Hotels Data Analysis Project

In [1]:

```
import pandas as pd
```

In [2]:

```
#After loading the data, I have done some basic data exploration(i.e Trying to make
df_booking = pd.read_csv("fact_bookings.csv")
df_booking.head()
```

Out[2]:

	booking_id	property_id	booking_date	check_in_date	checkout_date	no_guests	roon
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 [3]:

```
df_booking.shape
```

Out[3]:

```
(134590, 12)
```

In [4]:

```
df_booking.room_category.unique()
```

Out[4]:

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

In [5]:

```
df_booking.booking_platform.unique()
```

Out[5]:

```
array(['direct online', 'others', 'logtrip', 'tripster', 'makeyourtri
p',
       'journey', 'direct offline'], dtype=object)
```

In [6]:

```
# Finding the count of booking made per platform.
df_booking.booking_platform.value_counts()
```

Out[6]:

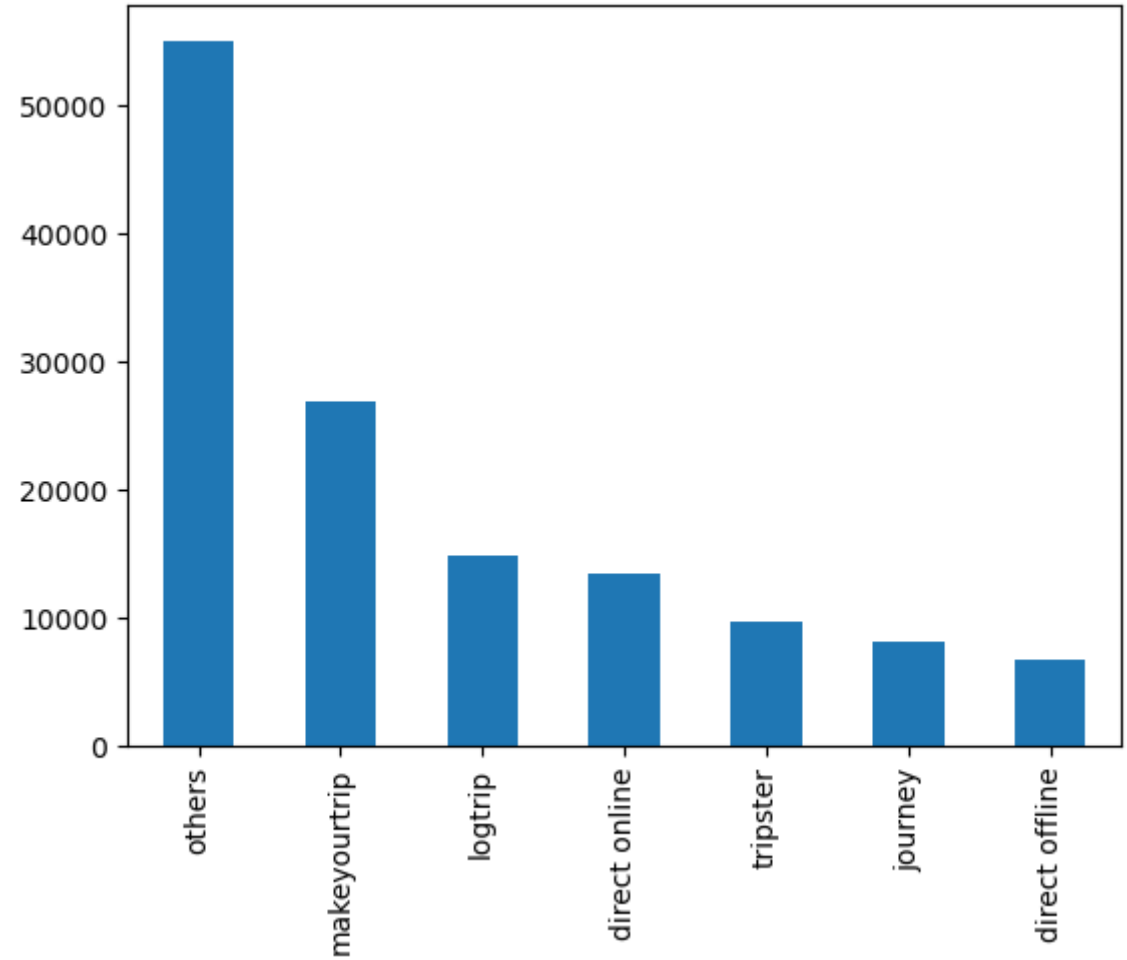
```
others            55066
makeyourtrip      26898
logtrip           14756
direct online     13379
tripster           9630
journey            8106
direct offline     6755
Name: booking_platform, dtype: int64
```

In [7]:

```
df_booking.booking_platform.value_counts().plot(kind="bar")
```

Out[7]:

```
<Axes: >
```



In [8]:

```
df_booking.describe()
```

Out[8]:

	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 [9]:

```
df_booking.revenue_generated.min(),df_booking.revenue_generated.max()
```

Out[9]:

```
(6500, 28560000)
```

In [10]:

```
df_date = pd.read_csv("dim_date.csv")
df_hotels = pd.read_csv("dim_hotels.csv")
df_rooms = pd.read_csv("dim_rooms.csv")
df_agg_bookings = pd.read_csv("fact_aggregated_bookings.csv")
```

In [11]:

```
df_hotels.shape
```

Out[11]:

```
(25, 4)
```

In [12]:

```
df_hotels.head()
```

Out[12]:

	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 [13]:

```
df_hotels.category.value_counts()
```

Out[13]:

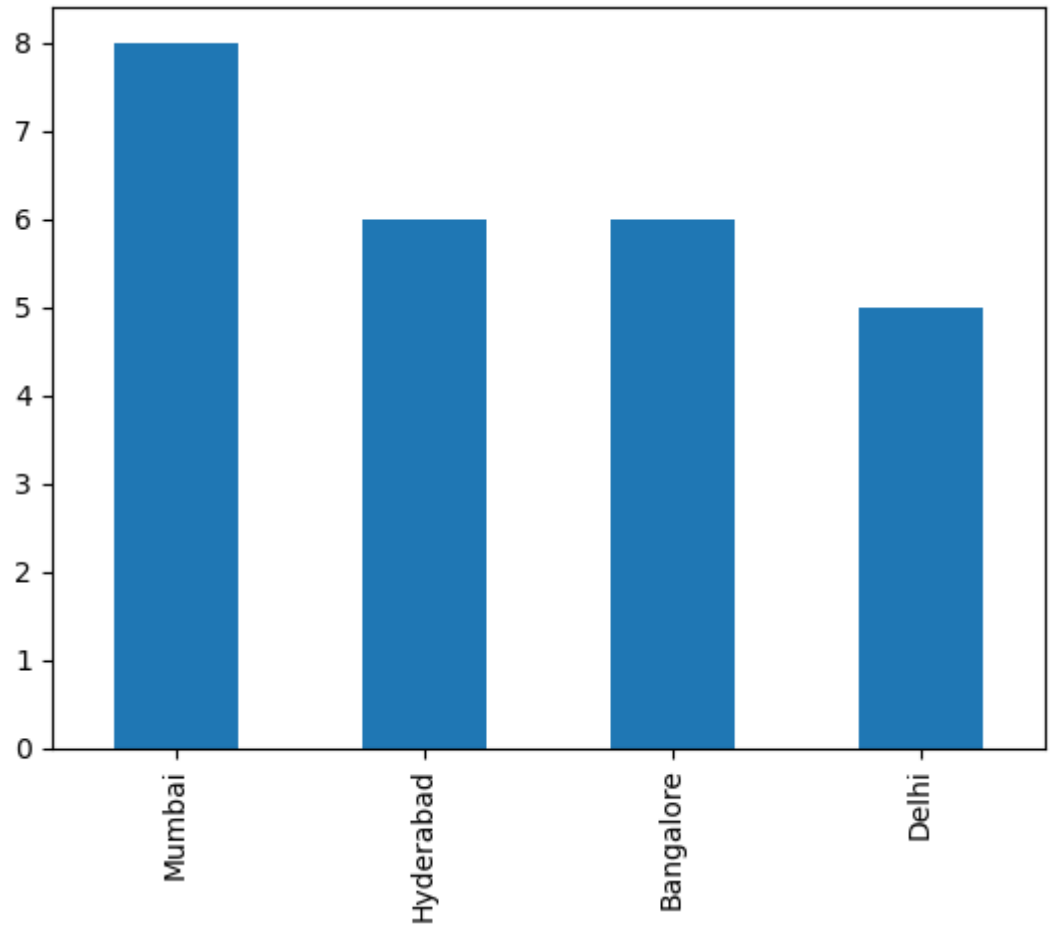
```
Luxury      16
Business     9
Name: category, dtype: int64
```

In [14]:

```
df_hotels.city.value_counts().sort_values(ascending=False).plot(kind = "bar")
```

Out[14]:

```
<Axes: >
```



Exploring aggregate bookings.

In [15]:

```
df_agg_bookings.head()
```

Out[15]:

	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 [16]:

```
df_agg_bookings.property_id.unique()
```

Out[16]:

```
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])
```

In [17]:

```
#Finding the property booking per property ID.
df_agg_bookings.groupby("property_id")["successful_bookings"].sum()
```

Out[17]:

```
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
```

In [18]:

```
# Finding out days on which bookings are greater than capacity.
```

In [19]:

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

Out[19]:

	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 [20]:

```
df_agg_bookings.capacity.max()
```

Out[20]:

```
50.0
```

In [21]:

```
# Finding out the property with higest capacity.
df_agg_bookings[df_agg_bookings.capacity == 50.0]
```

Out[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 × 5 columns

Data Cleaning

In [22]:

```
df_booking.describe()
```

Out[22]:

	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 [23]:

```
# Cleaning invalid guests**
df_booking[df_booking.no_guests<=0]
```

Out[23]:

	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
3	May012216558RT14	16558	28-04-22	1/5/2022	2/5/2022	-2.0
17924	May122218559RT44	18559	12/5/2022	12/5/2022	14-05-22	-10.0
18020	May122218561RT22	18561	8/5/2022	12/5/2022	14-05-22	-12.0
18119	May122218562RT311	18562	5/5/2022	12/5/2022	17-05-22	-6.0
18121	May122218562RT313	18562	10/5/2022	12/5/2022	17-05-22	-4.0
56715	Jun082218562RT12	18562	5/6/2022	8/6/2022	13-06-22	-17.0
119765	Jul202219560RT220	19560	19-07-22	20-07-22	22-07-22	-1.0
134586	Jul312217564RT47	17564	30-07-22	31-07-22	1/8/2022	-4.0

In [24]:

```
# As you can see above, number of guests having less than zero value represents data
df_booking = df_booking[df_booking.no_guests > 0]
```

In [25]:

```
df_booking.shape
```

Out[25]:

```
(134578, 12)
```

In [26]:

```
# Removing Outlier in revenue generated
df_booking.revenue_generated.min(), df_booking.revenue_generated.max()
```

Out[26]:

```
(6500, 28560000)
```

In [27]:

```
avg, std = df_booking.revenue_generated.mean(), df_booking.revenue_generated.std()
avg, std
```

Out[27]:

```
(15378.036937686695, 93040.15493143328)
```

In [28]:

```
higher_limit = avg + 3*std
higher_limit
```

Out[28]:

```
294498.50173198653
```

In [29]:

```
lower_limit = avg - 3*std
lower_limit
```

Out[29]:

```
-263742.4278566132
```

In [30]:

```
df_booking[df_booking.revenue_generated <= 0]
```

Out[30]:

booking_id	property_id	booking_date	check_in_date	checkout_date	no_guests	room_categor

In [31]:

```
df_booking[df_booking.revenue_generated > higher_limit]
```

Out[31]:

	booking_id	property_id	booking_date	check_in_date	checkout_date	no_guests
2	May012216558RT13	16558	28-04-22	1/5/2022	4/5/2022	2.0
111	May012216559RT32	16559	29-04-22	1/5/2022	2/5/2022	6.0
315	May012216562RT22	16562	28-04-22	1/5/2022	4/5/2022	2.0
562	May012217559RT118	17559	26-04-22	1/5/2022	2/5/2022	2.0
129176	Jul282216562RT26	16562	21-07-22	28-07-22	29-07-22	2.0

In [32]:

```
df_booking = df_booking[df_booking.revenue_generated<=higher_limit]
df_booking.shape
```

Out[32]:

```
(134573, 12)
```

In [33]:

```
df_booking.revenue_realized.describe()
```

Out[33]:

```
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
```

In [34]:

```
higher_limit = df_booking.revenue_realized.mean() + 3*df_booking.revenue_realized.st
higher_limit
```

Out[34]:

```
33479.3586618449
```

In [35]:

```
df_booking[df_booking.revenue_realized > higher_limit]
```

Out[35]:

	booking_id	property_id	booking_date	check_in_date	checkout_date	no_guests
137	May012216559RT41	16559	27-04-22	1/5/2022	7/5/2022	4.0
139	May012216559RT43	16559	1/5/2022	1/5/2022	2/5/2022	6.0
143	May012216559RT47	16559	28-04-22	1/5/2022	3/5/2022	3.0
149	May012216559RT413	16559	24-04-22	1/5/2022	7/5/2022	5.0
222	May012216560RT45	16560	30-04-22	1/5/2022	3/5/2022	5.0
...	...	...	...	...	...	..
134328	Jul312219560RT49	19560	31-07-22	31-07-22	2/8/2022	6.0
134331	Jul312219560RT412	19560	31-07-22	31-07-22	1/8/2022	6.0
134467	Jul312219562RT45	19562	28-07-22	31-07-22	1/8/2022	6.0
134474	Jul312219562RT412	19562	25-07-22	31-07-22	6/8/2022	5.0
134581	Jul312217564RT42	17564	31-07-22	31-07-22	1/8/2022	4.0

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 [36]:

```
df_rooms
```

Out[36]:

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

In [37]:

```
df_booking[df_booking.room_category == "RT4"].revenue_realized.describe()
```

Out[37]:

```
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 [38]:

```
23439 + 3*9048
```

Out[38]:

```
50583
```

In [39]:

```
df_booking.isnull().sum()
```

Out[39]:

```
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 is 134576. 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

In [40]:

```
df_agg_bookings.isnull().sum()
```

Out[40]:

```
property_id            0
check_in_date          0
room_category          0
successful_bookings    0
capacity               2
dtype: int64
```

In [41]:

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

Out[41]:

	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 [42]:

```
df_agg_bookings.capacity.median()
```

Out[42]:

```
25.0
```

In [43]:

```
df_agg_bookings.capacity.fillna(df_agg_bookings.capacity.median(),inplace=True)
```

In [44]:

```
df_agg_bookings.loc[[8,14]]
```

Out[44]:

	property_id	check_in_date	room_category	successful_bookings	capacity
8	17561	1-May-22	RT1	22	25.0
14	17562	1-May-22	RT1	12	25.0

In [45]:

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

Out[45]:

	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 [46]:

```
df_agg_bookings.shape
```

Out[46]:

```
(9200, 5)
```

In [47]:

```
df_agg_bookings = df_agg_bookings[df_agg_bookings.successful_bookings <= df_agg_book
df_agg_bookings.shape
```

Out[47]:

```
(9194, 5)
```

Data Transformation

In [48]:

```
# Creating occupancy percentage column: successful_bookings / capacity

df_agg_bookings["occ_pct"] = df_agg_bookings["successful_bookings"] / df_agg_booking
```

In [49]:

```
df_agg_bookings.head(3)
```

Out[49]:

	property_id	check_in_date	room_category	successful_bookings	capacity	occ_pct
0	16559	1-May-22	RT1	25	30.0	0.833333
1	19562	1-May-22	RT1	28	30.0	0.933333
2	19563	1-May-22	RT1	23	30.0	0.766667

In [50]:

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

Out[50]:

	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

Generating Insights

Q1. What is an average occupancy rate in each of the room categories?

In [51]:

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

Out[51]:

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

In [52]:

```
df_rooms
```

Out[52]:

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

In [53]:

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

Out[53]:

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

In [54]:

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

Out[54]:

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

In [55]:

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

Out[55]:

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

Q2. Printing average occupancy rate per city

In [56]:

```
df_hotels.head(2)
```

Out[56]:

	property_id	property_name	category	city
0	16558	Atliq Grands	Luxury	Delhi
1	16559	Atliq Exotica	Luxury	Mumbai

In [57]:

```
df = pd.merge(df, df_hotels, on="property_id")
df.head(3)
```

Out[57]:

	property_id	check_in_date	room_category	successful_bookings	capacity	occ_pct	room_clas
0	16559	1-May-22	RT1	25	30.0	83.33	Standar
1	16559	2-May-22	RT1	20	30.0	66.67	Standar
2	16559	3-May-22	RT1	17	30.0	56.67	Standar

In [60]:

```
df.groupby("city")["occ_pct"].mean()
```

Out[60]:

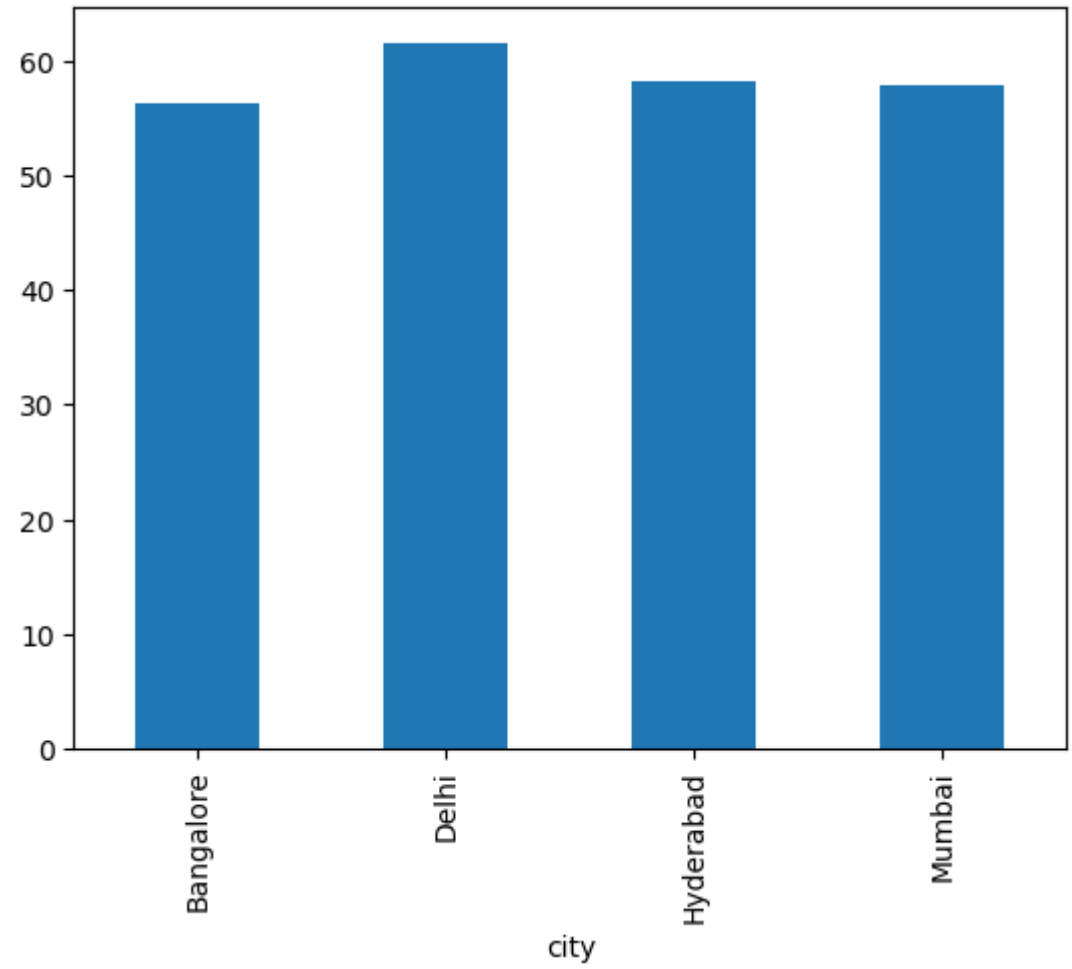
```
city
Bangalore    56.332376
Delhi        61.507341
Hyderabad    58.120652
Mumbai       57.909181
Name: occ_pct, dtype: float64
```

In [62]:

```
df.groupby("city")["occ_pct"].mean().plot(kind = "bar")
```

Out[62]:

```
<Axes: xlabel='city'>
```



Q3. When was the occupancy better? Weekday or Weekend?

