hotel-data-analytics

February 22, 2024

1 AtliQ Hotels Data Analysis Project

1.1 Exploratory Data Analysis

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

[189]:
```
df_date = pd.read_csv("D:/# Learning/4- Python/Atliq Hotel - datasets/dim_date.
 ↪csv")
df_hotels = pd.read_csv("D:/# Learning/4- Python/Atliq Hotel - datasets/
 ↪dim_hotels.csv")
df_rooms = pd.read_csv("D:/# Learning/4- Python/Atliq Hotel - datasets/
 ↪dim_rooms.csv")
df_bookings = pd.read_csv("D:/# Learning/4- Python/Atliq Hotel - datasets/
 ↪fact_bookings.csv")
df_aggregated_bookings = pd.read_csv("D:/# Learning/4- Python/Atliq Hotel -␣
 ↪datasets/fact_aggregated_bookings.csv")
```

[4]:
```
df_bookings.head()
```

[4]:

	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
0	May012216558RT11	16558	27-04-2022	01-05-2022	02-05-2022	-3.0	RT1	direct online	1.0	Checked Out	10010	10010
1	May012216558RT12	16558	30-04-2022	01-05-2022	02-05-2022	2.0	RT1	others	NaN	Cancelled	9100	3640
2	May012216558RT13	16558	28-04-2022	01-05-2022	04-05-2022	2.0	RT1	logtrip	5.0	Checked Out		
3	May012216558RT14	16558	28-04-2022	01-05-2022	02-05-2022	-2.0	RT1	others	NaN	Cancelled		
4	May012216558RT15	16558	27-04-2022	01-05-2022	02-05-2022	4.0	RT1	direct online	5.0	Checked Out		

```
2           9100000             9100
3              9100             3640
4             10920            10920
```

[7]:
```
df_bookings.columns
```

[7]:
```
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')
```

[9]:
```
df_bookings.room_category.unique()
```

[9]:
```
array(['RT1', 'RT2', 'RT3', 'RT4'], dtype=object)
```

[10]:
```
df_bookings.booking_platform.unique()
```

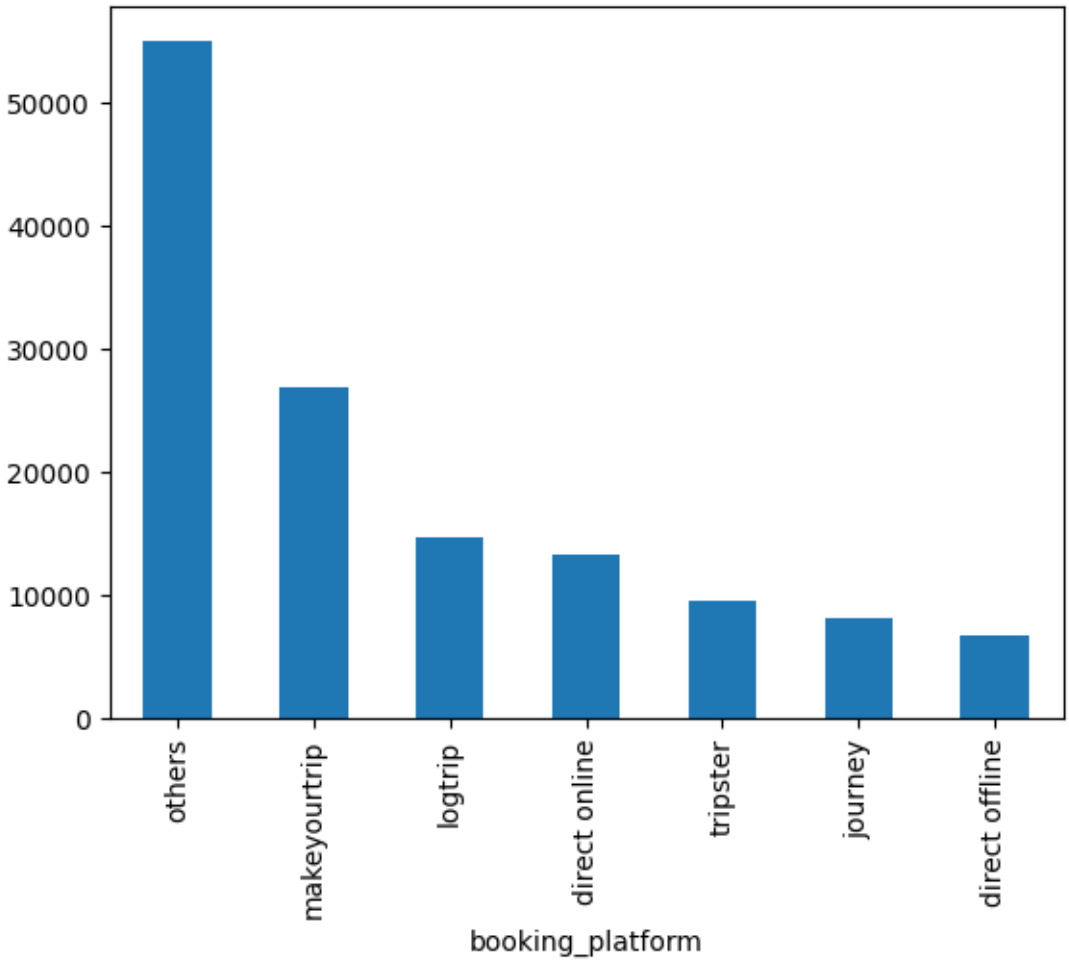
[10]:
```
array(['direct online', 'others', 'logtrip', 'tripster', 'makeyourtrip',
       'journey', 'direct offline'], dtype=object)
```

[11]:
```
df_bookings.booking_platform.value_counts()
```

[11]:
```
booking_platform
others            55066
makeyourtrip      26898
logtrip           14756
direct online     13379
tripster           9630
journey            8106
direct offline     6755
Name: count, dtype: int64
```

[12]:
```
df_bookings.booking_platform.value_counts().plot(kind = "bar")
```

[12]:
```
<Axes: xlabel='booking_platform'>
```



[13]:
```
df_bookings.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 134590 entries, 0 to 134589
Data columns (total 12 columns):
 #   Column             Non-Null Count   Dtype  
---  ------             --------------   -----  
 0   booking_id         134590 non-null  object 
 1   property_id        134590 non-null  int64  
 2   booking_date       134590 non-null  object 
 3   check_in_date      134590 non-null  object 
 4   checkout_date      134590 non-null  object 
 5   no_guests          134587 non-null  float64
 6   room_category      134590 non-null  object 
 7   booking_platform   134590 non-null  object 
 8   ratings_given      56683 non-null   float64
 9   booking_status     134590 non-null  object 
```

```
 10  revenue_generated  134590 non-null  int64
 11  revenue_realized   134590 non-null  int64
dtypes: float64(2), int64(3), object(7)
memory usage: 12.3+ MB
```

[16]:
```
df_hotels.shape
```

[16]:
```
(25, 4)
```

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

[17]:
```
   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
```

[18]:
```
df_hotels.property_name.unique()
```

[18]:
```
array(['Atliq Grands', 'Atliq Exotica', 'Atliq City', 'Atliq Blu',
       'Atliq Bay', 'Atliq Palace', 'Atliq Seasons'], dtype=object)
```

[19]:
```
df_hotels.category.unique()
```

[19]:
```
array(['Luxury', 'Business'], dtype=object)
```

[20]:
```
df_hotels.city.unique()
```

[20]:
```
array(['Delhi', 'Mumbai', 'Hyderabad', 'Bangalore'], dtype=object)
```

[21]:
```
df_hotels.category.value_counts()
```

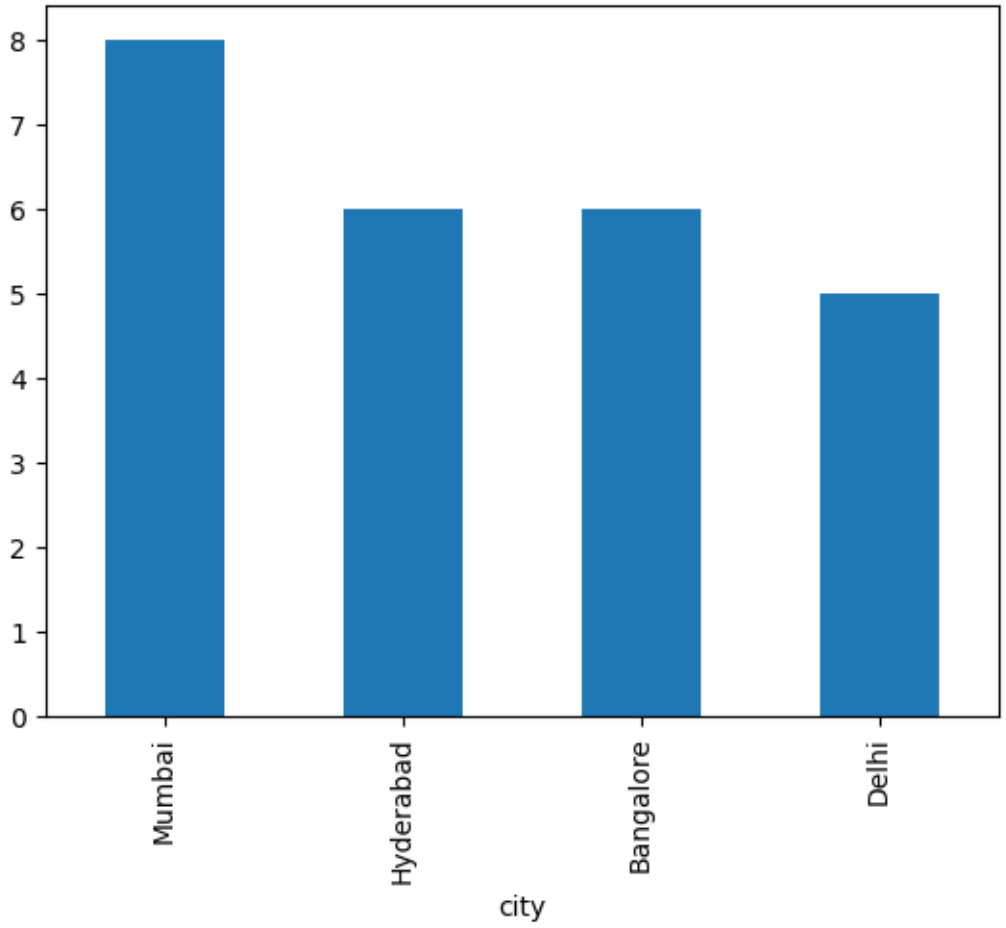
[21]:
```
category
Luxury      16
Business     9
Name: count, dtype: int64
```

[22]:
```
df_hotels.city.value_counts()
```

[22]:
```
city
Mumbai       8
Hyderabad    6
Bangalore    6
Delhi        5
Name: count, dtype: int64
```

[25]:
```
df_hotels.city.value_counts().plot(kind = "bar")
```

[25]: <Axes: xlabel='city'>



[24]:
```
df_hotels.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 25 entries, 0 to 24
Data columns (total 4 columns):
 #   Column         Non-Null Count  Dtype 
---  ------         --------------  ----- 
 0   property_id    25 non-null     int64 
 1   property_name  25 non-null     object
 2   category       25 non-null     object
 3   city           25 non-null     object
dtypes: int64(1), object(3)
memory usage: 932.0+ bytes
```

[26]:
```
df_aggregated_bookings.head()
```

[26]:

	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

[27]:
```
df_aggregated_bookings.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 9200 entries, 0 to 9199
Data columns (total 5 columns):
 #   Column               Non-Null Count  Dtype
---  ------               --------------  -----
 0   property_id          9200 non-null   int64
 1   check_in_date        9200 non-null   object
 2   room_category        9200 non-null   object
 3   successful_bookings  9200 non-null   int64
 4   capacity             9198 non-null   float64
dtypes: float64(1), int64(2), object(2)
memory usage: 359.5+ KB
```

[28]:
```
df_aggregated_bookings.describe()
```

[28]:

	property_id	successful_bookings	capacity
count	9200.000000	9200.000000	9198.000000
mean	18040.640000	14.655761	25.280496
std	1099.818325	7.736170	11.442080
min	16558.000000	1.000000	3.000000
25%	17558.000000	9.000000	18.000000
50%	17564.000000	14.000000	25.000000
75%	18563.000000	19.000000	34.000000
max	19563.000000	123.000000	50.000000

Exercise-1. Find out unique property ids in aggregate bookings dataset

[29]:
```
df_aggregated_bookings.property_id.unique()
```

[29]:
```
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], dtype=int64)
```

Exercise-2. Find out total bookings per property_id

[45]:
```
df_aggregated_bookings.groupby("property_id")['successful_bookings'].sum()
```

[45]:
```
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
```

Exercise-3. Find out days on which bookings are greater than capacity

[48]:
```
df_aggregated_bookings[df_aggregated_bookings.successful_bookings >␣
↪df_aggregated_bookings.capacity]
```

[48]:

	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

1.2 Data Cleaning

[50]:
```
df_bookings.describe()
```

[50]:

	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
```

We will filter data to include only values greater than 0 in the 'no-guests' column, as the minimum value of -17 seems incorrect.

[54]:
```
df_bookings = df_bookings[df_bookings.no_guests > 0]
df_bookings
```

[54]:
```
                booking_id  property_id booking_date check_in_date  \
1         May012216558RT12        16558   30-04-2022    01-05-2022
2         May012216558RT13        16558   28-04-2022    01-05-2022
4         May012216558RT15        16558   27-04-2022    01-05-2022
5         May012216558RT16        16558   01-05-2022    01-05-2022
6         May012216558RT17        16558   28-04-2022    01-05-2022
…                      …          …            …             …
134584   Jul312217564RT45        17564   30-07-2022    31-07-2022
134585   Jul312217564RT46        17564   29-07-2022    31-07-2022
134587   Jul312217564RT48        17564   30-07-2022    31-07-2022
134588   Jul312217564RT49        17564   29-07-2022    31-07-2022
134589  Jul312217564RT410        17564   31-07-2022    31-07-2022

       checkout_date  no_guests room_category booking_platform  ratings_given  \
1         02-05-2022        2.0           RT1           others            NaN
2         04-05-2022        2.0           RT1          logtrip            5.0
4         02-05-2022        4.0           RT1    direct online            5.0
5         03-05-2022        2.0           RT1           others            4.0
6         06-05-2022        2.0           RT1           others            NaN
…                ...        …             …              …              …
134584    01-08-2022        2.0           RT4           others            2.0
134585    03-08-2022        1.0           RT4     makeyourtrip            2.0
134587    02-08-2022        1.0           RT4         tripster            NaN
134588    01-08-2022        2.0           RT4          logtrip            2.0
134589    01-08-2022        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]
```

[55]:
```
df_bookings.shape
```

[55]:
```
(134578, 12)
```

[60]:
```
df_bookings.revenue_generated.min(),df_bookings.revenue_generated.
 ↪mean(),df_bookings.revenue_generated.median(),df_bookings.revenue_generated.
 ↪max()
```

[60]:
```
(6500, 15378.036937686695, 13500.0, 28560000)
```

Based on the insights provided, the average revenue per property for the hotel is 15,378, with a median of 13,500. While the minimum revenue per property is 6,500, the maximum revenue of 28,560,000 seems unusually high compared to both the mean and median. It suggests a potential typo or outlierrs

We will set a higher limit using three standard deviations, calculated as mean plus three times the standard deviation

[74]:
```
higher_limit = df_bookings.revenue_generated.mean() + (3*df_bookings.
 ↪revenue_generated.std())
higher_limit
```

[74]:
```
34273.98826948176
```

[71]:
```
df_bookings = df_bookings[df_bookings.revenue_generated < higher_limit]
df_bookings
```

[71]:
```
              booking_id  property_id booking_date check_in_date  \
1       May012216558RT12        16558   30-04-2022    01-05-2022
4       May012216558RT15        16558   27-04-2022    01-05-2022
5       May012216558RT16        16558   01-05-2022    01-05-2022
```

```
6         May012216558RT17         16558   28-04-2022    01-05-2022
7         May012216558RT18         16558   26-04-2022    01-05-2022
…                      …           …          …            …
134584   Jul312217564RT45          17564   30-07-2022    31-07-2022
134585   Jul312217564RT46          17564   29-07-2022    31-07-2022
134587   Jul312217564RT48          17564   30-07-2022    31-07-2022
134588   Jul312217564RT49          17564   29-07-2022    31-07-2022
134589  Jul312217564RT410          17564   31-07-2022    31-07-2022

       checkout_date  no_guests room_category booking_platform  ratings_given  \
1         02-05-2022        2.0           RT1           others            NaN
4         02-05-2022        4.0           RT1    direct online            5.0
5         03-05-2022        2.0           RT1           others            4.0
6         06-05-2022        2.0           RT1           others            NaN
7         03-05-2022        2.0           RT1          logtrip            NaN
…                ….        …           …              …            …
134584    01-08-2022        2.0           RT4           others            2.0
134585    03-08-2022        1.0           RT4     makeyourtrip            2.0
134587    02-08-2022        1.0           RT4         tripster            NaN
134588    01-08-2022        2.0           RT4          logtrip            2.0
134589    01-08-2022        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]
```

[72]:
```
df_bookings.revenue_realized.describe()
```

[72]:
```
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
```

[76]:
```
higher_limit = df_bookings.revenue_realized.mean() + (3*df_bookings.
 ↪revenue_realized.std())
higher_limit
```

[76]:
```
33479.358661845814
```

[77]:
```
df_bookings[df_bookings.revenue_realized > higher_limit]
```

[77]:

	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
137	May012216559RT41	16559	27-04-2022	01-05-2022	07-05-2022	4.0	RT4	others	NaN	Checked Out	38760	38760
139	May012216559RT43	16559	01-05-2022	01-05-2022	02-05-2022	6.0	RT4	tripster	3.0	Checked Out	45220	45220
143	May012216559RT47	16559	28-04-2022	01-05-2022	03-05-2022	3.0	RT4	others	5.0	Checked Out	35530	35530
149	May012216559RT413	16559	24-04-2022	01-05-2022	07-05-2022	5.0	RT4	logtrip	NaN	Checked Out	41990	41990
222	May012216560RT45	16560	30-04-2022	01-05-2022	03-05-2022	5.0	RT4	others	3.0	Checked Out	34580	34580
…	…	…	…	…	…	…	…	…	…	…	…	…
134328	Jul312219560RT49	19560	31-07-2022	31-07-2022	02-08-2022	6.0	RT4	direct online	5.0	Checked Out	39900	39900
134331	Jul312219560RT412	19560	31-07-2022	31-07-2022	01-08-2022	6.0	RT4	others	2.0	Checked Out	39900	39900
134467	Jul312219562RT45	19562	28-07-2022	31-07-2022	01-08-2022	6.0	RT4	makeyourtrip	4.0	Checked Out	39900	39900
134474	Jul312219562RT412	19562	25-07-2022	31-07-2022	06-08-2022	5.0	RT4	direct offline	5.0	Checked Out	37050	37050
134581	Jul312217564RT42	17564	31-07-2022	31-07-2022	01-08-2022	4.0	RT4	makeyourtrip	4.0			

```
134581    Checked Out              38760             38760

[1299 rows x 12 columns]
```

[78]:
```
df_rooms
```

[78]:
```
  room_id    room_class
0     RT1      Standard
1     RT2         Elite
2     RT3       Premium
3     RT4  Presidential
```

[80]:
```
df_bookings[df_bookings.room_category == 'RT4'].revenue_realized.describe()
```

[80]:
```
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
```

[81]:
```
23439 + 3*9048
```

[81]:
```
50583
```

[83]:
```
df_bookings.isnull().sum()
```

[83]:
```
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
```

1.3 Data Transformation

[112]:
```
df_aggregated_bookings.head()
```

[112]:

	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

We need to add Occupancy% column

[114]:
```
df_aggregated_bookings["Occu_%"] =␣
↪df_aggregated_bookings["successful_bookings"]*100/
↪df_aggregated_bookings["capacity"]
```

[115]:
```
df_aggregated_bookings
```

[115]:

	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	
…	…	…	…	…	…	
9195	16563	31-Jul-22	RT4	13	18.0	
9196	16559	31-Jul-22	RT4	13	18.0	
9197	17558	31-Jul-22	RT4	3	6.0	
9198	19563	31-Jul-22	RT4	3	6.0	
9199	17561	31-Jul-22	RT4	3	4.0	

	Occu_%
0	83.333333
1	93.333333
2	76.666667
3	157.894737
4	94.736842
…	…
9195	72.222222
9196	72.222222
9197	50.000000
9198	50.000000
9199	75.000000

[9200 rows x 6 columns]

1.4 Insights

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

[125]:
```
df_aggregated_bookings.groupby("room_category")["Occu_%"].mean().round(2)
```

[125]:
```
room_category
RT1    58.22
RT2    58.04
RT3    58.03
RT4    59.30
Name: Occu_%, dtype: float64
```

[126]:
```
df_rooms
```

[126]:

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

[127]:
```
df = pd.merge(df_aggregated_bookings,df_rooms, left_on = "room_category",␣
↪right_on = "room_id")
df
```

[127]:

	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	
…	…	…	…	…	…	
9195	16563	31-Jul-22	RT4	13	18.0	
9196	16559	31-Jul-22	RT4	13	18.0	
9197	17558	31-Jul-22	RT4	3	6.0	
9198	19563	31-Jul-22	RT4	3	6.0	
9199	17561	31-Jul-22	RT4	3	4.0	

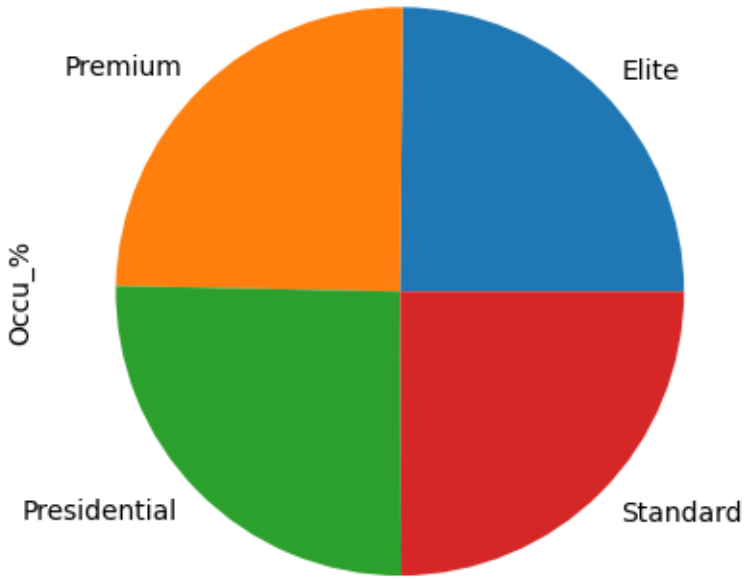
	Occu_%	room_id	room_class
0	83.333333	RT1	Standard
1	93.333333	RT1	Standard
2	76.666667	RT1	Standard
3	157.894737	RT1	Standard
4	94.736842	RT1	Standard
…	…	…	…
9195	72.222222	RT4	Presidential
9196	72.222222	RT4	Presidential
9197	50.000000	RT4	Presidential

```
9198   50.000000     RT4  Presidential
9199   75.000000     RT4  Presidential

[9200 rows x 8 columns]
```

[129]:
```
df.groupby("room_class")["Occu_%"].mean().round(2).plot(kind = "p")
```

[129]:
```
<Axes: ylabel='Occu_%'>
```



2. Print average occupancy rate per city

[149]:
```
df = pd.merge(df_aggregated_bookings,df_hotels, on = "property_id")
df.head()
```

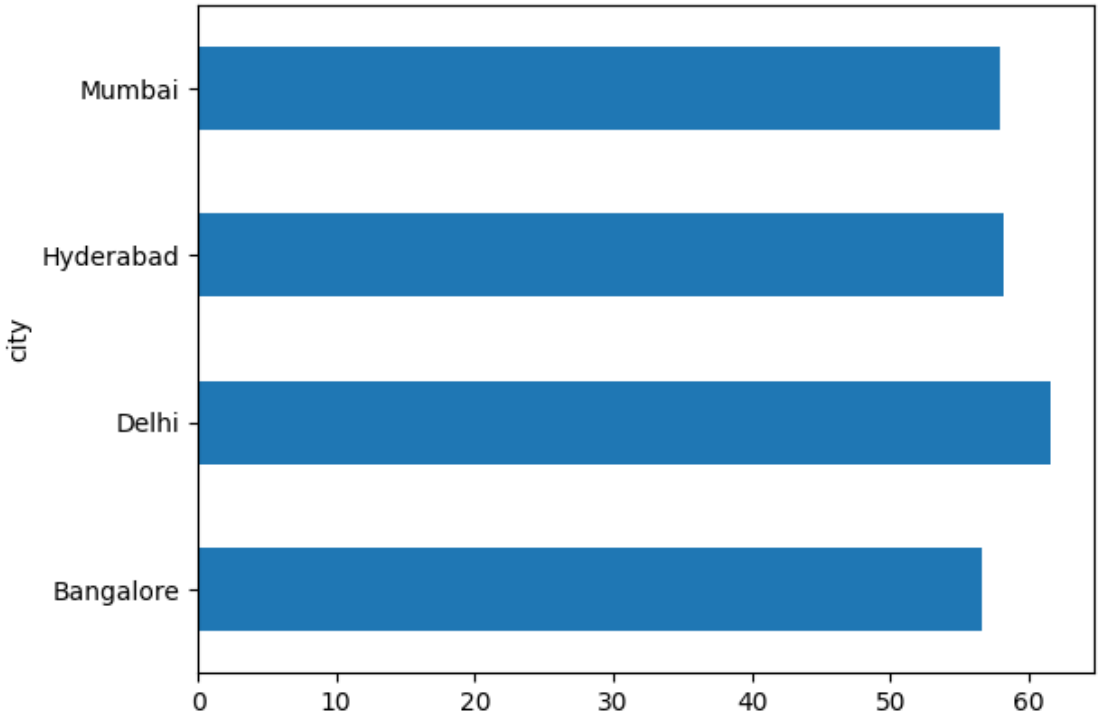
[149]:
```
   property_id check_in_date room_category  successful_bookings  capacity  \
0        16559      1-May-22           RT1                   25      30.0
1        19562      1-May-22           RT1                   28      30.0
2        19563      1-May-22           RT1                   23      30.0
3        17558      1-May-22           RT1                   30      19.0
4        16558      1-May-22           RT1                   18      19.0

       Occu_%  property_name  category       city
0   83.333333  Atliq Exotica    Luxury     Mumbai
1   93.333333      Atliq Bay    Luxury  Bangalore
```

```
2   76.666667   Atliq Palace  Business  Bangalore
3  157.894737   Atliq Grands    Luxury     Mumbai
4   94.736842   Atliq Grands    Luxury      Delhi
```

[150]:
```
df.groupby("city")["Occu_%"].mean().round(2).plot(kind = "barh")
```

[150]:
```
<Axes: ylabel='city'>
```



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

[151]:
```
df = pd.merge(df_aggregated_bookings,df_date,left_on = "check_in_date",right_on␣
↪= "date")
df.head()
```

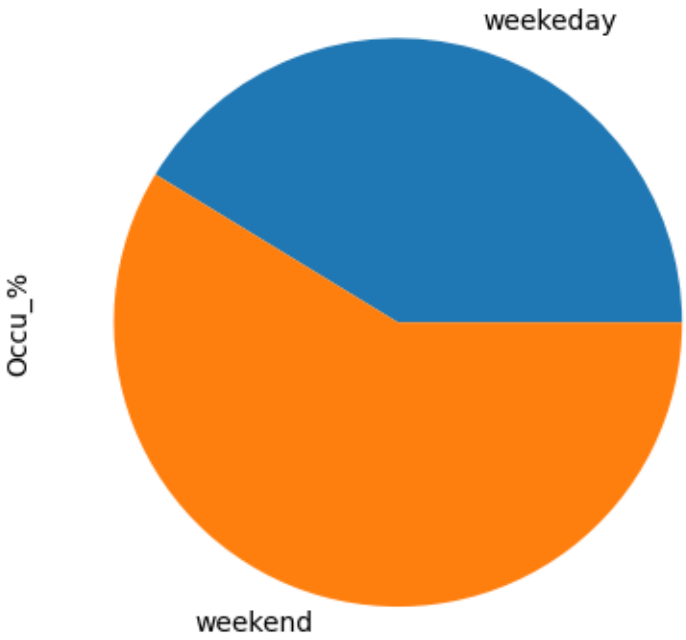
[151]:
```
   property_id check_in_date room_category  successful_bookings  capacity  \
0        19563     10-May-22           RT3                   15      29.0
1        18560     10-May-22           RT1                   19      30.0
2        19562     10-May-22           RT1                   18      30.0
3        19563     10-May-22           RT1                   16      30.0
4        17558     10-May-22           RT1                   11      19.0

      Occu_%       date  mmm yy week no  day_type
0  51.724138  10-May-22  May-22    W 20  weekeday
```

```
1  63.333333  10-May-22  May-22    W 20  weekeday
2  60.000000  10-May-22  May-22    W 20  weekeday
3  53.333333  10-May-22  May-22    W 20  weekeday
4  57.894737  10-May-22  May-22    W 20  weekeday
```

[148]:
```
df_combined_date.groupby("day_type")["Occu_%"].mean().round(2).plot(kind =␣
↪"pie")
```

[148]: <Axes: ylabel='Occu_%'>



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

[152]:
```
df
```

[152]:

	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	
…	…	…	…	…	…	
6495	16563	31-Jul-22	RT4	13	18.0	
6496	16559	31-Jul-22	RT4	13	18.0	
6497	17558	31-Jul-22	RT4	3	6.0	

```
6498        19563     31-Jul-22           RT4                    3       6.0
6499        17561     31-Jul-22           RT4                    3       4.0

         Occu_%       date  mmm yy week no  day_type
0     51.724138  10-May-22  May-22    W 20  weekeday
1     63.333333  10-May-22  May-22    W 20  weekeday
2     60.000000  10-May-22  May-22    W 20  weekeday
3     53.333333  10-May-22  May-22    W 20  weekeday
4     57.894737  10-May-22  May-22    W 20  weekeday
…           …          …       …       …        …
6495  72.222222  31-Jul-22  Jul-22    W 32   weekend
6496  72.222222  31-Jul-22  Jul-22    W 32   weekend
6497  50.000000  31-Jul-22  Jul-22    W 32   weekend
6498  50.000000  31-Jul-22  Jul-22    W 32   weekend
6499  75.000000  31-Jul-22  Jul-22    W 32   weekend

[6500 rows x 10 columns]
```

[155]:
```
df_june_22 = df[df["mmm yy"]=="Jun-22"]
df_june_22.shape
```

[155]:
```
(2100, 10)
```

[158]:
```
df_june = pd.merge(df_june_22,df_hotels,on = "property_id")
df_june.head()
```

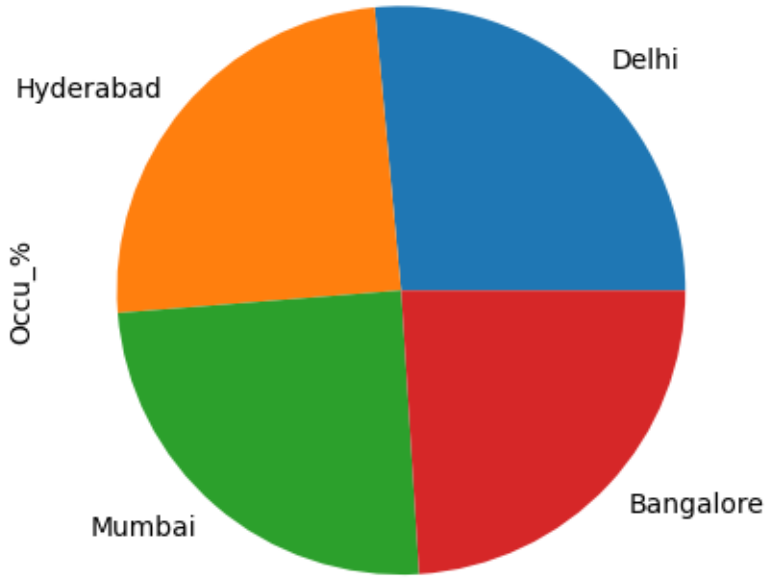
[158]:
```
   property_id check_in_date room_category  successful_bookings  capacity  \
0        16559     10-Jun-22           RT1                   20      30.0
1        19562     10-Jun-22           RT1                   19      30.0
2        19563     10-Jun-22           RT1                   17      30.0
3        17558     10-Jun-22           RT1                    9      19.0
4        16558     10-Jun-22           RT1                   11      19.0

      Occu_%       date  mmm yy week no  day_type  property_name  category  \
0  66.666667  10-Jun-22  Jun-22    W 24  weekeday  Atliq Exotica    Luxury
1  63.333333  10-Jun-22  Jun-22    W 24  weekeday      Atliq Bay    Luxury
2  56.666667  10-Jun-22  Jun-22    W 24  weekeday   Atliq Palace  Business
3  47.368421  10-Jun-22  Jun-22    W 24  weekeday   Atliq Grands    Luxury
4  57.894737  10-Jun-22  Jun-22    W 24  weekeday   Atliq Grands    Luxury

        city
0     Mumbai
1  Bangalore
2  Bangalore
3     Mumbai
4      Delhi
```

[164]:
```
df_june.groupby("city")["Occu_%"].mean().round(2).sort_values(ascending=False).
↪plot(kind = "pie")
```

[164]:
```
<Axes: ylabel='Occu_%'>
```



6. Print revenue realized per city

[171]:
```
df_booking_city = pd.merge(df_bookings,df_hotels,on = "property_id")
df_booking_city.head()
```

[171]:

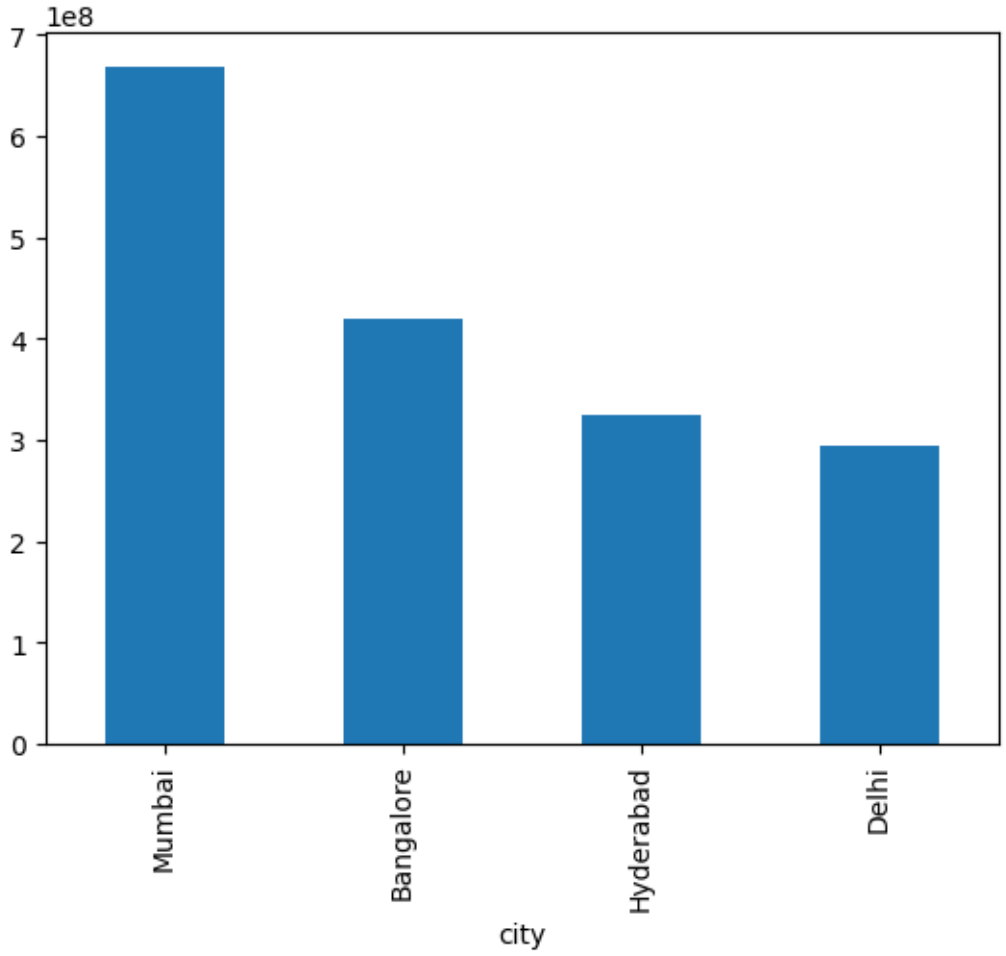
	booking_id	property_id	booking_date	check_in_date	checkout_date	\
0	May012216558RT11	16558	27-04-2022	01-05-2022	02-05-2022	
1	May012216558RT12	16558	30-04-2022	01-05-2022	02-05-2022	
2	May012216558RT13	16558	28-04-2022	01-05-2022	04-05-2022	
3	May012216558RT14	16558	28-04-2022	01-05-2022	02-05-2022	
4	May012216558RT15	16558	27-04-2022	01-05-2022	02-05-2022	

	no_guests	room_category	booking_platform	ratings_given	booking_status	\
0	-3.0	RT1	direct online	1.0	Checked Out	
1	2.0	RT1	others	NaN	Cancelled	
2	2.0	RT1	logtrip	5.0	Checked Out	
3	-2.0	RT1	others	NaN	Cancelled	
4	4.0	RT1	direct online	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	9100000	9100	Atliq Grands	Luxury	Delhi
3	9100	3640	Atliq Grands	Luxury	Delhi
4	10920	10920	Atliq Grands	Luxury	Delhi

[172]:
```
df_booking_city.groupby("city")["revenue_realized"].sum().sort_values(ascending
↪=False).plot(kind = "bar")
```

[172]: <Axes: xlabel='city'>



Exercise-1. Print revenue realized per hotel type

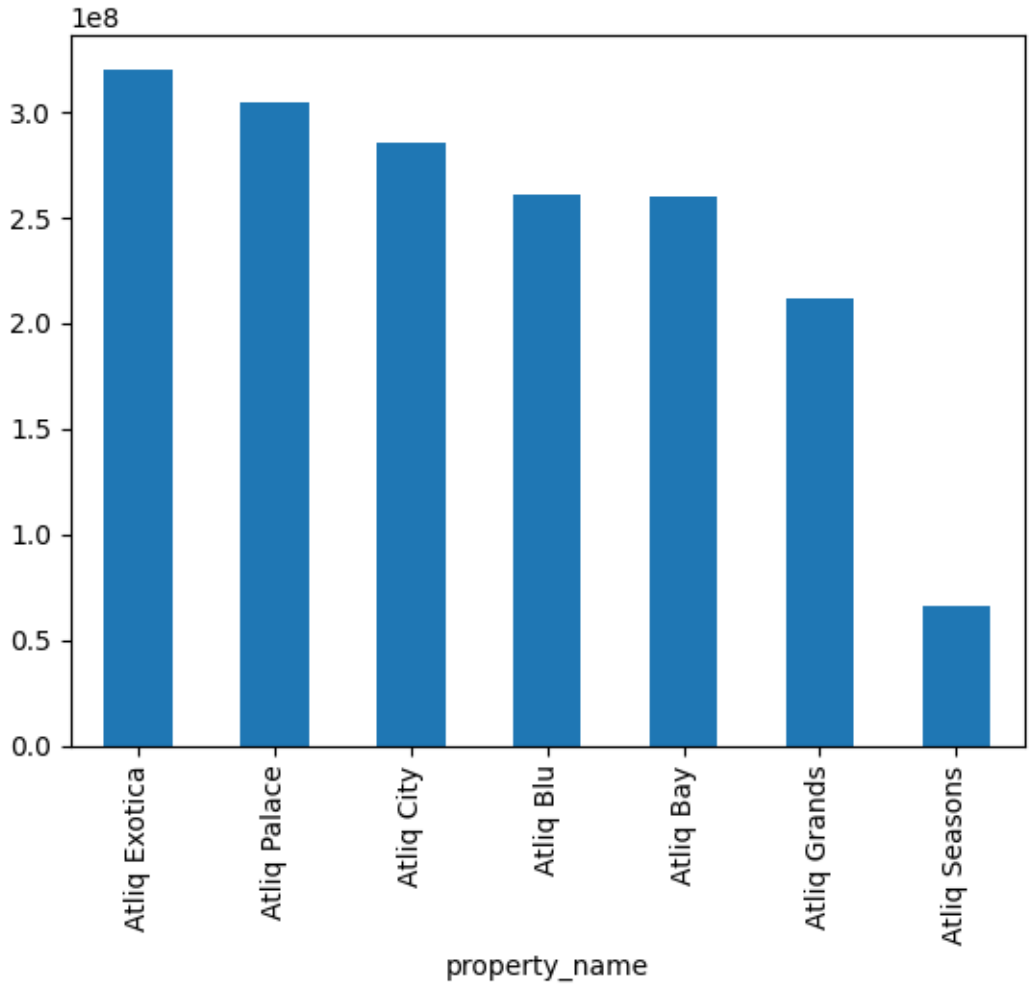
[194]:
```
df = pd.merge(df_bookings,df_hotels,on = "property_id")
df.head()
```

[194]:

	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	property_name	category	city
0	May012216558RT11	16558	27-04-2022	01-05-2022	02-05-2022	-3.0	RT1	direct online	1.0	Checked Out	10010	10010	Atliq Grands	Luxury	Delhi
1	May012216558RT12	16558	30-04-2022	01-05-2022	02-05-2022	2.0	RT1	others	NaN	Cancelled	9100	3640	Atliq Grands	Luxury	Delhi
2	May012216558RT13	16558	28-04-2022	01-05-2022	04-05-2022	2.0	RT1	logtrip	5.0	Checked Out	9100000	9100	Atliq Grands	Luxury	Delhi
3	May012216558RT14	16558	28-04-2022	01-05-2022	02-05-2022	-2.0	RT1	others	NaN	Cancelled	9100	3640	Atliq Grands	Luxury	Delhi
4	May012216558RT15	16558	27-04-2022	01-05-2022	02-05-2022	4.0	RT1	direct online	5.0	Checked Out	10920	10920	Atliq Grands	Luxury	Delhi

[199]:
```
df.groupby("property_name")["revenue_realized"].sum().sort_values(ascending =␣
↪False).plot(kind = "bar")
```

[199]:
```
<Axes: xlabel='property_name'>
```



Exercise-2 Print average rating per city

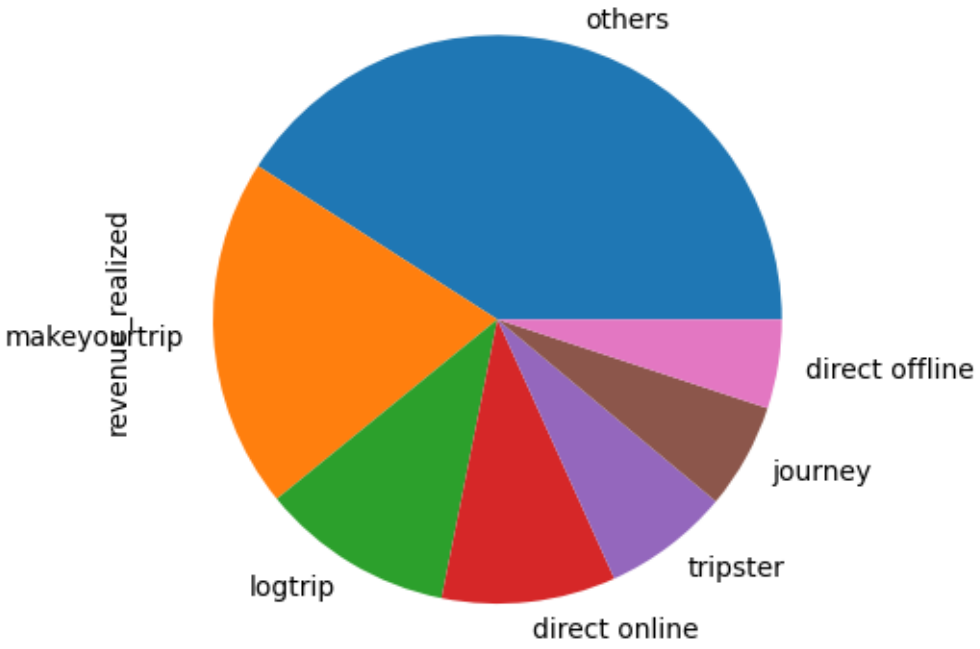
[203]:
```
df.groupby("city")["ratings_given"].mean().round(2).sort_values(ascending =␣
↪False)
```

[203]:
```
city
Delhi        3.78
Hyderabad    3.66
Mumbai       3.65
Bangalore    3.41
Name: ratings_given, dtype: float64
```

Exercise-3 Print a pie chart of revenue realized per booking platform

[208]:
```
df.groupby("booking_platform")["revenue_realized"].sum().round(2).
↪sort_values(ascending = False).plot(kind = "pie")
```

[208]: <Axes: ylabel='revenue_realized'>



[ ]: