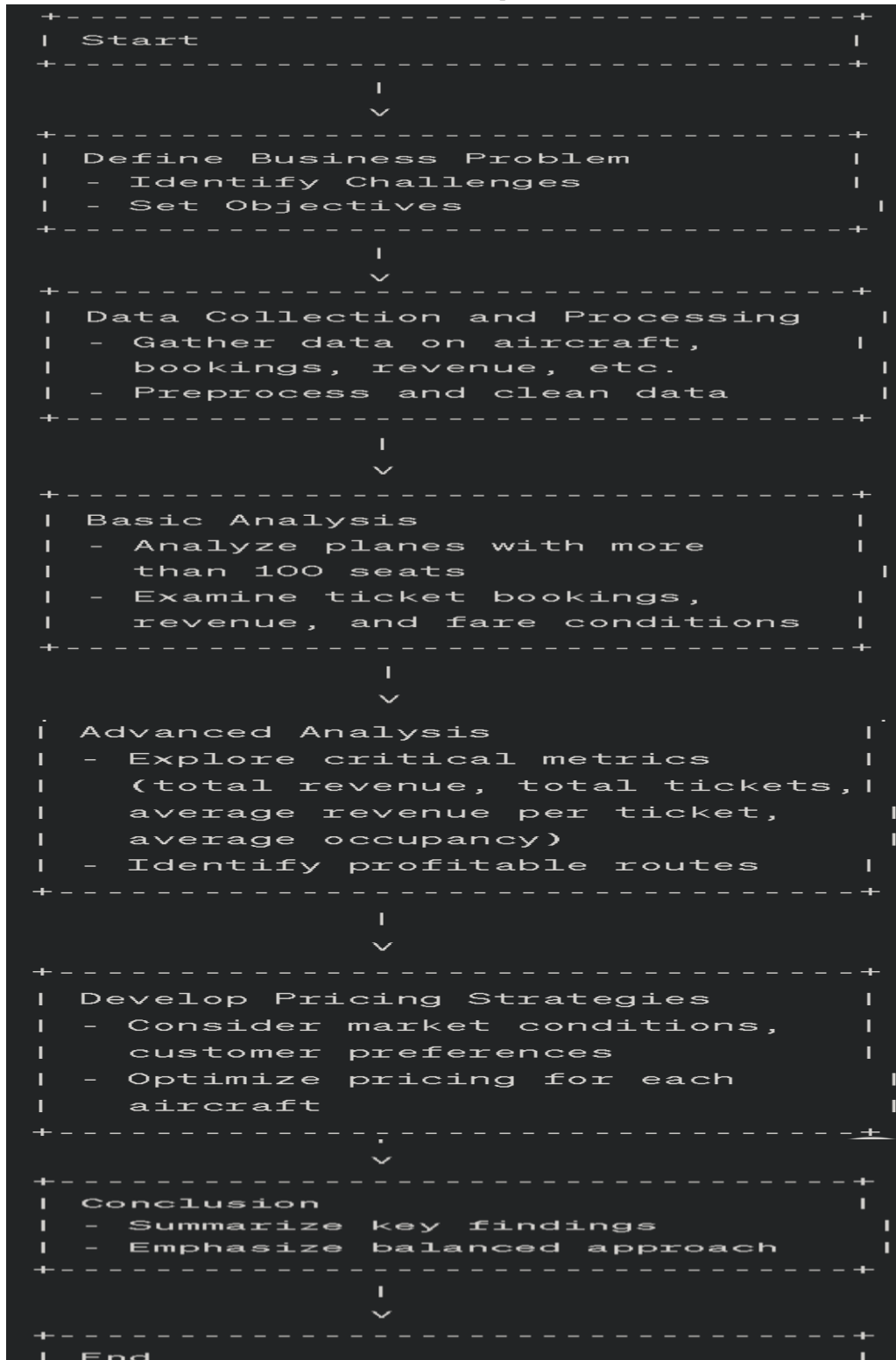
increased interest rates, rising fuel prices, and a tight labor market resulting in increased labor costs.

- As a **result**, the company's profitability is under pressure, and they are **seeking ways to address this issue**.

- To tackle this challenge, they are **looking to conduct an analysis** of their database **to find ways to increase their occupancy rate**, which can help boost the average profit earned per seat.

Process Flow Diagram

Representation of the sequential steps and processes involved in the data analysis.



Main Challenges:

1. **Stricter environmental regulations:** The demand in the airline industry to *decrease its carbon footprint* is growing, which has resulted in more stringent environmental laws that raise operating costs and restrict expansion potential.
2. **Higher flight taxes:** To solve environmental issues and increase money, governments all around the world are taxiing aircraft more heavily, which raises the cost of flying and decreases demand.



3. **Tight labor market resulting in increased labor costs:** The lack of trained people in the aviation sector has increased labor costs and increased turnover rates.

Objectives:

- **Increase occupancy rate:** By increasing the occupancy rate, we can boost the average profit earned per seat and mitigate the impact of the challenges we're facing.
- **Improve pricing strategy:** We need to develop a pricing strategy that takes into account the changing market conditions and customer preferences to attract and retain customers.
- **Enhance customer experience:** We need to focus on providing a seamless and convenient experience for our customers, from booking to arrival, to differentiate ourselves in a highly competitive industry and increase customer loyalty.

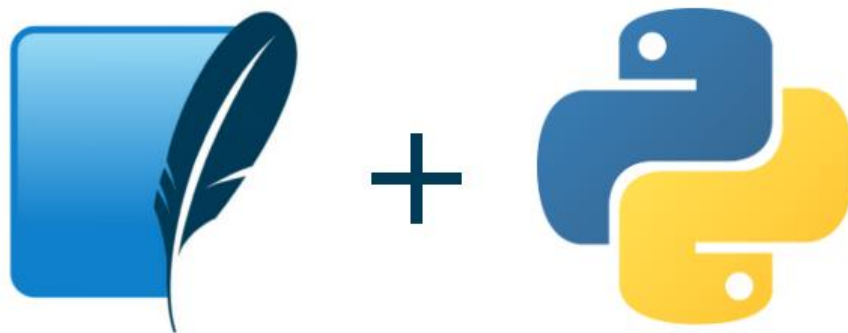
The **end goal** of this task would be to identify opportunities to increase the occupancy rate on low-performing flights, which can ultimately lead to increased profitability for the airline.

Libraries used:

SQLite3:

Creating and accessing databases:

SQLite3 can be used to create and access databases in Python. This can be useful for storing data that your Python application needs to access.



Running SQL queries:

SQLite3 can be used to run SQL queries in Python. This can be useful for retrieving data from a database, or for performing other database operations.

Creating database applications:

SQLite3 can be used to create database applications in Python. This can be useful for creating applications that need to store and retrieve data from a database

Python sqlite3 is an excellent module with which you can perform all possible DB operations with in-memory and persistent database in your applications.

This module implements the Python DB API interface to be a compliant solution for implementing SQL related operations in a program.

Using sqlite3 module

By the help of sqlite3 we can create databases and tables inside it and perform various DB operations on it.

Python SQLite3 module is used to integrate the SQLite database with Python. It is a standardized Python DBI API 2.0 and provides a straightforward and simple-to-use interface for interacting with SQLite databases. There is no need to install this module separately as it comes along with Python after the 2.5x version.

Pandas

Pandas is a Python library used for working with data sets. It has functions for analyzing, cleaning, exploring, and manipulating data.

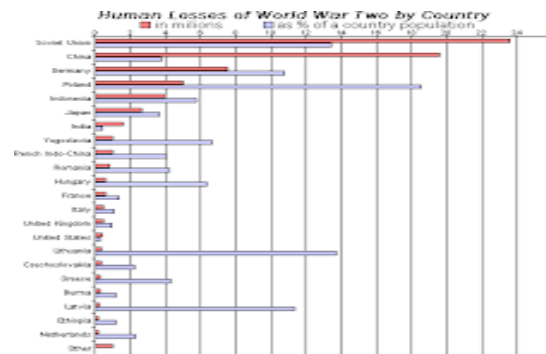
- ✓ Pandas is a powerful, versatile Python library that is used for data analysis. It is built on top of Numpy and provides high-performance, easy-to-use data structures and data analysis tools. Pandas are particularly well-suited for working with tabular data, such as spreadsheets or SQL tables.
- ✓ It uses Pandas for data analysis and data science projects. Pandas DataFrame enables them to manipulate the data and build machine learning models. Although the learning curve is a bit steep, it significantly increases the efficiency of data manipulation.
- ✓ Pandas is a popular open-source data analysis library for Python. It provides powerful data structures and data analysis tools, including data visualization capabilities. Pandas visualization is built on top of the matplotlib library, which provides a wide range of customizable plots.

Matplotlib

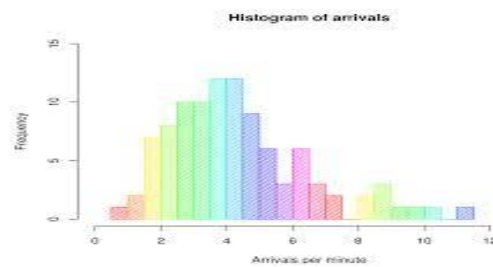
- Matplotlib is used comprehensive library for creating static, animated, and interactive visualizations in Python. Matplotlib makes easy things easy and hard things possible. Create publication quality plots.

- Matplotlib used to create a variety of different types of plots, including line charts,

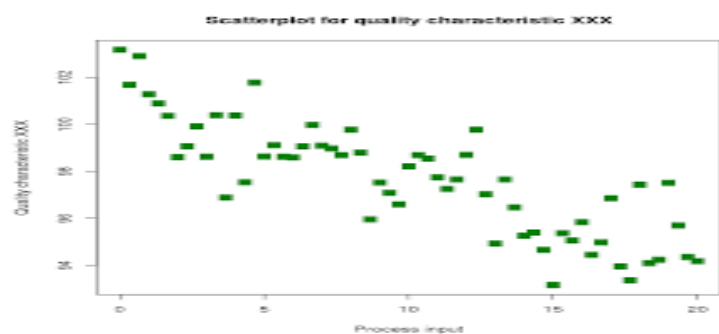
Bar charts



Histograms



scatter plots, and more. Matplotlib is a powerful tool for visualizing data in Python.



Matplotlib is a Python library that is used for data visualization. It is a powerful tool that can be used to create a variety of plots, including line plots, bar charts, histograms, and scatter plots. Matplotlib is often used in data analysis to help identify trends and patterns in data.

- **Identifying trends and patterns:**

Matplotlib can be used to create line plots that show how data changes over time. This can be helpful for identifying trends and patterns in the data. For example, a line plot could be used to show how sales have changed over the past year.

- **Comparing data:**

Matplotlib can be used to create bar charts and histograms to compare different sets of data. This can be helpful for identifying differences between the data sets. For example, a bar chart could be used to compare the sales of different products.

- **Understanding relationships:**

Matplotlib can be used to create scatter plots to show the relationship between two variables. This can be helpful for understanding how the variables are related to each other. For example, a scatter plot could be used to show the relationship between the price of a product and the number of units sold.

- **Matplotlib** is a powerful tool that can be used to create a variety of plots that can be helpful for data analysis. It is a popular tool among data scientists and is used in a variety of industries.

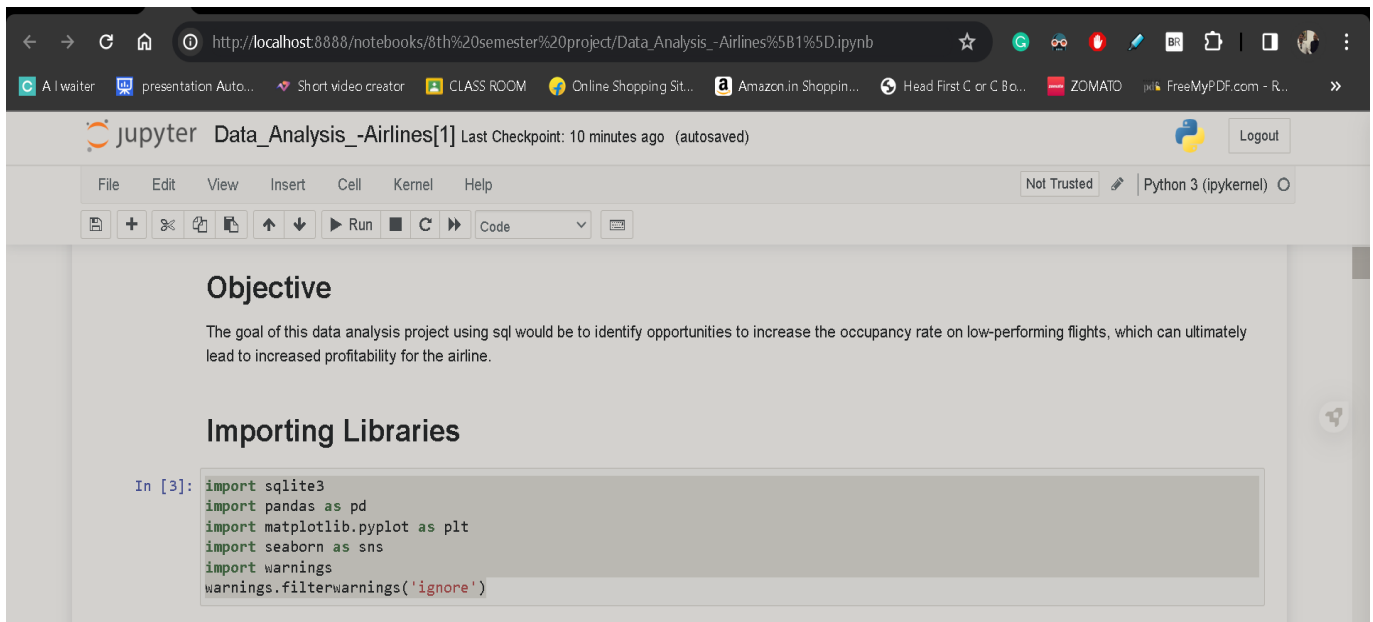
Warnings

- The warnings module in Python provides a way to control how warnings handled within a Python script. It allows developers to emit warning messages to alert users of potential issues or unexpected behavior in their code.
- The warnings module in Python provides a way to handle warnings that are generated by a program. It allows developers to control how warnings are displayed and to take specific actions when warnings occur.
- There are a number of built-in Warning subclasses, and user defined subclasses can also be defined. The default category for warn() is UserWarning.
- To turn off warning messages, you can use the warnings.filterwarnings() function.
- warnings.filterwarnings() function to filter warnings based on their category.
- It is useful to alert the user of some condition in a program, where that condition (normally) doesn't warrant raising an exception and terminating the program. For example, one might want to issue a warning when a program uses an obsolete module.

The warnings module is a powerful tool that can be used to handle warnings in a Python program. By using the warnings module, you can ensure that your program is robust and that warnings are handled in a way that is appropriate for your application.

Methodology:

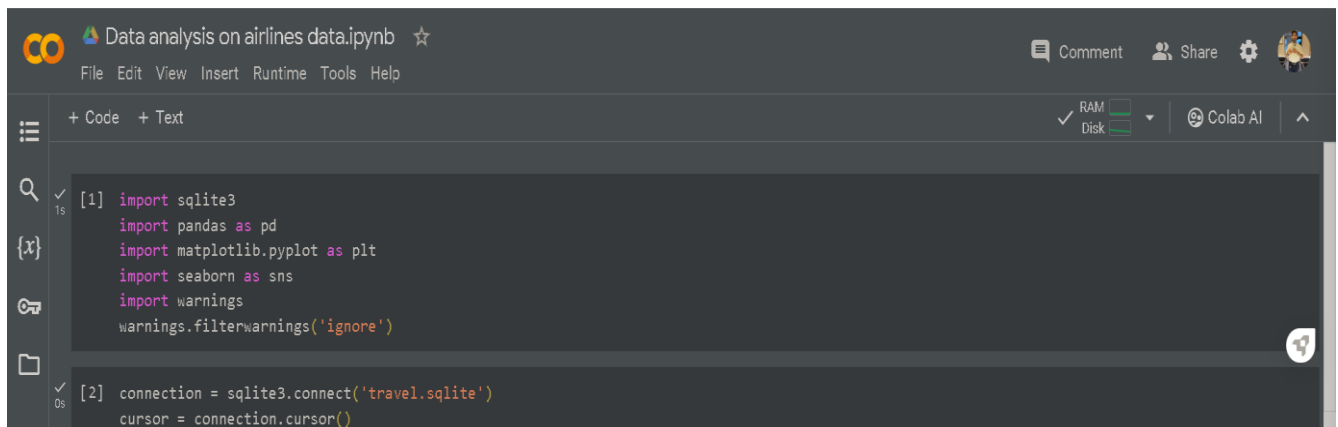
Importing dependencies:



So I'm using two platforms to implement this project first is: **Jupyter** second **Google Colab** using my account & **importing libraries**.

Data connection

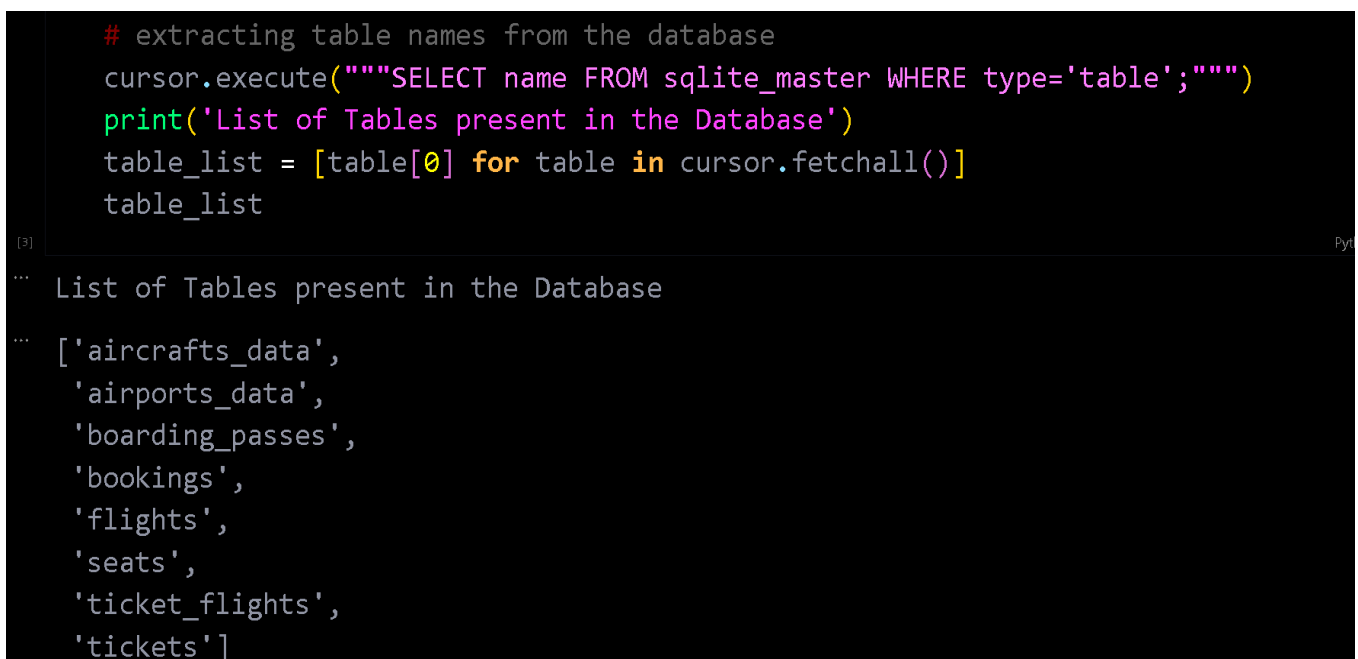
- Establish connection & cursor will help to execute the query.



```
[1] import sqlite3
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings('ignore')

[2] connection = sqlite3.connect('travel.sqlite')
cursor = connection.cursor()
```

