In [1]:
```
#importing necessery libraries for future analysis of the dataset
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import matplotlib.image as mpimg
%matplotlib inline
import seaborn as sns
```

In [2]:
```
spreadsheet = pd.read_csv('/Users/blacksheep/Downloads/AB_NYC_2019.csv')
```

In [3]:
```
spreadsheet.head()
```

Out[3]:

	id	name	host_id	host_name	neighbourhood_group	neighbourhood	latitude
0	2539	Clean & quiet apt home by the park	2787	John	Brooklyn	Kensington	40.64749
1	2595	Skylit Midtown Castle	2845	Jennifer	Manhattan	Midtown	40.75362
2	3647	THE VILLAGE OF HARLEM....NEW YORK !	4632	Elisabeth	Manhattan	Harlem	40.80902
3	3831	Cozy Entire Floor of Brownstone	4869	LisaRoxanne	Brooklyn	Clinton Hill	40.68514
4	5022	Entire Apt: Spacious Studio/Loft by central park	7192	Laura	Manhattan	East Harlem	40.79851

In [4]:
```
df = spreadsheet.sort_values('neighbourhood_group',ascending=True)
```

In [5]:
```
df.head()
```

Out[5]:

	id	name	host_id	host_name	neighbourhood_group	neighbourhood	l
7704	5824543	Private room for 2 (10 min to city) - Females ...	30232055	Qasim	Bronx	Longwood	40
45457	34756976	Gigi's Room	74633496	Justine	Bronx	University Heights	40
32797	25833266	Huge private & cozy room in the Bronx!	194102474	Digna	Bronx	Claremont Village	40
24924	19974905	Esteem's Place	141615596	Esteem	Bronx	Parkchester	40
32784	25816034	Bronx 2 Bedroom with View of Manhattan Skyline	179677211	Bettina	Bronx	Van Nest	40

In [6]:
```
#checking amount of rows in given dataset to understand the size we are working
len(spreadsheet)
```

Out[6]:
```
48895
```

In [7]:
```
#checking type of every column in the dataset
spreadsheet.dtypes
```

Out[7]:
```
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 [8]:
```
#after looking at the head of the dataset we already were able to notice some N

#looking to find out first what columns have null values
#using 'sum' function will show us how many nulls are found in each column in c
spreadsheet.isnull().sum()
```

Out[8]:
```
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
```

In [9]:
```
#dropping columns that are not significant or could be unethical to use for our
spreadsheet.drop(['id','host_name','last_review'], axis=1, inplace=True)
#examing the changes
spreadsheet.head(3)
```

Out[9]:

	name	host_id	neighbourhood_group	neighbourhood	latitude	longitude	room_ty
0	Clean & quiet apt home by the park	2787	Brooklyn	Kensington	40.64749	-73.97237	Priv rc
1	Skylit Midtown Castle	2845	Manhattan	Midtown	40.75362	-73.98377	En home/
2	THE VILLAGE OF HARLEM....NEW YORK !	4632	Manhattan	Harlem	40.80902	-73.94190	Priv rc

In [10]:
```
#replacing all NaN values in 'reviews_per_month' with 0
spreadsheet.fillna({'reviews_per_month':0}, inplace=True)
#examing changes
spreadsheet.reviews_per_month.isnull().sum()
```

Out[10]:
```
0
```

In [11]:
```
#let's proceed with examing some interesting categorical unique values

#examining the unique values of n_group as this column will appear very handy t
spreadsheet.neighbourhood_group.unique()
```

Out[11]:
```
array(['Brooklyn', 'Manhattan', 'Queens', 'Staten Island', 'Bronx'],
      dtype=object)
```

In [14]:
```
#we will skip first column for now and begin from host_id

#let's see what hosts (IDs) have the most listings on Airbnb platform and takin
top_host=spreadsheet.host_id.value_counts().head(10)
top_host
```

Out[14]:
```
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 [15]:
```
#coming back to our dataset we can confirm our fidnings with already existing c
top_host_check=spreadsheet.calculated_host_listings_count.max()
top_host_check
```

Out[15]:
```
327
```

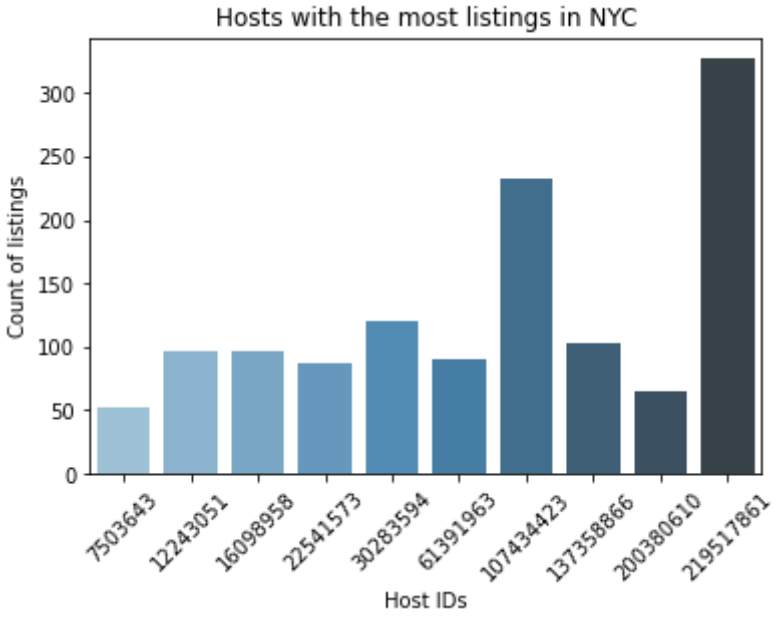
In [16]:
```
top_host_df=pd.DataFrame(top_host)
top_host_df.reset_index(inplace=True)
top_host_df.rename(columns={'index':'Host_ID', 'host_id':'P_Count'}, inplace=Tr
top_host_df
```

Out[16]:

	Host_ID	P_Count
0	219517861	327
1	107434423	232
2	30283594	121
3	137358866	103
4	16098958	96
5	12243051	96
6	61391963	91
7	22541573	87
8	200380610	65
9	7503643	52

In [17]:
```
viz_1=sns.barplot(x="Host_ID", y="P_Count", data=top_host_df,
                  palette='Blues_d')
viz_1.set_title('Hosts with the most listings in NYC')
viz_1.set_ylabel('Count of listings')
viz_1.set_xlabel('Host IDs')
viz_1.set_xticklabels(viz_1.get_xticklabels(), rotation=45)
```

Out[17]:
```
[Text(0, 0, '7503643'),
 Text(1, 0, '12243051'),
 Text(2, 0, '16098958'),
 Text(3, 0, '22541573'),
 Text(4, 0, '30283594'),
 Text(5, 0, '61391963'),
 Text(6, 0, '107434423'),
 Text(7, 0, '137358866'),
 Text(8, 0, '200380610'),
 Text(9, 0, '219517861')]
```



In [20]:
```
#let's find out more about our neiberhoods presented 'Brooklyn', 'Manhattan',

#Brooklyn
sub_1=spreadsheet.loc[spreadsheet['neighbourhood_group'] == 'Brooklyn']
price_sub1=sub_1[['price']]
#Manhattan
sub_2=spreadsheet.loc[spreadsheet['neighbourhood_group'] == 'Manhattan']
price_sub2=sub_2[['price']]
#Queens
sub_3=spreadsheet.loc[spreadsheet['neighbourhood_group'] == 'Queens']
price_sub3=sub_3[['price']]
#Staten Island
sub_4=spreadsheet.loc[spreadsheet['neighbourhood_group'] == 'Staten Island']
price_sub4=sub_4[['price']]
#Bronx
sub_5=spreadsheet.loc[spreadsheet['neighbourhood_group'] == 'Bronx']
price_sub5=sub_5[['price']]
#putting all the prices' dfs in the list
price_list_by_n=[price_sub1, price_sub2, price_sub3, price_sub4, price_sub5]
```

In [21]:
```
#creating an empty list that we will append later with price distributions for
p_l_b_n_2=[]
#creating list with known values in neighbourhood_group column
nei_list=['Brooklyn', 'Manhattan', 'Queens', 'Staten Island', 'Bronx']
#creating a for loop to get statistics for price ranges and append it to our em
for x in price_list_by_n:
    i=x.describe(percentiles=[.25, .50, .75])
    i=i.iloc[3:]
    i.reset_index(inplace=True)
    i.rename(columns={'index':'Stats'}, inplace=True)
    p_l_b_n_2.append(i)
#changing names of the price column to the area name for easier reading of the
p_l_b_n_2[0].rename(columns={'price':nei_list[0]}, inplace=True)
p_l_b_n_2[1].rename(columns={'price':nei_list[1]}, inplace=True)
p_l_b_n_2[2].rename(columns={'price':nei_list[2]}, inplace=True)
p_l_b_n_2[3].rename(columns={'price':nei_list[3]}, inplace=True)
p_l_b_n_2[4].rename(columns={'price':nei_list[4]}, inplace=True)
#finilizing our dataframe for final view
stat_df=p_l_b_n_2
```

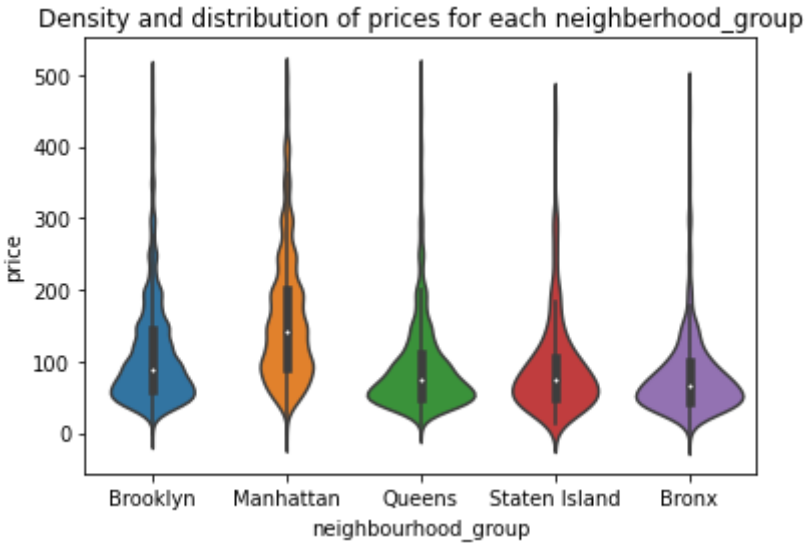
```
stat_df=[df.set_index('Stats') for df in stat_df]
stat_df=stat_df[0].join(stat_df[1:])
stat_df
```

Out[21]:

Stats	Brooklyn	Manhattan	Queens	Staten Island	Bronx
min	0.0	0.0	10.0	13.0	0.0
25%	60.0	95.0	50.0	50.0	45.0
50%	90.0	150.0	75.0	75.0	65.0
75%	150.0	220.0	110.0	110.0	99.0
max	10000.0	10000.0	10000.0	5000.0	2500.0

In [24]:
```
#we can see from our statistical table that we have some extreme values, there

#creating a sub-dataframe with no extreme values / less than 500
sub_6=spreadsheet[spreadsheet.price < 500]
#using violinplot to showcase density and distribtuion of prices
viz_2=sns.violinplot(data=sub_6, x='neighbourhood_group', y='price')
viz_2.set_title('Density and distribution of prices for each neighberhood_group
```

Out[24]:
```
Text(0.5, 1.0, 'Density and distribution of prices for each neighberhood_grou
p')
```

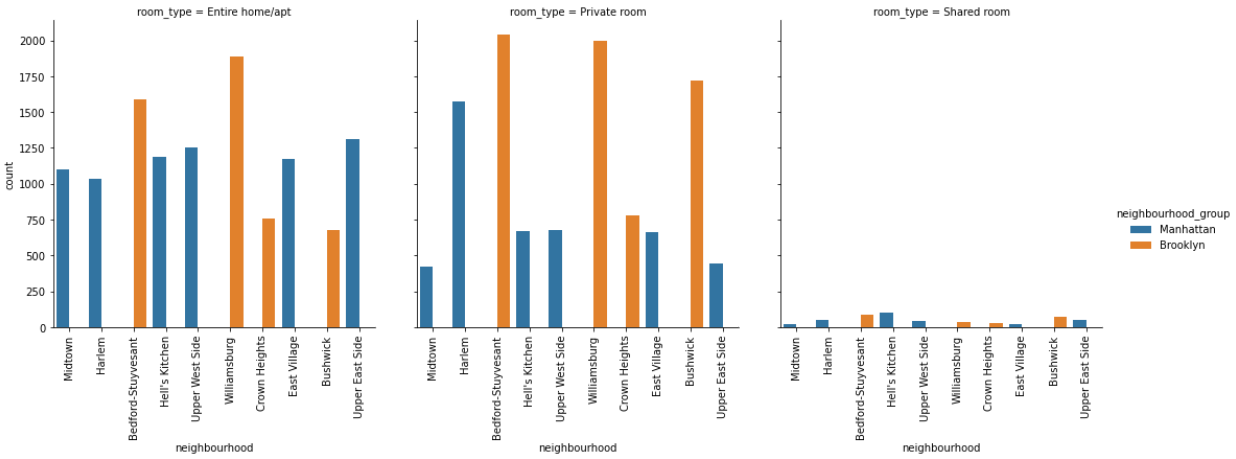


In [26]:
```
#as we saw earlier from unique values for neighbourhood there are way too many
#therefore, let's grab just top 10 neighbourhoods that have the most listings

#finding out top 10 neighbourhoods
spreadsheet.neighbourhood.value_counts().head(10)
```

Out[26]:
```
Williamsburg           3920
Bedford-Stuyvesant     3714
Harlem                 2658
Bushwick               2465
Upper West Side        1971
Hell's Kitchen         1958
East Village           1853
Upper East Side        1798
Crown Heights          1564
Midtown                1545
Name: neighbourhood, dtype: int64
```

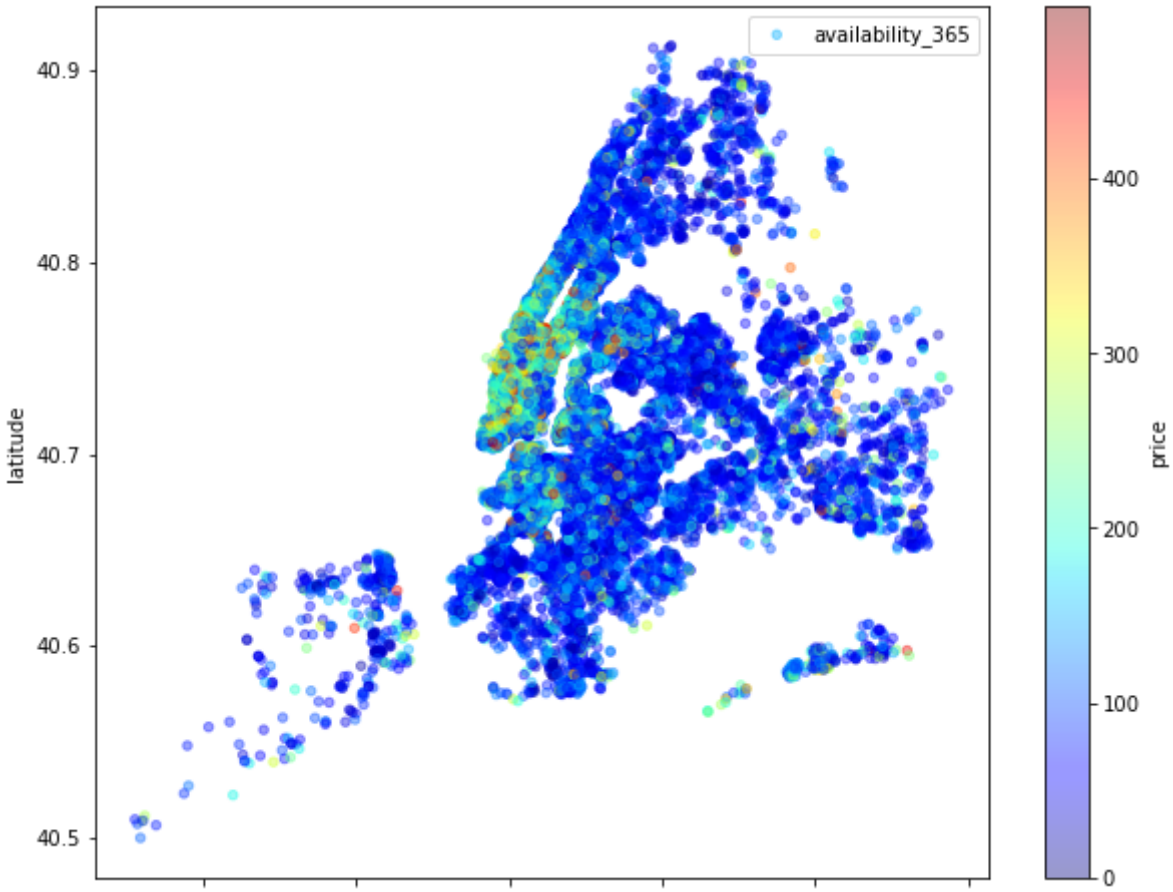
In [29]:
```
#let's now combine this with our boroughs and room type for a rich visualizatic

#grabbing top 10 neighbourhoods for sub-dataframe
sub_7=spreadsheet.loc[spreadsheet['neighbourhood'].isin(['Williamsburg','Bedfor
                  'Upper West Side','Hell\'s Kitchen','East Village','Upper East
#using catplot to represent multiple interesting attributes together and a cour
viz_3=sns.catplot(x='neighbourhood', hue='neighbourhood_group', col='room_type'
viz_3.set_xticklabels(rotation=90)
```

Out[29]:
```
<seaborn.axisgrid.FacetGrid at 0x11c548550>
```



In [30]:
```
#let's what we can do with our given longtitude and latitude columns

#let's see how scatterplot will come out
viz_4=sub_6.plot(kind='scatter', x='longitude', y='latitude', label='availabili
                 cmap=plt.get_cmap('jet'), colorbar=True, alpha=0.4, figsize=(
viz_4.legend()
```

Out[30]:
```
<matplotlib.legend.Legend at 0x11e0b8280>
```



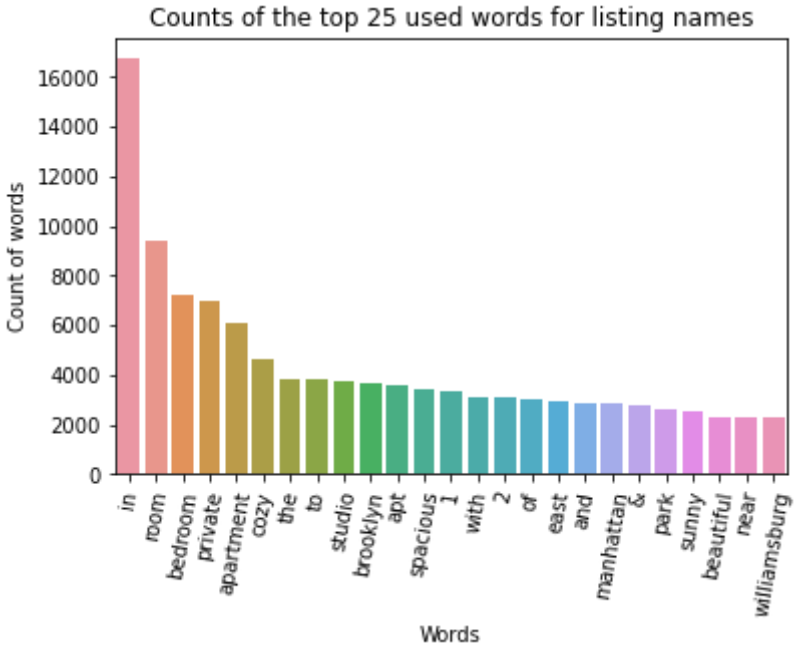
In [36]:
```
#let's comeback now to the 'name' column as it will require litte bit more codi

#initializing empty list where we are going to put our name strings
_names_=[]
#getting name strings from the column and appending it to the list
for name in spreadsheet.name:
    _names_.append(name)
#setting a function that will split those name strings into separate words
def split_name(name):
    spl=str(name).split()
    return spl
#initializing empty list where we are going to have words counted
_names_for_count_=[]
#getting name string from our list and using split function, later appending to
for x in _names_:
    for word in split_name(x):
        word=word.lower()
        _names_for_count_.append(word)
```

In [37]:
```
#we are going to use counter
from collections import Counter
#let's see top 25 used words by host to name their listing
_top_25_w=Counter(_names_for_count_).most_common()
_top_25_w=_top_25_w[0:25]
```

In [38]:
```
#now let's put our findings in dataframe for further visualizations
sub_w=pd.DataFrame(_top_25_w)
sub_w.rename(columns={0:'Words', 1:'Count'}, inplace=True)
```

In [39]:
```
#we are going to use barplot for this visualization
viz_5=sns.barplot(x='Words', y='Count', data=sub_w)
viz_5.set_title('Counts of the top 25 used words for listing names')
viz_5.set_ylabel('Count of words')
viz_5.set_xlabel('Words')
viz_5.set_xticklabels(viz_5.get_xticklabels(), rotation=80)
```

Out[39]:
```
[Text(0, 0, 'in'),
 Text(1, 0, 'room'),
 Text(2, 0, 'bedroom'),
 Text(3, 0, 'private'),
 Text(4, 0, 'apartment'),
 Text(5, 0, 'cozy'),
 Text(6, 0, 'the'),
 Text(7, 0, 'to'),
 Text(8, 0, 'studio'),
 Text(9, 0, 'brooklyn'),
 Text(10, 0, 'apt'),
 Text(11, 0, 'spacious'),
 Text(12, 0, '1'),
 Text(13, 0, 'with'),
 Text(14, 0, '2'),
 Text(15, 0, 'of'),
 Text(16, 0, 'east'),
 Text(17, 0, 'and'),
 Text(18, 0, 'manhattan'),
 Text(19, 0, '&'),
 Text(20, 0, 'park'),
 Text(21, 0, 'sunny'),
 Text(22, 0, 'beautiful'),
 Text(23, 0, 'near'),
 Text(24, 0, 'williamsburg')]
```



In [41]:
```
#last column we need to look at is 'number_of_reviews'

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

Out[41]:

	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 [42]:
```
price_avrg=top_reviewed_listings.price.mean()
print('Average price per night: {}'.format(price_avrg))
```

```
Average price per night: 65.4
```

In [ ]:
```
```