Analysis

May 17, 2025

AtliQ Hotels Data Analysis Project

[2]:
```
import pandas as pd
```

0.0.1 ==> 1. Data Import and Data Exploration

0.0.2 Datasets

We have 5 csv file

- dim_date.csv

- dim_hotels.csv
- dim_rooms.csv
- fact_aggregated_bookings
- fact_bookings.csv

Reading bookings data in a dataframe

[3]:
```
df_bookings = pd.read_csv("datasets/fact_bookings.csv")
df_bookings.head()
```

[3]:

	booking_id	property_id	booking_date	check_in_date	checkout_date	no_guests	room_category	booking_platform	ratings_given	booking_status
0	May012216558RT11	16558	27-04-22	1/5/2022	2/5/2022	-3.0	RT1	direct online	1.0	Checked Out
1	May012216558RT12	16558	30-04-22	1/5/2022	2/5/2022	2.0	RT1	others	NaN	Cancelled
2	May012216558RT13	16558	28-04-22	1/5/2022	4/5/2022	2.0	RT1	logtrip	5.0	Checked Out
3	May012216558RT14	16558	28-04-22	1/5/2022	2/5/2022	-2.0	RT1	others	NaN	Cancelled
4	May012216558RT15	16558	27-04-22	1/5/2022	2/5/2022	4.0	RT1	direct online	5.0	Checked Out

```
   revenue_generated  revenue_realized
0              10010             10010
1               9100              3640
2            9100000              9100
3               9100              3640
4              10920             10920
```

Exploring bookings data

[4]:
```
df_bookings.shape
```

[4]:
```
(134590, 12)
```

[5]:
```
df_bookings.room_category.unique()
```

[5]:
```
array(['RT1', 'RT2', 'RT3', 'RT4'], dtype=object)
```

[6]:
```
df_bookings.booking_platform.unique()
```

[6]:
```
array(['direct online', 'others', 'logtrip', 'tripster', 'makeyourtrip',
       'journey', 'direct offline'], dtype=object)
```

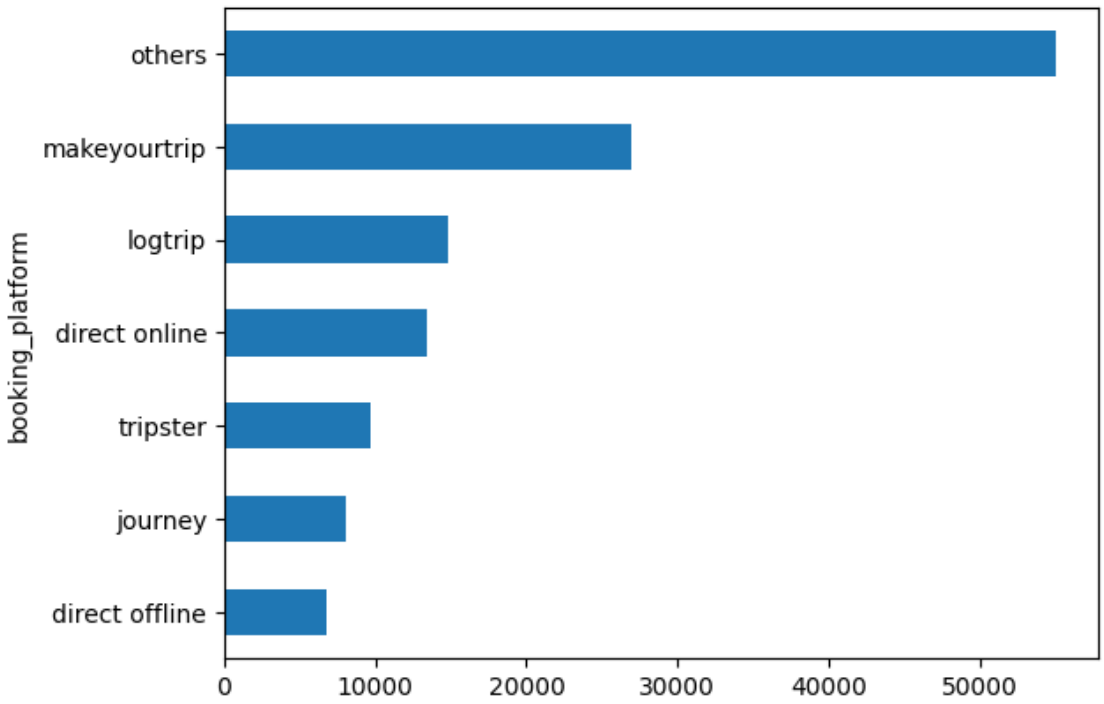
[7]:
```
df_bookings.booking_platform.value_counts()
```

[7]:
```
booking_platform
others            55066
makeyourtrip      26898
logtrip           14756
direct online     13379
tripster           9630
journey            8106
direct offline     6755
Name: count, dtype: int64
```

Number of bookings done through booking platforms

[8]:
```
df_bookings.booking_platform.value_counts().sort_values().plot(kind = "barh")
```

[8]:
```
<Axes: ylabel='booking_platform'>
```



Overall Aggregation of Bookings data

[9]:
```
df_bookings.describe()
```

[9]:

	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

[10]:
```
int(df_bookings.revenue_generated.min()), int(df_bookings.revenue_generated.
↪max())
```

[10]:
```
(6500, 28560000)
```

Reading rest of the files

[11]:
```
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')
```

[12]:
```
df_hotels.shape
```

[12]:
```
(25, 4)
```

[13]:
```
df_hotels.head()
```

[13]:

	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

[14]:
```
df_hotels.category.value_counts()
```

[14]:
```
category
Luxury      16
Business     9
Name: count, dtype: int64
```

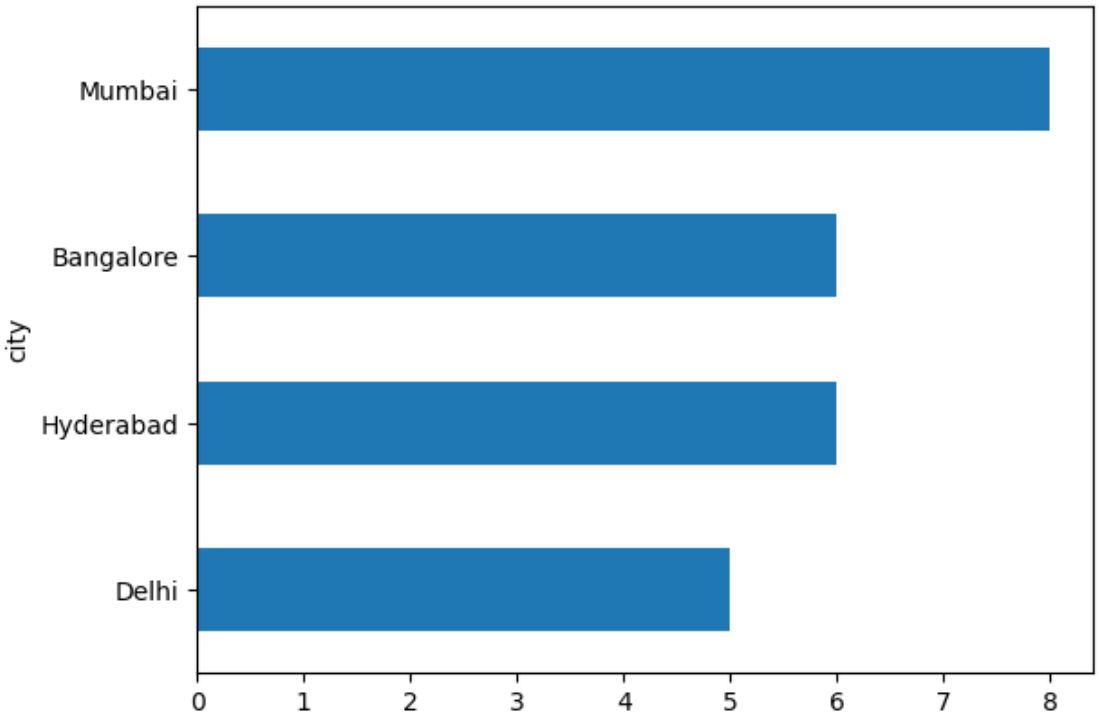
[15]:
```
df_hotels.city.value_counts().sort_values()
```

[15]:
```
city
Delhi        5
Hyderabad    6
Bangalore    6
Mumbai       8
Name: count, dtype: int64
```

Number of Hotels per City

[16]:
```
df_hotels.city.value_counts().sort_values().plot(kind = "barh")
```

[16]:
```
<Axes: ylabel='city'>
```



[17]:
```
df_agg_bookings.head()
```

[17]:

	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

[18]:
```
df_agg_bookings.property_id.unique()
```

[18]:
```
array([16559, 19562, 19563, 17558, 16558, 17560, 19558, 19560, 17561,
       16560, 16561, 16562, 16563, 17559, 17562, 17563, 18558, 18559,
       18561, 18562, 18563, 19559, 19561, 17564, 18560])
```

Total Successful Bookings done by each Property

[19]:
```
df_agg_bookings.groupby("property_id")['successful_bookings'].sum()
```

[19]:
```
property_id
16558    3153
16559    7338
16560    4693
16561    4418
```

```
16562    4820
16563    7211
17558    5053
17559    6142
17560    6013
17561    5183
17562    3424
17563    6337
17564    3982
18558    4475
18559    5256
18560    6638
18561    6458
18562    7333
18563    4737
19558    4400
19559    4729
19560    6079
19561    5736
19562    5812
19563    5413
Name: successful_bookings, dtype: int64
```

Properties where Successful Bookings crossed the Capacity

[20]:
```
df_agg_bookings[df_agg_bookings['successful_bookings'] >␣
↪df_agg_bookings['capacity']]
```

[20]:

	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

[21]:
```
df_agg_bookings[df_agg_bookings.capacity == df_agg_bookings.capacity.max()]
```

[21]:

	property_id	check_in_date	room_category	successful_bookings	capacity
27	17558	1-May-22	RT2	38	50.0
128	17558	2-May-22	RT2	27	50.0
229	17558	3-May-22	RT2	26	50.0
328	17558	4-May-22	RT2	27	50.0
428	17558	5-May-22	RT2	29	50.0
…	…	…	…	…	…
8728	17558	27-Jul-22	RT2	22	50.0
8828	17558	28-Jul-22	RT2	21	50.0
8928	17558	29-Jul-22	RT2	23	50.0

```
9028        17558     30-Jul-22           RT2                  32      50.0
9128        17558     31-Jul-22           RT2                  30      50.0

[92 rows x 5 columns]
```

0.0.3 ==> 2. Data Cleaning

---

[22]:
```
df_bookings.describe()
```

[22]:
```
         property_id      no_guests  ratings_given  revenue_generated  \
count  134590.000000  134587.000000   56683.000000       1.345900e+05
mean    18061.113493       2.036170       3.619004       1.537805e+04
std      1093.055847       1.034885       1.235009       9.303604e+04
min     16558.000000     -17.000000       1.000000       6.500000e+03
25%     17558.000000       1.000000       3.000000       9.900000e+03
50%     17564.000000       2.000000       4.000000       1.350000e+04
75%     18563.000000       2.000000       5.000000       1.800000e+04
max     19563.000000       6.000000       5.000000       2.856000e+07

       revenue_realized
count     134590.000000
mean       12696.123256
std         6928.108124
min         2600.000000
25%         7600.000000
50%        11700.000000
75%        15300.000000
max        45220.000000
```

(1) Cleaning invalid guests

[23]:
```
df_bookings[df_bookings.no_guests <= 0]
```

[23]:
```
              booking_id  property_id booking_date check_in_date  \
0        May012216558RT11        16558     27-04-22      1/5/2022
3        May012216558RT14        16558     28-04-22      1/5/2022
17924    May122218559RT44        18559    12/5/2022     12/5/2022
18020    May122218561RT22        18561     8/5/2022     12/5/2022
18119   May122218562RT311        18562     5/5/2022     12/5/2022
18121   May122218562RT313        18562    10/5/2022     12/5/2022
56715    Jun082218562RT12        18562     5/6/2022      8/6/2022
119765  Jul202219560RT220        19560     19-07-22      20-07-22
134586   Jul312217564RT47        17564     30-07-22      31-07-22

       checkout_date  no_guests room_category booking_platform  ratings_given  \
```

```
0          2/5/2022       -3.0           RT1    direct online            1.0
3          2/5/2022       -2.0           RT1           others            NaN
17924      14-05-22      -10.0           RT4    direct online            NaN
18020      14-05-22      -12.0           RT2     makeyourtrip            NaN
18119      17-05-22       -6.0           RT3   direct offline            5.0
18121      17-05-22       -4.0           RT3    direct online            NaN
56715      13-06-22      -17.0           RT1           others            NaN
119765     22-07-22       -1.0           RT2           others            NaN
134586     1/8/2022       -4.0           RT4          logtrip            2.0

       booking_status  revenue_generated  revenue_realized
0         Checked Out              10010             10010
3           Cancelled               9100              3640
17924         No Show              20900             20900
18020       Cancelled               9000              3600
18119     Checked Out              16800             16800
18121       Cancelled              14400              5760
56715     Checked Out               6500              6500
119765    Checked Out              13500             13500
134586    Checked Out              38760             38760
```

As you can see above, the number of guests having less than zero value represents data error. We can filter these records.

[24]:
```
df_bookings = df_bookings[df_bookings.no_guests > 0]
df_bookings
```

[24]:
```
              booking_id  property_id booking_date check_in_date  \
1        May012216558RT12        16558     30-04-22      1/5/2022
2        May012216558RT13        16558     28-04-22      1/5/2022
4        May012216558RT15        16558     27-04-22      1/5/2022
5        May012216558RT16        16558     1/5/2022      1/5/2022
6        May012216558RT17        16558     28-04-22      1/5/2022
…                     …           …           …             …
134584   Jul312217564RT45        17564     30-07-22      31-07-22
134585   Jul312217564RT46        17564     29-07-22      31-07-22
134587   Jul312217564RT48        17564     30-07-22      31-07-22
134588   Jul312217564RT49        17564     29-07-22      31-07-22
134589  Jul312217564RT410        17564     31-07-22      31-07-22

       checkout_date  no_guests room_category booking_platform  ratings_given  \
1          2/5/2022        2.0           RT1           others            NaN
2          4/5/2022        2.0           RT1          logtrip            5.0
4          2/5/2022        4.0           RT1    direct online            5.0
5          3/5/2022        2.0           RT1           others            4.0
6          6/5/2022        2.0           RT1           others            NaN
…              …           …             …                …              …
134584     1/8/2022        2.0           RT4           others            2.0
```

```
134585      3/8/2022        1.0           RT4     makeyourtrip            2.0
134587      2/8/2022        1.0           RT4          tripster           NaN
134588      1/8/2022        2.0           RT4           logtrip           2.0
134589      1/8/2022        2.0           RT4     makeyourtrip            NaN

       booking_status  revenue_generated  revenue_realized
1           Cancelled               9100              3640
2         Checked Out            9100000              9100
4         Checked Out              10920             10920
5         Checked Out               9100              9100
6           Cancelled               9100              3640
…                 …                  …                 …
134584    Checked Out              32300             32300
134585    Checked Out              32300             32300
134587      Cancelled              32300             12920
134588    Checked Out              32300             32300
134589      Cancelled              32300             12920

[134578 rows x 12 columns]
```

[25]:
```
df_bookings.shape
```

[25]:
```
(134578, 12)
```

(2) Outlier removal in revenue generated

[26]:
```
int(df_bookings.revenue_generated.min()), int(df_bookings.revenue_generated.
  ↪max())
```

[26]:
```
(6500, 28560000)
```

[27]:
```
avg, std = float(df_bookings.revenue_generated.mean()), float(df_bookings.
  ↪revenue_generated.std())
```

[28]:
```
avg, std
```

[28]:
```
(15378.036937686695, 93040.1549314641)
```

[29]:
```
higher_limit = avg + 3 * std
higher_limit
```

[29]:
```
294498.50173207896
```

[30]:
```
lower_limit = avg - 3 * std
lower_limit
```

[30]:
```
-263742.4278567056
```

[31]:
```
df_bookings[df_bookings.revenue_generated > higher_limit]
```

[31]:

	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
2	May012216558RT13	16558	28-04-22	1/5/2022	4/5/2022	2.0	RT1	logtrip	5.0	Checked Out	9100000	9100
111	May012216559RT32	16559	29-04-22	1/5/2022	2/5/2022	6.0	RT3	direct online	NaN	Checked Out	28560000	28560
315	May012216562RT22	16562	28-04-22	1/5/2022	4/5/2022	2.0	RT2	direct offline	3.0	Checked Out	12600000	12600
562	May012217559RT118	17559	26-04-22	1/5/2022	2/5/2022	2.0	RT1	others	NaN	Cancelled	2000000	4420
129176	Jul282216562RT26	16562	21-07-22	28-07-22	29-07-22	2.0	RT2	direct online	3.0	Checked Out	10000000	12600

[32]:
```
df_bookings = df_bookings[df_bookings.revenue_generated < higher_limit]
df_bookings
```

[32]:

	booking_id	property_id	booking_date	check_in_date	checkout_date	no_guests	room_category	booking_platform	ratings_given
1	May012216558RT12	16558	30-04-22	1/5/2022	2/5/2022	2.0	RT1	others	NaN
4	May012216558RT15	16558	27-04-22	1/5/2022	2/5/2022	4.0	RT1	direct online	5.0
5	May012216558RT16	16558	1/5/2022	1/5/2022	3/5/2022	2.0	RT1	others	4.0
6	May012216558RT17	16558	28-04-22	1/5/2022	6/5/2022	2.0	RT1	others	NaN
7	May012216558RT18	16558	26-04-22	1/5/2022	3/5/2022	2.0	RT1	logtrip	NaN
…	…	…	…	…	…	…	…	…	…
134584	Jul312217564RT45	17564	30-07-22	31-07-22	1/8/2022	2.0	RT4	others	2.0
134585	Jul312217564RT46	17564	29-07-22	31-07-22					
134587	Jul312217564RT48	17564	30-07-22	31-07-22					
134588	Jul312217564RT49	17564	29-07-22	31-07-22					
134589	Jul312217564RT410	17564	31-07-22	31-07-22					

```
134585      3/8/2022        1.0           RT4     makeyourtrip            2.0
134587      2/8/2022        1.0           RT4          tripster           NaN
134588      1/8/2022        2.0           RT4           logtrip           2.0
134589      1/8/2022        2.0           RT4     makeyourtrip            NaN

       booking_status  revenue_generated  revenue_realized
1           Cancelled               9100              3640
4         Checked Out              10920             10920
5         Checked Out               9100              9100
6           Cancelled               9100              3640
7             No Show               9100              9100
…                 …                  …                 …
134584    Checked Out              32300             32300
134585    Checked Out              32300             32300
134587      Cancelled              32300             12920
134588    Checked Out              32300             32300
134589      Cancelled              32300             12920

[134573 rows x 12 columns]
```

[33]:
```
df_bookings.shape
```

[33]:
```
(134573, 12)
```

[34]:
```
df_bookings.revenue_realized.describe()
```

[34]:
```
count    134573.000000
mean      12695.983585
std        6927.791692
min        2600.000000
25%        7600.000000
50%       11700.000000
75%       15300.000000
max       45220.000000
Name: revenue_realized, dtype: float64
```

[35]:
```
higher_limit = float(df_bookings.revenue_realized.mean()) + 3 *␣
  ↪float(df_bookings.revenue_realized.std())
higher_limit
```

[35]:
```
33479.358661845814
```

[36]:
```
df_bookings[df_bookings.revenue_realized > higher_limit]
```

[36]:
```
              booking_id  property_id booking_date check_in_date  \
137      May012216559RT41        16559     27-04-22      1/5/2022
139      May012216559RT43        16559     1/5/2022      1/5/2022
143      May012216559RT47        16559     28-04-22      1/5/2022
```

```
149      May012216559RT413        16559     24-04-22      1/5/2022
222       May012216560RT45        16560     30-04-22      1/5/2022
…                      …          …           …           …
134328    Jul312219560RT49        19560     31-07-22      31-07-22
134331   Jul312219560RT412        19560     31-07-22      31-07-22
134467    Jul312219562RT45        19562     28-07-22      31-07-22
134474   Jul312219562RT412        19562     25-07-22      31-07-22
134581    Jul312217564RT42        17564     31-07-22      31-07-22

       checkout_date  no_guests room_category booking_platform  ratings_given  \
137          7/5/2022        4.0           RT4           others            NaN
139          2/5/2022        6.0           RT4         tripster            3.0
143          3/5/2022        3.0           RT4           others            5.0
149          7/5/2022        5.0           RT4          logtrip            NaN
222          3/5/2022        5.0           RT4           others            3.0
…               …         …            …              …            …
134328       2/8/2022        6.0           RT4    direct online            5.0
134331       1/8/2022        6.0           RT4           others            2.0
134467       1/8/2022        6.0           RT4     makeyourtrip            4.0
134474       6/8/2022        5.0           RT4   direct offline            5.0
134581       1/8/2022        4.0           RT4     makeyourtrip            4.0

       booking_status  revenue_generated  revenue_realized
137       Checked Out              38760             38760
139       Checked Out              45220             45220
143       Checked Out              35530             35530
149       Checked Out              41990             41990
222       Checked Out              34580             34580
…                …                 …                 …
134328    Checked Out              39900             39900
134331    Checked Out              39900             39900
134467    Checked Out              39900             39900
134474    Checked Out              37050             37050
134581    Checked Out              38760             38760

[1299 rows x 12 columns]
```

One observation we get from the above dataframe is that all rooms are of RT4 category which means a presidential suite. Now, since RT4 is a luxurious room it is likely it's rent will be higher. To make a fair analysis, we need to do data analysis only on RT4 room types

[37]:
```
df_rooms
```

[37]:
```
  room_id    room_class
0     RT1      Standard
1     RT2         Elite
2     RT3       Premium
3     RT4  Presidential
```

[38]:
```
df_bookings[df_bookings.room_category == "RT4"].revenue_realized.describe()
```

[38]:
```
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
```

[39]:
```
# mean + 3*standard deviation
23439 + 3 * 9048
```

[39]:
```
50583
```

Here, the higher limit comes to be 50583 and in our dataframe above we can see that the max value for the revenue realized is 45220. Hence, we can conclude that there are no outliers in the room_category column and therefore, we don't need to do any data cleaning on it.

[40]:
```
df_bookings.isnull().sum()
```

[40]:
```
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
```

Total values in our dataframe are 1,34,573. Out of that 77,899 rows have 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.

[41]:
```
df_agg_bookings.head()
```

[41]:

	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

[42]:
```
df_agg_bookings.isnull().sum()
```

[42]:
```
property_id            0
check_in_date          0
room_category          0
successful_bookings    0
capacity               2
dtype: int64
```

[43]:
```
df_agg_bookings[df_agg_bookings.capacity.isna()]
```

[43]:

	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

[44]:
```
float(df_agg_bookings.capacity.median())
```

[44]:
```
25.0
```

[45]:
```
#df_agg_bookings['capacity'] = df_agg_bookings['capacity'].apply(lambda x:␣
 ↪df_agg_bookings['capacity'].mean() if pd.isna(x) else x)
df_agg_bookings.capacity = df_agg_bookings.capacity.fillna(df_agg_bookings.
 ↪capacity.median())
```

[46]:
```
df_agg_bookings.loc[[8, 15]]
```

[46]:

	property_id	check_in_date	room_category	successful_bookings	capacity
8	17561	1-May-22	RT1	22	25.0
15	17563	1-May-22	RT1	21	25.0

[47]:
```
df_agg_bookings[df_agg_bookings.successful_bookings > df_agg_bookings.capacity]
```

[47]:

	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

[48]:
```
df_agg_bookings.shape
```

[48]:
```
(9200, 5)
```

[49]:
```
df_agg_bookings = df_agg_bookings[df_agg_bookings.successful_bookings <=␣
 ↪df_agg_bookings.capacity]
df_agg_bookings.shape
```

[49]: (9194, 5)

0.0.4 ==> 3. Data Transformation

---

[50]:
```
df_agg_bookings.head()
```

[50]:

	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
4	16558	1-May-22	RT1	18	19.0
5	17560	1-May-22	RT1	28	40.0

Create occupancy percentage column

[51]:
```
df_agg_bookings.loc[:, "occ_pct"] = df_agg_bookings["successful_bookings"]/
 ↪df_agg_bookings["capacity"]
df_agg_bookings.head()
```

[51]:

	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	
4	16558	1-May-22	RT1	18	19.0	
5	17560	1-May-22	RT1	28	40.0	

	occ_pct
0	0.833333
1	0.933333
2	0.766667
4	0.947368
5	0.700000

Converting values to percentage format

[52]:
```
df_agg_bookings.loc[:, "occ_pct"] = df_agg_bookings["occ_pct"].apply(lambda x:␣
 ↪round(x*100, 2))
df_agg_bookings.head()
```

[52]:

	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	
4	16558	1-May-22	RT1	18	19.0	
5	17560	1-May-22	RT1	28	40.0	

```
   occ_pct
0    83.33
1    93.33
2    76.67
4    94.74
5    70.00
```

[53]:
```
df_agg_bookings.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 9194 entries, 0 to 9199
Data columns (total 6 columns):
 #   Column               Non-Null Count  Dtype
---  ------               --------------  -----
 0   property_id          9194 non-null   int64
 1   check_in_date        9194 non-null   object
 2   room_category        9194 non-null   object
 3   successful_bookings  9194 non-null   int64
 4   capacity             9194 non-null   float64
 5   occ_pct              9194 non-null   float64
dtypes: float64(2), int64(2), object(2)
memory usage: 502.8+ KB
```

0.0.5 ==> 4. Insights Generation

---

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

[54]:
```
df_agg_bookings.groupby("room_category")["occ_pct"].mean().round(2)
```

[54]:
```
room_category
RT1    57.89
RT2    58.01
RT3    58.03
RT4    59.28
Name: occ_pct, dtype: float64
```

Manager may say, "I don't understand RT1, RT2". Therefore: Printing room categories such as Standard, Premium, Elite etc along with average occupancy percentage

[55]:
```
df = pd.merge(df_agg_bookings, df_rooms, left_on = "room_category", right_on =␣
↪"room_id")
```

[56]:
```
df.head()
```

[56]:
```
   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        16558      1-May-22           RT1                   18      19.0
4        17560      1-May-22           RT1                   28      40.0

   occ_pct room_id room_class
0    83.33     RT1   Standard
1    93.33     RT1   Standard
2    76.67     RT1   Standard
3    94.74     RT1   Standard
4    70.00     RT1   Standard
```

[57]:
```
df.drop("room_id",axis = 1, inplace=True)
```

[58]:
```
df.head()
```

[58]:
```
   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        16558      1-May-22           RT1                   18      19.0
4        17560      1-May-22           RT1                   28      40.0

   occ_pct room_class
0    83.33   Standard
1    93.33   Standard
2    76.67   Standard
3    94.74   Standard
4    70.00   Standard
```

[59]:
```
df.groupby("room_class")["occ_pct"].mean().round(2)
```

[59]:
```
room_class
Elite           58.01
Premium         58.03
Presidential    59.28
Standard        57.89
Name: occ_pct, dtype: float64
```

2. Print average occupancy rate per city.

[60]:
```
df_hotels.head()
```

[60]:
```
   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
```

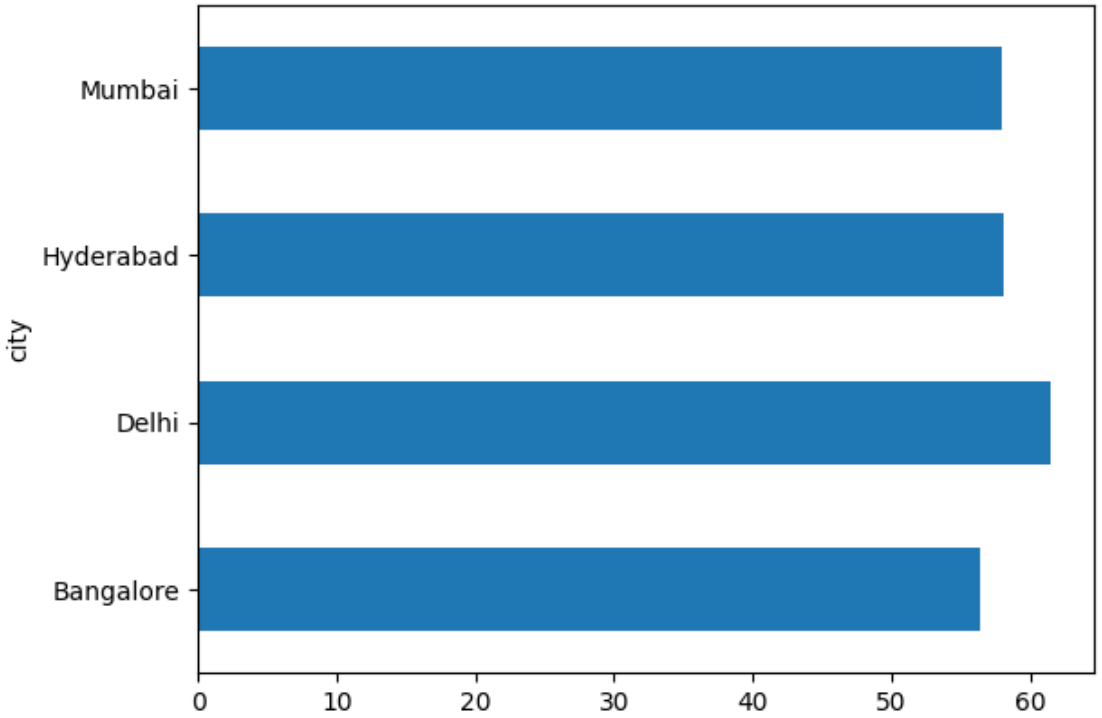
[61]:
```
df = pd.merge(df, df_hotels, on = "property_id")
df.head()
```

[61]:
```
   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        16558      1-May-22           RT1                   18      19.0
4        17560      1-May-22           RT1                   28      40.0

   occ_pct room_class  property_name  category       city
0    83.33   Standard  Atliq Exotica    Luxury     Mumbai
1    93.33   Standard      Atliq Bay    Luxury  Bangalore
2    76.67   Standard   Atliq Palace  Business  Bangalore
3    94.74   Standard   Atliq Grands    Luxury      Delhi
4    70.00   Standard     Atliq City  Business     Mumbai
```

[62]:
```
df.groupby("city")["occ_pct"].mean().plot(kind = "barh")
```

[62]:
```
<Axes: ylabel='city'>
```



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

[63]:
```
df_date.head()
```

[63]:
```
         date  mmm yy week no  day_type
0  01-May-22  May 22    W 19   weekend
1  02-May-22  May 22    W 19  weekeday
2  03-May-22  May 22    W 19  weekeday
3  04-May-22  May 22    W 19  weekeday
4  05-May-22  May 22    W 19  weekeday
```

[64]:
```
df = pd.merge(df, df_date, left_on = "check_in_date", right_on = "date")
df.head()
```

[64]:
```
   property_id check_in_date room_category  successful_bookings  capacity  \
0        19563     10-May-22           RT3                   15      29.0
1        18560     10-May-22           RT1                   19      30.0
2        19562     10-May-22           RT1                   18      30.0
3        19563     10-May-22           RT1                   16      30.0
4        17558     10-May-22           RT1                   11      19.0

   occ_pct room_class property_name  category       city       date  mmm yy  \
0    51.72    Premium  Atliq Palace  Business  Bangalore  10-May-22  May 22
1    63.33   Standard    Atliq City  Business  Hyderabad  10-May-22  May 22
2    60.00   Standard     Atliq Bay    Luxury  Bangalore  10-May-22  May 22
3    53.33   Standard  Atliq Palace  Business  Bangalore  10-May-22  May 22
4    57.89   Standard  Atliq Grands    Luxury     Mumbai  10-May-22  May 22

  week no  day_type
0    W 20  weekeday
1    W 20  weekeday
2    W 20  weekeday
3    W 20  weekeday
4    W 20  weekeday
```

[65]:
```
df.groupby("day_type")["occ_pct"].mean().round(2)
```

[65]:
```
day_type
weekeday    50.88
weekend     72.34
Name: occ_pct, dtype: float64
```

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

[66]:
```
df["mmm yy"].unique()
```

[66]:
```
array(['May 22', 'Jun 22', 'Jul 22'], dtype=object)
```

[67]:
```
df_june = df[df["mmm yy"] == "Jun 22"]
df_june.head()
```

[67]:
```
      property_id check_in_date room_category  successful_bookings  capacity  \
2200        16559     10-Jun-22           RT1                   20      30.0
2201        19562     10-Jun-22           RT1                   19      30.0
2202        19563     10-Jun-22           RT1                   17      30.0
2203        17558     10-Jun-22           RT1                    9      19.0
2204        16558     10-Jun-22           RT1                   11      19.0

      occ_pct room_class  property_name  category       city       date  \
2200    66.67   Standard  Atliq Exotica    Luxury     Mumbai  10-Jun-22
2201    63.33   Standard      Atliq Bay    Luxury  Bangalore  10-Jun-22
2202    56.67   Standard   Atliq Palace  Business  Bangalore  10-Jun-22
2203    47.37   Standard   Atliq Grands    Luxury     Mumbai  10-Jun-22
2204    57.89   Standard   Atliq Grands    Luxury      Delhi  10-Jun-22

      mmm yy week no  day_type
2200  Jun 22    W 24  weekeday
2201  Jun 22    W 24  weekeday
2202  Jun 22    W 24  weekeday
2203  Jun 22    W 24  weekeday
2204  Jun 22    W 24  weekeday
```

[68]:
```
df_june.groupby("city")["occ_pct"].mean().round(2).sort_values(ascending =␣
↪False)
```

[68]:
```
city
Delhi        62.47
Hyderabad    58.46
Mumbai       58.38
Bangalore    56.44
Name: occ_pct, dtype: float64
```

Reading and Exploring new data

[69]:
```
df_august = pd.read_csv("datasets/new_data_august.csv")
df_august.head()
```

[69]:
```
   property_id  property_name  category       city room_category room_class  \
0        16559  Atliq Exotica    Luxury     Mumbai            RT1   Standard
1        19562      Atliq Bay    Luxury  Bangalore            RT1   Standard
2        19563   Atliq Palace  Business  Bangalore            RT1   Standard
3        19558   Atliq Grands    Luxury  Bangalore            RT1   Standard
4        19560     Atliq City  Business  Bangalore            RT1   Standard

  check_in_date  mmm yy week no  day_type  successful_bookings  capacity  \
0     01-Aug-22  Aug-22    W 32  weekeday                    30        30
```

```
1     01-Aug-22  Aug-22    W 32  weekeday                    21         30
2     01-Aug-22  Aug-22    W 32  weekeday                    23         30
3     01-Aug-22  Aug-22    W 32  weekeday                    30         40
4     01-Aug-22  Aug-22    W 32  weekeday                    20         26

     occ%
0  100.00
1   70.00
2   76.67
3   75.00
4   76.92
```

[70]:
```
df_august.columns
```

[70]:
```
Index(['property_id', 'property_name', 'category', 'city', 'room_category',
       'room_class', 'check_in_date', 'mmm yy', 'week no', 'day_type',
       'successful_bookings', 'capacity', 'occ%'],
      dtype='object')
```

[71]:
```
df.columns
```

[71]:
```
Index(['property_id', 'check_in_date', 'room_category', 'successful_bookings',
       'capacity', 'occ_pct', 'room_class', 'property_name', 'category',
       'city', 'date', 'mmm yy', 'week no', 'day_type'],
      dtype='object')
```

[72]:
```
df_august.shape
```

[72]:
```
(7, 13)
```

[73]:
```
df.shape
```

[73]:
```
(6497, 14)
```

Dropping the extra date column created due to merge operation

[74]:
```
df.drop("date",axis = 1, inplace=True)
df.head()
```

[74]:
```
   property_id check_in_date room_category  successful_bookings  capacity  \
0        19563     10-May-22           RT3                   15      29.0
1        18560     10-May-22           RT1                   19      30.0
2        19562     10-May-22           RT1                   18      30.0
3        19563     10-May-22           RT1                   16      30.0
4        17558     10-May-22           RT1                   11      19.0

   occ_pct room_class property_name  category       city  mmm yy week no  \
0    51.72    Premium  Atliq Palace  Business  Bangalore  May 22    W 20
```

```
1    63.33   Standard    Atliq City  Business  Hyderabad  May 22    W 20
2    60.00   Standard     Atliq Bay    Luxury  Bangalore  May 22    W 20
3    53.33   Standard  Atliq Palace  Business  Bangalore  May 22    W 20
4    57.89   Standard  Atliq Grands    Luxury     Mumbai  May 22    W 20

   day_type
0  weekeday
1  weekeday
2  weekeday
3  weekeday
4  weekeday
```

[75]:
```
df.shape
```

[75]:
```
(6497, 13)
```

Updating the records

[76]:
```
latest_df = pd.concat([df, df_august], ignore_index = True, axis = 0)
latest_df.tail()
```

[76]:
```
      property_id check_in_date room_category  successful_bookings  capacity  \
6499        19563     01-Aug-22           RT1                   23      30.0
6500        19558     01-Aug-22           RT1                   30      40.0
6501        19560     01-Aug-22           RT1                   20      26.0
6502        17561     01-Aug-22           RT1                   18      26.0
6503        17564     01-Aug-22           RT1                   10      16.0

      occ_pct room_class  property_name  category       city  mmm yy week no  \
6499      NaN   Standard   Atliq Palace  Business  Bangalore  Aug-22    W 32
6500      NaN   Standard   Atliq Grands    Luxury  Bangalore  Aug-22    W 32
6501      NaN   Standard     Atliq City  Business  Bangalore  Aug-22    W 32
6502      NaN   Standard      Atliq Blu    Luxury     Mumbai  Aug-22    W 32
6503      NaN   Standard  Atliq Seasons  Business     Mumbai  Aug-22    W 32

      day_type   occ%
6499  weekeday  76.67
6500  weekeday  75.00
6501  weekeday  76.92
6502  weekeday  69.23
6503  weekeday  62.50
```

6. Print revenue realized per city

[77]:
```
df_bookings.head(3)
```

[77]:
```
          booking_id  property_id booking_date check_in_date checkout_date  \
1  May012216558RT12        16558     30-04-22      1/5/2022      2/5/2022
```

```
4  May012216558RT15        16558     27-04-22      1/5/2022      2/5/2022
5  May012216558RT16        16558     1/5/2022      1/5/2022      3/5/2022

   no_guests room_category booking_platform  ratings_given booking_status  \
1        2.0           RT1           others            NaN      Cancelled
4        4.0           RT1    direct online            5.0    Checked Out
5        2.0           RT1           others            4.0    Checked Out

   revenue_generated  revenue_realized
1               9100              3640
4              10920             10920
5               9100              9100
```

[78]:
```
df_hotels.head(3)
```

[78]:
```
   property_id  property_name  category    city
0        16558   Atliq Grands    Luxury   Delhi
1        16559  Atliq Exotica    Luxury  Mumbai
2        16560     Atliq City  Business   Delhi
```

[79]:
```
df_bookings_all = pd.merge(df_bookings, df_hotels, on = "property_id")
df_bookings_all.head(3)
```

[79]:
```
         booking_id  property_id booking_date check_in_date checkout_date  \
0  May012216558RT12        16558     30-04-22      1/5/2022      2/5/2022
1  May012216558RT15        16558     27-04-22      1/5/2022      2/5/2022
2  May012216558RT16        16558     1/5/2022      1/5/2022      3/5/2022

   no_guests room_category booking_platform  ratings_given booking_status  \
0        2.0           RT1           others            NaN      Cancelled
1        4.0           RT1    direct online            5.0    Checked Out
2        2.0           RT1           others            4.0    Checked Out

   revenue_generated  revenue_realized property_name category   city
0               9100              3640  Atliq Grands   Luxury  Delhi
1              10920             10920  Atliq Grands   Luxury  Delhi
2               9100              9100  Atliq Grands   Luxury  Delhi
```

[80]:
```
df_bookings_all.groupby("city")["revenue_realized"].sum()
```

[80]:
```
city
Bangalore    420383550
Delhi        294404488
Hyderabad    325179310
Mumbai       668569251
Name: revenue_realized, dtype: int64
```

7. Print month by month revenue

[81]:
```
df_date["mmm yy"].unique()
```

[81]:
```
array(['May 22', 'Jun 22', 'Jul 22'], dtype=object)
```

[82]:
```
df_date.head(2)
```

[82]:

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

[83]:
```
df_bookings_all.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 134573 entries, 0 to 134572
Data columns (total 15 columns):
 #   Column             Non-Null Count   Dtype
---  ------             --------------   -----
 0   booking_id         134573 non-null  object
 1   property_id        134573 non-null  int64
 2   booking_date       134573 non-null  object
 3   check_in_date      134573 non-null  object
 4   checkout_date      134573 non-null  object
 5   no_guests          134573 non-null  float64
 6   room_category      134573 non-null  object
 7   booking_platform   134573 non-null  object
 8   ratings_given      56676 non-null   float64
 9   booking_status     134573 non-null  object
 10  revenue_generated  134573 non-null  int64
 11  revenue_realized   134573 non-null  int64
 12  property_name      134573 non-null  object
 13  category           134573 non-null  object
 14  city               134573 non-null  object
dtypes: float64(2), int64(3), object(10)
memory usage: 15.4+ MB
```

[84]:
```
df_date.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 92 entries, 0 to 91
Data columns (total 4 columns):
 #   Column    Non-Null Count  Dtype
---  ------    --------------  -----
 0   date      92 non-null     object
 1   mmm yy    92 non-null     object
 2   week no   92 non-null     object
 3   day_type  92 non-null     object
dtypes: object(4)
memory usage: 3.0+ KB
```

[86]:
```
from dateutil import parser

df_date["date"] = df_date["date"].apply(
    lambda x: parser.parse(x, fuzzy=True) if pd.notnull(x) else pd.NaT
)
```

[87]:
```
df_date["date"] = pd.to_datetime(df_date["date"])
```

[88]:
```
df_date.head(2)
```

[88]:
```
        date  mmm yy week no  day_type
0 2022-05-01  May 22    W 19   weekend
1 2022-05-02  May 22    W 19  weekeday
```

[89]:
```
df_date.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 92 entries, 0 to 91
Data columns (total 4 columns):
 #   Column    Non-Null Count  Dtype
---  ------    --------------  -----
 0   date      92 non-null     datetime64[ns]
 1   mmm yy    92 non-null     object
 2   week no   92 non-null     object
 3   day_type  92 non-null     object
dtypes: datetime64[ns](1), object(3)
memory usage: 3.0+ KB
```

[90]:
```
from dateutil import parser

df_bookings_all["check_in_date"] = df_bookings_all["check_in_date"].apply(
    lambda x: parser.parse(x, fuzzy=True) if pd.notnull(x) else pd.NaT
)
```

[91]:
```
df_bookings_all["check_in_date"] = pd.
 ↪to_datetime(df_bookings_all["check_in_date"])
```

[93]:
```
df_bookings_all.head(3)
```

[93]:
```
         booking_id  property_id booking_date check_in_date checkout_date  \
0  May012216558RT12        16558     30-04-22    2022-01-05      2/5/2022
1  May012216558RT15        16558     27-04-22    2022-01-05      2/5/2022
2  May012216558RT16        16558     1/5/2022    2022-01-05      3/5/2022

   no_guests room_category booking_platform  ratings_given booking_status  \
0        2.0           RT1           others            NaN      Cancelled
1        4.0           RT1    direct online            5.0    Checked Out
2        2.0           RT1           others            4.0    Checked Out
```

```
   revenue_generated  revenue_realized property_name category   city
0               9100              3640  Atliq Grands   Luxury  Delhi
1              10920             10920  Atliq Grands   Luxury  Delhi
2               9100              9100  Atliq Grands   Luxury  Delhi
```

[94]:
```
df_bookings_all.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 134573 entries, 0 to 134572
Data columns (total 15 columns):
 #   Column             Non-Null Count   Dtype
---  ------             --------------   -----
 0   booking_id         134573 non-null  object
 1   property_id        134573 non-null  int64
 2   booking_date       134573 non-null  object
 3   check_in_date      134573 non-null  datetime64[ns]
 4   checkout_date      134573 non-null  object
 5   no_guests          134573 non-null  float64
 6   room_category      134573 non-null  object
 7   booking_platform   134573 non-null  object
 8   ratings_given      56676 non-null   float64
 9   booking_status     134573 non-null  object
 10  revenue_generated  134573 non-null  int64
 11  revenue_realized   134573 non-null  int64
 12  property_name      134573 non-null  object
 13  category           134573 non-null  object
 14  city               134573 non-null  object
dtypes: datetime64[ns](1), float64(2), int64(3), object(9)
memory usage: 15.4+ MB
```

[95]:
```
df_bookings_all = pd.merge(df_bookings_all, df_date, left_on="check_in_date",␣
 ↪right_on="date")
df_bookings_all.head(3)
```

[95]:
```
         booking_id  property_id booking_date check_in_date checkout_date  \
0  May052216558RT11        16558     15-04-22    2022-05-05      7/5/2022
1  May052216558RT12        16558     30-04-22    2022-05-05      7/5/2022
2  May052216558RT13        16558     1/5/2022    2022-05-05      6/5/2022

   no_guests room_category booking_platform  ratings_given booking_status  \
0        3.0           RT1         tripster            5.0    Checked Out
1        2.0           RT1           others            NaN      Cancelled
2        3.0           RT1   direct offline            5.0    Checked Out

   revenue_generated  revenue_realized property_name category   city  \
0              10010             10010  Atliq Grands   Luxury  Delhi
1               9100              3640  Atliq Grands   Luxury  Delhi
```

```
2              10010             10010  Atliq Grands   Luxury  Delhi

        date  mmm yy week no  day_type
0 2022-05-05  May 22    W 19  weekeday
1 2022-05-05  May 22    W 19  weekeday
2 2022-05-05  May 22    W 19  weekeday
```

[96]:
```
df_bookings_all.drop("date",axis = 1, inplace=True)
df_bookings_all.head(3)
```

[96]:
```
         booking_id  property_id booking_date check_in_date checkout_date  \
0  May052216558RT11        16558     15-04-22    2022-05-05      7/5/2022
1  May052216558RT12        16558     30-04-22    2022-05-05      7/5/2022
2  May052216558RT13        16558     1/5/2022    2022-05-05      6/5/2022

   no_guests room_category booking_platform  ratings_given booking_status  \
0        3.0           RT1         tripster            5.0    Checked Out
1        2.0           RT1           others            NaN      Cancelled
2        3.0           RT1   direct offline            5.0    Checked Out

   revenue_generated  revenue_realized property_name category   city  mmm yy  \
0              10010             10010  Atliq Grands   Luxury  Delhi  May 22
1               9100              3640  Atliq Grands   Luxury  Delhi  May 22
2              10010             10010  Atliq Grands   Luxury  Delhi  May 22

  week no  day_type
0    W 19  weekeday
1    W 19  weekeday
2    W 19  weekeday
```

[97]:
```
df_bookings_all.groupby("mmm yy")["revenue_realized"].sum()
```

[97]:
```
mmm yy
Jul 22    389940912
Jun 22    377191229
May 22    408375641
Name: revenue_realized, dtype: int64
```

Print revenue realized per hotel type

[98]:
```
df_bookings_all.groupby("property_name")["revenue_realized"].sum().sort_values()
```

[98]:
```
property_name
Atliq Seasons     45920757
Atliq Grands     145860641
Atliq Blu        179203544
Atliq Bay        179416721
Atliq City       196555383
```

```
Atliq Palace     209474575
Atliq Exotica    219076161
Name: revenue_realized, dtype: int64
```

Print average rating per city

[99]:
```
df_bookings_all.groupby("city")["ratings_given"].mean().round(2)
```

[99]:
```
city
Bangalore    3.40
Delhi        3.78
Hyderabad    3.66
Mumbai       3.64
Name: ratings_given, dtype: float64
```

Print a pie chart of revenue realized per booking platform

[100]:
```
df_bookings_all.groupby("booking_platform")["revenue_realized"].sum().plot(kind␣
↪= "pie")
```

[100]:
```
<Axes: ylabel='revenue_realized'>
```