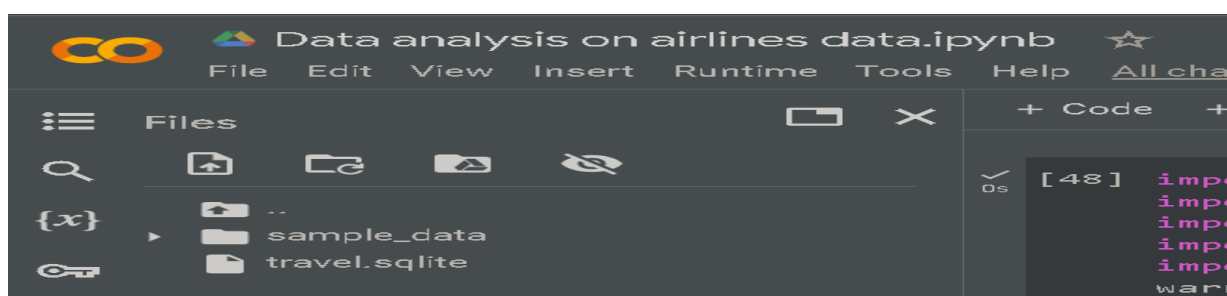
First SQL Query Which will show list of tables By list comprehension



```
# extracting table names from the database
cursor.execute("""SELECT name FROM sqlite_master WHERE type='table';""")
print('List of Tables present in the Database')
table_list = [table[0] for table in cursor.fetchall()]
table_list
```

... List of Tables present in the Database

... ['aircrafts_data',
'airports_data',
'boarding_passes',
'bookings',
'flights',
'seats',
'ticket_flights',
'tickets']



Files

- ..
- sample_data
- travel.sqlite

[48] import

Exploratory Data Analysis Or Data Exploration:

Let's see each column and value.

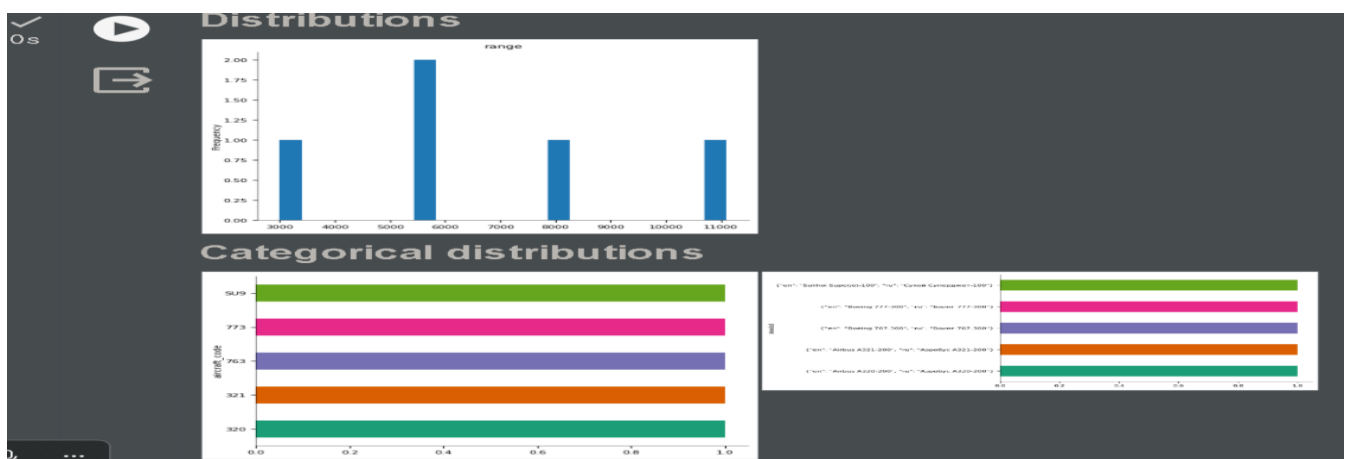
Using pandas will get a result in the form of a Data frame otherwise cursor will give in the form of a list.

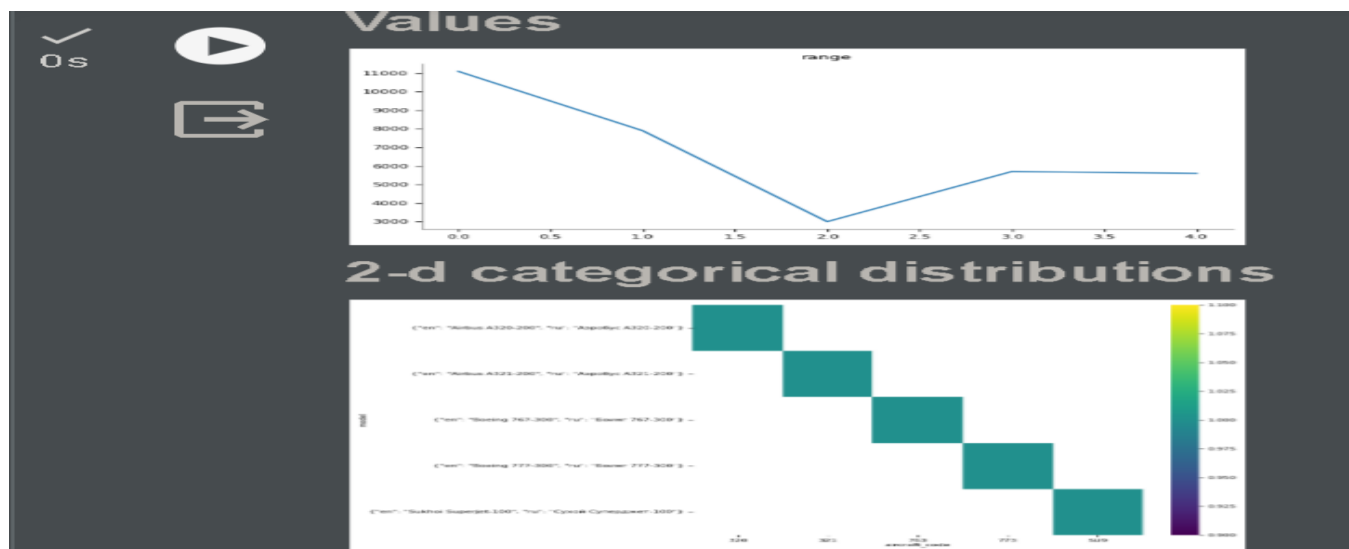
```
[26] aircrafts_data = pd.read_sql_query(f"SELECT * FROM aircrafts_data", connection)
      aircrafts_data.head()
```

index	aircraft_code	model	range
0	773	{"en": "Boeing 777-300", "ru": "Боинг 777-300"}	11100
1	763	{"en": "Boeing 767-300", "ru": "Боинг 767-300"}	7900
2	SU9	{"en": "Sukhoi Superjet-100", "ru": "Сухой Суперджет-100"}	3000
3	320	{"en": "Airbus A320-200", "ru": "Аэробус A320-200"}	5700
4	321	{"en": "Airbus A321-200", "ru": "Аэробус A321-200"}	5600

Show 25 per page

So as we observed from the table there are nine different aircraft with their model range and aircraft code.





These are the few different visualisations to understand the data set further we will see in our data analysis.


Airport Data there are 104 different airports by which aircraft takeoff and land

```
airports_data = pd.read_sql_query(f"SELECT * FROM airports_data", connection)
airports_data.head()
```


	airport_code	airport_name	city	coordinates	timezone
0	YKS	{"en": "Yakutsk Airport", "ru": "Якутск"}	{"en": "Yakutsk", "ru": "Якутск"}	(129.77099609375, 62.0932998657226562)	Asia/Yakutsk
1	MJZ	{"en": "Mirny Airport", "ru": "Мирный"}	{"en": "Mirny", "ru": "Мирный"}	(114.03900146484375, 62.534698486328125)	Asia/Yakutsk
2	KHV	{"en": "Khabarovsk-Novy Airport", "ru": "Хабаровск-Новый"}	{"en": "Khabarovsk", "ru": "Хабаровск"}	(135.18800354004, 48.5279998779300001)	Asia/Vladivostok
3	PKC	{"en": "Yelizovo Airport", "ru": "Елизово"}	{"en": "Petropavlovsk", "ru": "Петропавловск-Камчатский"}	(158.453994750976562, 53.1679000854492188)	Asia/Kamchatka
4	UUS	{"en": "Yuzhno-Sakhalinsk Airport", "ru": "Хомстовец"}	{"en": "Yuzhno-Sakhalinsk", "ru": "Южно-Сахалинск"}	(142.718002319335938, 46.8886985778808594)	Asia/Sakhalin

Boarding pass which is our third data set



✓
1s



```
boarding_passes = pd.read_sql_query(f"""SELECT * FROM boarding_passes""", connection)
boarding_passes.head()
```




	ticket_no	flight_id	boarding_no	seat_no
0	0005435212351	30625	1	2D
1	0005435212386	30625	2	3G
2	0005435212381	30625	3	4H
3	0005432211370	30625	4	5D
4	0005435212357	30625	5	11A





Booking table consist of date and total amount.



✓
1s



```
bookings = pd.read_sql_query(f"""SELECT * FROM bookings """, connection)
bookings.head()
```



	book_ref	book_date	total_amount
0	00000F	2017-07-05 03:12:00+03	265700
1	000012	2017-07-14 09:02:00+03	37900
2	000068	2017-08-15 14:27:00+03	18100
3	000181	2017-08-10 13:28:00+03	131800
4	0002D8	2017-08-07 21:40:00+03	23600

- Here Python will help us to manipulate the data and to bring out the insight from the data whereas SQL helps to establish the data connection because **we are not using CSV files we are extracting data from the database.**

```
flights = pd.read_sql_query(f"""SELECT * FROM flights """, connection)
flights.head()
```

light_no	scheduled_departure	scheduled_arrival	departure_airport	arrival_airport	status	aircraft_code	actual_departure	actual_arrival
PG0134	2017-09-10 09:50:00+03	2017-09-10 14:55:00+03	DME	BTK	Scheduled	319	IN	IN
PG0052	2017-08-25 14:50:00+03	2017-08-25 17:35:00+03	VKO	HMA	Scheduled	CR2	IN	IN
PG0561	2017-09-05 12:30:00+03	2017-09-05 14:15:00+03	VKO	AER	Scheduled	763	IN	IN
PG0529	2017-09-12 09:50:00+03	2017-09-12 11:20:00+03	SVO	UFA	Scheduled	763	IN	IN
PG0461	2017-09-04 12:25:00+03	2017-09-04 13:20:00+03	SVO	ULV	Scheduled	SU9	IN	IN

- In flight table we have flight ID Number schedule of arrival and departure Airport name status and many more
- Another is the **seat table** which consists of aircraft code, seat number and fare conditions.

```
seats = pd.read_sql_query(f"""SELECT * FROM seats """, connection)
seats.head()
```

	aircraft_code	seat_no	fare_conditions
0	319	2A	Business
1	319	2C	Business
2	319	2D	Business
3	319	2F	Business
4	319	3A	Business

- Ticket flights that use a **foreign key** from the flight table ticket-number & amount.

```

ticket_flights = pd.read_sql_query(f"""SELECT * FROM ticket_flights """, connection)
ticket_flights.head()

```

	ticket_no	flight_id	fare_conditions	amount
0	0005432159776	30625	Business	42100
1	0005435212351	30625	Business	42100
2	0005435212386	30625	Business	42100
3	0005435212381	30625	Business	42100
4	0005432211370	30625	Business	42100

- The Tickets table Consist of ticket number passenger ID, Booking reference from **booking table**.

```

✓ 0s tickets = pd.read_sql_query(f"""SELECT * FROM tickets """, connection)
tickets.head()

```

	ticket_no	book_ref	passenger_id
0	0005432000987	06B046	8149 604011
1	0005432000988	06B046	8499 420203
2	0005432000989	E170C3	1011 752484
3	0005432000990	E170C3	4849 400049
4	0005432000991	F313DD	6615 976589

Now we have to **select the column** that can help us in our **further analysis**.

Now let's look at the data type of columns that are present in different tables above we had looked at the table list so we need to Iterate it

PRAGMA query is used to get information about the table.

We used slicing to get the first and second index values.

```
✓ [34] for table in table_list:
0s      print("\ntable: " + table)
      columns_info = connection.execute("PRAGMA table_info({})".format(table))
      for column in columns_info.fetchall():
          print(column[1:3])
```

```
table: aircrafts_data
('aircraft_code', 'character(3)')
('model', 'jsonb')
('range', 'INTEGER')
```

```
table: airports_data
('airport_code', 'character(3)')
('airport_name', 'jsonb')
('city', 'jsonb')
('coordinates', 'point')
('timezone', 'TEXT')
```

```

table: bookings
('book_ref', 'character(6)')
('book_date', 'timestamp with time zone')
('total_amount', 'numeric(10,2)')

table: flights
('flight_id', 'INTEGER')
('flight_no', 'character(6)')
('scheduled_departure', 'timestamp with time zone')
('scheduled_arrival', 'timestamp with time zone')
('departure_airport', 'character(3)')
('arrival_airport', 'character(3)')
('status', 'character varying(20)')
('aircraft_code', 'character(3)')
('actual_departure', 'timestamp with time zone')
('actual_arrival', 'timestamp with time zone')

table: seats
('aircraft_code', 'character(3)')
('seat_no', 'character varying(4)')
('fare_conditions', 'character varying(10)')

table: ticket_flights
... ('ticket_no', 'character(13)')
('flight_id', 'INTEGER')
('flight_no', 'character(6)')
('fare_conditions', 'character varying(10)')
('amount', 'numeric(10,2)')

table: tickets
('ticket_no', 'character(13)')
('book_ref', 'character(6)')
('passenger_id', 'character varying(20)')

```

🌈 So are they integers, numeric, timestamp & character type of columns present in the table.

- **SQL** stands for Structured Query Language. It is a programming language used to store, organize, manage, and manipulate data in relational databases.
- The **GROUP BY** clause in SQL is used to organize related data into groups. It's one of the most commonly used SQL clauses.
- SQL Join statement is used to combine data or rows from two or more tables based on a common field between them.
- The **INNER JOIN** keyword **selects all rows from both the tables** as long as the condition is satisfied.
- This **LEFT JOIN** join returns all the rows of the table on the left side of the join and matches rows for the table on the right side of the join. For the rows for which there is no matching row on the right side, the result-set will contain null. **LEFT JOIN** is also known as **LEFT OUTER JOIN**.
- **RIGHT JOIN** is similar to **LEFT JOIN**. This join **returns all the rows of the table on the right side of the join and matching rows for the table on the left side of the join.** For the rows for which there is no matching row on the left side, the result-set will contain null. **RIGHT JOIN** is also known as **RIGHT OUTER JOIN**.
-

- **FULL JOIN** creates the result-set by **combining results of both LEFT JOIN and RIGHT JOIN**. The result-set will contain all the rows from both tables. For the rows for which there is no matching, the result-set will contain NULL values.

- **Natural join** can join tables based on the common columns in the tables being joined. A natural join returns all rows by matching values in common columns having the same name and data type of columns and that column should be present in both tables.

Data cleaning:

→ Before starting with our analysis we need to check

- Missing value in a table.

```
✓✓ 7s 7s [▶] status 0
[🔍] aircraft_code 0
actual_departure 0
actual_arrival 0
dtype: int64

Missing Values in table seats
aircraft_code 0
seat_no 0
fare_conditions 0
dtype: int64

Missing Values in table ticket_flights
ticket_no 0
flight_id 0
fare_conditions 0
amount 0
dtype: int64

Missing Values in table tickets
ticket_no 0
book_ref 0
passenger_id 0
dtype: int64

ticket_no 0
flight_id 0
boarding_no 0
seat_no 0
```

```
Missing Values in table bookings
book_ref 0
book_date 0
total_amount 0
dtype: int64

Missing Values in table flights
flight_id 0
flight_no 0
scheduled_departure 0
scheduled_arrival 0
departure_airport 0
arrival_airport 0
status 0
aircraft_code 0
actual_departure 0
actual_arrival 0
dtype: int64
```

- So zero signifies that the data is consistent and cleaned so Data cleaning not required.

Limitations

When working on a data analysis project focused on airline data using Python and SQL, it's crucial to acknowledge and address potential limitations.

External Factors:

- **Limitation:** External factors such as economic conditions, geopolitical events, or global health crises (as observed with COVID-19) can significantly impact airline data trends. The project may have limitations in accounting for unforeseen external influences.

Data Quality and Completeness:

- One significant limitation may be the quality and completeness of the airline data. Airlines generate massive amounts of data, but it may contain inconsistencies, missing values, or inaccuracies. Incomplete or inaccurate data can impact the reliability of the analysis and conclusions drawn from it. It's essential to recognize that the analysis is only as good as the data it relies on, and any limitations in data quality may introduce uncertainties in the findings.

LET'S START WITH DATA ANALYSIS NOW

The analysis of data provides insights into the number of planes with more than 100 seats, how the number of tickets booked and the total amount earned changed over time, and the average fare for each aircraft with different fare conditions.

- I will not be using all of the statistics and plots, I will be using flow which can help to insight.
- Let's create three questions to get a basic insight into the data that will help to solve the business problem.
- Let's look at how many planes we have which have more than a hundred seats.
- We can also use pandas by using group- by- function.

But we are using SQL and seats table, So we will get the number of seats per aircraft for all nine aircraft.

Performing analysis to get Insight for the client or business problem

Let look how many aircraft have more than 100 seats?

```
pd.read_sql_query(f"""SELECT aircraft_code, COUNT(*) as num_seats FROM seats
GROUP BY aircraft_code
HAVING num_seats > 100
ORDER BY num_seats DESC""", connection)
```

	aircraft_code	num_seats
0	773	402
1	763	222
2	321	170
3	320	140
4	733	130
5	319	116

- These findings will be useful in developing strategies to increase occupancy rates and optimize pricing for each aircraft. **Table 1** shows the aircraft with more than 100 seats and the actual count of the seats.

Aircraft code	Number of Seats
319	116
320	140
321	170
733	130
763	222
773	402


Table 1

Result six aircraft have more than a hundred seats.

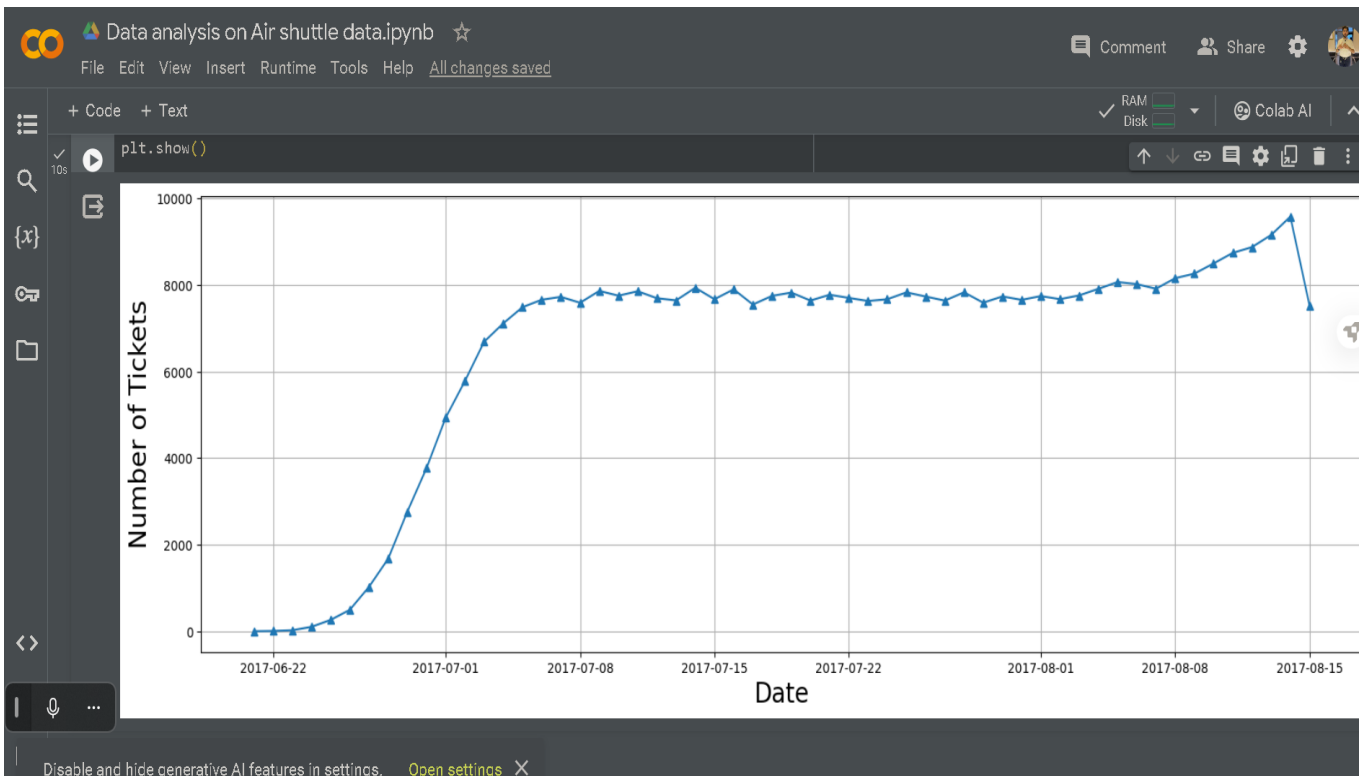
Now let's look at how several tickets are booked and change with the time with the help of a Line plot.

We need to use an inner joint for table Tickets and booking table.

Number of ticket booked and amount earned with time.

```
✓ 10s  tickets = pd.read_sql_query(f"""SELECT *  
                                FROM tickets  
                                INNER JOIN bookings  
                                ON tickets.book_ref=bookings.book_ref;""", connection)  
tickets['book_date'] = pd.to_datetime(tickets['book_date'])  
tickets['date'] = tickets['book_date'].dt.date  
x = tickets.groupby('date')[['date']].count()  
plt.figure(figsize = (18,6))  
plt.plot(x.index,x['date'], marker = '^')  
plt.xlabel('Date', fontsize = 20)  
plt.ylabel('Number of Tickets', fontsize = 20)  
plt.grid('b')  
plt.show()
```

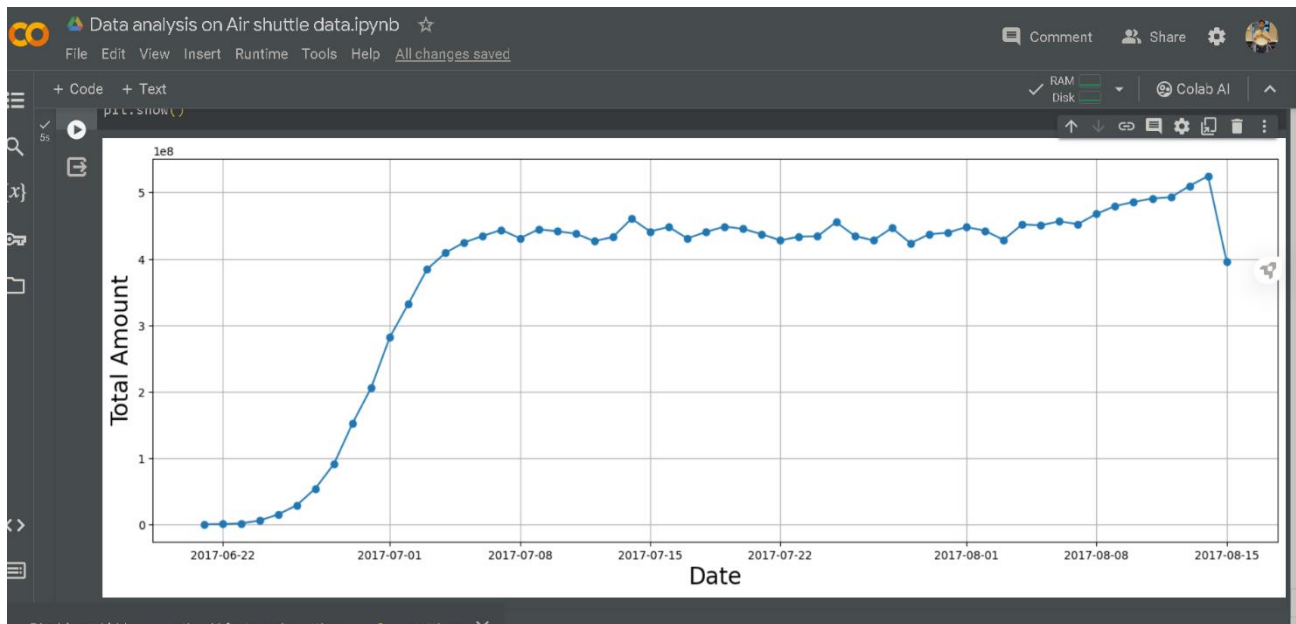
The Average ticket booked in a day is in the plot.



So on the X axis, there is a date and Y axis number of tickets Booked.

So if we **observe the plot** it starts with zero so we can see an **average approx 8000 per day** and we can also see the spike in the month of August approximate 10,000 per day.

```
bookings = pd.read_sql_query(f"""SELECT * FROM bookings""", connection)
bookings['book_date'] = pd.to_datetime(bookings['book_date'])
bookings['date'] = bookings['book_date'].dt.date
y = bookings.groupby('date')['total_amount'].sum()
plt.figure(figsize = (18,6))
plt.plot(y.index,y['total_amount'], marker = 'o')
plt.xlabel('Date', fontsize = 20)
plt.ylabel('Total Amount', fontsize = 20)
plt.grid('b')
plt.show()
```



So after performing this, we got some insights:

- To gain a deeper understanding of the trend of ticket bookings and revenue earned through those bookings, we have utilized a **line chart visualization**.
- Upon analysis of the chart, we observe that the number of tickets booked exhibits a **gradual increase from June 22nd to July 7th, followed by a relatively stable pattern from July 8th until August**, with a noticeable peak in ticket bookings where the highest number of tickets were booked on a single day.
- It is important to note that the revenue earned by the company from these bookings is closely **tied to the number of tickets booked**.
- Therefore, we can see a similar trend in the total revenue earned by the company throughout the analyzed period. These findings suggest that further exploration of the factors contributing to the **peak** in ticket bookings

may be beneficial for increasing overall revenue and optimizing operational strategies.

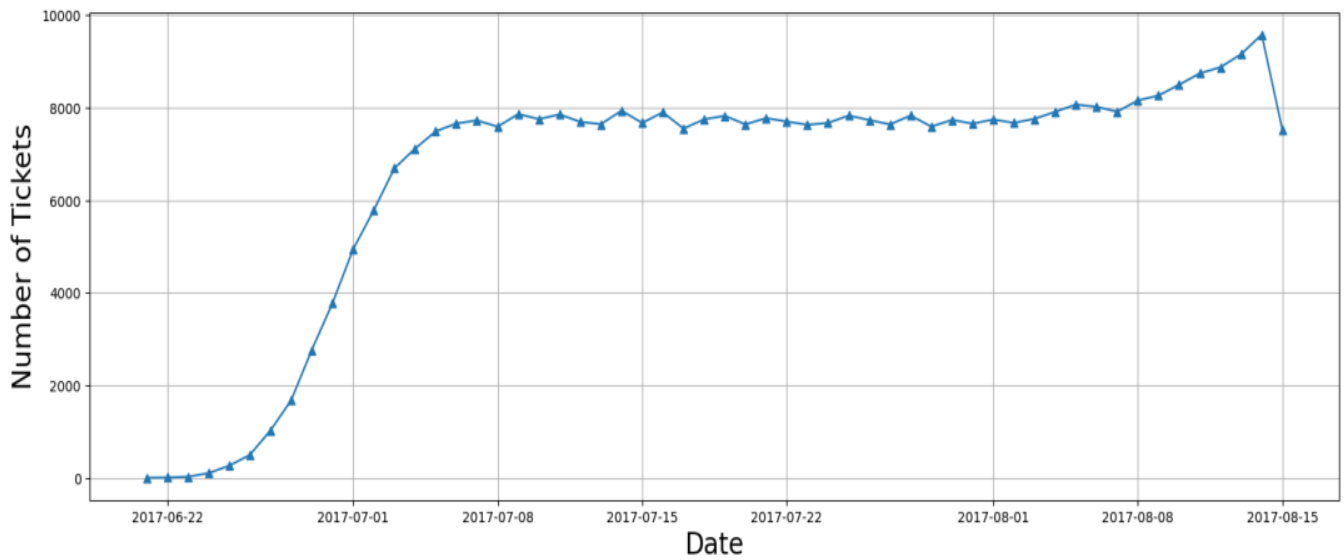


Figure 1

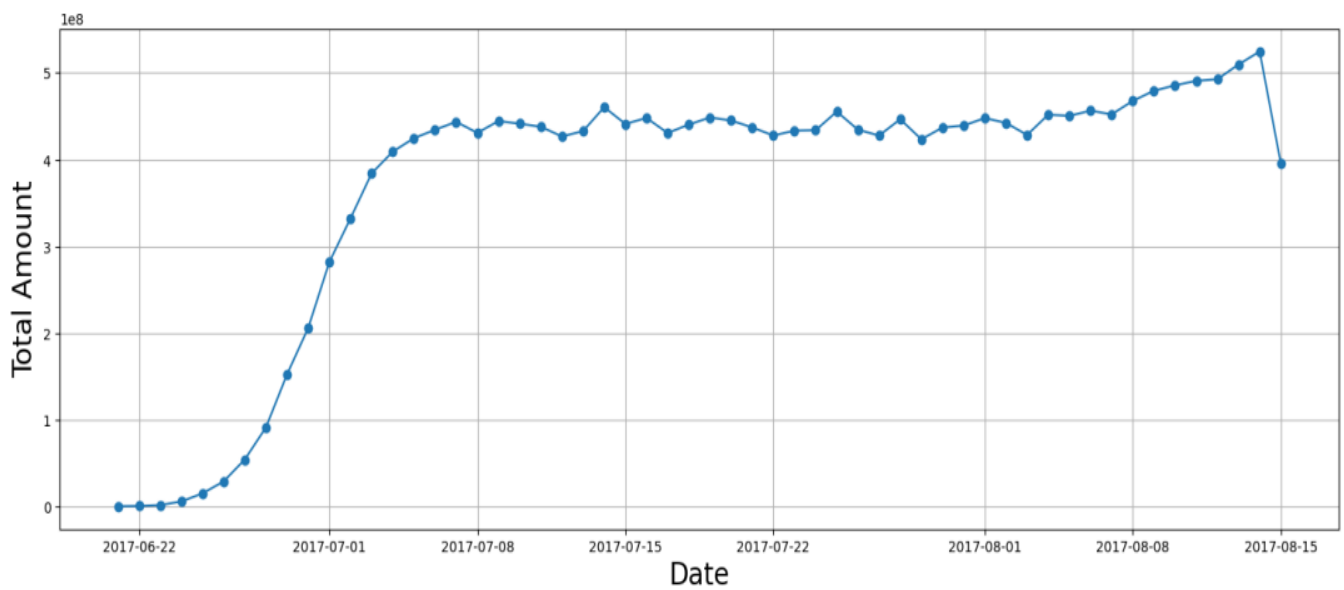


Figure 2

Now let's find the average charge for each Aircraft with different conditions.

- So we need to **join** two tables based on flight ID.
- So we will perform *Group by* on fare condition and aircraft code.
- So for this kind of data, we can use a bar chart.

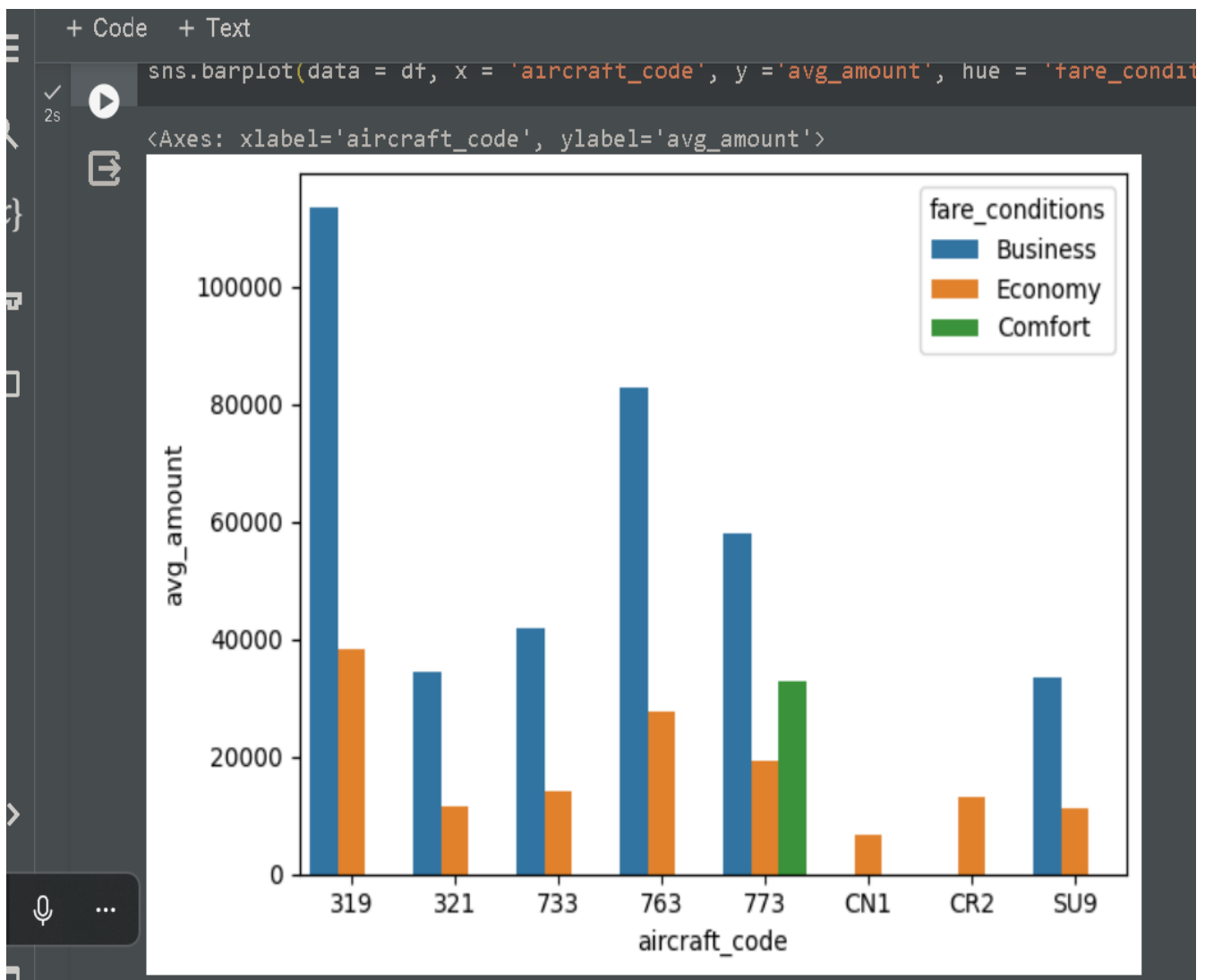
Let compute the average changes for each aeroplane with different fare conditions.

```
✓ 2s ▶ df = pd.read_sql_query(f"""SELECT fare_conditions, aircraft_code, AVG(amount) as avg_amount FROM ticket_flights
    JOIN flights
    ON ticket_flights.flight_id=flights.flight_id
    GROUP BY aircraft_code, fare_conditions""", connection)

sns.barplot(data = df, x = 'aircraft_code', y = 'avg_amount', hue = 'fare_conditions')
```

So we can get insight from the following chart that there are two aircraft out of nine which does not support business class.

And there is one aircraft which has all three fare conditions like business economy and comfort.



- Rest supports only business and economy-class
- We can observe there is one craft which has more fare for business and economy class as compared to other.

So what findings do we collect from this piece of analysis?

- ✓ We were able to generate a **bar graph** to graphically compare the data after we completed the computations for the average costs associated with different fare conditions for each aircraft.

The graph **Figure 3** shows data for three types of fares:

- 1) Business
- 2) Economy
- 3) comfort.

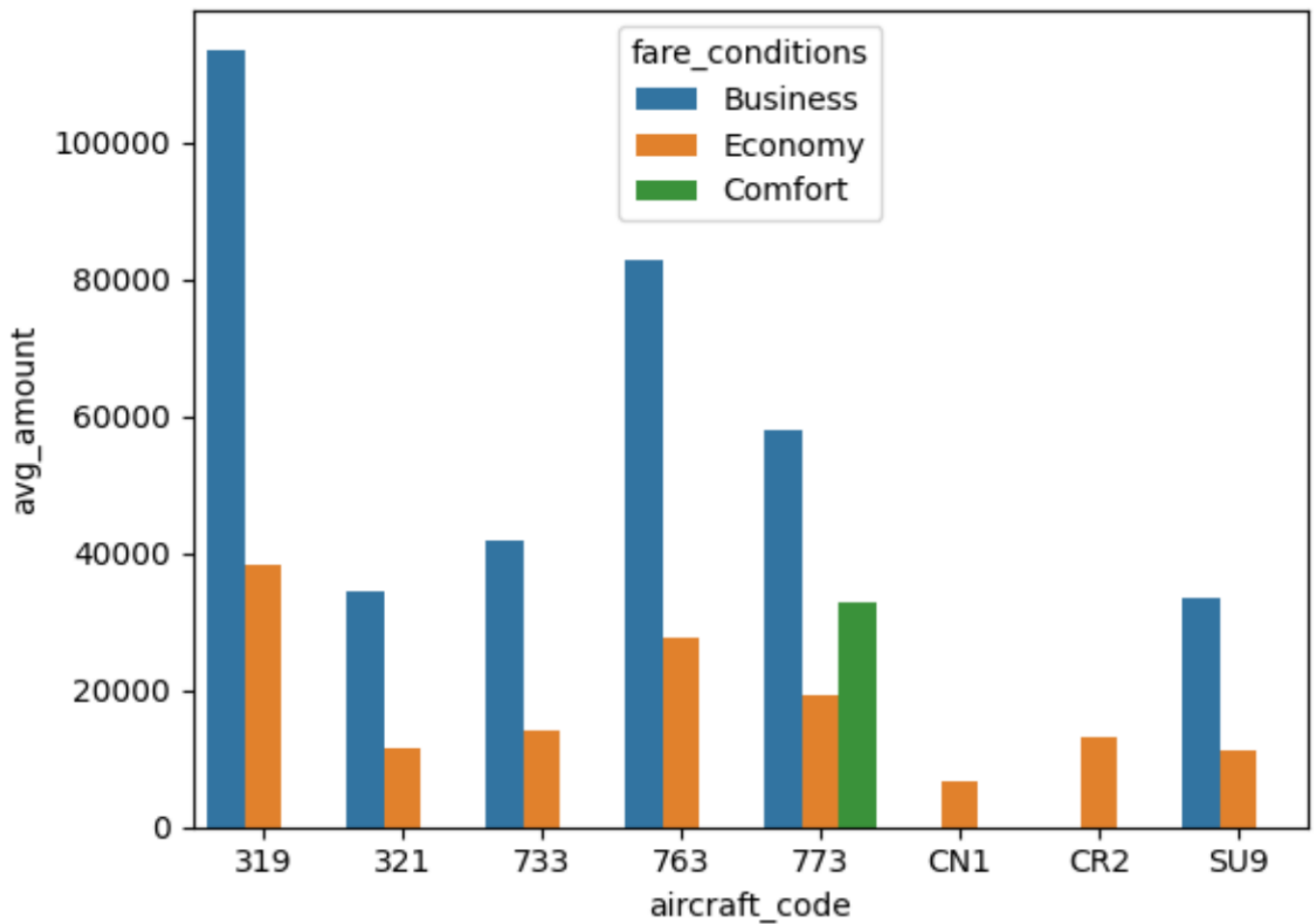


Figure 3

- ✓ It is worth mentioning that the comfort class is available on **only one aircraft, the 773**. The **CN1 and CR2 planes**, on the other hand, only **provide the economy class**.

- ✓ When different pricing circumstances within each aircraft are compared, the charges for business class are consistently greater than those for **economy** class. This trend may be seen across all planes, **regardless of fare conditions.**

TO INCREASE THE TURNOVER WE WILL ANALYSE THE OCCUPANCY RATE NOW:

In the airline industry, the term for occupancy rate is "load factor". It's the percentage of seats occupied by paying passengers on a flight.

→ Factors that affect flight occupancy include:

- ❖ Product offered by competitors
 - ❖ Seasonality
 - ❖ Flight frequency
 - ❖ Timings
- ❖ Direct/indirect operations
 - ❖ Traffic flows

Occupancy denotes the number of people in an aircraft cabin (half- vs. fully occupied cabin).

So we have three steps

So we need to compute total revenue and ticket count and then divide to get the average revenue per ticket.

Ticket flight and flight table will be used, with the help of subquery.

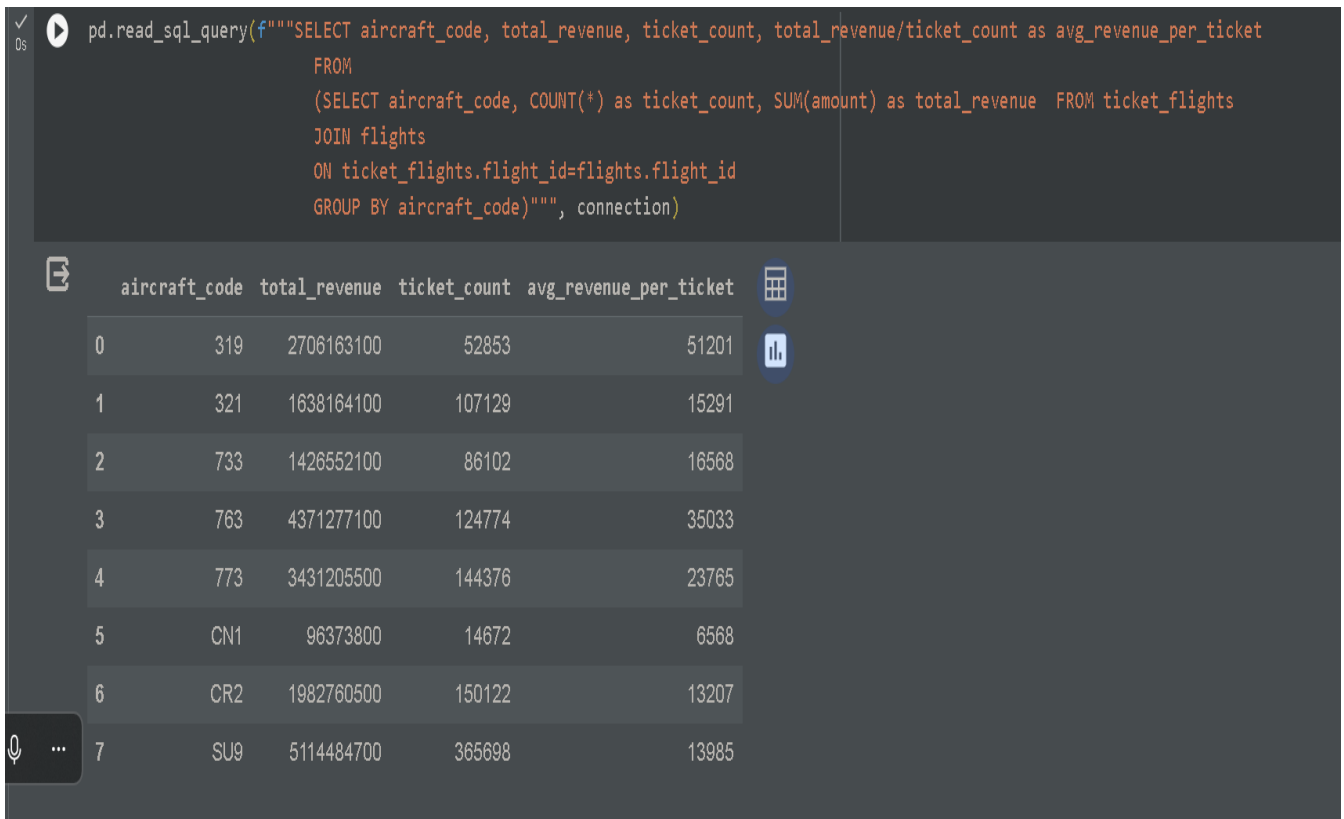
✓ Analyzing occupancy rate

For each Air shuttle, Compute the total revenue per year and the AVG revenue per ticket too.

```
✓ [3] import pandas as pd  
0s pd.set_option('display.float_format', str)
```



Start coding or generate with AI.



For each Air shuttle, the total revenue per year and the Average revenue per ticket too.

- ➔ Airlines must thoroughly analyze their revenue streams to maximize profitability.
- ➔ The overall income per year and average revenue per ticket for each aircraft are important metrics to consider.
- ➔ Airlines may use this information to determine which aircraft types and **itineraries** generate the most income and alter their operations appropriately.
- ➔ This research can also assist in identifying potential for pricing optimization and allocating resources to more profitable routes.
- ➔ The **below figure 4** shows the total revenue, total tickets and average revenue made per ticket for each aircraft.

	aircraft_code	total_revenue	ticket_count	avg_revenue_per_ticket
0	319	2706163100	52853	51201
1	321	1638164100	107129	15291
2	733	1426552100	86102	16568
3	763	4371277100	124774	35033
4	773	3431205500	144376	23765
5	CN1	96373800	14672	6568
6	CR2	1982760500	150122	13207
7	SU9	5114484700	365698	13985

Figure 4

The aircraft with **the highest total revenue is SU9** and from Figure 3 it can be seen that the price of the business class and economy class is the lowest in this aircraft.

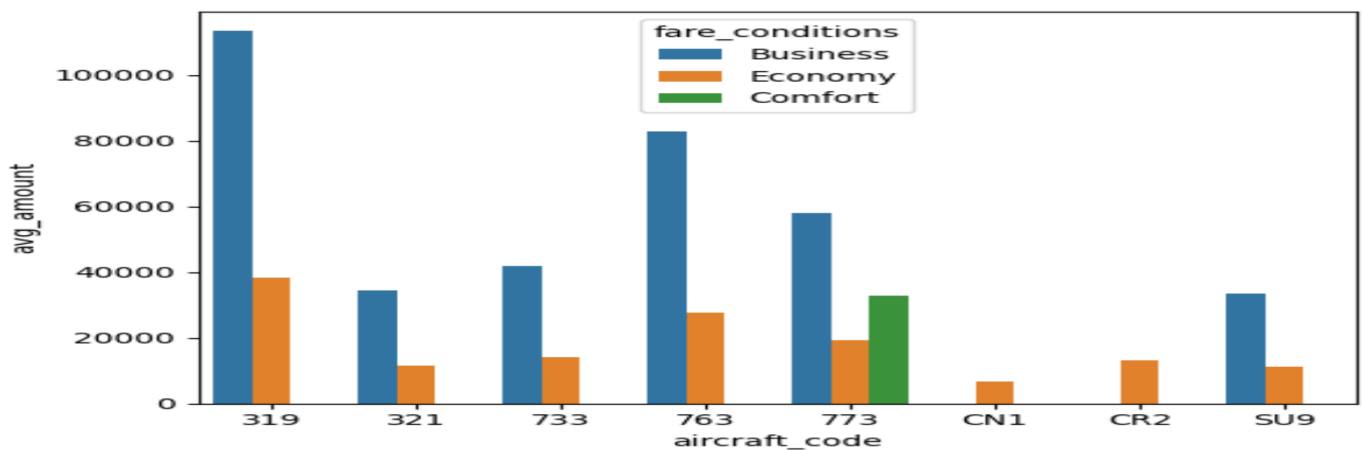


Figure 3

This can be the reason that most of the people **bought this aircraft ticket as its cost is less compared to others**. The aircraft with the **least total revenue is CN1**, and the possible **reason behind this is it only offers economy class with very least price** and it might be because of its **poor conditions or less facilities**.

Now let's calculate the average occupancy

per Aeroplane.

So first we fetch the data using SQL query.

So we need to join two tables boarding pass and a flight table so it will give seat count. Merging two queries to get seat count.

Finally, we got the occupancy rate now in the next step let's increase it by 10%

By creating a new field called **occupancy rate** & adding 0.1 which is 10%.

Find out the average occupancy per aircraft.

```
occupancy_rate = pd.read_sql_query(f"""SELECT a.aircraft_code, AVG(a.seats_count) as booked_seats, b.num_seats,
AVG(a.seats_count)/b.num_seats as occupancy_rate
FROM (
    SELECT aircraft_code, flights.flight_id, COUNT(*) as seats_count
    FROM boarding_passes
    INNER JOIN flights
    ON boarding_passes.flight_id=flights.flight_id
    GROUP BY aircraft_code, flights.flight_id
) as a INNER JOIN
(
    SELECT aircraft_code, COUNT(*) as num_seats FROM seats
    GROUP BY aircraft_code
) as b
ON a.aircraft_code = b.aircraft_code
GROUP BY a.aircraft_code""", connection)

occupancy_rate
```

The average occupancy

- Per aircraft is another **critical** number to consider. Airlines may measure how successfully they fill their seats and discover **chances to boost occupancy rates by using this metric.**
- Higher occupancy rates can help airlines increase revenue and profitability while **lowering operational expenses associated with vacant seats**. Pricing strategy, airline schedules, and customer satisfaction are all factors that might influence occupancy rates.
- The below **Figure 5** shows the average booked seats from the total number of seats for each aircraft.

	aircraft_code	booked_seats	num_seats	occupancy_rate
0	319	53.58318098720292	116	0.46192397402761143
1	321	88.80923076923077	170	0.5224072398190045
2	733	80.25546218487395	130	0.617349709114415
3	763	113.93729372937294	222	0.5132310528350132
4	773	264.9258064516129	402	0.659019419033863
5	CN1	6.004431314623338	12	0.5003692762186115
6	CR2	21.48284690220174	50	0.42965693804403476
7	SU9	56.81211267605634	97	0.5856918832583128

Figure 5

- The occupancy rate is calculated by dividing the booked seats by the total number of seats.
- **Higher occupancy rate** means the aircraft seats are more booked and only few seats are left unbooked.

Now we are at the stage where we need to calculate the total revenue:

And also calculate annual increased turnover.

Then subtraction between increased occupancy rate and increase total annual turnover.

Last find if we increase 10% occupancy rate than total annual turnover will be

```
occupancy_rate['Inc occupancy rate'] = occupancy_rate['occupancy_rate'] + occupancy_rate['occupancy_rate']*0.1
```

	aircraft_code	booked_seats	num_seats	occupancy_rate	Inc occupancy rate
0	319	53.58318098720292	116	0.46192397402761143	0.5081163714303726
1	321	88.80923076923077	170	0.5224072398190045	0.574647963800905
2	733	80.25546218487395	130	0.617349709114415	0.6790846800258565
3	763	113.93729372937294	222	0.5132310528350132	0.5645541581185146
4	773	264.9258064516129	402	0.659019419033863	0.7249213609372492
5	CN1	6.004431314623338	12	0.5003692762186115	0.5504062038404727
6	CR2	21.48284690220174	50	0.42965693804403476	0.4726226318484382
7	SU9	56.81211267605634	97	0.5856918832583128	0.644261071584144

```
total_revenue = pd.read_sql_query("""SELECT aircraft_code, SUM(amount) as total_revenue FROM ticket_flights
JOIN flights
ON ticket_flights.flight_id=flights.flight_id
GROUP BY aircraft_code""", connection)
```

```
occupancy_rate['Inc Total Annual Turnover'] = (total_revenue['total_revenue']/occupancy_rate['occupancy_rate'])*occupancy_rate['Inc occupancy rate']
```

	aircraft_code	booked_seats	num_seats	occupancy_rate	Inc occupancy rate	Inc Total Annual Turnover
0	319	53.58318098720292	116	0.46192397402761143	0.5081163714303726	2976779410.0
1	321	88.80923076923077	170	0.5224072398190045	0.574647963800905	1801980510.0
2	733	80.25546218487395	130	0.617349709114415	0.6790846800258565	1569207310.0000002
3	763	113.93729372937294	222	0.5132310528350132	0.5645541581185146	4808404810.0
4	773	264.9258064516129	402	0.659019419033863	0.7249213609372492	3774326050.0
5	CN1	6.004431314623338	12	0.5003692762186115	0.5504062038404727	106011180.00000001
6	CR2	21.48284690220174	50	0.42965693804403476	0.4726226318484382	2181036550.0
7	SU9	56.81211267605634	97	0.5856918832583128	0.644261071584144	5625933169.999999

Airlines can assess how much their total yearly turnover could improve by providing all aircraft with a **10% higher occupancy rate** to further examine the possible benefits of raising occupancy rates.

This research can assist airlines in determining the financial **impact of boosting occupancy rates** and if it is a **realistic strategy**.

Airlines may **enhance occupancy rates** and revenue while delivering greater value and service to consumers by optimizing pricing tactics and other operational considerations.

Figure 6 shows how the total revenue increased after increasing the occupancy rate by 10% and it gives the **result** that it will increase gradually so airlines should be more focused on the pricing strategies.

	aircraft_code	booked_seats	num_seats	occupancy_rate	Inc occupancy rate	Inc Total Annual Turnover
0	319	53.58318098720292	116	0.46192397402761143	0.5081163714303726	2976779410.0
1	321	88.80923076923077	170	0.5224072398190045	0.574647963800905	1801980510.0
2	733	80.25546218487395	130	0.617349709114415	0.6790846800258565	1569207310.0000002
3	763	113.93729372937294	222	0.5132310528350132	0.5645541581185146	4808404810.0
4	773	264.9258064516129	402	0.659019419033863	0.7249213609372492	3774326050.0
5	CN1	6.004431314623338	12	0.5003692762186115	0.5504062038404727	106011180.00000001
6	CR2	21.48284690220174	50	0.42965693804403476	0.4726226318484382	2181036550.0
7	SU9	56.81211267605634	97	0.5856918832583128	0.644261071584144	5625933169.999999

Figure 6

Conclusion/ Result/ Findings

To summarize, analyzing revenue data such as total revenue per year, average revenue per ticket, and average occupancy per aircraft is critical for airlines seeking to maximize profitability.

Airlines need to focus on those whose occupancy rate is low to increase occupancy rate they can do advertisement and marketing.

*Airlines can take insight from this and **make changes in their strategies***

Airlines can find areas for improvement and modify their pricing and route plans as a result of assessing these indicators. A greater occupancy rate is one important feature that can enhance profitability since it allows airlines to maximize revenue while minimizing costs associated with vacant seats.

The airline should revise the price for each aircraft as the lower price and high price are also the factor that people are not buying tickets from those aircrafts. They should decide the reasonable price according to the condition and facility of the aircraft and it should not be very cheap or high.

Furthermore, boosting occupancy rates should not come at the price of consumer happiness or safety. Airlines must strike a balance between the necessity for profit and the significance of delivering high-quality service and upholding safety regulations.

A SOLUTION TO THE PROBLEM\ RESULT

Economic Downturn or Recession:

During economic downturns or recessions, people tend to cut back on discretionary spending, including travel. Businesses may reduce corporate travel budgets, and individuals may postpone or cancel leisure trips. This can result in lower demand for airline tickets, leading to decreased occupancy rates.

Solution: During economic downturns, focus on cost-effective strategies and offer flexible pricing options. Consider targeted promotions or discounts to stimulate demand.

Seasonal Fluctuations:

Airlines often experience seasonal fluctuations in demand. For example, during off-peak seasons or specific months with lower travel demand, airlines may struggle to fill seats. Factors such as weather conditions, holidays, and school schedules can impact travel patterns, contributing to periods of low occupancy.

Solution: Implement dynamic pricing models that adjust ticket prices based on demand and seasonality. Offer attractive promotions and packages during off-peak seasons to incentivize travel. Explore new routes or destinations that may have higher demand during specific seasons.

Global Events or Health Crises:

Unforeseen global events, such as natural disasters, political instability, or health crises (as experienced with the COVID-19 pandemic), can significantly impact travel demand. Travel restrictions, lockdowns, and passenger concerns about safety can lead to a sharp decline in bookings, resulting in reduced occupancy rates for airlines.

Solution: Prioritize passenger safety and communicate transparently about health and safety measures implemented by the airline. Offer flexible booking and cancellation policies to instill confidence in travelers. Diversify routes to minimize the impact of regional crises.

Airlines need to carefully monitor market conditions, adapt pricing strategies, and implement responsive measures to navigate challenges and maintain healthy occupancy levels.

Airlines may achieve long-term success in a highly competitive business by adopting a data-driven strategy to revenue analysis and optimisation.

Future score

I am promising, with numerous opportunities for **further exploration** and enhancement.

Collaboration with Airport Operations:

- Collaborate with airport authorities to analyze data related to airport operations, such as baggage handling, security checks, and passenger flow. Improving efficiency in these areas can contribute to an overall enhanced travel experience.

Integration with IoT and Real-Time Data:

- Explore the integration of Internet of Things (IoT) devices to capture real-time data from aircraft, airport facilities, and passenger interactions. This could provide a more comprehensive and up-to-date dataset for analysis, leading to more accurate insights.

Carbon Emission Analysis and Sustainability:

- Extend your analysis to include environmental impact by incorporating data on carbon emissions from flights. Develop insights into the environmental footprint of different routes and explore strategies for airlines to enhance sustainability.

Bibliography

- Python programming language
SQL
- Anaconda Navigator
- Jupiter notebook
- Kaggle
- Google colaboratory
- Medium: Is a widely-used platform that allows users to publish articles and blog posts.

USEFUL LINKS

Code notebook

<https://drive.google.com/file/d/1UoCY3nf2y87YtDJl5ARnXLQtr6Wal8T/view?usp=sharing>

