Competitive Analysis On Rental Pricing

In [4]:

```python
import numpy as np
import pandas as pd
import pandas_profiling as pd_prof
from decimal import Decimal

data = pd.read_csv("C:\\Users\\Yat\\Documents\\MSDS\\MSDS 7331\\ML_Lab_Data\\listings.csv")
```

```
C:\Users\Yat\Anaconda3\lib\site-packages\IPython\core\interactiveshell.py:3058: DtypeWarning: Columns (61,62,94,95) have mixed types. Specify dtype option on import or set low_memory=False.
  interactivity=interactivity, compiler=compiler, result=result)
```

In [1]:

```python
#data.head()
```

After reiewing the raw dataset, columns contain ID, URLs and text descriptions will be dropped.

In [5]:

```python
sub=data.drop(['id','listing_url','scrape_id','last_scraped','summary','space','description','experiences_offered'
              , 'neighborhood_overview', 'notes', 'transit', 'access', 'interaction', 'house_rules',
              'thumbnail_url', 'medium_url', 'picture_url', 'xl_picture_url', 'host_url', 'host_thumbnail_url',
              'host_picture_url', 'country_code', 'country','amenities', 'minimum_minimum_nights',
              'maximum_minimum_nights','minimum_maximum_nights', 'maximum_maximum_nights','minimum_nights_avg_ntm',
              'maximum_nights_avg_ntm', 'availability_30', 'availability_365','availability_90','has_availability',
               'calculated_host_listings_count','calculated_host_listings_count_shared_rooms',
               'is_business_travel_ready','host_about', 'host_acceptance_rate', 'host_total_listings_count',
              'jurisdiction_names','license','monthly_price','square_feet','weekly_price', 'requires_license'], axis=1)
```

In [10]:

```python
#data.info()
```

In [7]:

```python
#functions courtesy of Karen
def money_to_decimal(x):
    x = x.replace("$", "").replace(",", "").replace(" ", "")
    return float(x)

def rem_percent(x):
    x=x.replace("%","")
    return float(x)/100
def truncate(n):
    return int(n * 1000) / 1000
```

In [8]:

```
#courtesy of Karen
#converts objects with money values into decimal values to become continous attribute
sub.cleaning_fee = sub.cleaning_fee.astype(str)
sub.extra_people = sub.extra_people.astype(str)
sub.security_deposit = sub.security_deposit.astype(str)
sub.price = sub.price.astype(str)
sub.loc[:,'price'] = sub.loc[:,'price'].apply(money_to_decimal)
sub.loc[:,'cleaning_fee'] = sub.loc[:,'cleaning_fee'].apply(money_to_decimal)
sub.loc[:,'extra_people'] = sub.loc[:,'extra_people'].apply(money_to_decimal)
sub.loc[:,'security_deposit'] = sub.loc[:,'security_deposit'].apply(money_to_decimal)

#imputations
sub['price']=sub.price.mask(sub.price == 0,sub.price.median())
sub.cleaning_fee=sub.cleaning_fee.fillna(sub.cleaning_fee.median())
sub.first_review=sub.first_review.fillna('2019-08-01')
sub['first_review'] =  pd.to_datetime(sub['first_review'],
                              format='%Y-%m-%d')
sub.host_response_rate = sub.host_response_rate.astype(str)
sub.loc[:,'host_response_rate'] = sub.loc[:, 'host_response_rate'].apply(rem_percent)
sub.host_response_rate=sub.host_response_rate.fillna(sub.host_response_rate.median())
sub['host_since'] =  pd.to_datetime(sub['host_since'],
                              format='%Y-%m-%d')
sub.last_review=sub.last_review.fillna('2019-08-01')
sub['last_review'] =  pd.to_datetime(sub['last_review'],
                              format='%Y-%m-%d')
sub.review_scores_accuracy=sub.review_scores_accuracy.fillna(truncate(sub.review_scores_accuracy.median()))
sub.review_scores_checkin=sub.review_scores_checkin.fillna(truncate(sub.review_scores_checkin.median()))
sub.review_scores_cleanliness=sub.review_scores_cleanliness.fillna(truncate(sub.review_scores_cleanliness.median()))
sub.review_scores_communication=sub.review_scores_communication.fillna(truncate(sub.review_scores_communication.median()))
sub.review_scores_location=sub.review_scores_location.fillna(truncate(sub.review_scores_location.median()))
sub.review_scores_rating=sub.review_scores_rating.fillna(truncate(sub.review_scores_rating.median()))
sub.review_scores_value=sub.review_scores_value.fillna(truncate(sub.review_scores_value.median()))
sub.reviews_per_month=sub.reviews_per_month.fillna(sub.reviews_per_month.median())
sub.security_deposit=sub.security_deposit.fillna(sub.security_deposit.median())
```

In [2]:

```
#sub.describe()
```

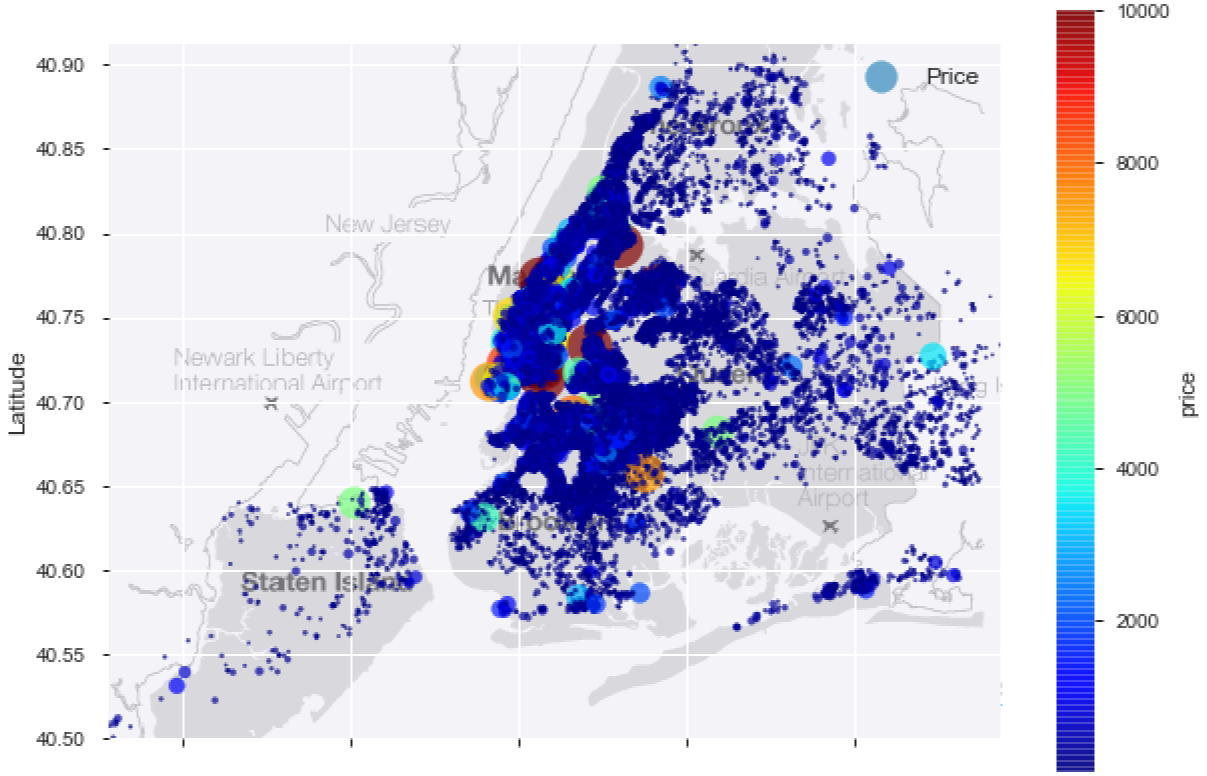
In [9]:

```
sub.info()
#checking the data ohjects after the conversion
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 48864 entries, 0 to 48863
Data columns (total 60 columns):
name                                            48848 non-null object
host_id                                         48864 non-null int64
host_name                                       48846 non-null object
host_since                                      48846 non-null datetime64[ns]
host_location                                   48694 non-null object
host_response_time                              32282 non-null object
host_response_rate                              48864 non-null float64
host_is_superhost                               48846 non-null object
host_neighbourhood                              42443 non-null object
host_listings_count                             48846 non-null float64
host_verifications                              48864 non-null object
host_has_profile_pic                            48846 non-null object
host_identity_verified                          48846 non-null object
street                                          48864 non-null object
neighbourhood                                   48853 non-null object
neighbourhood_cleansed                          48864 non-null object
neighbourhood_group_cleansed                    48864 non-null object
city                                            48802 non-null object
state                                           48859 non-null object
zipcode                                         48349 non-null object
market                                          48761 non-null object
smart_location                                  48864 non-null object
latitude                                        48864 non-null float64
longitude                                       48864 non-null float64
is_location_exact                               48864 non-null object
property_type                                   48864 non-null object
room_type                                       48864 non-null object
accommodates                                    48864 non-null int64
bathrooms                                       48808 non-null float64
bedrooms                                        48837 non-null float64
beds                                            48822 non-null float64
bed_type                                        48864 non-null object
price                                           48864 non-null float64
security_deposit                                48864 non-null float64
cleaning_fee                                    48864 non-null float64
guests_included                                 48864 non-null int64
extra_people                                    48864 non-null float64
minimum_nights                                  48864 non-null int64
maximum_nights                                  48864 non-null int64
calendar_updated                                48864 non-null object
availability_60                                 48864 non-null int64
calendar_last_scraped                           48864 non-null object
number_of_reviews                               48864 non-null int64
number_of_reviews_ltm                           48864 non-null int64
first_review                                    48864 non-null datetime64[ns]
last_review                                     48864 non-null datetime64[ns]
review_scores_rating                            48864 non-null float64
review_scores_accuracy                          48864 non-null float64
review_scores_cleanliness                       48864 non-null float64
review_scores_checkin                           48864 non-null float64
review_scores_communication                     48864 non-null float64
review_scores_location                          48864 non-null float64
review_scores_value                             48864 non-null float64
instant_bookable                                48864 non-null object
cancellation_policy                             48863 non-null object
require_guest_profile_picture                   48864 non-null object
require_guest_phone_verification                48864 non-null object
calculated_host_listings_count_entire_homes     48864 non-null int64
```

```
calculated_host_listings_count_private_rooms     48864 non-null int64
reviews_per_month                                48864 non-null float64
dtypes: datetime64[ns](3), float64(19), int64(10), object(28)
memory usage: 22.4+ MB
```

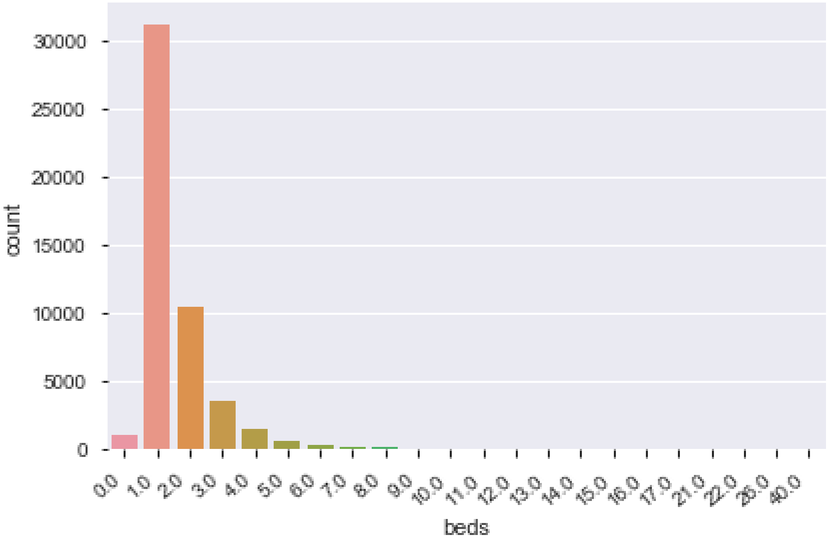
In [91]:
```
import geopandas as gpd
import matplotlib.pyplot as plt
import matplotlib.image as mpimg

%matplotlib inline


NYC_img=mpimg.imread("C:\\Users\\Yat\\Documents\\MSDS\\MSDS 7331\\ML_Lab_Data\\nyc-b
oros.png")
ax = sub.plot(kind="scatter", x="longitude", y="latitude",
    s=sub['price']/20, label="Price",
    c="price", cmap=plt.get_cmap("jet"),
    colorbar=True, alpha=0.7, figsize=(10,7),
)
plt.imshow(NYC_img, extent=[-74.244, -73.713, 40.5, 40.912], alpha=0.5)
plt.ylabel("Latitude", fontsize=12)
plt.xlabel("Longitude", fontsize=12)

plt.legend(fontsize=12)
plt.show()
```



Projecting Rental Price Onto The NYC Boros Map

In the scatter plot, a high concentration of rentals were from the Manhattan and Brooklyn. Majority of the rental prices were below the $2000 price range. However there are some high rental price (8000$ and above) spots within Manhattan area. Our preliminary analysis suggest that the highly concentrated rental areas correspond to certain landmarks and mass transit locations such as Time Square or La Guardia Airport

In [113]:
```
import seaborn as sns
ax=sns.countplot(x="beds", data=sub)
ax.set_xticklabels(ax.get_xticklabels(), rotation=40, ha="right")
plt.tight_layout()
plt.show()
```

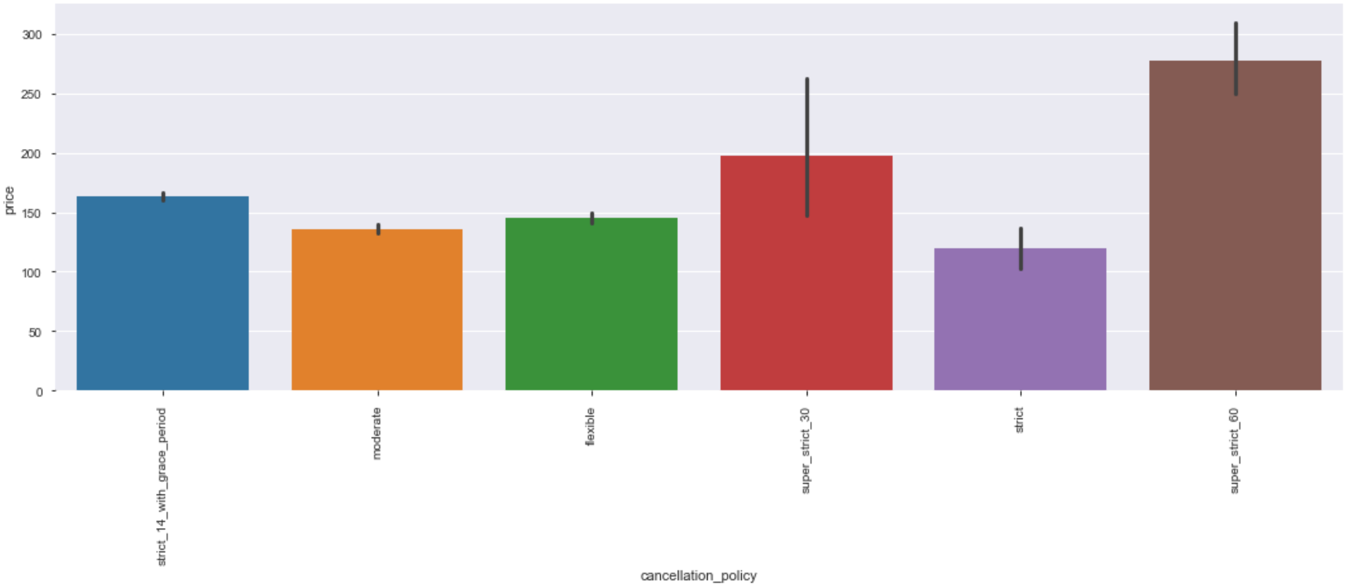


Beds

Most units have between 1 to 2 beds with price ranges from sub $2000to$10000. 0 bed option was shown in the plot and there could be reason such as these units may have alternative sleeping arrangement and will need to investigate further for explanations. The beds versus price plot is right skewed and there are outliers are identified at beds = 21, 22, 26 and 40.

In [155]:
```
ax=sns.catplot(x="cancellation_policy", y="price", kind="bar", data=sub,height=5, as
pect=3)
ax.set_xticklabels(rotation=90)
```

Out[155]:
```
<seaborn.axisgrid.FacetGrid at 0x1d01c50e7f0>
```

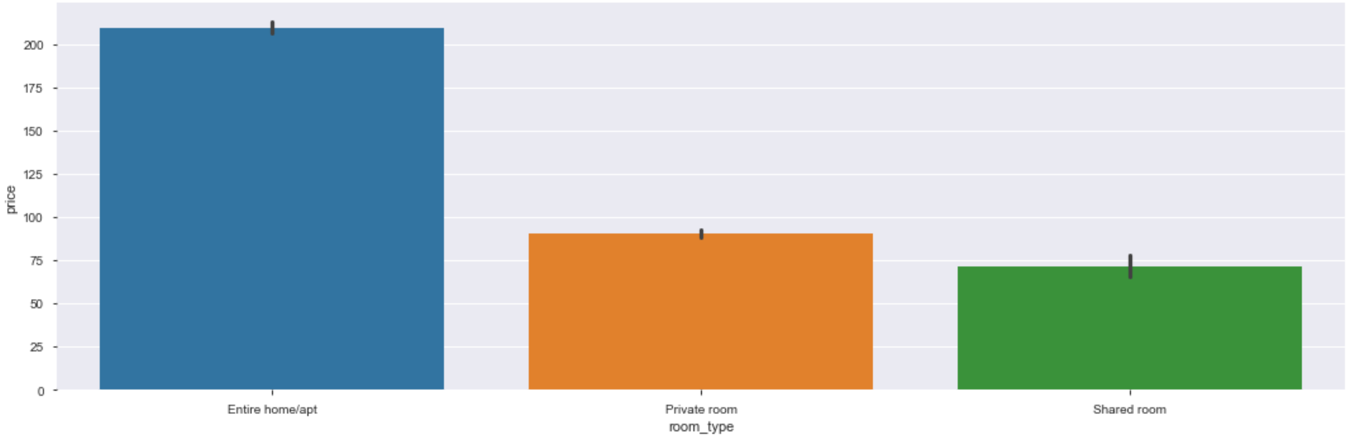


Cancellation Policy and Price

Rental Price over $200usuallyhasa(superstrict60)60dayscancellationpolicywhilethe$140-$150unitsusuallyhaveamoderatetoflixiblecancallation.Unitswith>$150 to < $200haveа14daysgraceperiodandat$200, the cencellation policy is 30 days (super strict 20). Properties with a $<150 price and strict cancellation policy will need to be further examine to determine if this could be grouped into either the super strict 30 or super strict 60 group.

```
In [156]: sns.catplot(x="room_type", y="price", kind="bar", data=sub,height=5, aspect=3)
```

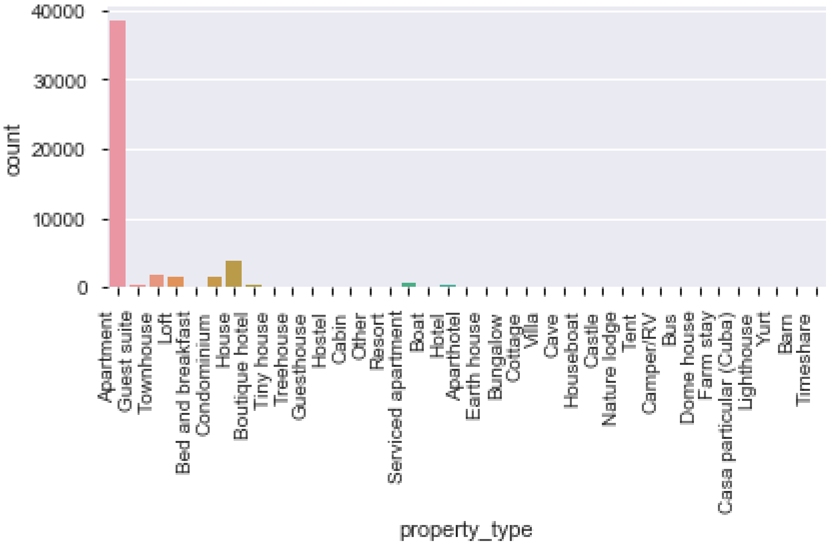
```
Out[156]: <seaborn.axisgrid.FacetGrid at 0x1d0f9782748>
```



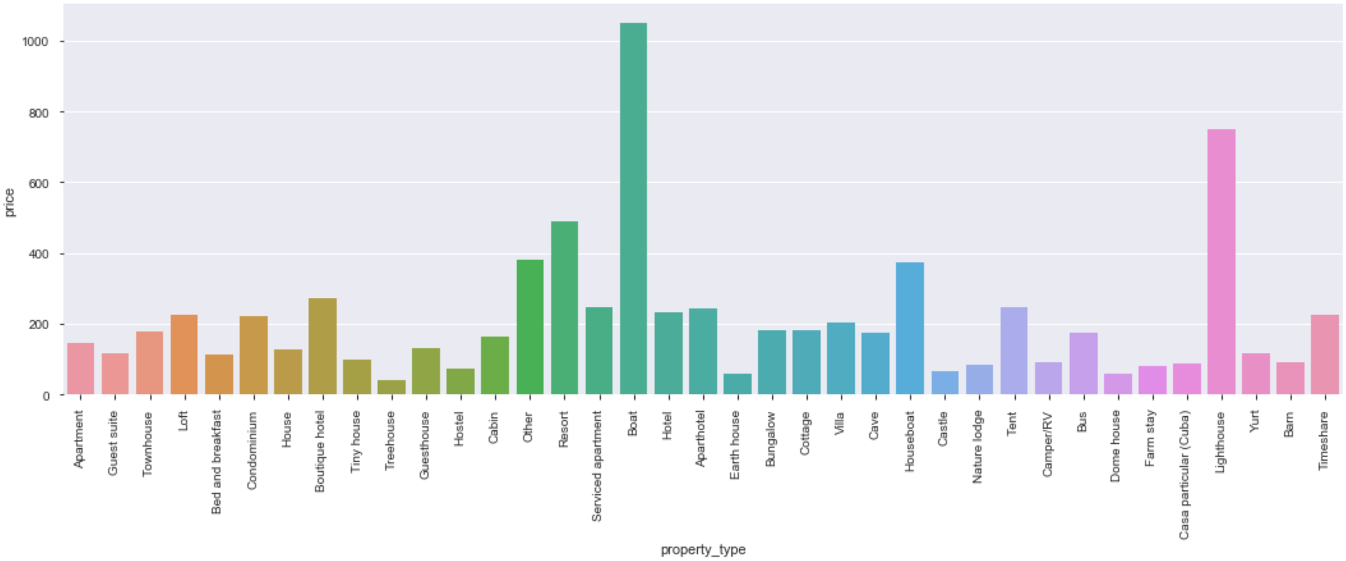
Room Type

In terms of room type, most prefer the entire home/ apartment option which costs approxmiately $200(Forreference,standarddeviationofpriceis236.576536).Followbyprivateroomat$90 and shared room at $70.

```
In [157]: ax=sns.countplot(x="property_type", data=sub)
          ax.set_xticklabels(ax.get_xticklabels(), rotation=90, ha="right")
          plt.tight_layout()
          plt.show()

          ax=sns.catplot(x="property_type", y="price", kind="bar", data=sub, ci=None,height=5,
          aspect=3)
          ax.set_xticklabels(rotation=90)
```



```
Out[157]: <seaborn.axisgrid.FacetGrid at 0x1d03d5aeef0>
```

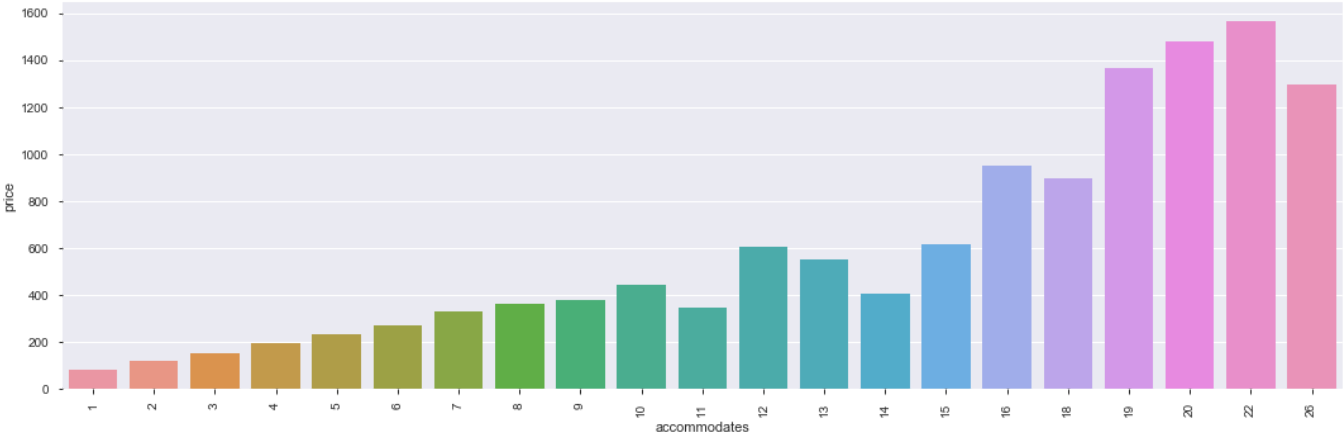


Property Type

In the property type by count plot, we see that the most popular and common type is an apartment. In the plot we can also see that boutique hotel is also a common choice for Airbnb renters after apartment. In the property type versus price, we can see that the rental price variable ranged from the $100stoover$1000. Most of the rental properties were in the sub $500rangehoweverwealsoseeunusualpropertiessuchasboatandlighthouseswhichcouldcostrentersfrom >$500 to >$1000.

```
In [158]: ax=sns.catplot(x="accommodates", y="price", kind="bar", data=sub, ci=None,height=5,
          aspect=3)
          ax.set_xticklabels(rotation=90)
```

```
Out[158]: <seaborn.axisgrid.FacetGrid at 0x1d0ce3bca90>
```

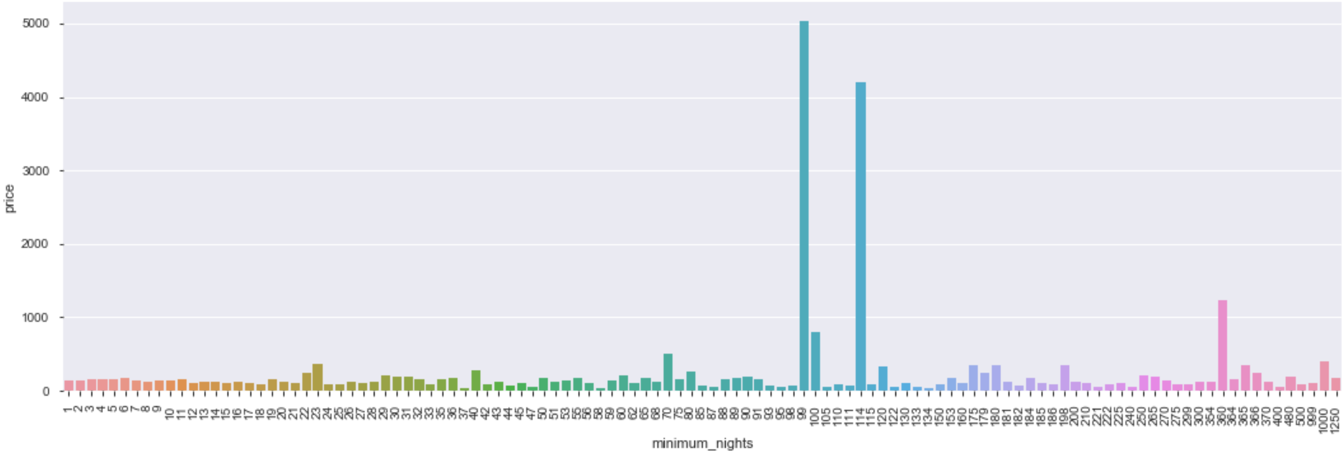


Accommodates

Properties that can accomodates between 1-4 guests were usually in the >= $200 range. For 5 to 9 guests, the cost is betweem >$200 to $400. In general, the rent price costs more if it can accommodate more guests. There were exceptions where a 11-guest, 14-guest, 18-guest and 26-guest units would cost less than a smaller units (10,13,16,22-guest variants).

```
In [154]: plt.figure(figsize=(20,15))
          ax=sns.catplot(x="minimum_nights", y="price", kind="bar", data=sub, ci=None,height=5
          , aspect=3)
          ax.set_xticklabels(rotation=90)
```

```
Out[154]: <seaborn.axisgrid.FacetGrid at 0x1d014facf98>

          <Figure size 1440x1080 with 0 Axes>
```

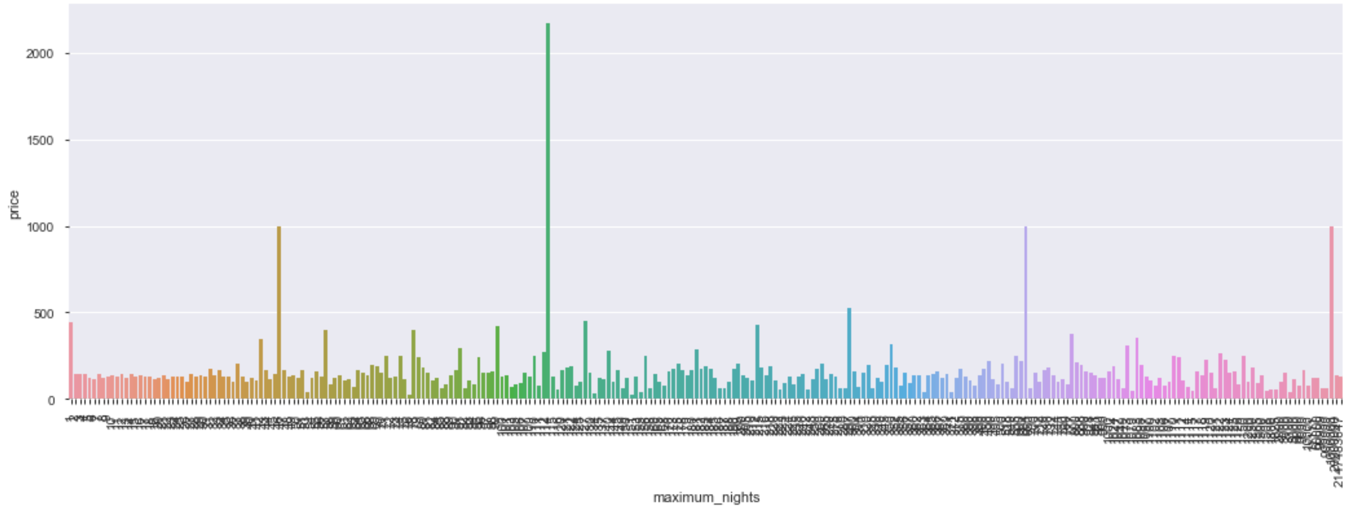


Minimum Nights

We see that two rental units with 99 or 114 minimum nights stay requirement had a price of >$4000. Assuming most renters were staying for a short period of time from 1 day to months, these two points could be the total cost of renting a certain property type.

```
In [150]: plt.figure(figsize=(20,15))
          ax=sns.catplot(x="maximum_nights", y="price", kind="bar", data=sub, ci=None,height=5
          , aspect=3)
          ax.set_xticklabels(rotation=90)
```

```
Out[150]: <seaborn.axisgrid.FacetGrid at 0x1d01891f940>

          <Figure size 1440x1080 with 0 Axes>
```

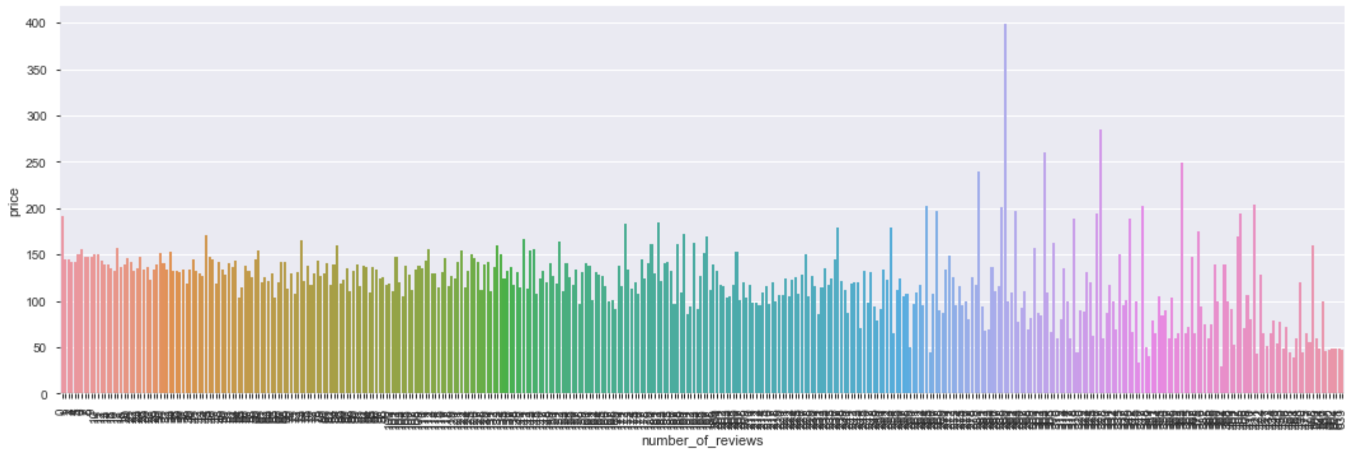


Maximum Nights

Simialr to the minimum nights stay, some of the maximum nights data points had $1000 to over 2000$ rental price that that could either be the total cost of renting the unit upto the maximum days or they were pricing for a long term stay.

```
In [143]: plt.figure(figsize=(20,15))
          ax=sns.catplot(x="number_of_reviews", y="price", kind="bar", data=sub , ci=None,heig
          ht=5, aspect=3)
          ax.set_xticklabels(rotation=90)
```

```
Out[143]: <seaborn.axisgrid.FacetGrid at 0x1d02d38a128>

          <Figure size 1440x1080 with 0 Axes>
```

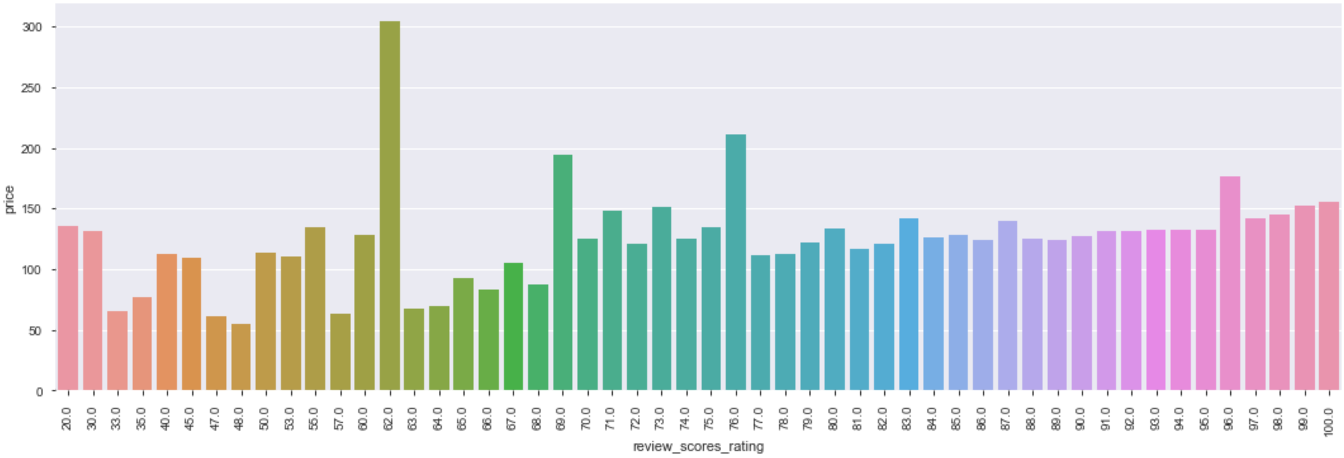


In [147]:

```python
plt.figure(figsize=(20,15))
ax=sns.catplot(x="review_scores_rating", y="price", kind="bar", data=sub , ci=None,height=5, aspect=3)
ax.set_xticklabels(rotation=90)
```

Out[147]:

```
<seaborn.axisgrid.FacetGrid at 0x1d0d5586240>

<Figure size 1440x1080 with 0 Axes>
```



Review Rating Score vs Price

Rating Score between 80 to 100 were associated primarily with the rental units between >$100to <$200. There was a low score of 62 that was associated with a rental price of $300 and this needs to be examine to reference the property type and location to see if we could perhaps isolate the point.

In [ ]: