airlines-data-analysis

June 21, 2024

1 Objective

The goal of this SQL-based data analysis project is to identify opportunities to increase the occupancy rate on low-performing flights, ultimately leading to increased profitability for the airline.

2 Importing Libraries

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

3 Database Connection

[2]:
```
file_path='C:\\Users\\abc\\Desktop\\travel.sqlite'
```

[3]:
```
connection=sqlite3.connect(file_path)
cursor=connection.cursor()
```

[4]:
```
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
```

[4]:
```
['aircrafts_data',
 'airports_data',
 'boarding_passes',
 'bookings',
 'flights',
 'seats',
 'ticket_flights',
 'tickets']
```

4 Data Exploration

[5]:
```
aircrafts_data=pd.read_sql_query("select * from aircrafts_data",connection)
aircrafts_data
```

[5]:
```
  aircraft_code                                          model  range
0           773    {"en": "Boeing 777-300", "ru": "    777-300"}  11100
1           763    {"en": "Boeing 767-300", "ru": "    767-300"}   7900
2           SU9  {"en": "Sukhoi Superjet-100", "ru": "      …   3000
3           320  {"en": "Airbus A320-200", "ru": "     A320-…   5700
4           321  {"en": "Airbus A321-200", "ru": "     A321-…   5600
5           319  {"en": "Airbus A319-100", "ru": "     A319-…   6700
6           733    {"en": "Boeing 737-300", "ru": "    737-300"}   4200
7           CN1  {"en": "Cessna 208 Caravan", "ru": "    208…   1200
8           CR2  {"en": "Bombardier CRJ-200", "ru": "      …   2700
```

[6]:
```
airports_data=pd.read_sql_query("select * from airports_data",connection)
airports_data
```

[6]:
```
    airport_code                                      airport_name  \
0            YKS          {"en": "Yakutsk Airport", "ru": "    "}
1            MJZ            {"en": "Mirny Airport", "ru": "    "}
2            KHV  {"en": "Khabarovsk-Novy Airport", "ru": "   …
3            PKC        {"en": "Yelizovo Airport", "ru": "    "}
4            UUS  {"en": "Yuzhno-Sakhalinsk Airport", "ru": "  …
..           …                                                 …
99           MMK       {"en": "Murmansk Airport", "ru": "     "}
100          ABA           {"en": "Abakan Airport", "ru": "    "}
101          BAX         {"en": "Barnaul Airport", "ru": "    "}
102          AAQ  {"en": "Anapa Vityazevo Airport", "ru": "   …
103          CNN         {"en": "Chulman Airport", "ru": "    "}

                                           city  \
0                   {"en": "Yakutsk", "ru": "    "}
1                    {"en": "Mirnyj", "ru": "    "}
2              {"en": "Khabarovsk", "ru": "     "}
3    {"en": "Petropavlovsk", "ru": "       - …
4    {"en": "Yuzhno-Sakhalinsk", "ru": "  -   …
..                                             …
99               {"en": "Murmansk", "ru": "     "}
100                  {"en": "Abakan", "ru": "    "}
101                {"en": "Barnaul", "ru": "    "}
102                    {"en": "Anapa", "ru": "   "}
103              {"en": "Neryungri", "ru": "     "}

                                coordinates          timezone
0       (129.77099609375,62.0932998657226562)      Asia/Yakutsk
```

```
1      (114.03900146484375,62.534698486328125)      Asia/Yakutsk
2        (135.18800354004,48.5279998779300001)  Asia/Vladivostok
3    (158.453994750976562,53.1679000854492188)    Asia/Kamchatka
4    (142.718002319335938,46.8886985778808594)     Asia/Sakhalin
..                                          …               …
99   (32.7508010864257812,68.7817001342773438)     Europe/Moscow
100  (91.3850021362304688,53.7400016784667969)  Asia/Krasnoyarsk
101   (83.5384979248046875,53.363800048828125)  Asia/Krasnoyarsk
102   (37.3473014831539984,45.002101898192997)     Europe/Moscow
103  (124.914001464839998,56.9138984680179973)      Asia/Yakutsk

[104 rows x 5 columns]
```

[7]:
```
boarding_passes=pd.read_sql_query("select * from boarding_passes",connection)
boarding_passes
```

[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 x 4 columns]
```

[8]:
```
bookings=pd.read_sql_query("select * from bookings",connection)
bookings
```

[8]:
```
        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 x 3 columns]
```

[9]:
```
flights=pd.read_sql_query("select * from flights",connection)
flights
```

[9]:
```
       flight_id flight_no     scheduled_departure       scheduled_arrival  \
0           1185    PG0134  2017-09-10 09:50:00+03  2017-09-10 14:55:00+03
1           3979    PG0052  2017-08-25 14:50:00+03  2017-08-25 17:35:00+03
2           4739    PG0561  2017-09-05 12:30:00+03  2017-09-05 14:15:00+03
3           5502    PG0529  2017-09-12 09:50:00+03  2017-09-12 11:20:00+03
4           6938    PG0461  2017-09-04 12:25:00+03  2017-09-04 13:20:00+03
…            …       …                       …                       …
33116      33117    PG0063  2017-08-02 19:25:00+03  2017-08-02 20:10:00+03
33117      33118    PG0063  2017-07-28 19:25:00+03  2017-07-28 20:10:00+03
33118      33119    PG0063  2017-09-08 19:25:00+03  2017-09-08 20:10:00+03
33119      33120    PG0063  2017-08-01 19:25:00+03  2017-08-01 20:10:00+03
33120      33121    PG0063  2017-08-26 19:25:00+03  2017-08-26 20:10:00+03

      departure_airport arrival_airport     status aircraft_code  \
0                   DME             BTK  Scheduled           319
1                   VKO             HMA  Scheduled           CR2
2                   VKO             AER  Scheduled           763
3                   SVO             UFA  Scheduled           763
4                   SVO             ULV  Scheduled           SU9
…                  …               …       …            …
33116               SKX             SVO    Arrived           CR2
33117               SKX             SVO    Arrived           CR2
33118               SKX             SVO  Scheduled           CR2
33119               SKX             SVO    Arrived           CR2
33120               SKX             SVO  Scheduled           CR2

             actual_departure          actual_arrival
0                          \N                      \N
1                          \N                      \N
2                          \N                      \N
3                          \N                      \N
4                          \N                      \N
…                         …                       …
33116  2017-08-02 19:25:00+03  2017-08-02 20:10:00+03
33117  2017-07-28 19:30:00+03  2017-07-28 20:15:00+03
33118                      \N                      \N
33119  2017-08-01 19:26:00+03  2017-08-01 20:12:00+03
33120                      \N                      \N

[33121 rows x 10 columns]
```

[10]:
```
seats=pd.read_sql_query("select * from seats",connection)
seats
```

[10]:

	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 x 3 columns]

[11]:
```
ticket_flights=pd.read_sql_query("select * from ticket_flights",connection)
ticket_flights
```

[11]:

	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 x 4 columns]

[12]:
```
tickets=pd.read_sql_query("select * from tickets",connection)
tickets
```

[12]:

	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
…	…	…	…
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 x 3 columns]
```

[13]:
```
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)')
```

[14]:
```
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
```

5 Basic Analysis

6 How many planes have more than 100 seats?

[15]:
```
pd.read_sql_query("""select aircraft_code,count(*) as num_seats from seats
                     group by aircraft_code having num_seats >␣
 ↪100""",connection)
```

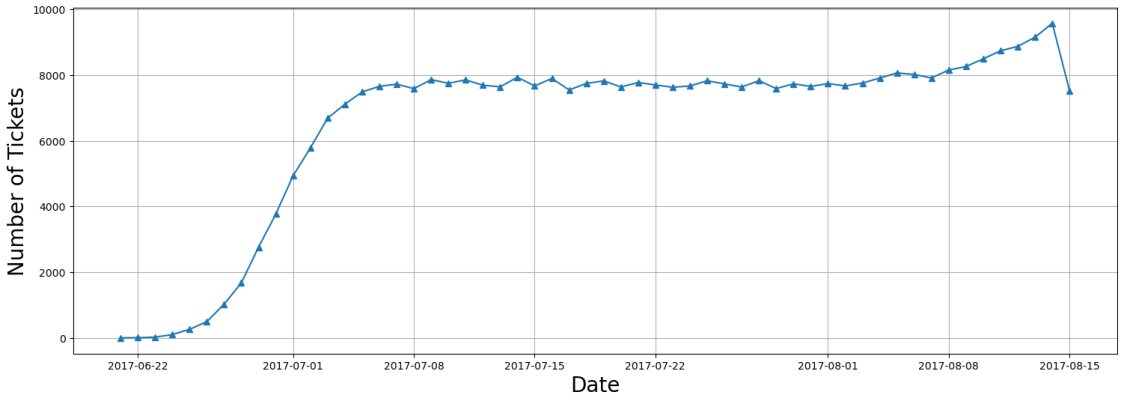
[15]:

	aircraft_code	num_seats
0	319	116
1	320	140

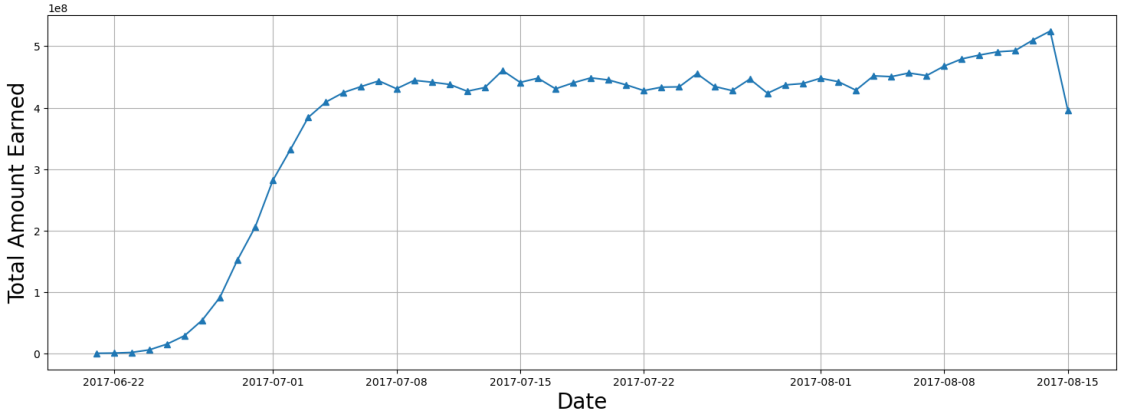
2	321	170
3	733	130
4	763	222
5	773	402

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

[16]:
```
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').size().reset_index(name='count')
plt.figure(figsize=(18,6))
plt.plot(x['date'],x['count'],marker='^')
plt.xlabel('Date',fontsize=20)
plt.ylabel('Number of Tickets',fontsize=20)
plt.grid('b')
plt.show()
```



[17]:
```
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()
```

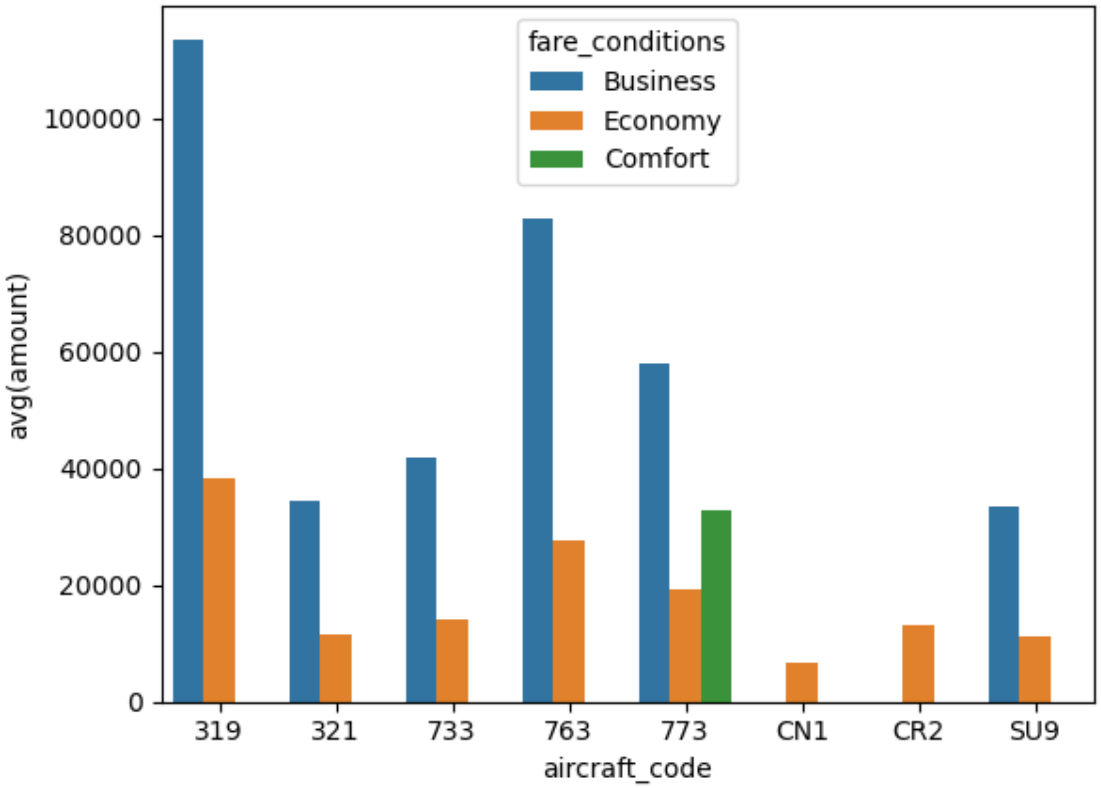


8 Calculate the average charges for each aircraft with different fare conditions.

[18]:
```
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)
```

[19]:
```
sns.barplot(data=df, x='aircraft_code', y='avg(amount)', hue='fare_conditions')
```

[19]:
```
<AxesSubplot:xlabel='aircraft_code', ylabel='avg(amount)'>
```



9 Analyzing occupancy rate

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

[20]:
```
pd.read_sql_query("""select aircraft_code, tickets_count, total_revenue,␣
 ↪total_revenue/tickets_count as avg_revenue_per_ticket from
(select aircraft_code, count(*) as tickets_count, sum(amount) as total_revenue␣
 ↪from ticket_flights
                    join flights on ticket_flights.flight_id=flights.flight_id␣
 ↪group by aircraft_code)""",connection)
```

[20]:

	aircraft_code	tickets_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
```

11 Calculate the average occupancy per aircraft.

[21]:
```
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
```

[21]:

	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

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

[22]:
```
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
0	319	53.583181	116	0.461924	0.508116
1	321	88.809231	170	0.522407	0.574648
2	733	80.255462	130	0.617350	0.679085
3	763	113.937294	222	0.513231	0.564554
4	773	264.925806	402	0.659019	0.724921
5	CN1	6.004431	12	0.500369	0.550406
6	CR2	21.482847	50	0.429657	0.472623
7	SU9	56.812113	97	0.585692	0.644261

[23]:
```
pd.set_option("display.float_format",str)
```

[24]:
```
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']
occupancy_rate
```

[24]:

	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	

	Inc occupancy_rate	Inc Total Annual Turnover
0	0.5081163714303726	2976779410.0
1	0.574647963800905	1801980510.0
2	0.6790846800258565	1569207310.0000002
3	0.5645541581185146	4808404810.0
4	0.7249213609372492	3774326050.0
5	0.5504062038404727	106011180.00000001
6	0.4726226318484382	2181036550.0

```
7   0.644261071584144          5625933169.999999
```

[ ]: