OBJECTIVE

To Identify opportunity to increase the occupancy rate on low-performing flights, which can ultimately lead to increased profitability for the Airlines

In [1]:

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

Database connection

In [2]:

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

In [3]:

```
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
```

Out[3]:

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

Data Exploration

In [5]:

```
aircraft_data = pd.read_sql_query("select * from aircrafts_data", connection)
aircraft_data
```

Out[5]:

	aircraft_code	model	range
0	773	{"en": "Boeing 777-300", "ru": "Боинг 777-300"}	11100

	aircraft_code	model	range
1	763	{"en": "Boeing 767-300", "ru": "Боинг 767-300"}	7900
2	SU9	{"en": "Sukhoi Superjet-100", "ru": "Сухой Суп...	3000
3	320	{"en": "Airbus A320-200", "ru": "Аэробус A320-...	5700
4	321	{"en": "Airbus A321-200", "ru": "Аэробус A321-...	5600
5	319	{"en": "Airbus A319-100", "ru": "Аэробус A319-...	6700
6	733	{"en": "Boeing 737-300", "ru": "Боинг 737-300"}	4200
7	CN1	{"en": "Cessna 208 Caravan", "ru": "Сессна 208...	1200
8	CR2	{"en": "Bombardier CRJ-200", "ru": "Бомбардье ...	2700

In [6]:

```
airports_data = pd.read_sql_query("select * from airports_data", connection)
airports_data
```

Out[6]:

	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": "Mirnyj", "ru": "Мирный"}	(114.03900146484375,62.534698486328125)	Asia/Yakutsk
2	KHV	{"en": "Khabarovsk-Novy Airport",	{"en": "Khabarovsk", "ru": "Хабаровск"}	(135.18800354004,48.5279998779300001)	Asia/Vladivostok

	airport_code	airport_name	city	coordinates	timezone
		"ru": "Хабар...			
3	PKC	{"en": "Yelizovo Airport", "ru": "Елизово"}	{"en": "Petropavlovsk", "ru": "Петропавловск-К...	(158.453994750976562,53.16790008544921 88)	Asia/Kamchatka
4	UUS	{"en": "Yuzhno-Sakhalinsk Airport", "ru": "Хом...	{"en": "Yuzhno-Sakhalinsk", "ru": "Южно-Сахали...	(142.718002319335938,46.88869857788085 94)	Asia/Sakhalin
...	...	...	...	...	...
99	MMK	{"en": "Murmansk Airport", "ru": "Мурманск" }	{"en": "Murmansk", "ru": "Мурманск"}	(32.7508010864257812,68.78170013427734 38)	Europe/Moscow
10 0	ABA	{"en": "Abakan Airport", "ru": "Абакан"}	{"en": "Abakan", "ru": "Абакан"}	(91.3850021362304688,53.74000167846679 69)	Asia/Krasnoyarsk
10 1	BAX	{"en": "Barnaul Airport", "ru": "Барнаул"}	{"en": "Barnaul", "ru": "Барнаул"}	(83.5384979248046875,53.36380004882812 5)	Asia/Krasnoyarsk
10 2	AAQ	{"en": "Anapa Vityazevo Airport", "ru": "Витяз...	{"en": "Anapa", "ru": "Анапа"}	(37.3473014831539984,45.00210189819299 7)	Europe/Moscow
10 3	CNN	{"en": "Chulman Airport",	{"en": "Neryungri",	(124.914001464839998,56.91389846801799 73)	Asia/Yakutsk

airport_code	airport_name	city	coordinates	timezone
	"ru": "Чульман"}	"ru": "Нерюнгри"}		

104 rows × 5 columns

In [7]:

```
boarding_passes = pd.read_sql_query("select * from boarding_passes",
connection)
boarding_passes
```

Out[7]:

	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
...	...	...	...	...
579681	0005434302871	19945	85	20F
579682	0005432892791	19945	86	21C
579683	0005434302869	19945	87	20E
579684	0005432802476	19945	88	21F
579685	0005432802482	19945	89	21E

579686 rows × 4 columns

In [9]:

```
bookings = pd.read_sql_query("select * from bookings", connection)
```

```
bookings
```

Out[9]:

	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
...	...	...	...
262783	FFFEF3	2017-07-17 07:23:00+03	56000
262784	FFFF2C	2017-08-08 05:55:00+03	10800
262785	FFFF43	2017-07-20 20:42:00+03	78500
262786	FFFFA8	2017-08-08 04:45:00+03	28800
262787	FFFFF7	2017-07-01 22:12:00+03	73600

262788 rows × 3 columns

In [10]:

```
flights = pd.read_sql_query("select * from flights", connection)
flights
```

Out[10]:

	flight_id	flight_no	scheduled_departure	scheduled_arrival	departure_airport	arrival_airport	status	aircraft_code	actual_departure	actual_arrival
0	1185	PG0134	2017-09-10 09:50:00+03	2017-09-10 14:55:00+03	DME	BTK	Scheduled	319	\N	\N

	flight_id	flight_no	scheduled_departure	scheduled_arrival	departure_airport	arrival_airport	status	aircraft_code	actual_departure	actual_arrival
1	3979	PG0052	2017-08-25 14:50:00+03	2017-08-25 17:35:00+03	VKO	HMA	Scheduled	CR2	\N	\N
2	4739	PG0561	2017-09-05 12:30:00+03	2017-09-05 14:15:00+03	VKO	AER	Scheduled	763	\N	\N
3	5502	PG0529	2017-09-12 09:50:00+03	2017-09-12 11:20:00+03	SVO	UFA	Scheduled	763	\N	\N
4	6938	PG0461	2017-09-04 12:25:00+03	2017-09-04 13:20:00+03	SVO	ULV	Scheduled	SU9	\N	\N
...	...	...	...	...	...	...	...	...	...	...
3311 6	33117	PG0063	2017-08-02 19:25:00+03	2017-08-02 20:10:00+03	SKX	SVO	Arrived	CR2	2017-08-02 19:25:00+03	2017-08-02 20:10:00+03
3311 7	33118	PG0063	2017-07-28 19:25:00+03	2017-07-28 20:10:00+03	SKX	SVO	Arrived	CR2	2017-07-28 19:30:00+03	2017-07-28 20:15:00+03
3311 8	33119	PG0063	2017-09-08 19:25:00+03	2017-09-08 20:10:00+03	SKX	SVO	Scheduled	CR2	\N	\N
3311 9	33120	PG0063	2017-08-01 19:25:00+03	2017-08-01 20:10:00+03	SKX	SVO	Arrived	CR2	2017-08-01 19:26:00+03	2017-08-01 20:12:00+03

	flight_id	flight_no	scheduled_departure	scheduled_arrival	departure_airport	arrival_airport	status	aircraft_code	actual_departure	actual_arrival
33120	33121	PG0063	2017-08-26 19:25:00+03	2017-08-26 20:10:00+03	SKX	SVO	Scheduled	CR2	\N	\N

33121 rows × 10 columns

In [11]:

```
seats = pd.read_sql_query("select * from seats", connection)
seats
```

Out[11]:

	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
...	...	...	...
1334	773	48H	Economy
1335	773	48K	Economy
1336	773	49A	Economy
1337	773	49C	Economy
1338	773	49D	Economy

1339 rows × 3 columns

In [12]:

```
ticket_flights = pd.read_sql_query("select * from ticket_flights",
connection)
ticket_flights
```

Out[12]:

	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
...	...	...	...	...
1045721	0005435097522	32094	Economy	5200
1045722	0005435097521	32094	Economy	5200
1045723	0005435104384	32094	Economy	5200
1045724	0005435104352	32094	Economy	5200
1045725	0005435104389	32094	Economy	5200

1045726 rows × 4 columns

In [13]:

```
tickets = pd.read_sql_query("select * from tickets", connection)
tickets
```

Out[13]:

	ticket_no	book_ref	passenger_id
0	0005432000987	06B046	8149 604011

	ticket_no	book_ref	passenger_id
1	0005432000988	06B046	8499 420203
2	0005432000989	E170C3	1011 752484
3	0005432000990	E170C3	4849 400049
4	0005432000991	F313DD	6615 976589
...	...	...	...
366728	0005435999869	D730BA	0474 690760
366729	0005435999870	D730BA	6535 751108
366730	0005435999871	A1AD46	1596 156448
366731	0005435999872	7B6A53	9374 822707
366732	0005435999873	7B6A53	7380 075822

366733 rows × 3 columns

In [16]:

```
for table in table_list:
    print('\ntable:',table)
    column_info = connection.execute("PRAGMA table_info({})".format(table))
    for column in column_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: boarding_passes
```

```
('ticket_no', 'character(13)')
('flight_id', 'INTEGER')
('boarding_no', 'INTEGER')
('seat_no', 'character varying(4)')

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')
('fare_conditions', 'character varying(10)')
('amount', 'numeric(10,2)')

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

In [19]:

```
for table in table_list:
    print('\ntable:',table)
    df_table = pd.read_sql_query(f"select * from {table}",connection)
    print(df_table.isnull().sum())
```

```
table: aircrafts_data
aircraft_code    0
model            0
range            0
dtype: int64

table: airports_data
airport_code    0
airport_name    0
city            0
```

```
coordinates     0
timezone        0
dtype: int64

table: boarding_passes
ticket_no      0
flight_id      0
boarding_no    0
seat_no        0
dtype: int64

table: bookings
book_ref        0
book_date       0
total_amount    0
dtype: int64

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

table: seats
aircraft_code      0
seat_no            0
fare_conditions    0
dtype: int64

table: ticket_flights
ticket_no          0
flight_id          0
fare_conditions    0
amount             0
dtype: int64

table: tickets
ticket_no       0
book_ref        0
passenger_id    0
dtype: int64
```

Basic Analysis

1. How many planes have more than 100 seats?

In [20]:

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

Out[20]:

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

2. How many number of tickets booked and total amount earned changed with the time?

In [22]:

```
pd.read_sql_query("""select * from tickets inner join bookings on
tickets.book_ref = bookings.book_ref""", connection)
```

Out[22]:

	ticket_no	book_ref	passenger_id	book_ref	book_date	total_amount
0	0005432000987	06B046	8149 604011	06B046	2017-07-05 20:19:00+03	12400
1	0005432000988	06B046	8499 420203	06B046	2017-07-05 20:19:00+03	12400
2	0005432000989	E170C3	1011 752484	E170C3	2017-06-29 01:55:00+03	24700
3	0005432000990	E170C3	4849 400049	E170C3	2017-06-29 01:55:00+03	24700

	ticket_no	book_ref	passenger_id	book_ref	book_date	total_amount
4	0005432000991	F313DD	6615 976589	F313DD	2017-07-03 04:37:00+03	30900
...	...	...	...	...	...	...
366728	0005435999869	D730BA	0474 690760	D730BA	2017-08-14 11:50:00+03	210600
366729	0005435999870	D730BA	6535 751108	D730BA	2017-08-14 11:50:00+03	210600
366730	0005435999871	A1AD46	1596 156448	A1AD46	2017-08-13 03:49:00+03	45900
366731	0005435999872	7B6A53	9374 822707	7B6A53	2017-08-15 15:54:00+03	219400
366732	0005435999873	7B6A53	7380 075822	7B6A53	2017-08-15 15:54:00+03	219400

366733 rows × 6 columns

In [30]:

```
tickets = pd.read_sql_query("""select * from tickets inner join bookings on
tickets.book_ref = bookings.book_ref""", connection)
tickets.dtypes
```

Out[30]:

```
ticket_no       object
book_ref        object
passenger_id    object
book_ref        object
book_date       object
total_amount     int64
dtype: object
```

In [47]:

```
tickets
```

Out[47]:

	ticket_no	book_ref	passenger_id	book_ref	book_date	total_amount
0	0005432000987	06B046	8149 604011	06B046	2017-07-05 20:19:00+03	12400
1	0005432000988	06B046	8499 420203	06B046	2017-07-05 20:19:00+03	12400
2	0005432000989	E170C3	1011 752484	E170C3	2017-06-29 01:55:00+03	24700

	ticket_no	book_ref	passenger_id	book_ref	book_date	total_amount
3	0005432000990	E170C3	4849 400049	E170C3	2017-06-29 01:55:00+03	24700
4	0005432000991	F313DD	6615 976589	F313DD	2017-07-03 04:37:00+03	30900
...	...	...	...	...	...	...
366728	0005435999869	D730BA	0474 690760	D730BA	2017-08-14 11:50:00+03	210600
366729	0005435999870	D730BA	6535 751108	D730BA	2017-08-14 11:50:00+03	210600
366730	0005435999871	A1AD46	1596 156448	A1AD46	2017-08-13 03:49:00+03	45900
366731	0005435999872	7B6A53	9374 822707	7B6A53	2017-08-15 15:54:00+03	219400
366732	0005435999873	7B6A53	7380 075822	7B6A53	2017-08-15 15:54:00+03	219400

366733 rows × 6 columns

In [48]:

```
tickets = pd.read_sql_query("""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
tickets
```

Out[48]:

	ticket_no	book_ref	passenger_id	book_ref	book_date	total_amount	date
0	0005432000987	06B046	8149 604011	06B046	2017-07-05 20:19:00+03:00	12400	2017-07-05
1	0005432000988	06B046	8499 420203	06B046	2017-07-05 20:19:00+03:00	12400	2017-07-05
2	0005432000989	E170C3	1011 752484	E170C3	2017-06-29 01:55:00+03:00	24700	2017-06-29
3	0005432000990	E170C3	4849 400049	E170C3	2017-06-29 01:55:00+03:00	24700	2017-06-29

	ticket_no	book_ref	passenger_id	book_ref	book_date	total_amount	date
4	0005432000991	F313DD	6615 976589	F313DD	2017-07-03 04:37:00+03:00	30900	2017-07-03
...	...	...	...	...	...	...	...
366728	0005435999869	D730BA	0474 690760	D730BA	2017-08-14 11:50:00+03:00	210600	2017-08-14
366729	0005435999870	D730BA	6535 751108	D730BA	2017-08-14 11:50:00+03:00	210600	2017-08-14
366730	0005435999871	A1AD46	1596 156448	A1AD46	2017-08-13 03:49:00+03:00	45900	2017-08-13
366731	0005435999872	7B6A53	9374 822707	7B6A53	2017-08-15 15:54:00+03:00	219400	2017-08-15
366732	0005435999873	7B6A53	7380 075822	7B6A53	2017-08-15 15:54:00+03:00	219400	2017-08-15

366733 rows × 7 columns

In [50]:

```
tickets.groupby('date')[['date']].count()
```

Out[50]:

	date
date	
2017-06-21	6
2017-06-22	12
2017-06-23	28
2017-06-24	106
2017-06-25	266

date	
2017-06-26	499
2017-06-27	1028
2017-06-28	1678
2017-06-29	2765
2017-06-30	3772
2017-07-01	4936
2017-07-02	5780
2017-07-03	6686
2017-07-04	7112
2017-07-05	7484
2017-07-06	7656
2017-07-07	7722
2017-07-08	7586
2017-07-09	7860
2017-07-10	7749
2017-07-11	7852

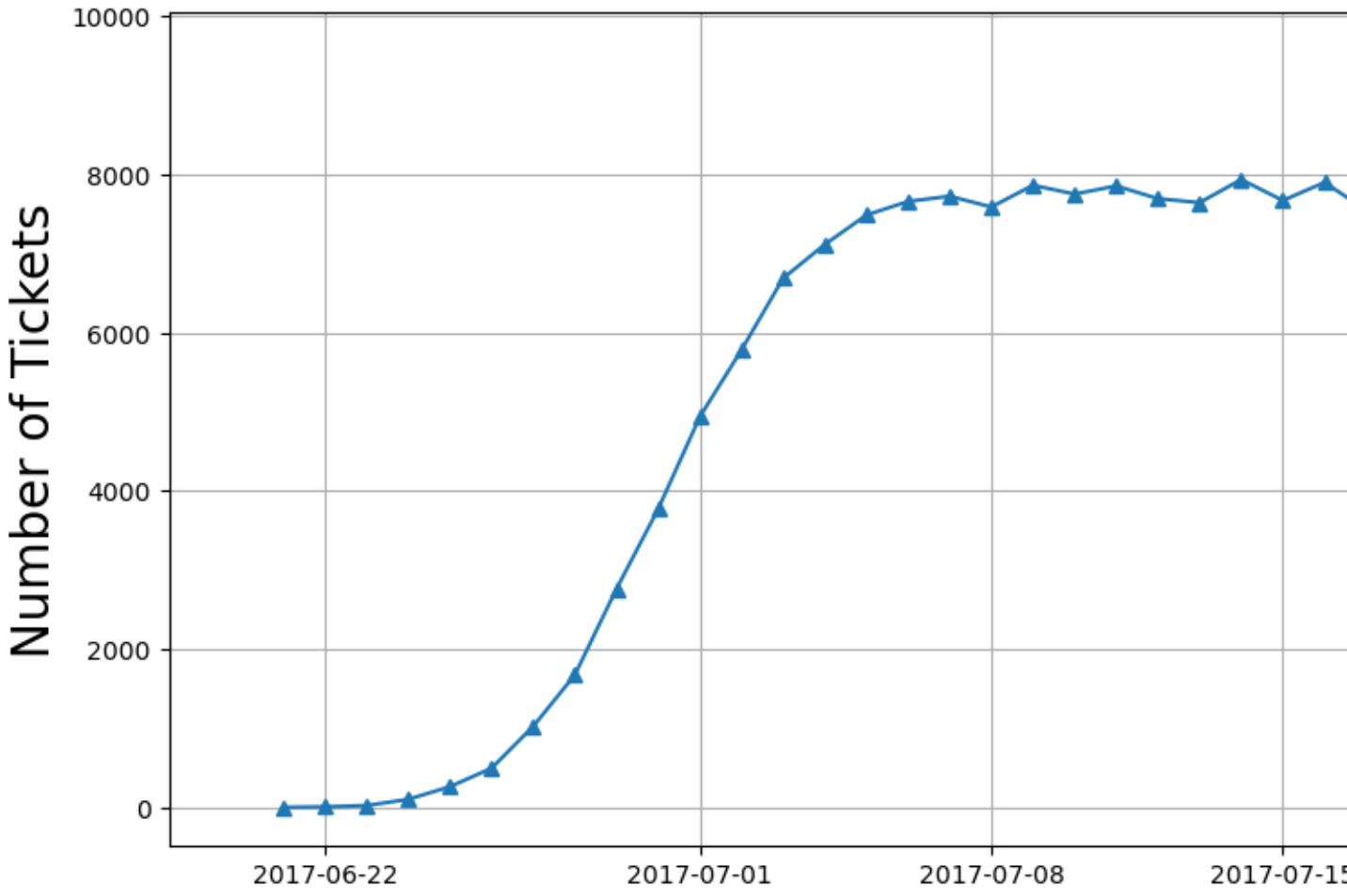
date	
date	
2017-07-12	7691
2017-07-13	7641
2017-07-14	7932
2017-07-15	7668
2017-07-16	7896
2017-07-17	7546
2017-07-18	7745
2017-07-19	7821
2017-07-20	7637
2017-07-21	7771
2017-07-22	7698
2017-07-23	7627
2017-07-24	7667
2017-07-25	7826
2017-07-26	7730
2017-07-27	7636

date	
date	
2017-07-28	7827
2017-07-29	7588
2017-07-30	7732
2017-07-31	7653
2017-08-01	7740
2017-08-02	7669
2017-08-03	7756
2017-08-04	7908
2017-08-05	8064
2017-08-06	8016
2017-08-07	7910
2017-08-08	8153
2017-08-09	8258
2017-08-10	8493
2017-08-11	8737
2017-08-12	8870

	date
date	
2017-08-13	9151
2017-08-14	9574
2017-08-15	7519

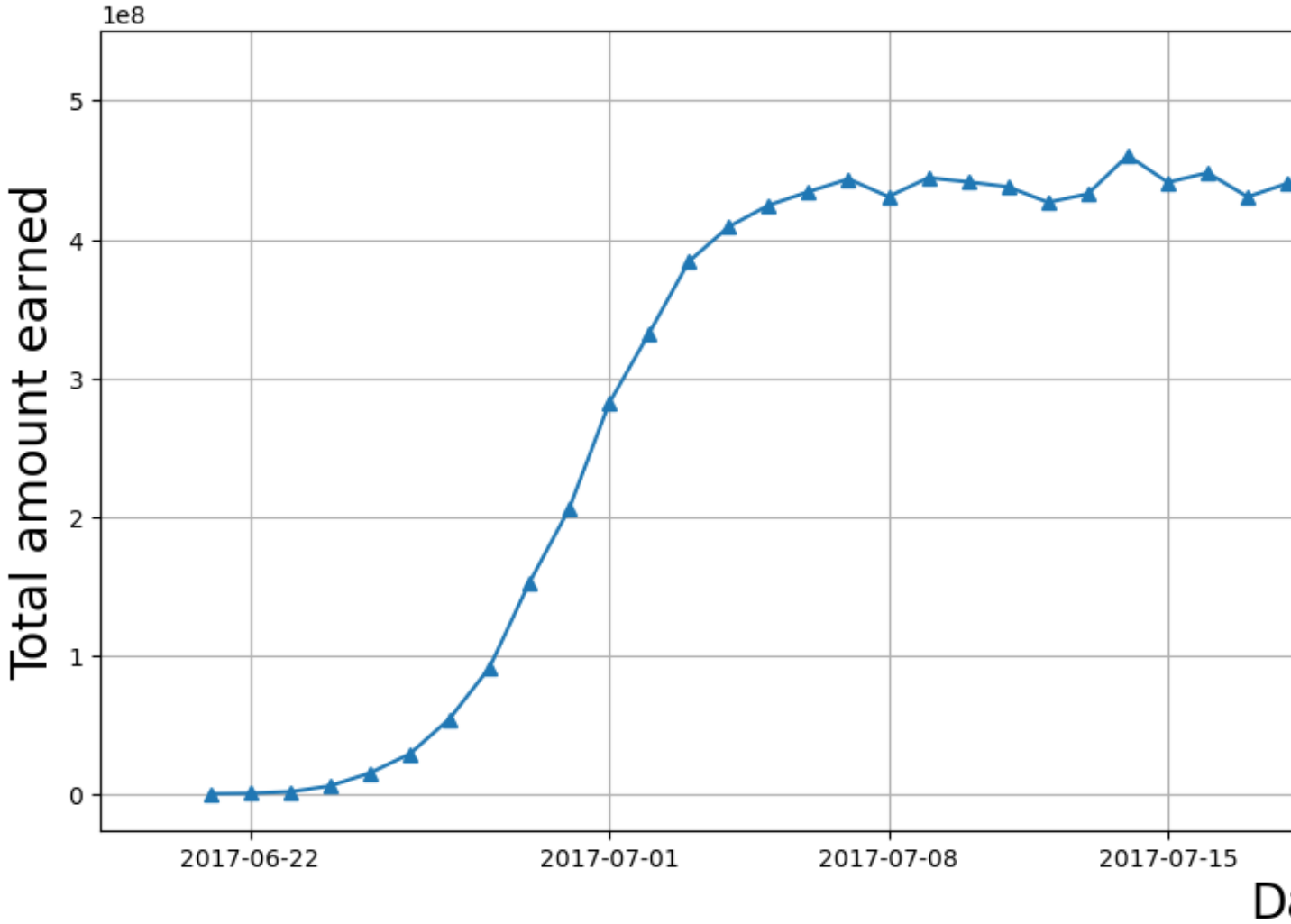
In [51]:

```
tickets = pd.read_sql_query("""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()
```



In [54]:

```
bookings = pd.read_sql_query("select * from bookings", connection)
bookings['book_date'] = pd.to_datetime(bookings['book_date'])
bookings['date'] = bookings['book_date'].dt.date
x = bookings.groupby('date')[['total_amount']].sum()

plt.figure(figsize = (18,6))
plt.plot(x.index, x['total_amount'],marker = '^')
plt.xlabel('Date', fontsize = 20)
plt.ylabel('Total amount earned', fontsize = 20)
plt.grid('b')
plt.show()
```



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

In [55]:

```
df = pd.read_sql_query("""select* from ticket_flights
                  join flights on ticket_flights.flight_id =
flights.flight_id""", connection)
```

In [56]:

```
df
```

Out[56]:

	ticket_no	flight_id	fare_conditions	amount	flight_id	flight_no	scheduled_departure	scheduled_arrival	departure_airport	arrival_airport	status	aircraft_code	actual_departure	actual_arrival
0	0005432159776	30625	Business	42100	30625	PG0013	2017-07-16 18:15:00+03	2017-07-16 20:00:00+03	AER	SVO	Arrived	773	2017-07-16 18:18:00+03	2017-07-16 20:04:00+03
1	0005435212351	30625	Business	42100	30625	PG0013	2017-07-16 18:15:00+03	2017-07-16 20:00:00+03	AER	SVO	Arrived	773	2017-07-16 18:18:00+03	2017-07-16 20:04:00+03
2	0005435212386	30625	Business	42100	30625	PG0013	2017-07-16 18:15:00+03	2017-07-16 20:00:00+03	AER	SVO	Arrived	773	2017-07-16 18:18:00+03	2017-07-16 20:04:00+03
3	0005435212381	30625	Business	42100	30625	PG0013	2017-07-16 18:15:00+03	2017-07-16 20:00:00+03	AER	SVO	Arrived	773	2017-07-16 18:18:00+03	2017-07-16 20:04:00+03
4	0005432211370	30625	Business	42100	30625	PG0013	2017-07-16 18:15:00+03	2017-07-16 20:00:00+03	AER	SVO	Arrived	773	2017-07-16 18:18:00+03	2017-07-16 20:04:00+03
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
1045721	0005435097522	32094	Economy	5200	32094	PG0708	2017-09-14 17:15:00+03	2017-09-14 18:00:00+03	SGC	OVS	Scheduled	733	\N	\N

	ticket_no	flight_id	fare_conditions	amount	flight_id	flight_no	scheduled_departure	scheduled_arrival	departure_airport	arrival_airport	status	aircraft_code	actual_departure	actual_arrival
1045722	0005435097521	32094	Economy	5200	32094	PG0708	2017-09-14 17:15:00+03	2017-09-14 18:00:00+03	SGC	OVS	Scheduled	733	\N	\N
1045723	0005435104384	32094	Economy	5200	32094	PG0708	2017-09-14 17:15:00+03	2017-09-14 18:00:00+03	SGC	OVS	Scheduled	733	\N	\N
1045724	0005435104352	32094	Economy	5200	32094	PG0708	2017-09-14 17:15:00+03	2017-09-14 18:00:00+03	SGC	OVS	Scheduled	733	\N	\N
1045725	0005435104389	32094	Economy	5200	32094	PG0708	2017-09-14 17:15:00+03	2017-09-14 18:00:00+03	SGC	OVS	Scheduled	733	\N	\N

1045726 rows × 14 columns

In [57]:

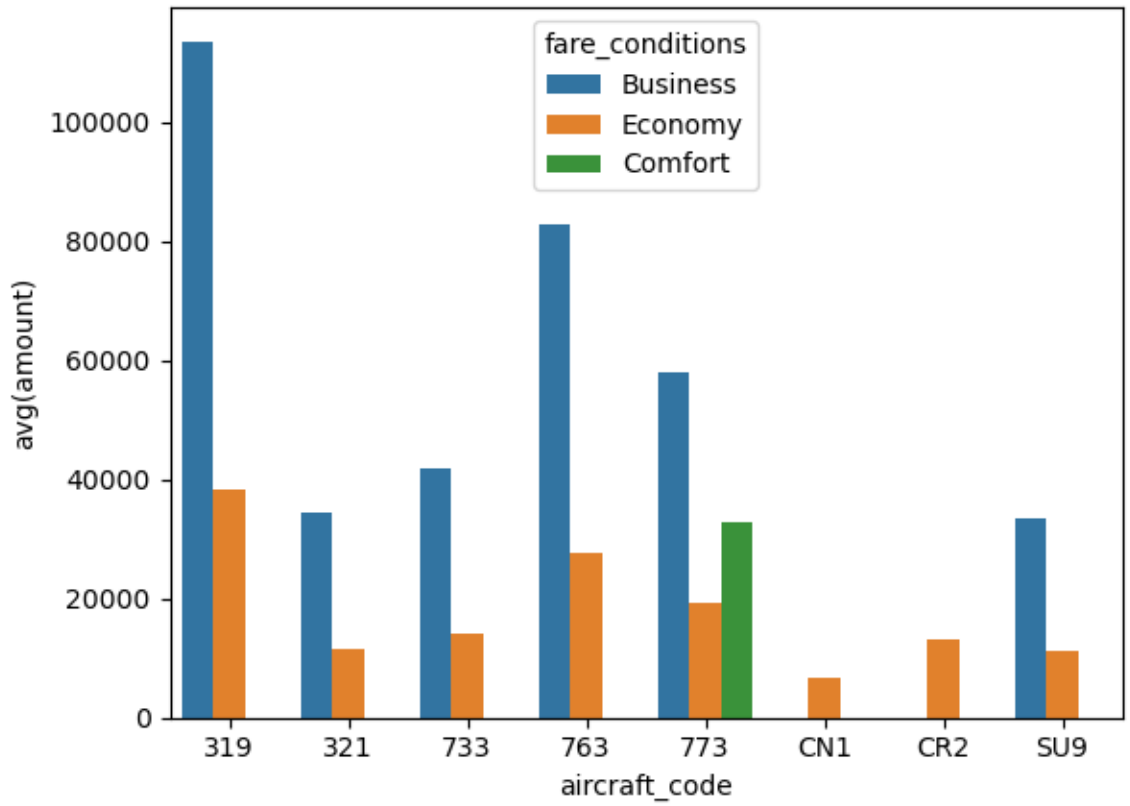
```
df = pd.read_sql_query("""select fare_conditions, aircraft_code, avg(amount)
from ticket_flights
                        join flights on ticket_flights.flight_id =
flights.flight_id group by aircraft_code, fare_conditions""",connection)
```

In [58]:

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

Out[58]:

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



Analyzing occupancy rate

1. For each aircraft, calculate the total revenue per year and the average revenue per ticket

In [59]:

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

Out[59]:

	aircraft_code	ticket_count	total_revenue	total_revenue/ticket_count
0	319	52853	2706163100	51201
1	321	107129	1638164100	15291

	aircraft_code	ticket_count	total_revenue	total_revenue/ticket_count
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

2. Calculate the average occupancy per aircraft

In [60]:

```
pd.read_sql_query("""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""", connection)
```

Out[60]:

	aircraft_code	flight_id	seats_count
0	319	1162	51
1	319	1166	54
2	319	1167	57
3	319	1168	60
4	319	1170	58
...	...	...	...

	aircraft_code	flight_id	seats_count
11513	SU9	32925	12
11514	SU9	32928	25
11515	SU9	32931	12
11516	SU9	32933	16
11517	SU9	32937	6

11518 rows × 3 columns

In [62]:

```
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)
```

Out[62]:

	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

	aircraft_code	booked_seats	num_seats	occupancy_rate
5	CN1	6.004431	12	0.500369
6	CR2	21.482847	50	0.429657
7	SU9	56.812113	97	0.585692

In [64]:

```
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
```

Out[64]:

	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

	aircraft_code	booked_seats	num_seats	occupancy_rate
7	SU9	56.812113	97	0.585692

3. Calculate by how much the total annual turnover could increase by giving all aircraft a 10% higher occupancy rate.

In [70]:

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

Out[70]:

	aircraft_code	booked_seats	num_seats	occupancy_rate	Inc occupancy rate	Inc occupancy_rate
0	319	53.583181	116	0.461924	0.508116	0.508116
1	321	88.809231	170	0.522407	0.574648	0.574648
2	733	80.255462	130	0.617350	0.679085	0.679085
3	763	113.937294	222	0.513231	0.564554	0.564554
4	773	264.925806	402	0.659019	0.724921	0.724921
5	CN1	6.004431	12	0.500369	0.550406	0.550406
6	CR2	21.482847	50	0.429657	0.472623	0.472623
7	SU9	56.812113	97	0.585692	0.644261	0.644261

In [71]:

```
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)
total_revenue
```

Out[71]:

	aircraft_code	total_revenue
0	319	2706163100
1	321	1638164100
2	733	1426552100
3	763	4371277100
4	773	3431205500
5	CN1	96373800
6	CR2	1982760500
7	SU9	5114484700

In [72]:
```
pd.set_option("display.float_format", str)
```

In [74]:
```
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_r
ate['Inc occupancy rate']
occupancy_rate
```

Out[74]:

	aircraft_code	booked_seats	num_seats	occupancy_rate	Inc occupancy rate	Inc occupancy_rate	Inc Total Annual turnover
0	319	53.583180987202292	116	0.46192397402761143	0.5081163714303726	0.5081163714303726	2976779410.0
1	321	88.809230769230770 77	170	0.52240723981900450	0.5746479638009050	0.5746479638009050	1801980510.0

	aircraft_code	booked_seats	num_seats	occupancy_rate	Inc occupancy rate	Inc occupancy_rate	Inc Total Annual turnover
2	733	80.25546218487395	130	0.617349709114415	0.6790846800258565	0.6790846800258565	1569207310.000000002
3	763	113.93729372937294	222	0.5132310528350132	0.5645541581185146	0.5645541581185146	4808404810.0
4	773	264.9258064516129	402	0.659019419033863	0.7249213609372492	0.7249213609372492	3774326050.0
5	CN1	6.004431314623338	12	0.5003692762186115	0.5504062038404727	0.5504062038404727	106011180.000000001
6	CR2	21.482846902200174	50	0.42965693804403476	0.4726226318484382	0.4726226318484382	2181036550.0
7	SU9	56.81211267605634	97	0.5856918832583128	0.6442610715844144	0.6442610715844144	5625933169.999999