

AtliQ Hotels Data Analysis Project

1) Data Import and Data Exploration

Datasets

We have 5 csv files:

- dim_date.csv
- dim_hotels.csv
- dim_rooms.csv
- fact_aggregated_bookings.csv
- fact_bookings.csv
- new_data_aug

Read all data files in different dataframes

In [108...]

```
import pandas as pd

# importing all tables
df_bookings = pd.read_csv("datasets/fact_bookings.csv")
df_date = pd.read_csv("datasets/dim_date.csv")
df_hotels = pd.read_csv("datasets/dim_hotels.csv")
df_rooms = pd.read_csv("datasets/dim_rooms.csv")
df_agg_bookings = pd.read_csv("datasets/fact_aggregated_bookings.csv")
df_new_aug = pd.read_csv("datasets/new_data_august.csv")
```

Explore fact_bookings data

In [108...]

```
df_bookings.describe()
```

Out[108...]

	property_id	no_guests	ratings_given	revenue_generated	revenue_realized
count	134590.000000	134587.000000	56683.000000	1.345900e+05	134590.000000
mean	18061.113493	2.036170	3.619004	1.537805e+04	12696.123256
std	1093.055847	1.034885	1.235009	9.303604e+04	6928.108124
min	16558.000000	-17.000000	1.000000	6.500000e+03	2600.000000
25%	17558.000000	1.000000	3.000000	9.900000e+03	7600.000000
50%	17564.000000	2.000000	4.000000	1.350000e+04	11700.000000
75%	18563.000000	2.000000	5.000000	1.800000e+04	15300.000000
max	19563.000000	6.000000	5.000000	2.856000e+07	45220.000000

In [108...]

df_bookings.columns

Out[108...]

```
Index(['booking_id', 'property_id', 'booking_date', 'check_in_date',
       'checkout_date', 'no_guests', 'room_category', 'booking_platform',
       'ratings_given', 'booking_status', 'revenue_generated',
       'revenue_realized'],
      dtype='object')
```

In [108...]

df_bookings.head(5)

Out[108...]

	booking_id	property_id	booking_date	check_in_date	checkout_date	no_guests
0	May012216558RT11	16558	27-04-22	1/5/2022	2/5/2022	-3.0
1	May012216558RT12	16558	30-04-22	1/5/2022	2/5/2022	2.0
2	May012216558RT13	16558	28-04-22	1/5/2022	4/5/2022	2.0
3	May012216558RT14	16558	28-04-22	1/5/2022	2/5/2022	-2.0
4	May012216558RT15	16558	27-04-22	1/5/2022	2/5/2022	4.0

In [108...]

df_bookings.room_category.value_counts()

Out[108...]

```
room_category
RT2    49505
RT1    38446
RT3    30566
RT4    16073
Name: count, dtype: int64
```

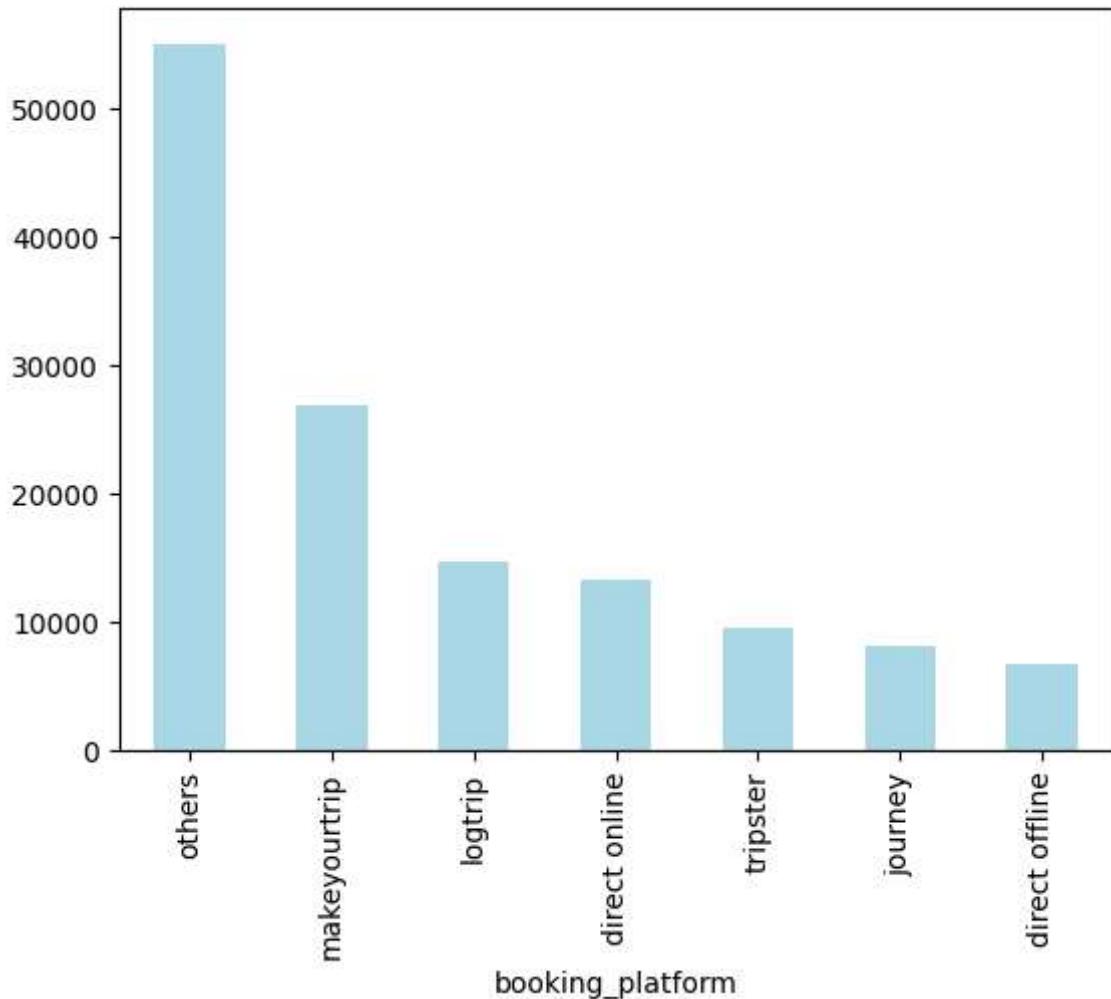
In [109...]

df_bookings.booking_platform.value_counts()

```
Out[109... booking_platform  
others      55066  
makeyourtrip 26898  
logtrip     14756  
direct online 13379  
tripster     9630  
journey      8106  
direct offline 6755  
Name: count, dtype: int64
```

```
In [116... df_bookings.booking_platform.value_counts().plot(kind="bar", color="lightblue")
```

```
Out[116... <Axes: xlabel='booking_platform'>
```



Exploring remaining datasets

```
In [109... df_hotels.head()
```

Out[109...]

	property_id	property_name	category	city
0	16558	Atliq Grands	Luxury	Delhi
1	16559	Atliq Exotica	Luxury	Mumbai
2	16560	Atliq City	Business	Delhi
3	16561	Atliq Blu	Luxury	Delhi
4	16562	Atliq Bay	Luxury	Delhi

In [109...]

df_rooms.head()

Out[109...]

	room_id	room_class
0	RT1	Standard
1	RT2	Elite
2	RT3	Premium
3	RT4	Presidential

In [109...]

df_agg_bookings.head()

Out[109...]

	property_id	check_in_date	room_category	successful_bookings	capacity
0	16559	1-May-22	RT1	25	30.0
1	19562	1-May-22	RT1	28	30.0
2	19563	1-May-22	RT1	23	30.0
3	17558	1-May-22	RT1	30	19.0
4	16558	1-May-22	RT1	18	19.0

In [109...]

df_new_aug.head()

Out[109...]

	property_id	property_name	category	city	room_category	room_class	check_in_date
0	16559	Atliq Exotica	Luxury	Mumbai	RT1	Standard	01-Aug-2023
1	19562	Atliq Bay	Luxury	Bangalore	RT1	Standard	01-Aug-2023
2	19563	Atliq Palace	Business	Bangalore	RT1	Standard	01-Aug-2023
3	19558	Atliq Grands	Luxury	Bangalore	RT1	Standard	01-Aug-2023
4	19560	Atliq City	Business	Bangalore	RT1	Standard	01-Aug-2023

2) Data Cleaning

In [109...]

```
#here we removed all 7 rows with negative no_guests
df_bookings = df_bookings[df_bookings.no_guests > 0]

#There are outliers for revenue_generated column. So to detect them we use 3*std approach
#First check if there are any negative values

df_bookings[df_bookings.revenue_generated < 0]
#There are no negative values
```

Out[109...]

	booking_id	property_id	booking_date	check_in_date	checkout_date	no_guests	room_c...
2	May012216558RT13	16558	28-04-22	1/5/2022	4/5/2022	4	

In [109...]

```
avg, std = df_bookings.revenue_generated.mean(), df_bookings.revenue_generated.std()
upper_limit_1 = avg + 3*std
#Lower Limit is not required. Logically, 0 is the Lower Limit.
df_bookings[df_bookings.revenue_generated > upper_limit_1]
```

Out[109...]

	booking_id	property_id	booking_date	check_in_date	checkout_date	no_...
2	May012216558RT13	16558	28-04-22	1/5/2022	4/5/2022	4
111	May012216559RT32	16559	29-04-22	1/5/2022	2/5/2022	2
315	May012216562RT22	16562	28-04-22	1/5/2022	4/5/2022	4
562	May012217559RT118	17559	26-04-22	1/5/2022	2/5/2022	2
129176	Jul282216562RT26	16562	21-07-22	28-07-22	29-07-22	1

```
In [109... #removing all 5 rows with invalid revenue_generated values
df_bookings = df_bookings[df_bookings.revenue_generated < upper_limit_1]
upper_limit_1
```

```
Out[109... np.float64(294498.50173207896)
```

```
In [109... df_bookings.shape
```

```
Out[109... (134573, 12)
```

```
In [110... #Checking for column revenue_realized
df_bookings[df_bookings.revenue_realized < 0]
```

```
Out[110... booking_id property_id booking_date check_in_date checkout_date no_guests room_c
```

```
In [110... #Check the outliers
upper_limit_2 = df_bookings.revenue_realized.mean() + 3 * df_bookings.revenue_realized.std()
print("Upper Limit:", upper_limit_2)
df_bookings[df_bookings.revenue_realized > upper_limit_2]
```

Upper Limit: 33479.358661845814

	booking_id	property_id	booking_date	check_in_date	checkout_date	no_guests	room_category
137	May012216559RT41	16559	27-04-22	1/5/2022	7/5/2022	1	RT4
139	May012216559RT43	16559	1/5/2022	1/5/2022	2/5/2022	1	RT4
143	May012216559RT47	16559	28-04-22	1/5/2022	3/5/2022	1	RT4
149	May012216559RT413	16559	24-04-22	1/5/2022	7/5/2022	1	RT4
222	May012216560RT45	16560	30-04-22	1/5/2022	3/5/2022	1	RT4
...
134328	Jul312219560RT49	19560	31-07-22	31-07-22	2/8/2022	1	RT4
134331	Jul312219560RT412	19560	31-07-22	31-07-22	1/8/2022	1	RT4
134467	Jul312219562RT45	19562	28-07-22	31-07-22	1/8/2022	1	RT4
134474	Jul312219562RT412	19562	25-07-22	31-07-22	6/8/2022	1	RT4
134581	Jul312217564RT42	17564	31-07-22	31-07-22	1/8/2022	1	RT4

1299 rows × 12 columns

One observation we can have in above dataframe is that all rooms are RT4 which means presidential suit. Now since RT4 is a luxurious room it is likely their rent will be higher. To make a fair analysis, we need to do data analysis only on RT4 room types

```
In [110... df_bookings.room_category.value_counts()
```

```
Out[110... room_category
RT2    49500
RT1    38441
RT3    30561
RT4    16071
Name: count, dtype: int64
```

```
In [110... df_bookings[df_bookings.room_category == "RT4"].revenue_realized.describe()
```

```
Out[110... count    16071.000000
mean     23439.308444
std      9048.599076
min      7600.000000
25%     19000.000000
50%     26600.000000
75%     32300.000000
max     45220.000000
Name: revenue_realized, dtype: float64
```

```
In [110... 23439 + 3 * 9048
```

```
Out[110... 50583
```

Here higher limit comes to be 50583 and in our dataframe above we can see that max value for revenue realized is 45220. Hence we can conclude that there is no outlier and we don't need to do any data cleaning on this particular column.

```
In [110... df_bookings.isnull().sum()
```

```
Out[110... booking_id          0
property_id         0
booking_date        0
check_in_date       0
checkout_date       0
no_guests           0
room_category        0
booking_platform    0
ratings_given      77897
booking_status       0
revenue_generated    0
revenue_realized     0
dtype: int64
```

```
In [110... df_bookings.shape
```

```
Out[110... (134573, 12)
```

Total values in our dataframe is 134573. Out of that 77899 rows has null rating. Since there are many rows with null rating, we should not filter these values. Also we should not replace this rating with a median or mean rating etc.

Now, Let's work on fact_aggregated_bookings

In [110... df_agg_bookings.head()

	property_id	check_in_date	room_category	successful_bookings	capacity
0	16559	1-May-22	RT1	25	30.0
1	19562	1-May-22	RT1	28	30.0
2	19563	1-May-22	RT1	23	30.0
3	17558	1-May-22	RT1	30	19.0
4	16558	1-May-22	RT1	18	19.0

In [110... df_agg_bookings.isnull().sum()

Out[110... property_id 0
check_in_date 0
room_category 0
successful_bookings 0
capacity 2
dtype: int64

In [110... df_agg_bookings[df_agg_bookings.capacity.isna()]

	property_id	check_in_date	room_category	successful_bookings	capacity
8	17561	1-May-22	RT1	22	NaN
14	17562	1-May-22	RT1	12	NaN

In [111... df_agg_bookings.capacity.median()

Out[111... np.float64(25.0)

In [111... df_agg_bookings.fillna(df_agg_bookings.capacity.median(), inplace = True)
df_agg_bookings.isnull().sum()

Out[111... property_id 0
check_in_date 0
room_category 0
successful_bookings 0
capacity 0
dtype: int64

**Finding out records that have successful_bookings value greater than capacity. Filtering those records.

In [111... df_agg_bookings[df_agg_bookings.successful_bookings > df_agg_bookings.capacity]

Out[111...]

	property_id	check_in_date	room_category	successful_bookings	capacity
3	17558	1-May-22	RT1	30	19.0
12	16563	1-May-22	RT1	100	41.0
4136	19558	11-Jun-22	RT2	50	39.0
6209	19560	2-Jul-22	RT1	123	26.0
8522	19559	25-Jul-22	RT1	35	24.0
9194	18563	31-Jul-22	RT4	20	18.0

In [111...]

df_agg_bookings.shape

Out[111...]

(9200, 5)

In [111...]

df_agg_bookings = df_agg_bookings[df_agg_bookings.successful_bookings <= df_agg_bookings.capacity] df_agg_bookings.shape

Out[111...]

(9194, 5)

3) Data Transformation

Creating occupancy percentage column

In [111...]

```
df_agg_bookings["occ_pct"] = df_agg_bookings.successful_bookings / df_agg_bookings.capacity
df_agg_bookings["occ_pct"] = df_agg_bookings["occ_pct"].apply(lambda x: round(x * 100))
df_agg_bookings.head()
```

Out[111...]

	property_id	check_in_date	room_category	successful_bookings	capacity	occ_pct
0	16559	1-May-22	RT1	25	30.0	83.33
1	19562	1-May-22	RT1	28	30.0	93.33
2	19563	1-May-22	RT1	23	30.0	76.67
4	16558	1-May-22	RT1	18	19.0	94.74
5	17560	1-May-22	RT1	28	40.0	70.00

4) Insights Generation

1. What is the average occupancy rate in each of the room categories?

```
In [111... df_agg_bookings.groupby("room_category")["occ_pct"].mean()
```

```
Out[111... room_category
RT1    57.889643
RT2    58.009756
RT3    58.028213
RT4    59.277925
Name: occ_pct, dtype: float64
```

```
In [111... df_rooms
```

	room_id	room_class
0	RT1	Standard
1	RT2	Elite
2	RT3	Premium
3	RT4	Presidential

```
In [111... #merging these two dataframes
```

```
pd.merge(df_rooms, df_agg_bookings.groupby("room_category")["occ_pct"].mean(), left
```

	room_id	room_class	occ_pct
0	RT1	Standard	57.889643
1	RT2	Elite	58.009756
2	RT3	Premium	58.028213
3	RT4	Presidential	59.277925

```
In [111... #merging df_agg_bookings and df_rooms using the key room_id
```

```
df = pd.merge(df_agg_bookings, df_rooms, left_on = "room_category", right_on = "room
df.head()
```

	property_id	check_in_date	room_category	successful_bookings	capacity	occ_pct	room
0	16559	1-May-22	RT1		25	30.0	83.33
1	19562	1-May-22	RT1		28	30.0	93.33
2	19563	1-May-22	RT1		23	30.0	76.67
3	16558	1-May-22	RT1		18	19.0	94.74
4	17560	1-May-22	RT1		28	40.0	70.00

◀ ▶

```
In [112... df.shape
```

```
Out[112... (9194, 8)
```

```
In [112... df.drop("room_id", axis = 1, inplace= True)
df.head()
```

```
Out[112...   property_id  check_in_date  room_category  successful_bookings  capacity  occ_pct  room_class
0           16559      1-May-22          RT1                  25       30.0    83.33  Standard
1           19562      1-May-22          RT1                  28       30.0    93.33  Standard
2           19563      1-May-22          RT1                  23       30.0    76.67  Standard
3           16558      1-May-22          RT1                  18       19.0    94.74  Standard
4           17560      1-May-22          RT1                  28       40.0    70.00  Standard
```

```
In [112... df.groupby("room_class")["occ_pct"].mean()
```

```
Out[112...   room_class
Elite            58.009756
Premium          58.028213
Presidential     59.277925
Standard          57.889643
Name: occ_pct, dtype: float64
```

2. What is the average occupancy rate per city?

```
In [112... df_hotels.head()
```

```
Out[112...   property_id  property_name  category  city
0           16558      Atliq Grands  Luxury    Delhi
1           16559      Atliq Exotica  Luxury    Mumbai
2           16560      Atliq City    Business  Delhi
3           16561      Atliq Blu     Luxury    Delhi
4           16562      Atliq Bay     Luxury    Delhi
```

```
In [112... df = pd.merge(df,df_hotels, on="property_id")
df.head()
```

Out[112...]

	property_id	check_in_date	room_category	successful_bookings	capacity	occ_pct	room_type
0	16559	1-May-22	RT1	25	30.0	83.33	Standard
1	19562	1-May-22	RT1	28	30.0	93.33	Standard
2	19563	1-May-22	RT1	23	30.0	76.67	Standard
3	16558	1-May-22	RT1	18	19.0	94.74	Standard
4	17560	1-May-22	RT1	28	40.0	70.00	Standard



In [112...]

df.shape

Out[112...]

(9194, 10)

In [112...]

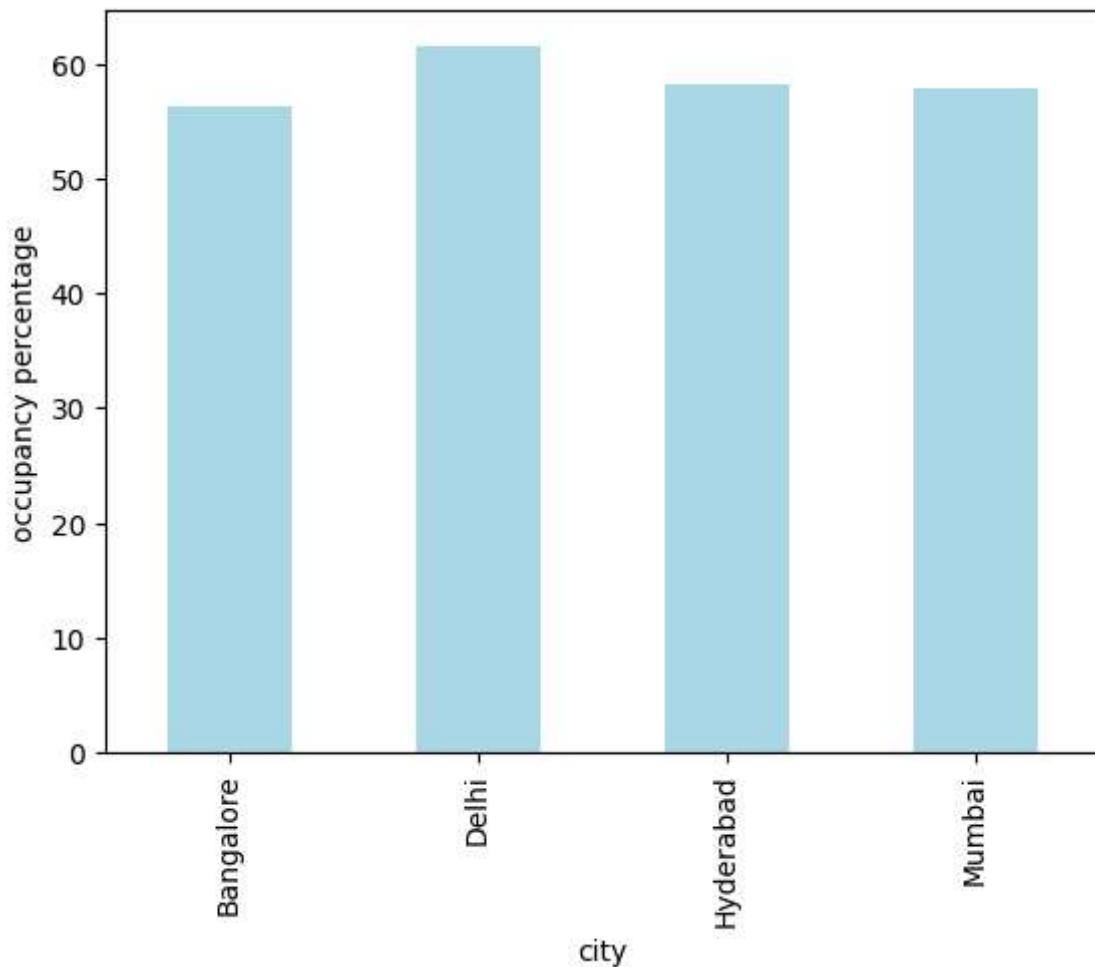
```
print(df.groupby("city")["occ_pct"].mean())
df.groupby("city")["occ_pct"].mean().plot(kind="bar", ylabel = "occupancy percentage")
```

city	
Bangalore	56.332376
Delhi	61.507341
Hyderabad	58.120652
Mumbai	57.909181

Name: occ_pct, dtype: float64

Out[112...]

<Axes: xlabel='city', ylabel='occupancy percentage'>



3. When was the occupancy better? Weekday or Weekend?

In [112...]

```
df_date.head()
```

Out[112...]

	date	mmm yy	week no	day_type
0	01-May-22	May 22	W 19	weekend
1	02-May-22	May 22	W 19	weekday
2	03-May-22	May 22	W 19	weekday
3	04-May-22	May 22	W 19	weekday
4	05-May-22	May 22	W 19	weekday

In [112...]

```
df.head()
```

Out[112...]

	property_id	check_in_date	room_category	successful_bookings	capacity	occ_pct	roon
0	16559	1-May-22	RT1	25	30.0	83.33	St
1	19562	1-May-22	RT1	28	30.0	93.33	St
2	19563	1-May-22	RT1	23	30.0	76.67	St
3	16558	1-May-22	RT1	18	19.0	94.74	St
4	17560	1-May-22	RT1	28	40.0	70.00	St

While merging df and df_date most of the rows might be dropped as date types in both the dataframes are inconsistent. So, we will first change them in the same date type.

In [112...]

```
# Convert 'date' column in df_date to datetime
df_date["date"] = pd.to_datetime(df_date["date"], format="%d-%b-%y")

# Reformat the 'date' column in df_date to 'dd-MMM-yy' format
df_date['date'] = df_date['date'].dt.strftime('%d-%b-%y')

df_date.head()
```

Out[112...]

	date	mmm yy	week no	day_type
0	01-May-22	May 22	W 19	weekend
1	02-May-22	May 22	W 19	weekday
2	03-May-22	May 22	W 19	weekday
3	04-May-22	May 22	W 19	weekday
4	05-May-22	May 22	W 19	weekday

In [113...]

```
# Convert 'check_in_date' to datetime
df["check_in_date"] = pd.to_datetime(df["check_in_date"], format="%d-%b-%y")

# Reformat the 'check_in_date' to 'dd-MMM-yy' format
df['check_in_date'] = df['check_in_date'].dt.strftime('%d-%b-%y')

print(df.shape)
df.head()
```

(9194, 10)

Out[113...]

	property_id	check_in_date	room_category	successful_bookings	capacity	occ_pct	roon
0	16559	01-May-22	RT1	25	30.0	83.33	St
1	19562	01-May-22	RT1	28	30.0	93.33	St
2	19563	01-May-22	RT1	23	30.0	76.67	St
3	16558	01-May-22	RT1	18	19.0	94.74	St
4	17560	01-May-22	RT1	28	40.0	70.00	St



In [113...]

```
df = pd.merge(df, df_date, left_on = "check_in_date", right_on = "date", how = "left")
print(df.shape)
df.head()
```

(9194, 14)

Out[113...]

	property_id	check_in_date	room_category	successful_bookings	capacity	occ_pct	roon
0	16559	01-May-22	RT1	25	30.0	83.33	St
1	19562	01-May-22	RT1	28	30.0	93.33	St
2	19563	01-May-22	RT1	23	30.0	76.67	St
3	16558	01-May-22	RT1	18	19.0	94.74	St
4	17560	01-May-22	RT1	28	40.0	70.00	St



In [113...]

```
df.drop("date", axis = 1, inplace = True)
df.head(3)
```

Out[113...]

	property_id	check_in_date	room_category	successful_bookings	capacity	occ_pct	room	
0	16559	01-May-22	RT1		25	30.0	83.33	St
1	19562	01-May-22	RT1		28	30.0	93.33	St
2	19563	01-May-22	RT1		23	30.0	76.67	St

◀ ▶

In [113...]

df.groupby("day_type")["occ_pct"].mean()

Out[113...]

```
day_type
weekday    51.807975
weekend    73.960616
Name: occ_pct, dtype: float64
```

4: In the month of June, what is the mean occupancy for different cities?

In [113...]

df["mmm yy"].unique()

Out[113...]

array(['May 22', 'Jun 22', 'Jul 22'], dtype=object)

In [113...]

df.check_in_date.unique()

Out[113...]

```
array(['01-May-22', '02-May-22', '03-May-22', '04-May-22', '05-May-22',
       '06-May-22', '07-May-22', '08-May-22', '09-May-22', '10-May-22',
       '11-May-22', '12-May-22', '13-May-22', '14-May-22', '15-May-22',
       '16-May-22', '17-May-22', '18-May-22', '19-May-22', '20-May-22',
       '21-May-22', '22-May-22', '23-May-22', '24-May-22', '25-May-22',
       '26-May-22', '27-May-22', '28-May-22', '29-May-22', '30-May-22',
       '31-May-22', '01-Jun-22', '02-Jun-22', '03-Jun-22', '04-Jun-22',
       '05-Jun-22', '06-Jun-22', '07-Jun-22', '08-Jun-22', '09-Jun-22',
       '10-Jun-22', '11-Jun-22', '12-Jun-22', '13-Jun-22', '14-Jun-22',
       '15-Jun-22', '16-Jun-22', '17-Jun-22', '18-Jun-22', '19-Jun-22',
       '20-Jun-22', '21-Jun-22', '22-Jun-22', '23-Jun-22', '24-Jun-22',
       '25-Jun-22', '26-Jun-22', '27-Jun-22', '28-Jun-22', '29-Jun-22',
       '30-Jun-22', '01-Jul-22', '02-Jul-22', '03-Jul-22', '04-Jul-22',
       '05-Jul-22', '06-Jul-22', '07-Jul-22', '08-Jul-22', '09-Jul-22',
       '10-Jul-22', '11-Jul-22', '12-Jul-22', '13-Jul-22', '14-Jul-22',
       '15-Jul-22', '16-Jul-22', '17-Jul-22', '18-Jul-22', '19-Jul-22',
       '20-Jul-22', '21-Jul-22', '22-Jul-22', '23-Jul-22', '24-Jul-22',
       '25-Jul-22', '26-Jul-22', '27-Jul-22', '28-Jul-22', '29-Jul-22',
       '30-Jul-22', '31-Jul-22'], dtype=object)
```

In [113...]

df[df["mmm yy"]=="Jun 22"].groupby("city")["occ_pct"].mean()

```
Out[113... city
Bangalore    55.849527
Delhi        61.456367
Hyderabad   57.688917
Mumbai       57.789542
Name: occ_pct, dtype: float64
```

5: We got new data for the month of august. Append that to existing data

In [113... df_new_aug

	property_id	property_name	category	city	room_category	room_class	check_in_date
0	16559	Atliq Exotica	Luxury	Mumbai	RT1	Standard	01-Aug-22
1	19562	Atliq Bay	Luxury	Bangalore	RT1	Standard	01-Aug-22
2	19563	Atliq Palace	Business	Bangalore	RT1	Standard	01-Aug-22
3	19558	Atliq Grands	Luxury	Bangalore	RT1	Standard	01-Aug-22
4	19560	Atliq City	Business	Bangalore	RT1	Standard	01-Aug-22
5	17561	Atliq Blu	Luxury	Mumbai	RT1	Standard	01-Aug-22
6	17564	Atliq Seasons	Business	Mumbai	RT1	Standard	01-Aug-22

In [113... df.head(3)

	property_id	check_in_date	room_category	successful_bookings	capacity	occ_pct	room_class
0	16559	01-May-22	RT1	25	30.0	83.33	Standard
1	19562	01-May-22	RT1	28	30.0	93.33	Standard
2	19563	01-May-22	RT1	23	30.0	76.67	Standard

In [113... print(df_new_aug.shape, df.shape)

(7, 13) (9194, 13)

```
In [114... df_new_aug.rename(columns={"occ%": "occ_pct"}, inplace=True)
```

```
In [114... df_latest = pd.concat([df, df_new_aug], ignore_index = True)
df_latest.head(3)
```

Out[114...

	property_id	check_in_date	room_category	successful_bookings	capacity	occ_pct	room	
0	16559	01-May-22	RT1		25	30.0	83.33	St
1	19562	01-May-22	RT1		28	30.0	93.33	St
2	19563	01-May-22	RT1		23	30.0	76.67	St

◀ ▶

```
In [114... df_latest.shape
```

Out[114... (9201, 13)

6. Finding revenue realized per city

```
In [114... print(df_bookings.shape)
df_bookings.head(3)
```

(134573, 12)

```
Out[114... booking_id property_id booking_date check_in_date checkout_date no_guests
```

	booking_id	property_id	booking_date	check_in_date	checkout_date	no_guests
1	May012216558RT12	16558	30-04-22	1/5/2022	2/5/2022	2.0
4	May012216558RT15	16558	27-04-22	1/5/2022	2/5/2022	4.0
5	May012216558RT16	16558	1/5/2022	1/5/2022	3/5/2022	2.0

◀ ▶

```
In [114... df_hotels.head(3)
```

```
Out[114... property_id property_name category city
```

	property_id	property_name	category	city
0	16558	Atliq Grands	Luxury	Delhi
1	16559	Atliq Exotica	Luxury	Mumbai
2	16560	Atliq City	Business	Delhi

```
In [114... df_bookings_merged = pd.merge(df_bookings, df_hotels, on = "property_id")
print(df_bookings_merged.shape)
df_bookings_merged.head(3)
```

(134573, 15)

Out[114...]

	booking_id	property_id	booking_date	check_in_date	checkout_date	no_guests
0	May012216558RT12	16558	30-04-22	1/5/2022	2/5/2022	2.0
1	May012216558RT15	16558	27-04-22	1/5/2022	2/5/2022	4.0
2	May012216558RT16	16558	1/5/2022	1/5/2022	3/5/2022	2.0

◀ ▶

In [114...]

```
df_bookings_merged.groupby("city")["revenue_realized"].sum()
```

Out[114...]

city	revenue_realized
Bangalore	420383550
Delhi	294404488
Hyderabad	325179310
Mumbai	668569251

Name: revenue_realized, dtype: int64

7. Print month by month revenue

In [114...]

```
df_date.head(3)
```

Out[114...]

	date	mmm yy	week no	day_type
0	01-May-22	May 22	W 19	weekend
1	02-May-22	May 22	W 19	weekday
2	03-May-22	May 22	W 19	weekday

In [114...]

```
df_bookings_merged.head(3)
```

Out[114...]

	booking_id	property_id	booking_date	check_in_date	checkout_date	no_guests
0	May012216558RT12	16558	30-04-22	1/5/2022	2/5/2022	2.0
1	May012216558RT15	16558	27-04-22	1/5/2022	2/5/2022	4.0
2	May012216558RT16	16558	1/5/2022	1/5/2022	3/5/2022	2.0

◀ ▶

In [114...]

```
# Convert 'check_in_date' to datetime
df_bookings_merged["check_in_date"] = pd.to_datetime(df_bookings_merged["check_in_d

# Reformat the 'check_in_date' to 'dd-MMM-yy' format
df_bookings_merged['check_in_date'] = df_bookings_merged['check_in_date'].dt.strftime("%d-%b-%y")
```

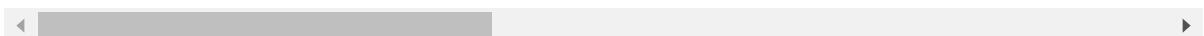
In [115...]

```
print(df_bookings_merged.shape)
df_bookings_merged.head(3)
```

(134573, 15)

Out[115...]

	booking_id	property_id	booking_date	check_in_date	checkout_date	no_guests
0	May012216558RT12	16558	30-04-22	01-May-22	2/5/2022	2.0
1	May012216558RT15	16558	27-04-22	01-May-22	2/5/2022	4.0
2	May012216558RT16	16558	1/5/2022	01-May-22	3/5/2022	2.0



In [115...]

```
df_bookings_merged = pd.merge(df_bookings_merged, df_date, left_on = "check_in_date",
```

In [115...]

```
df_bookings_merged.drop("date", axis = 1, inplace = True)
```

In [115...]

```
df_bookings_merged.head(3)
```

Out[115...]

	booking_id	property_id	booking_date	check_in_date	checkout_date	no_guests
0	May012216558RT12	16558	30-04-22	01-May-22	2/5/2022	2.0
1	May012216558RT15	16558	27-04-22	01-May-22	2/5/2022	4.0
2	May012216558RT16	16558	1/5/2022	01-May-22	3/5/2022	2.0



In [115...]

```
df_bookings_merged.groupby("mmm yy")["revenue_realized"].sum()
```

Out[115...]

```
mmm yy
Jul 22    572843348
Jun 22    553925855
May 22    581767396
Name: revenue_realized, dtype: int64
```

In [115...]

```
df_bookings_merged.revenue_realized.sum()
```

Out[115...]

```
np.int64(1708536599)
```

8. Finding average ratings and revenue realized per hotel name.

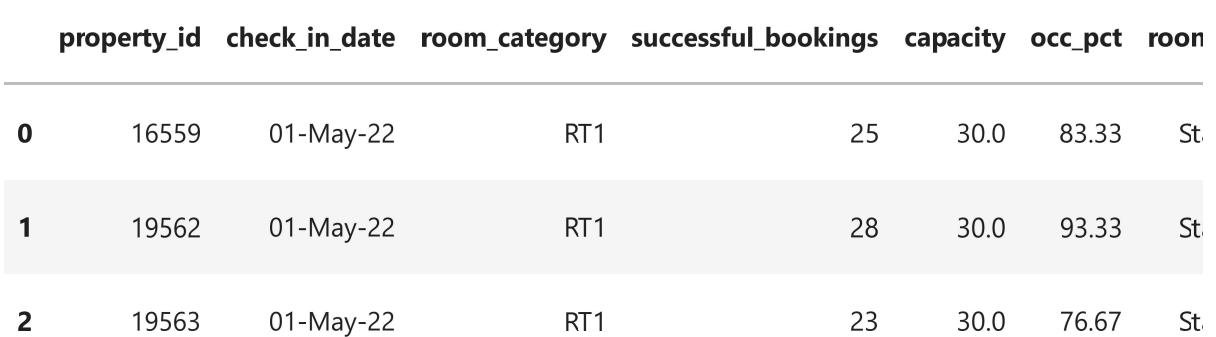
In [115...]

```
print(df_bookings_merged.groupby("property_name")["ratings_given"].mean())
print()
print(df_bookings_merged.groupby("property_name")["revenue_realized"].sum())
```

```
property_name
Atliq Bay      3.708943
Atliq Blu       3.959650
Atliq City      3.694799
Atliq Exotica   3.619241
Atliq Grands    3.099779
Atliq Palace     3.749545
Atliq Seasons   2.295033
Name: ratings_given, dtype: float64
```

```
property_name
Atliq Bay      259996918
Atliq Blu       260851922
Atliq City      285798439
Atliq Exotica   320258588
Atliq Grands    211462134
Atliq Palace     304081863
Atliq Seasons   66086735
Name: revenue_realized, dtype: int64
```

In [115... df.head(3)

Out[115... 

	property_id	check_in_date	room_category	successful_bookings	capacity	occ_pct	room	
0	16559	01-May-22	RT1		25	30.0	83.33	St
1	19562	01-May-22	RT1		28	30.0	93.33	St
2	19563	01-May-22	RT1		23	30.0	76.67	St



In [115... print(df_bookings_merged.shape)
df_bookings_merged.head(3)

(134573, 18)

Out[115... 

	booking_id	property_id	booking_date	check_in_date	checkout_date	no_guests
0	May012216558RT12	16558	30-04-22	01-May-22	2/5/2022	2.0
1	May012216558RT15	16558	27-04-22	01-May-22	2/5/2022	4.0
2	May012216558RT16	16558	1/5/2022	01-May-22	3/5/2022	2.0



In [115... df_bookings_merged = pd.merge(df_bookings_merged, df_rooms, left_on = "room_category")

```
In [116... print(df_bookings_merged.shape)
df_bookings_merged.head(3)
```

(134573, 20)

Out[116...

	booking_id	property_id	booking_date	check_in_date	checkout_date	no_guests
0	May012216558RT12	16558	30-04-22	01-May-22	2/5/2022	2.0
1	May012216558RT15	16558	27-04-22	01-May-22	2/5/2022	4.0
2	May012216558RT16	16558	1/5/2022	01-May-22	3/5/2022	2.0

◀		▶
---	--	---

In [116...

```
df_bookings_merged.drop("room_id", axis=1, inplace=True)
print(df_bookings_merged.shape)
df_bookings_merged.head(3)
```

(134573, 19)

Out[116...

	booking_id	property_id	booking_date	check_in_date	checkout_date	no_guests
0	May012216558RT12	16558	30-04-22	01-May-22	2/5/2022	2.0
1	May012216558RT15	16558	27-04-22	01-May-22	2/5/2022	4.0
2	May012216558RT16	16558	1/5/2022	01-May-22	3/5/2022	2.0

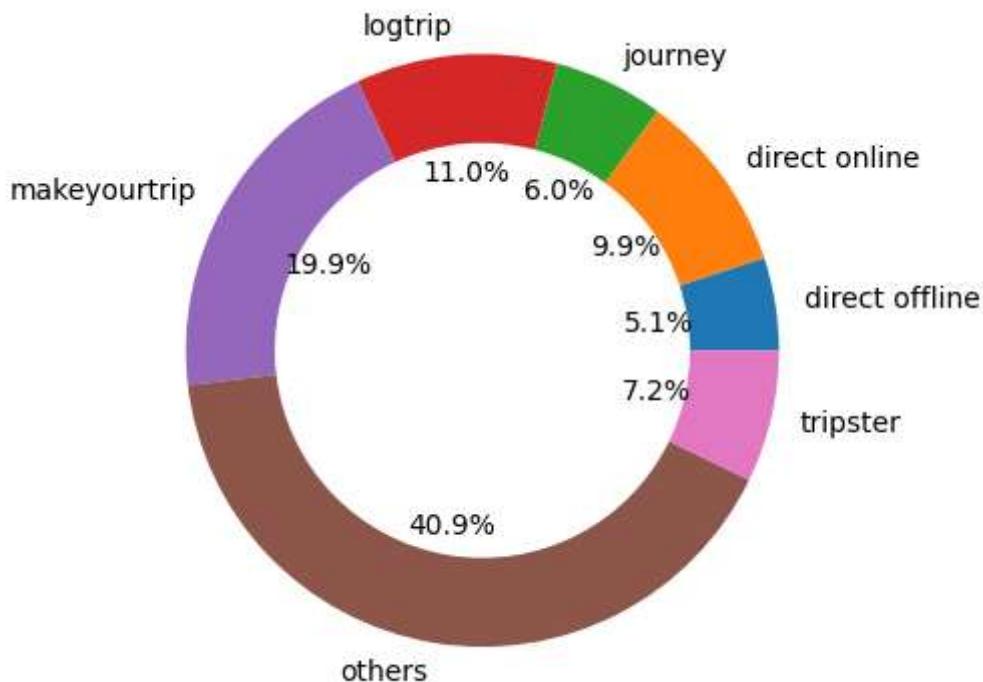
◀		▶
---	--	---

5) Data Visualization

1. Pie chart of revenue realized per booking platform

```
In [118... df_bookings_merged.groupby("booking_platform")["revenue_realized"].sum().plot(kind=
```

Out[118... <Axes: >

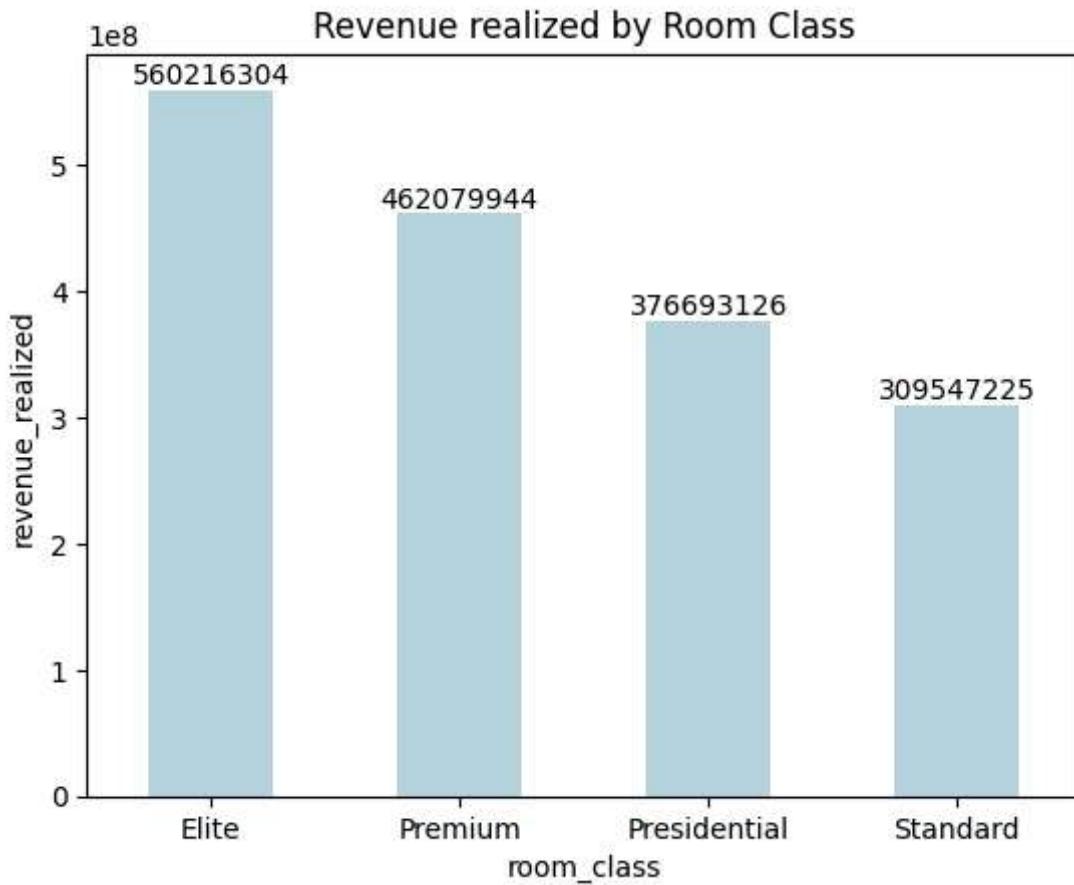


```
In [117... a = (df_bookings_merged.groupby('room_class')['revenue_realized'].sum())/1000000
print(a.iloc[1])
print(a)
```

```
462.079944
room_class
Elite      560.216304
Premium    462.079944
Presidential 376.693126
Standard    309.547225
Name: revenue_realized, dtype: float64
```

```
In [118... import seaborn as sns

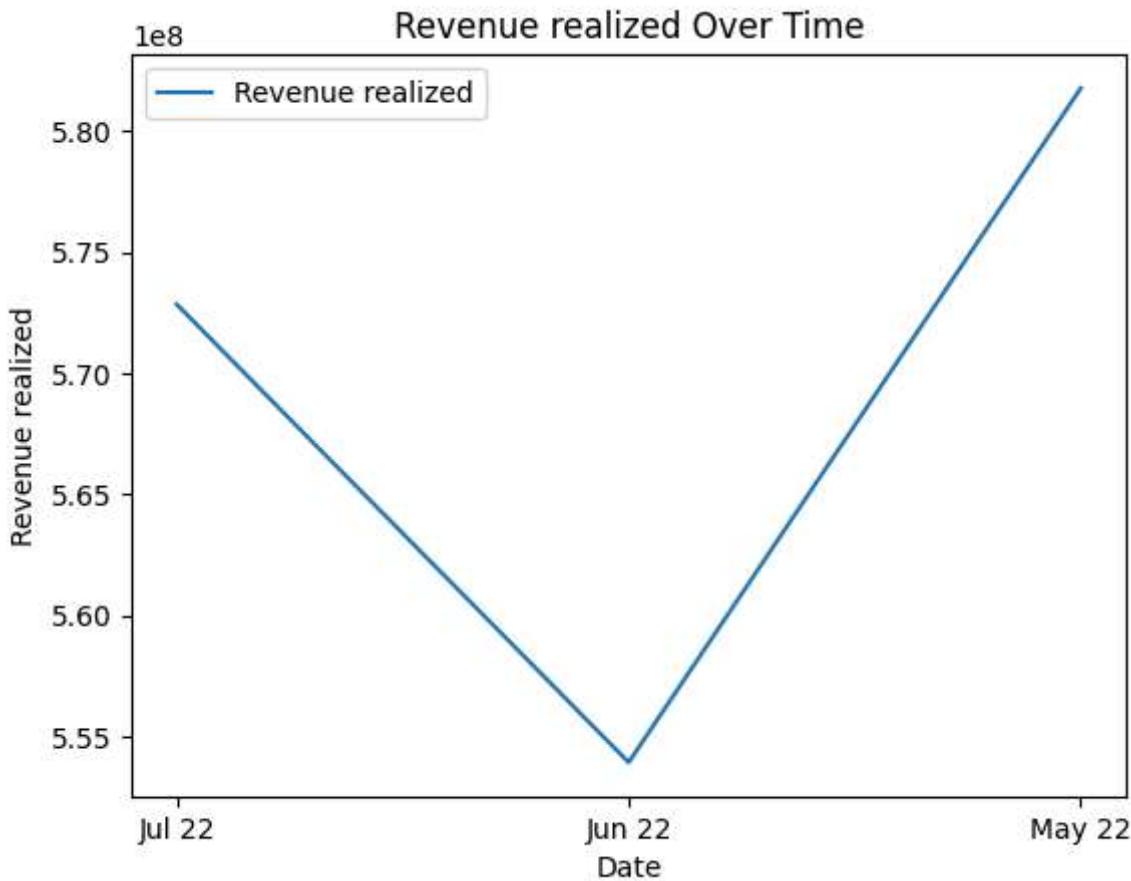
rcbyrr = df_bookings_merged.groupby('room_class',as_index=False)['revenue_realized']
# rcbyrr["revenue_realized"] = rcbyrr["revenue_realized"]/1000000
ax = sns.barplot(x='room_class', y="revenue_realized", data=rcbyrr, width = 0.5, co
plt.title('Revenue realized by Room Class')
for bar in ax.containers:
    ax.bar_label(bar, fmt='%.0f')
plt.show()
```



In [118...]

```
import matplotlib.pyplot as plt

mybyrr = df_bookings_merged.groupby('mmm yy',as_index=False)[['revenue_realized']].sum()
plt.plot(mybyrr['mmm yy'], mybyrr['revenue_realized'], label='Revenue realized')
plt.xlabel('Date')
plt.ylabel('Revenue realized')
plt.title('Revenue realized Over Time')
plt.legend()
plt.show()
```



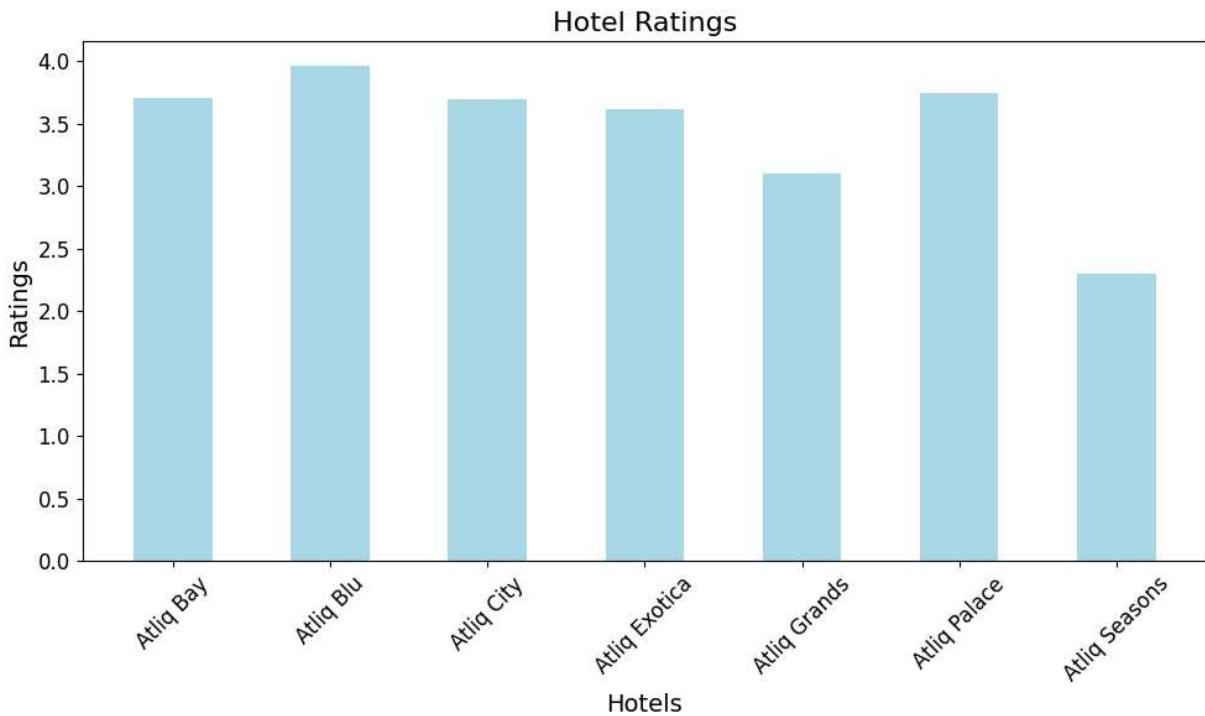
```
In [120]: plt.figure(figsize=(10, 6)) # Set the figure size

pnbyrg = df_bookings_merged.groupby("property_name", as_index = False)[["ratings_given"]]
ax3 = plt.bar(pnbyrg["property_name"], pnbyrg["ratings_given"], color='lightblue', width=0.5)

# Add Labels and title
plt.title('Hotel Ratings', fontsize=16)
plt.xlabel('Hotels', fontsize=14)
plt.ylabel('Ratings', fontsize=14)

# Rotate x-axis Labels for better readability
plt.xticks(rotation=45, fontsize=12)
plt.yticks(fontsize=12)

# Show the chart
plt.tight_layout() # Adjust layout to prevent overlap
plt.show()
```



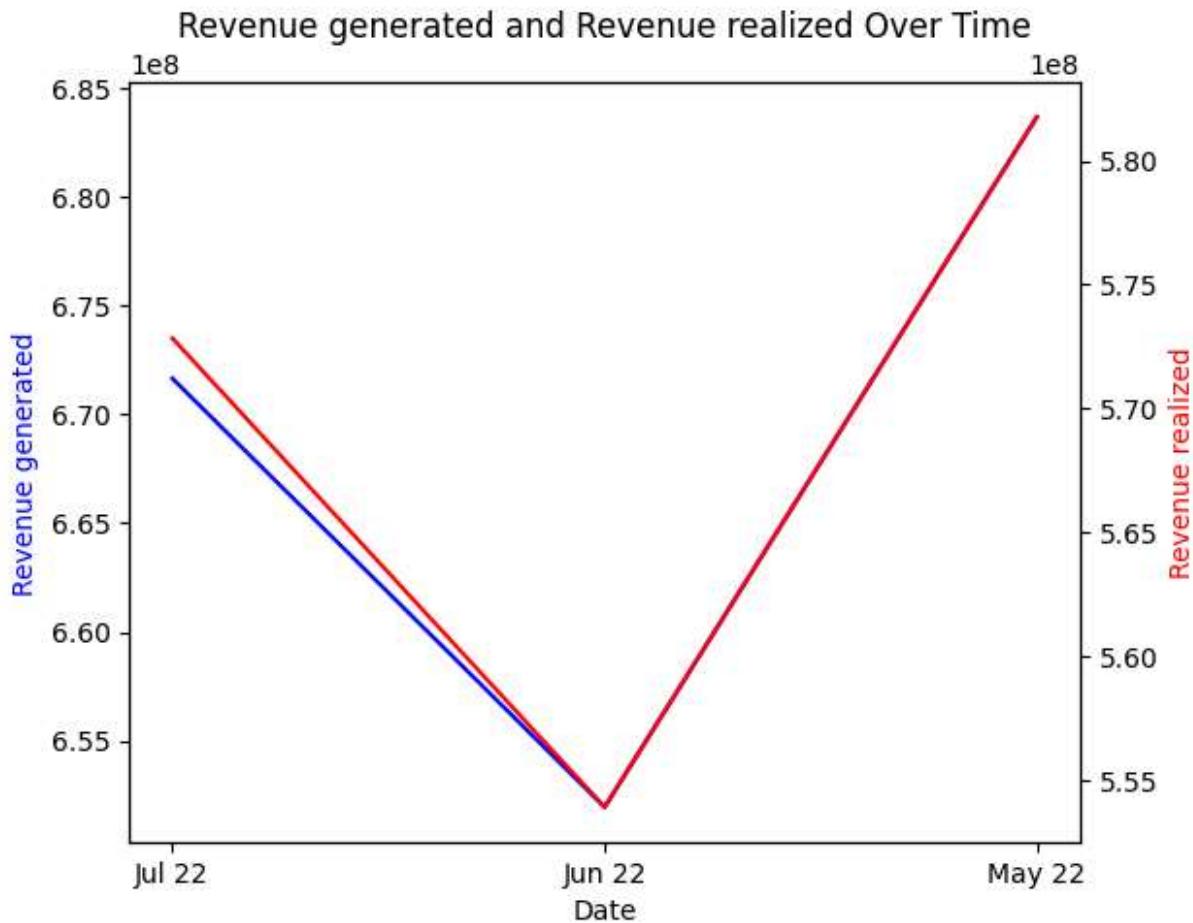
In [119...]

```
fig, ax1 = plt.subplots()

mybyrg = df_bookings_merged.groupby('mmm yy',as_index=False)[['revenue_generated']].sum()
ax1.plot(mybyrg['mmm yy'], mybyrg['revenue_generated'], 'b-', label='Revenue Generated')
ax1.set_xlabel('Date')
ax1.set_ylabel('Revenue generated', color='b')

mybyrr = df_bookings_merged.groupby('mmm yy',as_index=False)[['revenue_realized']].sum()
ax2 = ax1.twinx()
ax2.plot(mybyrr['mmm yy'], mybyrr['revenue_realized'], 'r-', label='Revenue Realized')
ax2.set_ylabel('Revenue realized', color='r')

fig.tight_layout()
plt.title('Revenue generated and Revenue realized Over Time')
plt.show()
```



In []: