Airlines Data Analysis using SQL and Python

The Airline Database consists of following tables: -

- aircrafts_data
- airports_data
- boarding_passes
- bookings
- flights
- seats
- ticket_flights
- tickets

Objective of this project: -

1. To increase the occupancy rate, so that we can boost the average profit earned per seat.
2. To develop a pricing strategy that considers the changing market conditions and customer preferences to attract and retain customers.

Observations based on Analysis of the tables: -

1.**Tickets and Bookings: -**

This is the most important observation which shows the strong relationship between the bookings and tickets tables. The **book_ref** column serves as a key connecting these tables. Analysis of this relationship can provide insights into the booking patterns and the number of tickets associated with each booking.

```
import sqlite3
import pandas as pd

database_path = r'C:\Users\14695\OneDrive\Desktop\python_projects\travel.sqlite'

# Establish the database connection using a with statement
with sqlite3.connect(database_path) as connection:
    # Query to get the number of tickets booked and total amount earned over time
    query = """
    SELECT
        bookings.book_ref,
        COUNT(tickets.ticket_no) AS num_tickets
    FROM
        bookings
        INNER JOIN tickets ON bookings.book_ref = tickets.book_ref
    GROUP BY
        bookings.book_ref
    ORDER BY
        num_tickets DESC;
    """

    # Execute the query and read into a DataFrame
    df = pd.read_sql_query(query, connection)

    # Print the DataFrame
    print(df)
```

```
       book_ref  num_tickets
0        E4EE9A            5
1        E4BF84            5
2        C4AC71            5
3        B7D627            5
4        9BF4CE            5
...         ...          ...
262783   00034E            1
262784   0002D8            1
262785   000068            1
262786   000012            1
262787   00000F            1
```

Observation: - From this analysis it is clearly understood that the maximum tickets associated with a booking is 5 and minimum is 1.

2.Flights and Airlines: -

The flights table contains information about flights, and the airlines table provides details about the associated airlines. Analyzing the frequency of flights by airline can reveal which airlines are most active.

```
import sqlite3
import pandas as pd

database_path = r'C:\Users\14695\OneDrive\Desktop\python_projects\travel.sqlite'

# Establish the database connection using a with statement
with sqlite3.connect(database_path) as connection:
    # Query to get the booking reference with the maximum number of tickets booked
    query = """
    SELECT aircrafts_data.aircraft_code,
    COUNT(flights.flight_id) AS num_flights
    FROM
    aircrafts_data
    INNER JOIN flights ON aircrafts_data.aircraft_code = flights.aircraft_code
    GROUP BY
    aircrafts_data.model
    ORDER BY
    num_flights DESC;

    """

    # Execute the query and read into a DataFrame
    df = pd.read_sql_query(query, connection)

    # Print the DataFrame
    print("Booking Reference with Max Tickets Booked:")
    print(df)
```

```
Booking Reference with Max Tickets Booked:
  aircraft_code  num_flights
0           CN1         9273
1           CR2         9048
2           SU9         8504
3           321         1952
4           733         1274
5           319         1239
6           763         1221
7           773          610
```

Observation:- It is clear that aircraft code CN1 has highest number of flights.

How many planes have more than 100 seats?

```
pd.read_sql_query("""select aircraft_code, count(*) as num_seats from seats group by aircraft_code having num_seats>100""",
                  connection)
```

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

Find the total number of airacrafts based on the aircraft code.

```
import sqlite3

# Connect to the SQLite database
database_path = r'C:\Users\14695\OneDrive\Desktop\python_projects\travel.sqlite'
connection = sqlite3.connect(database_path)
cursor = connection.cursor()

# Execute SQL query to count the number of aircraft based on aircraft_code
cursor.execute("SELECT aircraft_code, COUNT(*) FROM seats GROUP BY aircraft_code;")
aircraft_count_by_code = cursor.fetchall()

# Print the result
print("Number of Aircrafts based on Aircraft Code:")
for aircraft_code, count in aircraft_count_by_code:
    print(f"Aircraft Code: {aircraft_code}, Count: {count}")

# Close the connection
connection.close()
```

```
Number of Aircrafts based on Aircraft Code:
Aircraft Code: 319, Count: 116
Aircraft Code: 320, Count: 140
Aircraft Code: 321, Count: 170
Aircraft Code: 733, Count: 130
Aircraft Code: 763, Count: 222
Aircraft Code: 773, Count: 402
Aircraft Code: CN1, Count: 12
Aircraft Code: CR2, Count: 50
Aircraft Code: SU9, Count: 97
```

Observation:

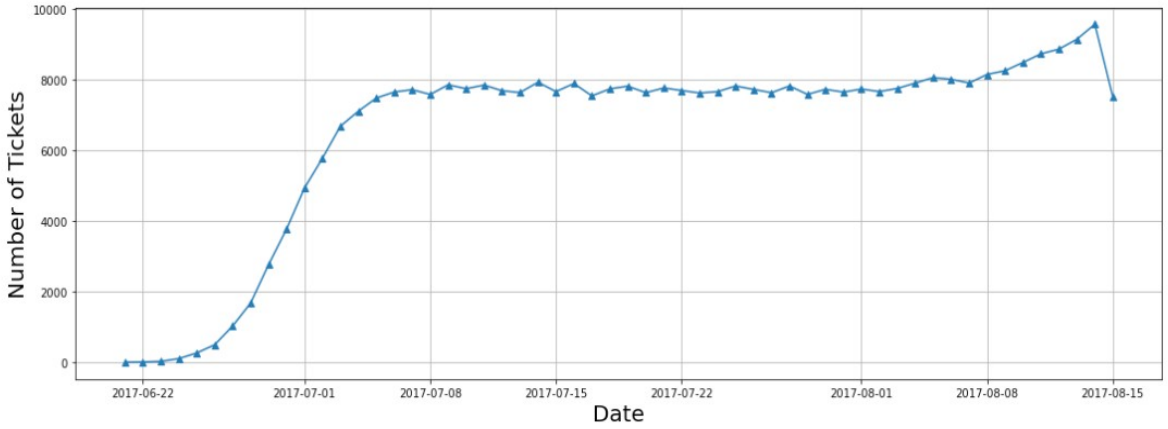
1)From the above observation we can understand that there are a total of 9 aircrafts.

2)Aircraft Code 773 has the highest number of aircrafts.

```
. How the number of tickets booked and total amount earned changed with time.
```

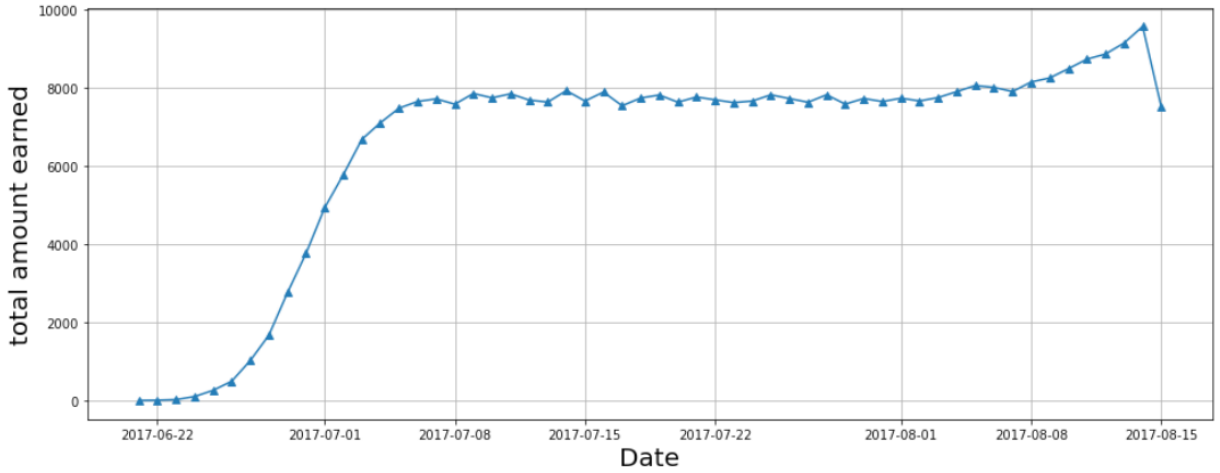
```
import sqlite3
import pandas as pd

database_path = r'C:\Users\14695\OneDrive\Desktop\python_projects\travel.sqlite'

# Establish the database connection using a with statement
with sqlite3.connect(database_path) as connection:
    # Query to get the number of tickets booked and total amount earned over time
    query = """
    SELECT
    *FROM tickets
    INNER JOIN bookings ON tickets.book_ref = bookings.book_ref;
    """

    # Execute the query and read into a DataFrame
    df = pd.read_sql_query(query, connection)

    # Print the DataFrame
    print(df)
```

```
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()
```



Observation: Here this plot shows the variation in number of Tickets over period of time. From June till Aug the avg tickets sold were around 8000 and 10000 being the highest number of tickets.

```
bookings= pd.read_sql_query("select *from bookings", connection)
df['book_date']= pd.to_datetime(df['book_date'])
df['date']=df['book_date'].dt.date
x=df.groupby('date')[["date"]].count()
plt.figure(figsize=(16,6))
plt.plot(x.index,x['date'], marker= '^')
plt.xlabel('Date', fontsize=20)
plt.ylabel('total amount earned', fontsize=20)
plt.grid('b')
plt.show()
```



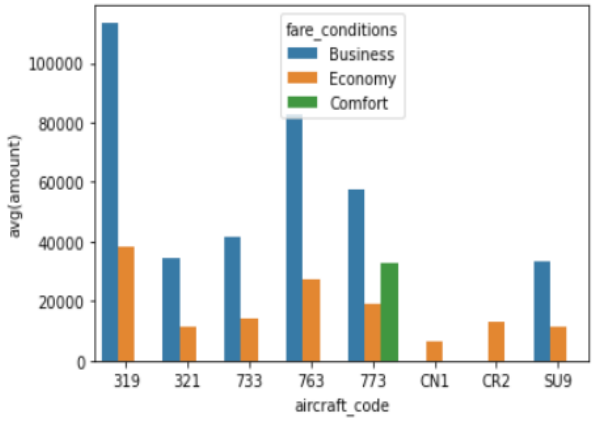
Observation: This plot is similar to the above plot as the total amount earned depends on the number of Tickets.

```
Calculate the average charges for each aircraft with different fare conditions.
```

```
df=pd.read_sql_query("""select aircraft_code,fare_conditions, avg(amount) from ticket_flights inner 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')
```

```
: <AxesSubplot:xlabel='aircraft_code', ylabel='avg(amount)'>
```



Observation:

1. It is clearly understood that only one aircraft i.e 773 has the comfort class.
2. Both CN1 and CR2 have only Economy classes.
3. The fare_conditions of Business class is higher than the Economy class in all aircrafts.
4. The fare_conditions are higher in aircraft 319 for both Business and economy compared to other aircrafts.

```
For each aircraft calculate the total revenue per year and the avaerage revenue per ticket.
```

```
df=pd.read_sql_query("""select aircraft_code,ticket_count,total_revenue, total_revenue/ticket_count as avg_revenue_per_ticket
 select aircraft_code, count(*) as ticket_count, sum(amount) as total_revenue from ticket_flights
oin flights on ticket_flights.flight_id=flights.flight_id
                        group by aircraft_code)""", connection)
```

```
df
```

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

Observation: -

From the results it is clear that SU9 has highest number of tickets booked as its fare conditions are lower and its total revenue is higher than others.

```
. Calculate the average occupancy per aircraft.
```

```
occupancy_rate= pd.read_sql_query("""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
```

	aircraft_code	booked_seats	num_seats	occupancy_rate
0	319	53.583181	116	0.461924
1	321	88.809231	170	0.522407
2	733	80.255462	130	0.617350
3	763	113.937294	222	0.513231
4	773	264.925806	402	0.659019
5	CN1	6.004431	12	0.500369
6	CR2	21.482847	50	0.429657
7	SU9	56.812113	97	0.585692

Observation: -

It is clear that the occupancy rate is higher for aircraft 773 as it has 3 classes , Comfort, Business and Economy.

```
7. calculate by how much the total annual turnover could increase by giving all aircraft a 10% higher occupancy rate.
```

```
pd.set_option("display.float_format", str)
```

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

.22]:

	aircraft_code	booked_seats	num_seats	occupancy_rate	Inc occupancy rate	Increase 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	0.5645541581185145	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

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

```
occupancy_rate
```

.24]:

	aircraft_code	booked_seats	num_seats	occupancy_rate	Inc occupancy rate	Increase 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	0.5645541581185145	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

Observation: -

The Total annual Turnover is highest for SU9 aircraft as its total revenue is higher than others.

Summary: -

- Analyzing total revenue per year, average revenue per ticket, and average occupancy per aircraft is helpful for airlines to maximize their profitability.
- A higher occupancy rate is one important feature that maximizes profitability since it allows airlines to maximize their revenues.
- Also, the increase in occupancy rates should not come at the price of customer happiness and their safety.