In [63]:

```
df.head(2)
```

Out[63]:

	property_id	check_in_date	room_category	successful_bookings	capacity	occ_pct	room_clas
0	16559	1-May-22	RT1	25	30.0	83.33	Standar
1	16559	2-May-22	RT1	20	30.0	66.67	Standar

In [64]:

```
df_date.head(2)
```

Out[64]:

	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

In [65]:

```
df = pd.merge(df,df_date,left_on="check_in_date",right_on="date")
df.head(2)
```

Out[65]:

	property_id	check_in_date	room_category	successful_bookings	capacity	occ_pct	room_clas
0	16559	10-May-22	RT1	18	30.0	60.00	Standar
1	16559	10-May-22	RT2	25	41.0	60.98	Elit

In [66]:

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

Out[66]:

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

Q4. In the month of June, what is the occupancy for different cities

In [67]:

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

Out[67]:

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

In [72]:

```
df_jun_22 = df[df["mmm yy"] == "Jun 22"]
df_jun_22.head(3)
```

Out[72]:

	property_id	check_in_date	room_category	successful_bookings	capacity	occ_pct	room_
2200	16559	10-Jun-22	RT1	20	30.0	66.67	Star
2201	16559	10-Jun-22	RT2	26	41.0	63.41	
2202	16559	10-Jun-22	RT3	20	32.0	62.50	Pre

In [81]:

```
df_jun_22.groupby("city")["occ_pct"].mean().round(2).sort_values(ascending = False)
```

Out[81]:

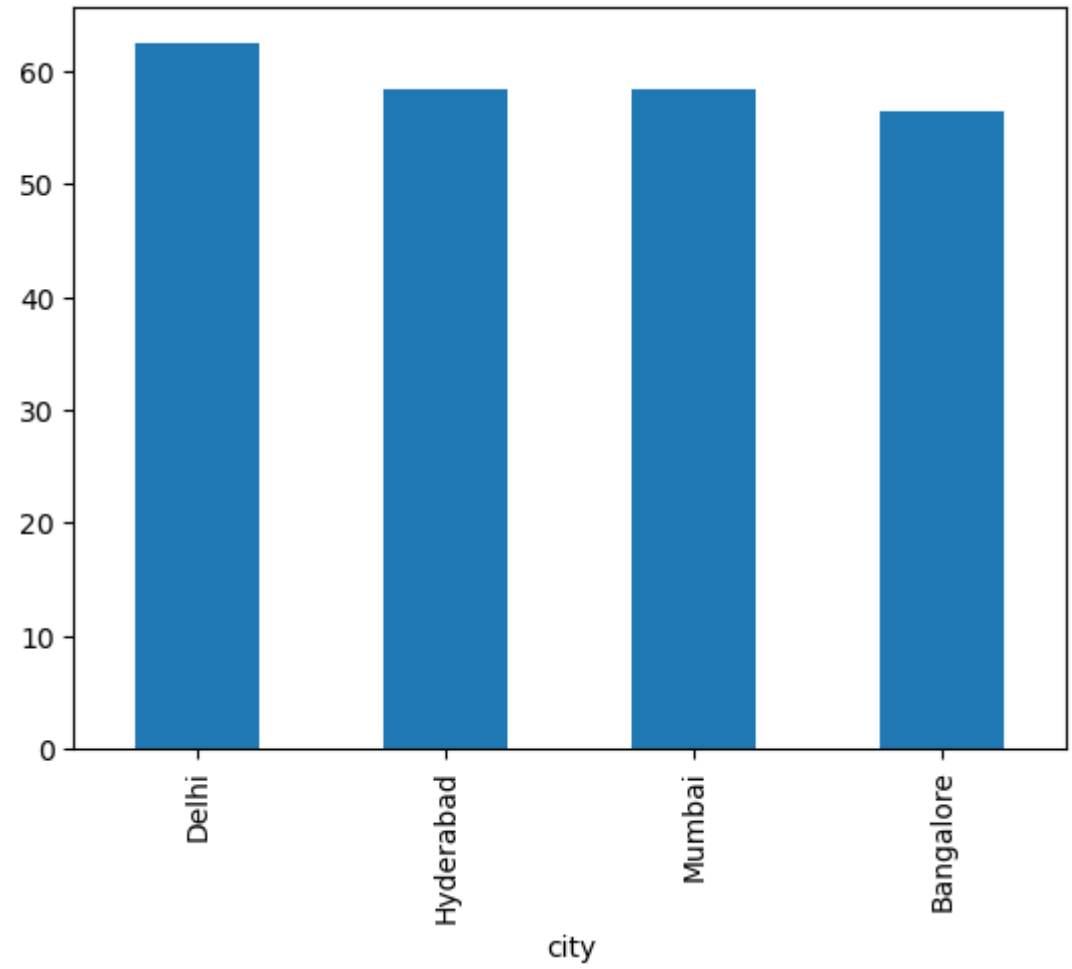
```
city
Delhi        62.47
Hyderabad    58.46
Mumbai       58.38
Bangalore    56.44
Name: occ_pct, dtype: float64
```

In [83]:

```
df_jun_22.groupby('city')['occ_pct'].mean().round(2).sort_values(ascending=False).pl
```

Out[83]:

```
<Axes: xlabel='city'>
```



Q5. We got new data for the month of august. Append that to existing data

In [86]:

```
df_august = pd.read_csv("new_data_august.csv")
df_august.head(2)
```

Out[86]:

	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	

In [87]:

```
df.columns
```

Out[87]:

```
Index(['property_id', 'check_in_date', 'room_category', 'successful_bo
okings',
       'capacity', 'occ_pct', 'room_class', 'property_name', 'categor
y',
       'city', 'date', 'mmm yy', 'week no', 'day_type'],
      dtype='object')
```

In [92]:

```
df_august.columns
```

Out[92]:

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

In [93]:

```
df.shape
```

Out[93]:

```
(6497, 14)
```

In [91]:

```
df_august.shape
```

Out[91]:

```
(7, 13)
```

In [97]:

```
latest_df = pd.concat([df, df_august], ignore_index = True, axis = 0)
latest_df.tail(10)
```

Out[97]:

	property_id	check_in_date	room_category	successful_bookings	capacity	occ_pct	room_
6494	16563	31-Jul-22	RT2	32	38.0	84.21	
6495	16563	31-Jul-22	RT3	14	20.0	70.00	Pre
6496	16563	31-Jul-22	RT4	13	18.0	72.22	Presid
6497	16559	01-Aug-22	RT1	30	30.0	NaN	Star
6498	19562	01-Aug-22	RT1	21	30.0	NaN	Star
6499	19563	01-Aug-22	RT1	23	30.0	NaN	Star
6500	19558	01-Aug-22	RT1	30	40.0	NaN	Star
6501	19560	01-Aug-22	RT1	20	26.0	NaN	Star
6502	17561	01-Aug-22	RT1	18	26.0	NaN	Star
6503	17564	01-Aug-22	RT1	10	16.0	NaN	Star

In [98]:

```
latest_df.shape
```

Out[98]:

```
(6504, 15)
```

Q6. Print revenue realized per city

In [100]:

```
df_booking.head(2)
```

Out[100]:

	booking_id	property_id	booking_date	check_in_date	checkout_date	no_guests	roon
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	

In [101]:

```
df_hotels.head(2)
```

Out[101]:

	property_id	property_name	category	city
0	16558	Atliq Grands	Luxury	Delhi
1	16559	Atliq Exotica	Luxury	Mumbai

In [103]:

```
df_booking_all = pd.merge(df_booking,df_hotels,on="property_id")
df_booking_all.head()
```

Out[103]:

	booking_id	property_id	booking_date	check_in_date	checkout_date	no_guests	roon
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	
3	May012216558RT17	16558	28-04-22	1/5/2022	6/5/2022	2.0	
4	May012216558RT18	16558	26-04-22	1/5/2022	3/5/2022	2.0	

In [105]:

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

Out[105]:

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

Q7. Print month by month revenue

In [106]:

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

Out[106]:

	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

In [107]:

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

Out[107]:

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

In [109]:

```
df_booking_all.head(3)
```

Out[109]:

	booking_id	property_id	booking_date	check_in_date	checkout_date	no_guests	roon
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 [110]:

```
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
```

In [111]:

```
df_date["date"] = pd.to_datetime(df_date["date"])
df_date.head(3)
```

Out[111]:

	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
2	2022-05-03	May 22	W 19	weekeday

In [113]:

```
df_booking_all.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 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: 16.4+ MB
```

In [115]:

```
df_booking_all["check_in_date"] = pd.to_datetime(df_booking_all["check_in_date"])
df_booking_all.head(4)
```

Out[115]:

	booking_id	property_id	booking_date	check_in_date	checkout_date	no_guests	roon
0	May012216558RT12	16558	30-04-22	2022-01-05	2/5/2022	2.0	
1	May012216558RT15	16558	27-04-22	2022-01-05	2/5/2022	4.0	
2	May012216558RT16	16558	1/5/2022	2022-01-05	3/5/2022	2.0	
3	May012216558RT17	16558	28-04-22	2022-01-05	6/5/2022	2.0	

In [117]:

```
df_booking_all = pd.merge(df_booking_all, df_date, left_on="check_in_date", right_or
df_booking_all.head(3)
```

Out[117]:

	booking_id	property_id	booking_date	check_in_date	checkout_date	no_guests	roon
0	May052216558RT11	16558	15-04-22	2022-05-05	7/5/2022	3.0	
1	May052216558RT12	16558	30-04-22	2022-05-05	7/5/2022	2.0	
2	May052216558RT13	16558	1/5/2022	2022-05-05	6/5/2022	3.0	

In [118]:

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

Out[118]:

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

Q8. Print revenue realized per hotel type

In [119]:

```
df_booking_all.property_name.unique()
```

Out[119]:

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

In [120]:

```
df_booking_all.groupby("property_name")["revenue_realized"].sum().round(2).sort_valu
```

Out[120]:

```
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
```

Q9. Print average rating per city

In [121]:

```
df_booking_all.groupby("city")["ratings_given"].mean().round(2)
```

Out[121]:

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

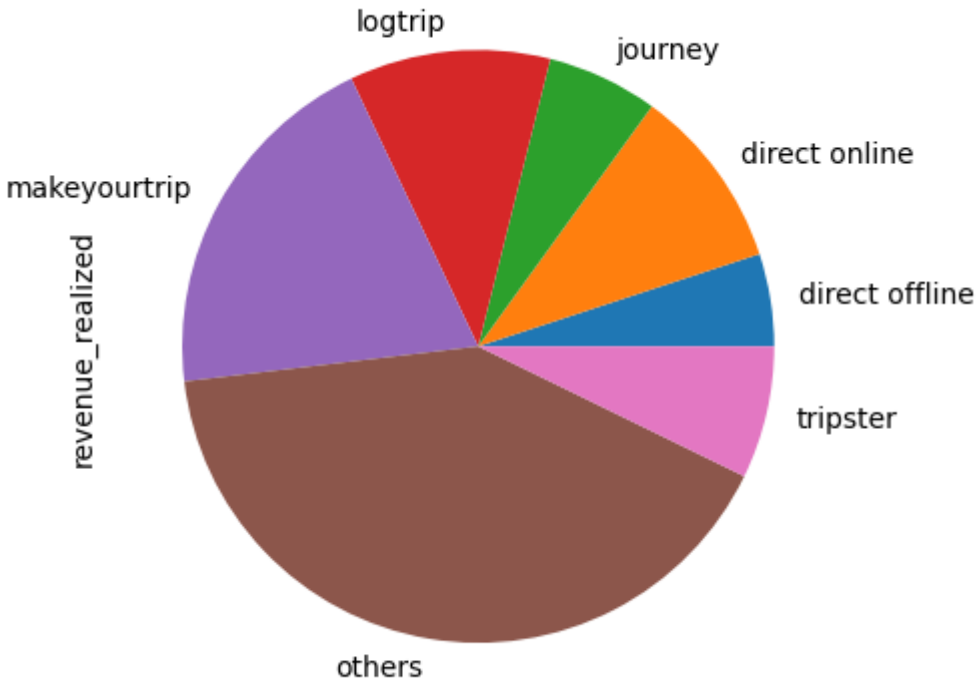
Q10. Print a pie chart of revenue realized per booking platform

In [122]:

```
df_booking_all.groupby("booking_platform")["revenue_realized"].sum().plot(kind="pie"
```

Out[122]:

```
<Axes: ylabel='revenue_realized'>
```



In [ ]: