MINI PROJECT

Topic - Data analysis and Visualization of AIRBNB at New York city

Name:Pranav Kalambe **Roll No. 1913023** **Branch:SY-EXTC**

Name:Ishika Bhatt **Roll No. 1914073** **Branch:SY-IT**

Link For the dataset:
https://www.kaggle.com/dgomonov/new-york-city-airbnb-open-data (https://www.kaggle.com/dgomonov/new-york-city-airbnb-open-data)

Airbnb: Inc. is an American vacation rental online marketplace company based in San Francisco, California,United States. Airbnb offers arrangement for lodging, primarily homestays, or tourism experiences.

Since 2008, guests and hosts have used Airbnb to expand on traveling possibilities and present more unique, personalized way of experiencing the world.This dataset describes the listing activity and metrics in NYC, NY for 2019.

This data file includes all needed information to find out more about hosts, geographical availability, necessary metrics to make predictions and draw conclusions.

 image here

Getting Started

Importing all required libraries

In [1]:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
```

Loading and Observing dataset

In [2]:

```
original_df = pd.read_csv("AB_NYC_2019.csv")
original_df.head()
```

Out[2]:

	id	name	host_id	host_name	neighbourhood_group	neighbourhood	latitu
0	2539	Clean & quiet apt home by the park	2787	John	Brooklyn	Kensington	40.647
1	2595	Skylit Midtown Castle	2845	Jennifer	Manhattan	Midtown	40.753
2	3647	THE VILLAGE OF HARLEM....NEW YORK !	4632	Elisabeth	Manhattan	Harlem	40.809
3	3831	Cozy Entire Floor of Brownstone	4869	LisaRoxanne	Brooklyn	Clinton Hill	40.685
4	5022	Entire Apt: Spacious Studio/Loft by central park	7192	Laura	Manhattan	East Harlem	40.798

In [3]:

```
original_df.shape
```

Out[3]:

```
(48895, 16)
```

There are total of 48895 rows and 16 colums

In [4]:

```
original_df.dtypes
```

Out[4]:

```
id                                 int64
name                              object
host_id                            int64
host_name                         object
neighbourhood_group               object
neighbourhood                     object
latitude                         float64
longitude                        float64
room_type                         object
price                              int64
minimum_nights                     int64
number_of_reviews                  int64
last_review                       object
reviews_per_month                float64
calculated_host_listings_count     int64
availability_365                   int64
dtype: object
```

In [5]:

```
original_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 48895 entries, 0 to 48894
Data columns (total 16 columns):
 #   Column                          Non-Null Count  Dtype  
---  ------                          --------------  -----  
 0   id                              48895 non-null  int64  
 1   name                            48879 non-null  object 
 2   host_id                         48895 non-null  int64  
 3   host_name                       48874 non-null  object 
 4   neighbourhood_group             48895 non-null  object 
 5   neighbourhood                   48895 non-null  object 
 6   latitude                        48895 non-null  float64
 7   longitude                       48895 non-null  float64
 8   room_type                       48895 non-null  object 
 9   price                           48895 non-null  int64  
 10  minimum_nights                  48895 non-null  int64  
 11  number_of_reviews               48895 non-null  int64  
 12  last_review                     38843 non-null  object 
 13  reviews_per_month               38843 non-null  float64
 14  calculated_host_listings_count  48895 non-null  int64  
 15  availability_365                48895 non-null  int64  
dtypes: float64(3), int64(7), object(6)
memory usage: 6.0+ MB
```

Finding No. of Missing data in dataset

In [6]:

```
original_df.isnull().sum()
```

Out[6]:

```
id                                    0
name                                 16
host_id                               0
host_name                            21
neighbourhood_group                   0
neighbourhood                         0
latitude                              0
longitude                             0
room_type                             0
price                                 0
minimum_nights                        0
number_of_reviews                     0
last_review                       10052
reviews_per_month                 10052
calculated_host_listings_count        0
availability_365                      0
dtype: int64
```

Cleaning of Dataset

Droping last_review column - It consist of dtype Object (to be specify Date) and alot of missing data
Droping id - Since it is not much significant
Droping host_name because of ethical reasons

In [7]:

```
original_df.drop(['id','host_name','last_review'],axis = 1,inplace = True)
original_df.tail()
```

Out[7]:

	name	host_id	neighbourhood_group	neighbourhood	latitude	longitude
48890	Charming one bedroom - newly renovated rowhouse	8232441	Brooklyn	Bedford-Stuyvesant	40.67853	-73.94995
48891	Affordable room in Bushwick/East Williamsburg	6570630	Brooklyn	Bushwick	40.70184	-73.93317
48892	Sunny Studio at Historical Neighborhood	23492952	Manhattan	Harlem	40.81475	-73.94867
48893	43rd St. Time Square-cozy single bed	30985759	Manhattan	Hell's Kitchen	40.75751	-73.99112
48894	Trendy duplex in the very heart of Hell's Kitchen	68119814	Manhattan	Hell's Kitchen	40.76404	-73.98933

In [8]:

```
original_df.isnull().sum()
```

Out[8]:

```
name                                16
host_id                              0
neighbourhood_group                  0
neighbourhood                        0
latitude                             0
longitude                            0
room_type                            0
price                                0
minimum_nights                       0
number_of_reviews                    0
reviews_per_month                10052
calculated_host_listings_count       0
availability_365                     0
dtype: int64
```

In [9]:

```
original_df.dtypes['reviews_per_month']
```

Out[9]:

```
dtype('float64')
```

In [10]:

```
original_df.fillna({'reviews_per_month':0},inplace = True)
#examing changes
original_df.reviews_per_month.isnull().sum()
```

Out[10]:

```
0
```

Dropping the rows which have Name column as 'NA'

In [11]:

```
original_df.dropna(how='any',inplace=True)
```

In [12]:

```
original_df.isnull().sum()
```

Out[12]:

```
name                            0
host_id                         0
neighbourhood_group             0
neighbourhood                   0
latitude                        0
longitude                       0
room_type                       0
price                           0
minimum_nights                  0
number_of_reviews               0
reviews_per_month               0
calculated_host_listings_count  0
availability_365                0
dtype: int64
```

Now the dataset has been Cleaned

In [13]:

```
len(original_df)
```

Out[13]:

```
48879
```

Note: At the start No. of rows were 48894, now we have 48879 rows, means 16 rows has been removed which has name column as 'NA'

Since Original dataset is now manupulated and cleaned, let's assign it to a new variable and Save it !

In [14]:

```
air_df = original_df.copy()
```

In [15]:

```
air_df.to_csv("Updated_Airbnb.csv")
```

Examine Continous Variables

In [16]:

```
air_df.describe()
```

Out[16]:

	host_id	latitude	longitude	price	minimum_nights	number_of_
count	4.887900e+04	48879.000000	48879.000000	48879.000000	48879.000000	4887
mean	6.763013e+07	40.728945	-73.952168	152.722355	7.011027	2
std	7.862070e+07	0.054529	0.046160	240.186804	20.016000	4
min	2.438000e+03	40.499790	-74.244420	0.000000	1.000000	
25%	7.816856e+06	40.690090	-73.983070	69.000000	1.000000	
50%	3.079133e+07	40.723080	-73.955680	106.000000	3.000000	
75%	1.074344e+08	40.763110	-73.936280	175.000000	5.000000	2
max	2.743213e+08	40.913060	-73.712990	10000.000000	1250.000000	62

Let's have a closer look at individual features and relation between them

In [17]:

```
air_df.columns
```

Out[17]:

```
Index(['name', 'host_id', 'neighbourhood_group', 'neighbourhood', 'latitud
e',
       'longitude', 'room_type', 'price', 'minimum_nights',
       'number_of_reviews', 'reviews_per_month',
       'calculated_host_listings_count', 'availability_365'],
      dtype='object')
```

In [18]:

```
len(air_df['host_id'].unique()) # Since there alot of unique values not displaying them
```

Out[18]:

```
37443
```

In [19]:

```
top_host=air_df.host_id.value_counts().head(10)
top_host
```

Out[19]:

```
219517861    327
107434423    232
30283594     121
137358866    103
16098958      96
12243051      96
61391963      91
22541573      87
200380610     65
7503643       52
Name: host_id, dtype: int64
```

In [20]:

```
#coming back to our dataset we can confirm our findings with already existing column ca
lled 'calculated_host_listings_count'
top_host_check=air_df.calculated_host_listings_count.max()
top_host_check
```

Out[20]:

```
327
```

In [21]:

```
air_df['neighbourhood_group'].unique()
```

Out[21]:

```
array(['Brooklyn', 'Manhattan', 'Queens', 'Staten Island', 'Bronx'],
      dtype=object)
```

No. of Rooms used at different Neighbourhood_groups

In [22]:

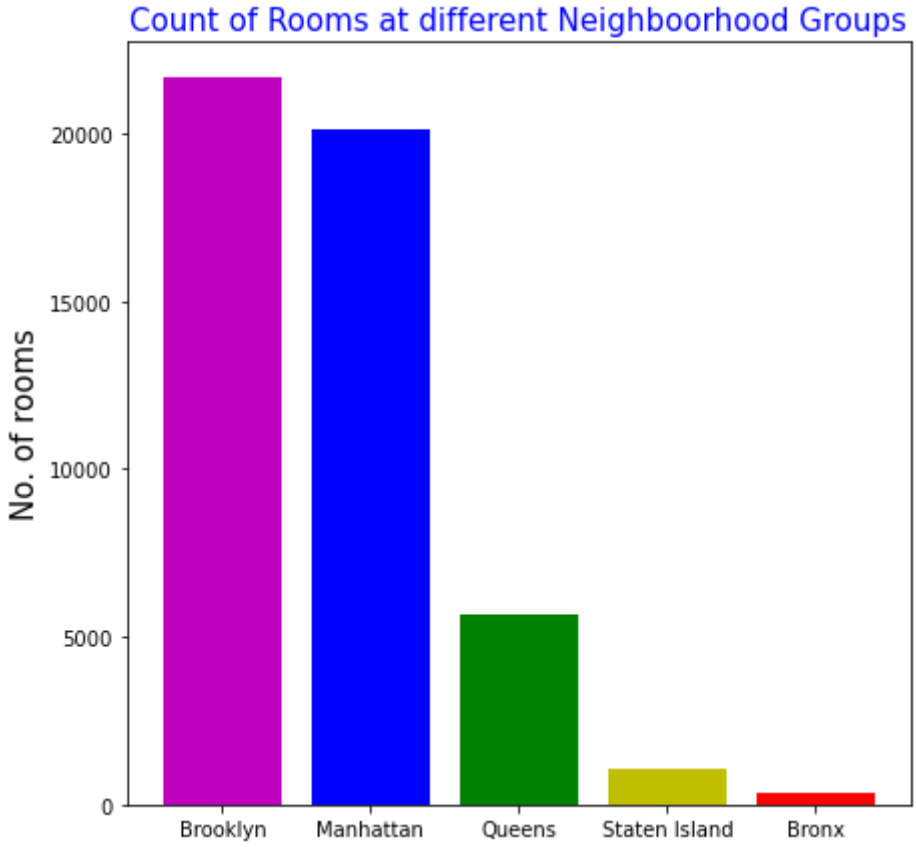
```
air_df.neighbourhood_group.value_counts()
```

Out[22]:

```
Manhattan        21652
Brooklyn         20098
Queens            5666
Bronx             1090
Staten Island      373
Name: neighbourhood_group, dtype: int64
```

In [23]:

```
plt.figure(figsize=(7,7))
plt.bar(air_df['neighbourhood_group'].unique(),air_df.neighbourhood_group.value_counts
(),color = ('m','b','g','y','r') )
plt.title('Count of Rooms at different Neighboorhood Groups',color = 'blue',fontsize =
15)
plt.ylabel('No. of rooms',fontsize=15)
plt.show()
```



In [24]:

```
len(air_df['neighbourhood'].unique()) # Since there alot of unique values not displayin
g them
```

Out[24]:

221

Room Type

In [25]:

```
air_df['room_type'].unique()
```

Out[25]:

```
array(['Private room', 'Entire home/apt', 'Shared room'], dtype=object)
```

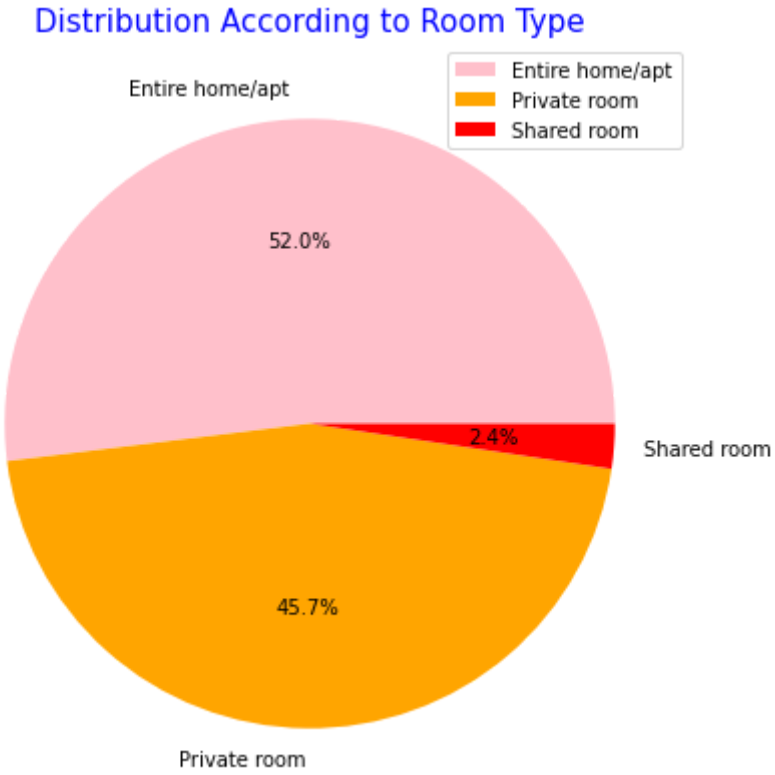
In [26]:

```
print(air_df.room_type.value_counts())
```

```
Entire home/apt    25402
Private room       22318
Shared room         1159
Name: room_type, dtype: int64
```

In [27]:

```
labels = air_df.room_type.value_counts().index
colors = ['pink','orange','red']
explode = [0,0,0]
sizes =air_df.room_type.value_counts().values

plt.figure(0,figsize = (7,7))
plt.pie(sizes, explode=explode, labels=labels, colors=colors, autopct='%1.1f%%')
plt.title('Distribution According to Room Type',color = 'blue',fontsize = 15)
plt.legend()
plt.show()
```



Prices

In [28]:

```
len(air_df['price'].unique())
```

Out[28]:

674

In [29]:

```
air_df['price'].describe()
```

Out[29]:

```
count    48879.000000
mean       152.722355
std        240.186804
min          0.000000
25%         69.000000
50%        106.000000
75%        175.000000
max      10000.000000
Name: price, dtype: float64
```

Sorting Maximum Price Hotel Rooms

In [30]:

```
air_df = air_df.sort_values(by=["price"], ascending=False)
air_df.head()
```

Out[30]:

	name	host_id	neighbourhood_group	neighbourhood	latitude	longitude	roor
9151	Furnished room in Astoria apartment	20582832	Queens	Astoria	40.76810	-73.91651	
29238	1-BR Lincoln Center	72390391	Manhattan	Upper West Side	40.77213	-73.98665	hc
17692	Luxury 1 bedroom apt. - stunning Manhattan views	5143901	Brooklyn	Greenpoint	40.73260	-73.95739	hc
12342	Quiet, Clean, Lit @ LES & Chinatown	3906464	Manhattan	Lower East Side	40.71355	-73.98507	
6530	Spanish Harlem Apt	1235070	Manhattan	East Harlem	40.79264	-73.93898	hc

Minimum Price Hotel Rooms

In [31]:

```
min_price = air_df['price'].min()
air_df.loc[(air_df.price == min_price)]
```

Out[31]:

	name	host_id	neighbourhood_group	neighbourhood	latitude	longitude
25796	Cozy yet spacious private brownstone bedroom	86327101	Brooklyn	Bedford-Stuyvesant	40.68258	-73.91284
25794	Spacious comfortable master bedroom with nice ...	86327101	Brooklyn	Bedford-Stuyvesant	40.68173	-73.91342
25795	Contemporary bedroom in brownstone with nice view	86327101	Brooklyn	Bedford-Stuyvesant	40.68279	-73.91170
26259	the best you can find	13709292	Manhattan	Murray Hill	40.75091	-73.97597
25634	MARTIAL LOFT 3: REDEMPTION (upstairs, 2nd room)	15787004	Brooklyn	Bushwick	40.69467	-73.92433
26866	Best Coliving space ever! Shared room.	101970559	Brooklyn	Bushwick	40.69166	-73.90928
23161	Huge Brooklyn Brownstone Living, Close to it all.	8993084	Brooklyn	Bedford-Stuyvesant	40.69023	-73.95428
25433	★Hostel Style Room | Ideal Traveling Buddies★	131697576	Bronx	East Morrisania	40.83296	-73.88668
25778	Modern apartment in the heart of Williamsburg	10132166	Brooklyn	Williamsburg	40.70838	-73.94645
25753	Sunny, Quiet Room in Greenpoint	1641537	Brooklyn	Greenpoint	40.72462	-73.94072
26841	Coliving in Brooklyn! Modern design / Shared room	101970559	Brooklyn	Bushwick	40.69211	-73.90670

Highest Price of different neighbourhood_groups

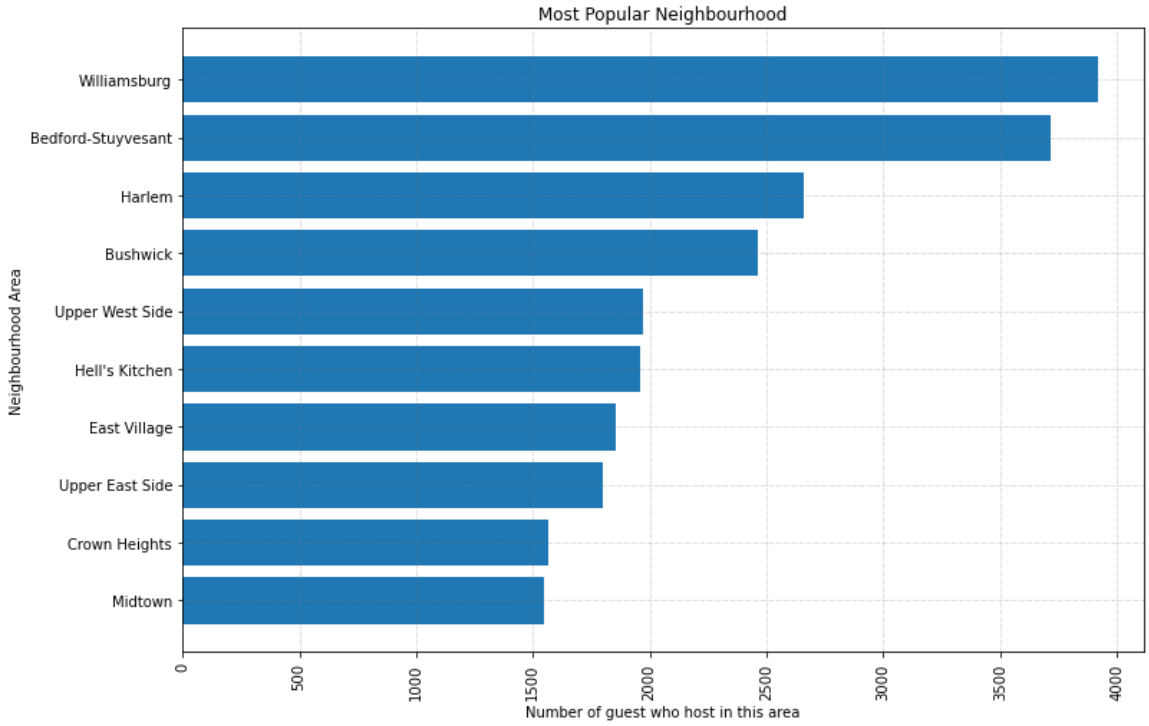
In [32]:

```
groups = air_df.neighbourhood_group.unique()
for i in groups:
        temp = air_df.loc[(air_df.neighbourhood_group == i)]
        group_max = temp['price'].max()
        print('{:<25} {}'.format(i,group_max))
```

```
Queens                    10000
Manhattan                 10000
Brooklyn                  10000
Staten Island             5000
Bronx                     2500
```

In [33]:

```
data = air_df.neighbourhood.value_counts()[:10]
plt.figure(figsize=(12, 8))
x = list(data.index)
y = list(data.values)
x.reverse()
y.reverse()

plt.title("Most Popular Neighbourhood")
plt.ylabel("Neighbourhood Area")
plt.xlabel("Number of guest who host in this area")
plt.barh(x, y)
plt.xticks(rotation=90)

plt.grid(b = True, color ='grey',
        linestyle ='-.', linewidth = 0.5,
        alpha = 0.5)
plt.show()
```



Minimum Number of Nights spend

In [34]:

```
#Maximum No. of nights spend
nights_max = air_df['minimum_nights'].unique().max()
print("Maximum No. of Nights spend is",nights_max)
```

```
Maximum No. of Nights spend is 1250
```

In [35]:

```
total = air_df['minimum_nights'].sum()
n = air_df['minimum_nights'].count()
average = total/n
print("Average No. of Nights spend is",int(average))
```

```
Average No. of Nights spend is 7
```

In [36]:

```
air_df['minimum_nights'].describe()
```

Out[36]:

```
count    48879.000000
mean         7.011027
std         20.016000
min          1.000000
25%          1.000000
50%          3.000000
75%          5.000000
max       1250.000000
Name: minimum_nights, dtype: float64
```

In [37]:

```
#last column we need to look at is 'number_of_reviews'

#let's grab 10 most reviewed listings in NYC
top_reviewed_listings=air_df.nlargest(10,'number_of_reviews')
top_reviewed_listings
```

Out[37]:

	name	host_id	neighbourhood_group	neighbourhood	latitude	lon
11759	Room near JFK Queen Bed	47621202	Queens	Jamaica	40.66730	-73
2031	Great Bedroom in Manhattan	4734398	Manhattan	Harlem	40.82085	-73
2030	Beautiful Bedroom in Manhattan	4734398	Manhattan	Harlem	40.82124	-73
2015	Private Bedroom in Manhattan	4734398	Manhattan	Harlem	40.82264	-73
13495	Room Near JFK Twin Beds	47621202	Queens	Jamaica	40.66939	-73
10623	Steps away from Laguardia airport	37312959	Queens	East Elmhurst	40.77006	-73
1879	Manhattan Lux Loft.Like.Love.Lots.Look !	2369681	Manhattan	Lower East Side	40.71921	-73
20403	Cozy Room Family Home LGA Airport NO CLEANING FEE	26432133	Queens	East Elmhurst	40.76335	-73
4870	Private brownstone studio Brooklyn	12949460	Brooklyn	Park Slope	40.67926	-73
471	LG Private Room/Family Friendly	792159	Brooklyn	Bushwick	40.70283	-73

In [38]:

```
price_avrg=top_reviewed_listings.price.mean()
print('Average price per night at top 10 rooms is {}'.format(price_avrg))
print('9/10 are private rooms')
```

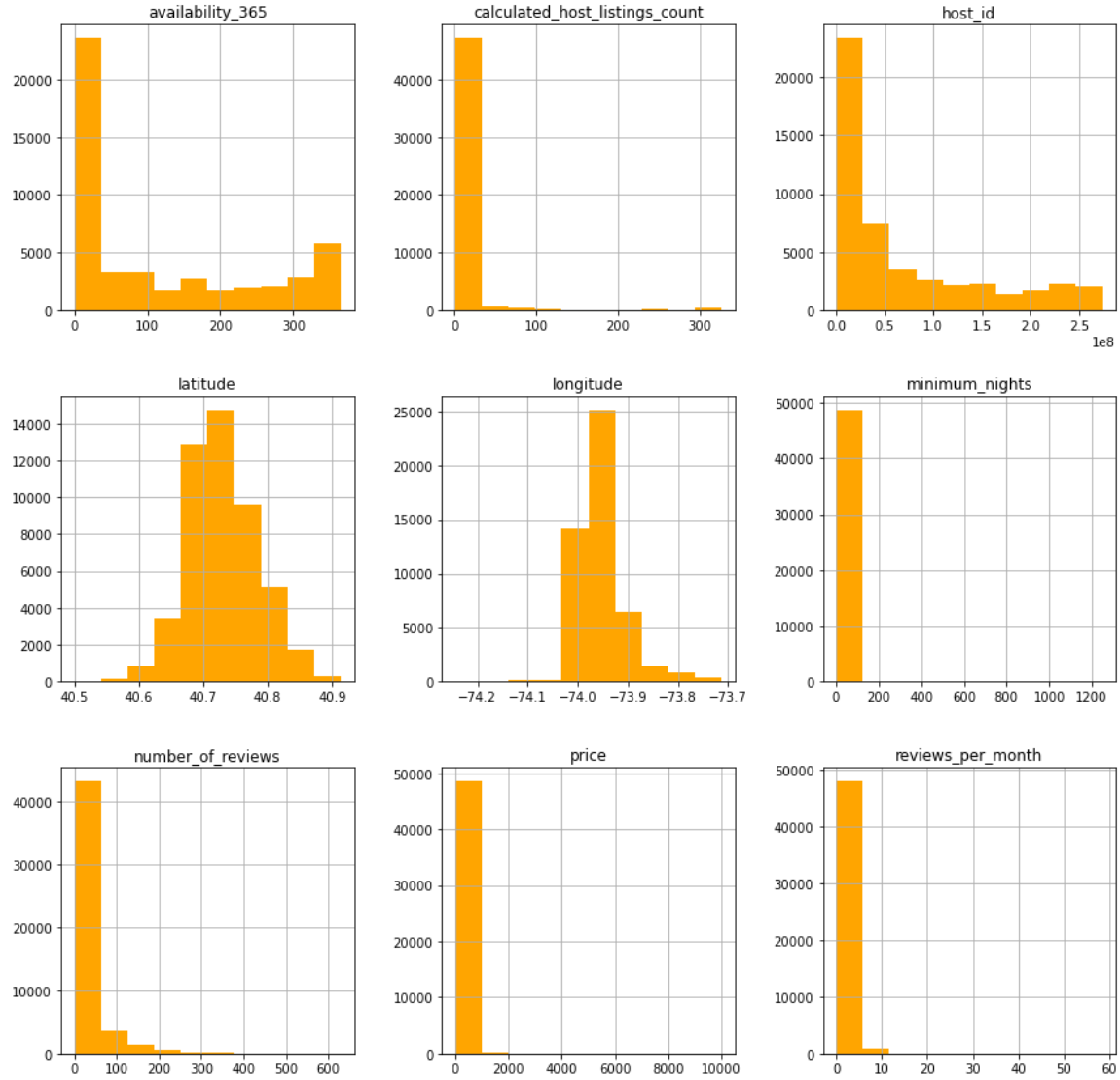
```
Average price per night at top 10 rooms is 65.4
9/10 are private rooms
```

Visualizing the distribution for every feature (Histogram)

In [39]:

```
fig = plt.figure(figsize = (15,15))
ax = fig.gca()
air_df.hist(ax=ax,color='orange')
plt.show()
```

```
<ipython-input-39-3847014b5728>:3: UserWarning: To output multiple subplot
s, the figure containing the passed axes is being cleared
  air_df.hist(ax=ax,color='orange')
```



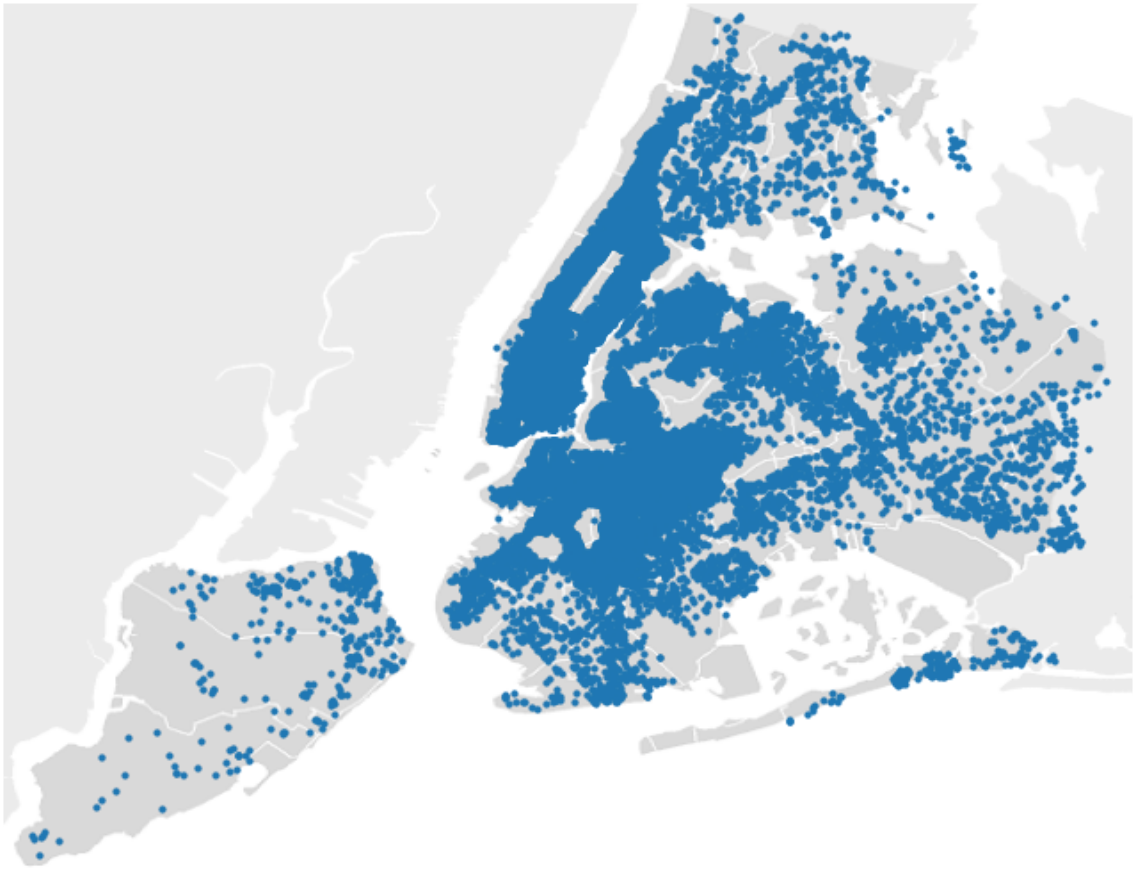
Plotting Different Rooms Location on New York Map

In [40]:

```
plt.figure(figsize=(15,9))
img = plt.imread("map_new_york.png")
plt.imshow(img,zorder=0,extent=[-74.258, -73.7, 40.49,40.92],alpha=0.3)
plt.scatter(air_df['longitude'],air_df['latitude'],s=10)
plt.axis('off')
```

Out[40]:

```
(-74.258, -73.7, 40.49, 40.92)
```



Plotting Longitude and Latitude of Rooms at different Neighbourhood Groups

In [41]:

```
air_df['neighbourhood_group'].unique()
```

Out[41]:

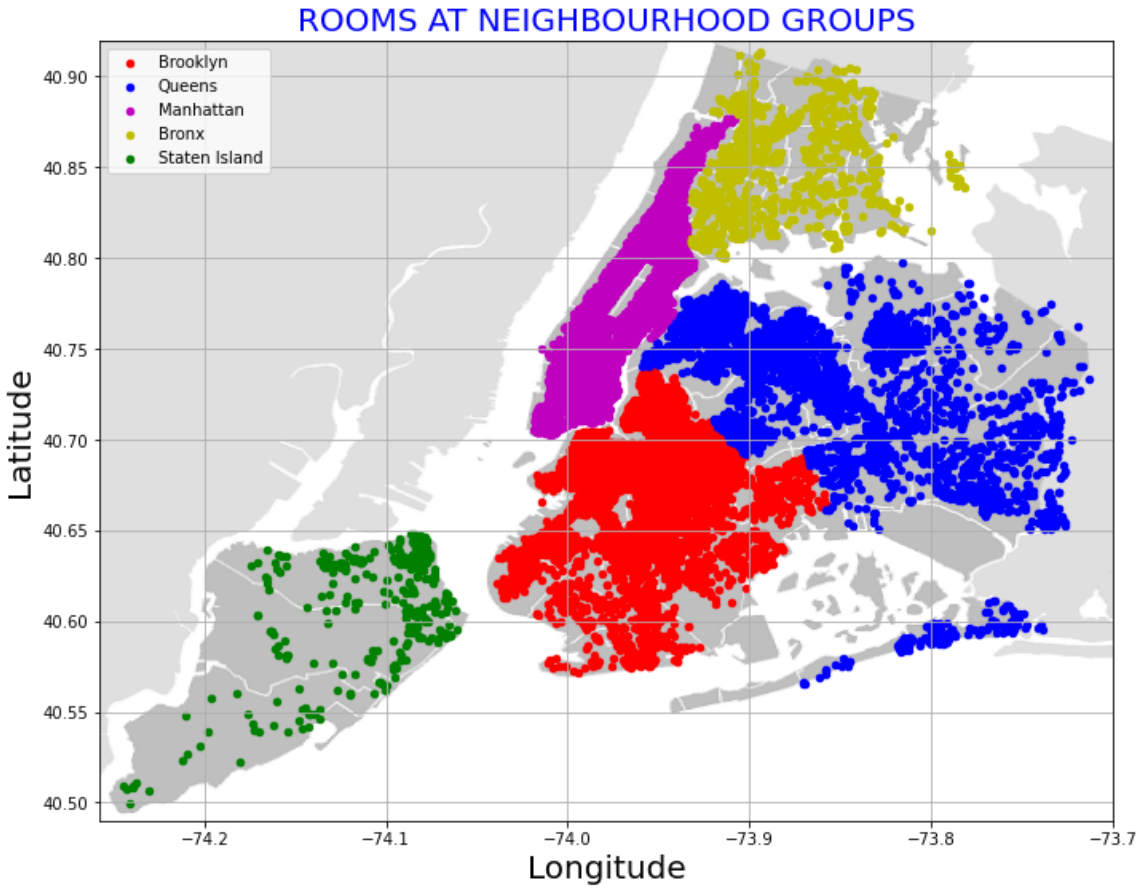
```
array(['Queens', 'Manhattan', 'Brooklyn', 'Staten Island', 'Bronx'],
      dtype=object)
```

In [42]:

```
data_brooklyn = air_df.loc[(air_df['neighbourhood_group'] == 'Brooklyn')]
lat_brooklyn = data_brooklyn['latitude']
long_brooklyn = data_brooklyn['longitude']
color_brooklyn = 'r'

data_queens = air_df.loc[(air_df['neighbourhood_group'] == 'Queens')]
lat_queens = data_queens['latitude']
long_queens = data_queens['longitude']
color_queens = 'b'

data_Manhattan = air_df.loc[(air_df['neighbourhood_group'] == 'Manhattan')]
lat_Manhattan = data_Manhattan['latitude']
long_Manhattan = data_Manhattan['longitude']
color_Manhattan = 'm'

data_Bronx = air_df.loc[(air_df['neighbourhood_group'] == 'Bronx')]
lat_Bronx = data_Bronx['latitude']
long_Bronx = data_Bronx['longitude']
color_Bronx = 'y'


data_Staten_Island = air_df.loc[(air_df['neighbourhood_group'] == 'Staten Island')]
lat_Staten_Island = data_Staten_Island['latitude']
long_Staten_Island = data_Staten_Island['longitude']
color_Staten_Island = 'g'
```

In [43]:

```
places_long_lat_color = [(long_brooklyn,lat_brooklyn,color_brooklyn,'Brooklyn'),(long_q
ueens,lat_queens,color_queens,'Queens'),
                         (long_Manhattan,lat_Manhattan,color_Manhattan,'Manhattan'),(lon
g_Bronx,lat_Bronx,color_Bronx,'Bronx'),
                         (long_Staten_Island,lat_Staten_Island,color_Staten_Island,'Stat
en Island')]
plt.figure(figsize=(15,9))
img = plt.imread("map_new_york.png")
plt.imshow(img,zorder=0,extent=[-74.258, -73.7, 40.49,40.92],alpha=0.5)
for (long,lat,pcolor,place) in places_long_lat_color:
    plt.scatter(long,lat,color=pcolor ,label= place ,s=20)

plt.grid(True)
plt.title("ROOMS AT NEIGHBOURHOOD GROUPS", color = 'b', fontsize=20)
plt.xlabel('Longitude',fontsize=20)
plt.ylabel('Latitude',fontsize=20)
plt.legend()
plt.show()
```



Plotting Longitude and Latitude of different Types of rooms in New york

In [44]:

```
air_df['room_type'].unique()
```

Out[44]:

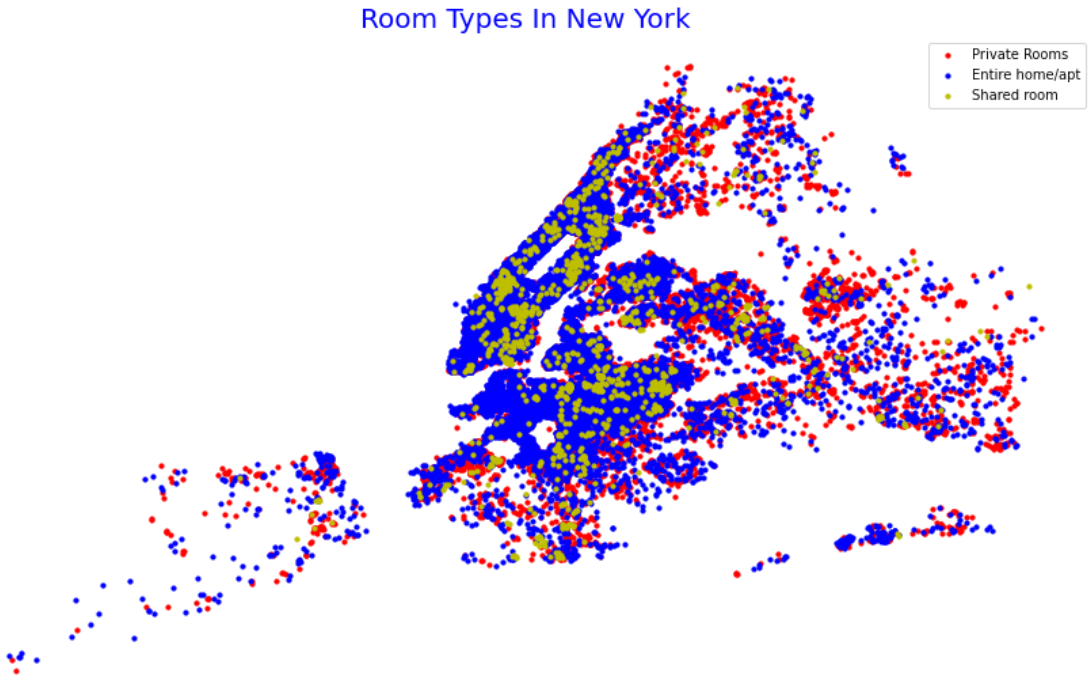
```
array(['Private room', 'Entire home/apt', 'Shared room'], dtype=object)
```

In [45]:

```
data_private = air_df.loc[(air_df['room_type'] == 'Private room')]
lat_private = data_private['latitude']
long_private = data_private['longitude']
color_private = 'r'

data_Entire = air_df.loc[(air_df['room_type'] == 'Entire home/apt')]
lat_Entire = data_Entire['latitude']
long_Entire = data_Entire['longitude']
color_Entire = 'b'

data_Shared_room = air_df.loc[(air_df['room_type'] == 'Shared room')]
lat_Shared_room = data_Shared_room['latitude']
long_Shared_room = data_Shared_room['longitude']
color_Shared_room = 'y'
```

In [46]:

```
places_long_lat_color = [(long_private,lat_private,color_private,'Private Rooms'),
                         (long_Entire,lat_Entire,color_Entire,'Entire home/apt'),
                        (long_Shared_room,lat_Shared_room,color_Shared_room,'Shared roo
m')]
plt.figure(figsize=(15,9))
for (long,lat,pcolor,place) in places_long_lat_color:
    plt.scatter(long,lat,color=pcolor ,label= place,s=10)
plt.title('Room Types In New York',fontsize=20,color='b')
plt.axis('off')
plt.legend()
plt.show()
```



Relation Between Neighbouring Group And Availability of days

In [47]:

```
air_df.neighbourhood_group.unique()
```

Out[47]:

```
array(['Queens', 'Manhattan', 'Brooklyn', 'Staten Island', 'Bronx'],
      dtype=object)
```

In [48]:

```
air_df['availability_365'].count()
```

Out[48]:

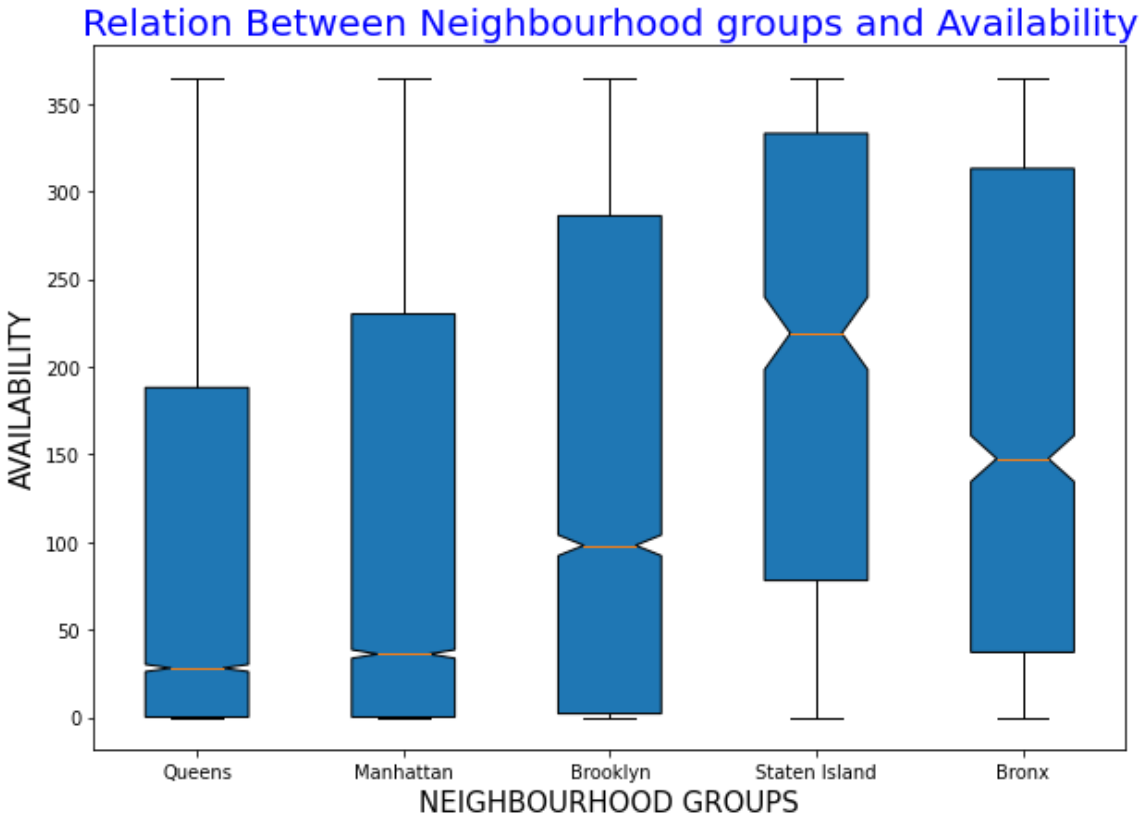
```
48879
```

In [49]:

```
data_brooklyn = air_df.loc[(air_df['neighbourhood_group'] == 'Brooklyn')]
avail_brooklyn = data_brooklyn['availability_365']

data_queens = air_df.loc[(air_df['neighbourhood_group'] == 'Queens')]
avail_queens = data_queens['availability_365']

data_Manhattan = air_df.loc[(air_df['neighbourhood_group'] == 'Manhattan')]
avail_Manhattan = data_Manhattan['availability_365']

data_Staten_Island = air_df.loc[(air_df['neighbourhood_group'] == 'Staten Island')]
avail_Staten_Island = data_Staten_Island['availability_365']

data_Bronx = air_df.loc[(air_df['neighbourhood_group'] == 'Bronx')]
avail_Bronx = data_Bronx['availability_365']
```

In [50]:

```
available_days = [avail_brooklyn,avail_Manhattan,avail_queens,avail_Staten_Island,avail
_Bronx]
plt.figure(figsize=(10,7))
plt.boxplot(available_days,patch_artist=True,notch='True',vert=1,labels=air_df.neighbou
rhood_group.unique())
plt.title('Relation Between Neighbourhood groups and Availability',fontsize=20,color=
'b')
plt.xlabel('NEIGHBOURHOOD GROUPS',fontsize=15)
plt.ylabel('AVAILABILITY',fontsize=15)
plt.show()
```



_This Airbnb ('AB_NYC*2019') dataset for the 2019 year appeared to be a very rich dataset with a variety of columns that allowed us to do deep data exploration on each significant column presented.*

In [ ]: