Inference Document for SML -CT2

1.Read the dataset – Using Pandas

2. Data Understanding

a. What are the number of rows; no. & types of variables (continuous, categorical etc.)

Data Overview:

a. Number of Rows, Columns, and Variable Types:

- The dataset contains a total of **num_rows** rows and **num_cols** columns.
- Variable types are categorized into different data types (**int64**, **float64**, **object**).

b. Calculate five point summary for numerical variables

Descriptive Statistics:

b. Five-point Summary for Numerical Variables:

- Descriptive statistics (mean, standard deviation, min, 25%, 50%, 75%, max) for numerical variables are provided, including **price**, **minimum_nights**, **number_of_reviews**, etc.

c. Summarize observations for categorical variables – no. of categories, % observations in each category

c. Summary for Categorical Variables:

- Descriptive statistics for categorical variables, including count, unique values, top value, and frequency.

Categorical Variables Analysis:

- For each categorical variable, the number of categories and the percentage of observations in each category are calculated and displayed.

Selected Analyses:

1. Host-Level Analysis:

- Average number of listings per host is calculated.

2. Price Distribution by Room Type:

- A boxplot is used to visualize the distribution of prices for different room types.

3. Availability by Room Type:

- Average availability (in days) is calculated for each room type.

4. Hosts with Multiple Listings:

- Identifies hosts with more than one listing and displays their number of listings.

5. Distribution of Minimum Nights:

- Histogram and kernel density estimate are used to visualize the distribution of minimum nights.

6. Popular Neighbourhoods:

- Lists the top 10 popular neighbourhoods based on the number of listings.

7. Average Price by Neighbourhood:

- Displays the average price for each neighbourhood, sorted in descending order.

8. Price vs. Number of Reviews:

- Scatter plot to explore the relationship between listing price and the number of reviews.

9. Average Price by Neighbourhood Group and Room Type:

- Bar plot visualizing the average price for each neighbourhood group, categorized by room type.

10. Hosts with Longest and Shortest Minimum Nights:

- Identifies hosts with the longest and shortest minimum nights.

Visualizations:

- Various visualizations, including boxplots, histograms, scatter plots, and bar plots, are used to enhance the interpretation of the data.

3. Data Preparation

a. Do variables have missing/null values?

b. Do variables have outliers?

c. Is the data normally distributed? Is it a defect? Why or why not?

Handling Missing Values:

a. Variables with Missing/Null Values:

- Identifies variables with missing values and displays the count of missing values for each variable.

b. Handling Missing Values:

- A common method for handling missing values is demonstrated by dropping rows with missing values using **dropna()**. This approach may be suitable when the number of missing values is relatively small.

Handling Outliers:

- **Example: Capping Values Based on Z-Score**
  - Outliers are identified using z-scores, and rows with values beyond a certain threshold are excluded.

Checking for Normal Distribution:

- Histograms are plotted to visually inspect the distribution of numerical variables. This helps in understanding the data distribution and identifying potential skewness.

Addressing Non-Normal Distribution:

- **Example: Applying Log Transformation**
  - A log transformation is applied to numerical variables to address non-normal distribution and reduce the impact of outliers.

Handling Duplicates:

- **Duplicate Rows:**
  - Identifies and displays the count of duplicate rows in the dataset.

Checking Unique Values:

- **Example: Checking Unique Values in Categorical Columns**
  - Lists unique values for each categorical column to identify potential issues or inconsistencies.

Checking for Unrealistic Values or Outliers:

- **Example: Unexpected Patterns in 'availability_365'**
  - Identifies and displays rows where 'availability_365' is greater than 365, which might indicate an issue.

Checking for Date-Related Issues:

- **Example: Ensuring 'last_review' is in a Valid Date Format**
  - Converts the 'last_review' column to a datetime format, addressing potential date-related issues.

Handling Unique Values in Categorical Columns:

- **Example: Grouping Values with Low Frequency into 'Other'**
  - If a categorical column has too many unique values, values with low frequency are grouped into an 'Other' category.

Handling Unrealistic or Out-of-Range Values:

- **Example: Replacing Negative Values in 'minimum_nights' with Absolute Values**
  - Negative values in 'minimum_nights' are replaced with their absolute values.

Handling Unexpected Patterns:

- **Example: Replacing 'availability_365' Values Greater Than 365**
  - Values in 'availability_365' greater than 365 are replaced with 365.

Handling Date-Related Issues:

- **Example: Coercing 'last_review' to DateTime Format**
  - The 'last_review' column is coerced into a datetime format with errors handled using 'coerce'.

Handling Data Distributions:

- **Example: Applying Log Transformation to 'price'**
  - A custom log transformation function is applied to the 'price' variable to address skewness.

4. Summarize relationships among variables

a. **Plot correlation plots. Which are the variables most correlated with Target? Which independent variables are correlated among themselves? Do you want to exclude some variables from the model based on this analysis? What other actions will you take?**

The provided code conducts correlation analysis on the Airbnb dataset, focusing on understanding the relationships between variables, especially with the target variable 'price.' Here's an inference document summarizing the key steps and findings:

Correlation Matrix for All Variables:

- A heatmap of the correlation matrix is plotted for all variables (including the target variable 'price').
- The heatmap visually represents the strength and direction of linear relationships between different pairs of variables.

Variables Most Correlated with the Target ('price'):

- Correlation coefficients between 'price' and other numeric variables are calculated and displayed in descending order.
- High positive correlation coefficients indicate a positive linear relationship, while high negative coefficients indicate a negative linear relationship.

Correlation Matrix for Independent Variables:

- A heatmap of the correlation matrix is plotted for independent variables, excluding the target variable 'price.'
- This analysis focuses on relationships among independent variables to identify potential multicollinearity.

Exclusion of Highly Correlated Variables:

- A threshold is set for correlation coefficients to identify highly correlated variables.
- Variables with correlation coefficients exceeding the threshold are considered for exclusion.

Decision on Variable Exclusion:

- The example code suggests excluding variables highly correlated with the target variable 'price.'
- The threshold is set at 0.7, meaning variables with correlation coefficients greater than 0.7 or less than -0.7 with 'price' are considered highly correlated.

Inferred Decisions:

- The decision to exclude highly correlated variables depends on the chosen threshold and the specific requirements of the analysis.
- Variables with high correlations may introduce multicollinearity in regression models, affecting the model's interpretability and stability.

5. Fit a base model. Please write your key observations (5 marks)

a. Fit the Linear Regression Model

b. What is the overall R2? Please comment on whether it is good or not.

c. Which variables are significant?

Data Preparation:

- The dataset is split into features (X) and the target variable 'price' (y).
- The target variable is excluded from the feature set.

Handling Missing Values:

- Missing values in the feature set are imputed using the mean strategy provided by **SimpleImputer**.

Data Splitting:

- The data is split into training and testing sets using a test size of 20% and a random seed for reproducibility.

Linear Regression Modeling:

- A linear regression model is fitted using the training set (**X_train**, **y_train**).
- The model is used to make predictions on the test set (**X_test**), and the predicted values are stored in **y_pred**.

Model Evaluation:

- The overall performance of the model is evaluated using the coefficient of determination (R^2) on the test set.
- The R^2 value provides an indication of how well the model explains the variance in the target variable.

Results and Metrics:

- The overall R^2 value is calculated and printed, indicating the proportion of variance in the target variable that the model explains.

Variable Significance Analysis:

- The significance of each variable in the model is assessed using the **statsmodels** library.
- The summary statistics from the OLS (Ordinary Least Squares) regression results are printed, including coefficients, standard errors, t-values, and p-values.

Inferred Decisions:

- The R^2 value provides insights into the predictive performance of the linear regression model.
- The significance of each variable helps identify which features contribute significantly to predicting the target variable.

Inference Of Overall Model Fit:

Overall Model Fit:

The linear regression model was fitted to predict the 'price' variable based on selected independent variables. The overall R^2 value of 0.1225 indicates that the model explains approximately 12.25% of the variability in the 'price' variable. This suggests a limited ability of the model to capture and predict the observed variations. Variable Significance:

The statsmodels summary provides insights into the significance of individual variables. Variables with lower p-values are considered more statistically significant. Review these p-values to identify variables contributing significantly to the prediction of 'price'. High p-values for certain variables may indicate that they are not providing significant information and may be candidates for further analysis or exclusion. Coefficient Interpretation:

Examine the coefficients of significant variables to understand their impact on 'price'. A positive coefficient indicates a positive correlation with 'price', while a negative coefficient indicates a negative correlation. The magnitude of the coefficients reflects the strength of the relationship between each independent variable and the target variable. Assumption Checking:

Assess the assumptions of linear regression, including linearity, independence, homoscedasticity, and normality of residuals. Examine residuals for random patterns and systematic trends. Any deviations from assumptions should be addressed to enhance the model's reliability.

Output :

SML CT-2

Data_set:

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.

1. id = listing ID
2. name = name of the listing
3. host_id = host ID
4. host_name = name of the host
5. neighbourhood_group = location
6. neighbourhood = area
7. latitude = latitude coordinates
8. longitude = longitude coordinates
9. room_type = listing space type
10. price = price in dollars (Target variable)
11. minimum_nights = amount of nights minimum
12. number_of_reviews = number of reviews
13. last_review = latest review
14. reviews_per_month = number of reviews per month
15. calculated_host_listings_count = amount of listing per host
16. availability_365 = number of days when listing is available for booking

1. Read the dataset (tab, csv, xls, txt, inbuilt dataset)

In [1]:
```
# Kindly change the below cells from markdown to code and execute it
```

```
import pandas as pd

import csv

with open("data_set.csv","r")as file:

    reader=csv.reader(file)

df=pd.read_csv("data_set.csv")

df.head()
```

In [2]:
```
import numpy as np
import pandas as pd
import matplotlib
import matplotlib.pyplot as plt
import seaborn as sns
plt.style.use('fivethirtyeight')
matplotlib.rcParams['font.family'] = "Arial"
import collections
import itertools
import scipy.stats as stats
from scipy.stats import norm
from scipy.special import boxcox1p
import statsmodels
import statsmodels.api as sm
#print(statsmodels.__version__)
from sklearn.preprocessing import scale, StandardScaler, RobustScaler, OneHotEncode
from sklearn.compose import ColumnTransformer
from sklearn.model_selection import train_test_split, cross_val_score, GridSearchCV
from sklearn.linear_model import Ridge, RidgeCV, Lasso, LassoCV, LinearRegression,
from sklearn.metrics import mean_squared_error, r2_score
from sklearn.gaussian_process import GaussianProcessRegressor
from sklearn.ensemble import RandomForestRegressor, GradientBoostingRegressor
from sklearn.utils import resample
```

2. Data Understanding

Summarize important observations from the data set (5 Marks)

Some pointers which would help you, but don't be limited by these

a. What are the number of rows; no. & types of variables (continuous, categorical etc.)

b. Calculate five point summary for numerical variables

c. Summarize observations for categorical variables – no. of categories, % observations in each category

In [3]:
```
# Load the dataset
file_path = 'new_york_city_air.csv'
airbnb_data = pd.read_csv(file_path)

# a. Number of rows, number and types of variables
num_rows, num_cols = airbnb_data.shape
variable_types = airbnb_data.dtypes.value_counts()

print(f'a. Number of rows: {num_rows}')
print(f'   Number of columns: {num_cols}')
print(f'   Types of variables:\n{variable_types}')

# b. Five-point summary for numerical variables
numerical_summary = airbnb_data.describe()
print('\nb. Five-point summary for numerical variables:')
print(numerical_summary)
```

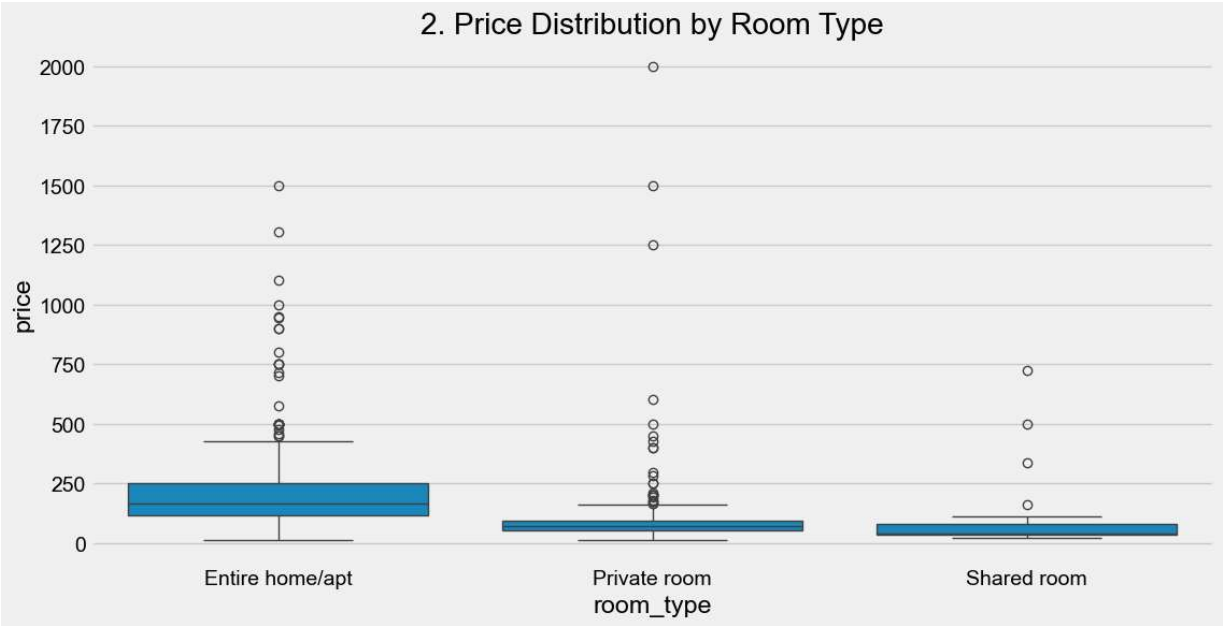
```
# c. Summary for categorical variables
categorical_summary = airbnb_data.select_dtypes(include='object').describe(include=
print('\nc. Summary for categorical variables:')
print(categorical_summary)

# Calculate the number of categories and % observations for each categorical variab
for column in airbnb_data.select_dtypes(include='object').columns:
    num_categories = airbnb_data[column].nunique()
    percentage_observation = airbnb_data[column].value_counts(normalize=True) * 100
    print(f'\nVariable: {column}')
    print(f'Number of categories: {num_categories}')
    print(f'Percentage of observations in each category:\n{percentage_observation}'

# Below are the analyses i have chosen

# 1. Host-Level Analysis
average_listings_per_host = airbnb_data.groupby('host_id')['id'].count().mean()
print(f'1. Average number of listings per host: {average_listings_per_host:.2f}')

# 2. Price Distribution by Room Type
plt.figure(figsize=(12, 6))
sns.boxplot(x='room_type', y='price', data=airbnb_data)
plt.title('2. Price Distribution by Room Type')
plt.show()

# 3. Availability by Room Type
average_availability_by_room_type = airbnb_data.groupby('room_type')['availability_
print(f'3. Average availability by room type:\n{average_availability_by_room_type}'

# 4. Hosts with Multiple Listings
multiple_listings_hosts = airbnb_data['host_id'].value_counts().reset_index()
multiple_listings_hosts.columns = ['host_id', 'num_listings']
multiple_listings_hosts = multiple_listings_hosts[multiple_listings_hosts['num_list
print(f'4. Hosts with Multiple Listings:\n{multiple_listings_hosts.head()}')

# 5. Distribution of Minimum Nights
plt.figure(figsize=(10, 6))
sns.histplot(airbnb_data['minimum_nights'], bins=30, kde=True)
plt.title('5. Distribution of Minimum Nights')
plt.xlabel('Minimum Nights')
plt.ylabel('Frequency')
plt.show()

# 6. Popular Neighbourhoods
popular_neighbourhoods = airbnb_data['neighbourhood'].value_counts().head(10)
print(f'6. Top 10 Popular Neighbourhoods:\n{popular_neighbourhoods}')

# 7. Average Price by Neighbourhood
average_price_by_neighbourhood = airbnb_data.groupby('neighbourhood')['price'].mean
print(f'7. Average Price by Neighbourhood:\n{average_price_by_neighbourhood}')

# 8. Price vs. Number of Reviews
plt.figure(figsize=(10, 6))
sns.scatterplot(x='price', y='number_of_reviews', data=airbnb_data)
plt.title('8. Price vs. Number of Reviews')
```

```
plt.show()

# 9. Average Price by Neighbourhood Group and Room Type
plt.figure(figsize=(12, 8))
sns.barplot(x='neighbourhood_group', y='price', hue='room_type', data=airbnb_data)
plt.title('9. Average Price by Neighbourhood Group and Room Type')
plt.show()

# 10. Hosts with Longest and Shortest Minimum Nights
longest_min_nights_hosts = airbnb_data.loc[airbnb_data.groupby('host_id')['minimum_
shortest_min_nights_hosts = airbnb_data.loc[airbnb_data.groupby('host_id')['minimum
print(f'10. Hosts with Longest Minimum Nights:\n{longest_min_nights_hosts.head()}')
print(f'   Hosts with Shortest Minimum Nights:\n{shortest_min_nights_hosts.head()}'
```

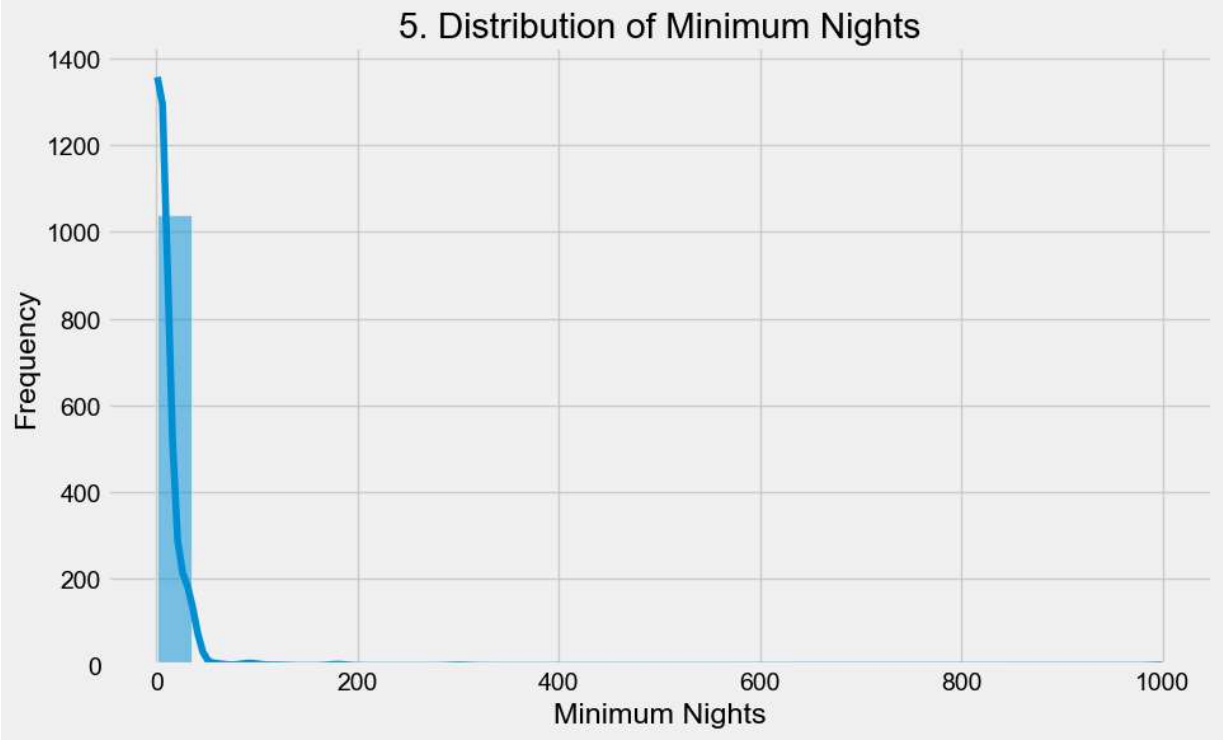
```
a. Number of rows: 1054
   Number of columns: 16
   Types of variables:
int64      7
object     6
float64    3
Name: count, dtype: int64

b. Five-point summary for numerical variables:
                 id       host_id     latitude    longitude        price  \
count  1.054000e+03  1.054000e+03  1054.000000  1054.000000  1054.000000
mean   1.911356e+07  6.652685e+07    40.728377   -73.951802   149.699241
std    1.096046e+07  7.757195e+07     0.056092     0.049034   156.239782
min    1.229900e+04  2.787000e+03    40.499790   -74.240840    10.000000
25%    9.391594e+06  6.983334e+06    40.689253   -73.983770    70.000000
50%    1.997263e+07  3.013617e+07    40.721665   -73.956930   104.000000
75%    2.929557e+07  1.074344e+08    40.761120   -73.935523   179.000000
max    3.648561e+07  2.722480e+08    40.898730   -73.731700  2000.000000

       minimum_nights  number_of_reviews  reviews_per_month  \
count     1054.000000        1054.000000         844.000000
mean         8.002846          24.620493           1.347690
std         34.580253          49.178939           1.675154
min          1.000000           0.000000           0.020000
25%          1.000000           1.000000           0.190000
50%          3.000000           6.000000           0.730000
75%          5.000000          24.000000           1.990000
max        999.000000         426.000000          15.320000

       calculated_host_listings_count  availability_365
count                     1054.000000       1054.000000
mean                         6.959203        111.572106
std                         31.266268        129.835933
min                          1.000000          0.000000
25%                          1.000000          0.000000
50%                          1.000000         43.500000
75%                          2.000000        223.750000
max                        327.000000        365.000000

c. Summary for categorical variables:
                                name host_name neighbourhood_group  \
count                           1053      1054                1054
unique                          1050       779                   5
top     New york Multi-unit building      Alex           Manhattan
freq                               2        13                 449

       neighbourhood        room_type last_review
count           1054             1054         844
unique           123                3         360
top     Williamsburg  Entire home/apt  01-07-2019
freq              92              539          32

Variable: name
Number of categories: 1050
Percentage of observations in each category:
name
```

```
New york Multi-unit building                          0.189934
Spacious and Bright Williamsburg Loft                 0.189934
New York Apartment                                    0.189934
Sunny shared apartment in Harlem                      0.094967
Perfect Cobble Hill 1-bed, Brooklyn's best spot       0.094967
                                                        ...
Sunny studio apt in heart of SoHo                     0.094967
Bright, Williamsburg bedroom with private balcony     0.094967
2 Queen Bed Spacious Room near Flatiron & Union Sq    0.094967
BEAUTIFUL SMALL BEDROOM/TIMES SQUARE/8 AVENUE         0.094967
Flex 2BR Loft ( 1br +Sleep Loft +Sofa Bed )!          0.094967
Name: proportion, Length: 1050, dtype: float64

Variable: host_name
Number of categories: 779
Percentage of observations in each category:
host_name
Alex               1.233397
Sarah              0.948767
Michael            0.664137
David              0.664137
Jennifer           0.664137
                     ...
Siwen              0.094877
Jolene And Ryan    0.094877
Antonia            0.094877
Rl                 0.094877
Vida               0.094877
Name: proportion, Length: 779, dtype: float64

Variable: neighbourhood_group
Number of categories: 5
Percentage of observations in each category:
neighbourhood_group
Manhattan        42.599620
Brooklyn         40.891841
Queens           12.333966
Bronx             3.225806
Staten Island     0.948767
Name: proportion, dtype: float64

Variable: neighbourhood
Number of categories: 123
Percentage of observations in each category:
neighbourhood
Williamsburg          8.728653
Bedford-Stuyvesant    7.210626
Bushwick              5.028463
Upper West Side       4.648956
Hell's Kitchen        4.364326
                        ...
Tribeca               0.094877
Kew Gardens           0.094877
Morris Park           0.094877
Randall Manor         0.094877
Huguenot              0.094877
```

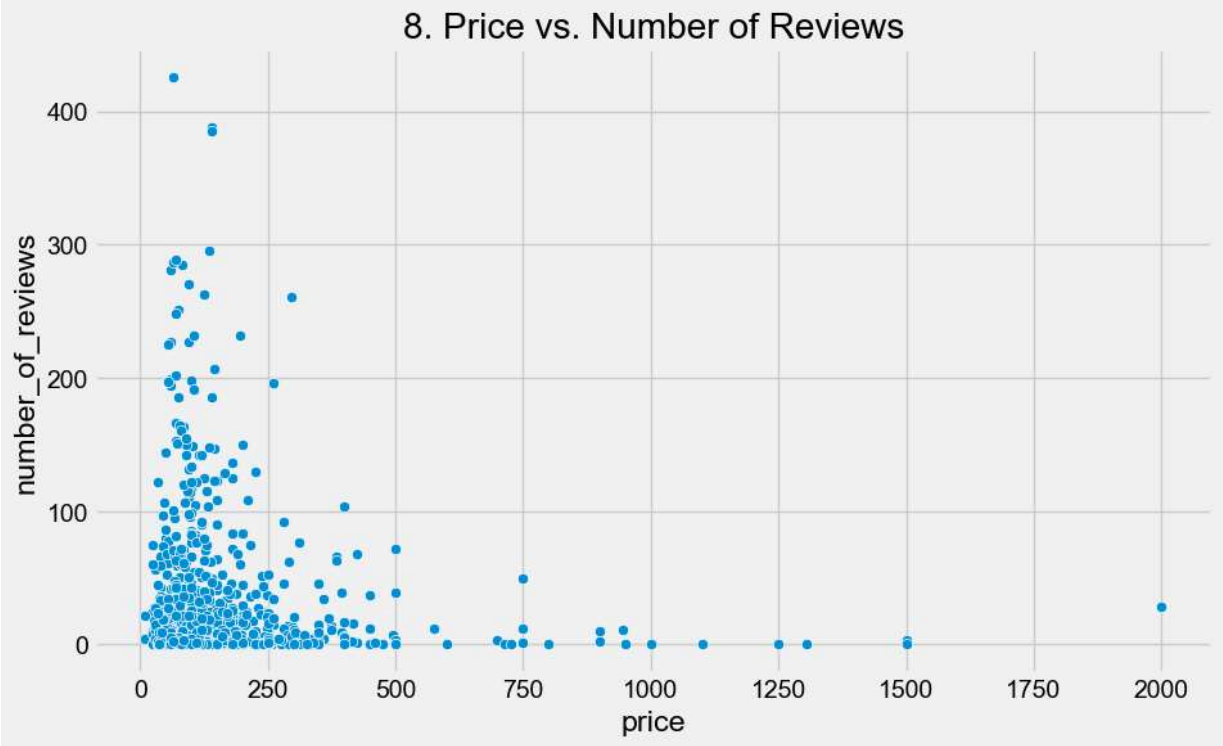
```
Name: proportion, Length: 123, dtype: float64

Variable: room_type
Number of categories: 3
Percentage of observations in each category:
room_type
Entire home/apt    51.13852
Private room       45.73055
Shared room         3.13093
Name: proportion, dtype: float64

Variable: last_review
Number of categories: 360
Percentage of observations in each category:
last_review
01-07-2019    3.791469
24-06-2019    3.436019
30-06-2019    3.199052
23-06-2019    3.080569
07-07-2019    2.962085
                ...
23-02-2019    0.118483
14-09-2016    0.118483
20-05-2019    0.118483
04-07-2016    0.118483
07-10-2018    0.118483
Name: proportion, Length: 360, dtype: float64
1. Average number of listings per host: 1.04
```

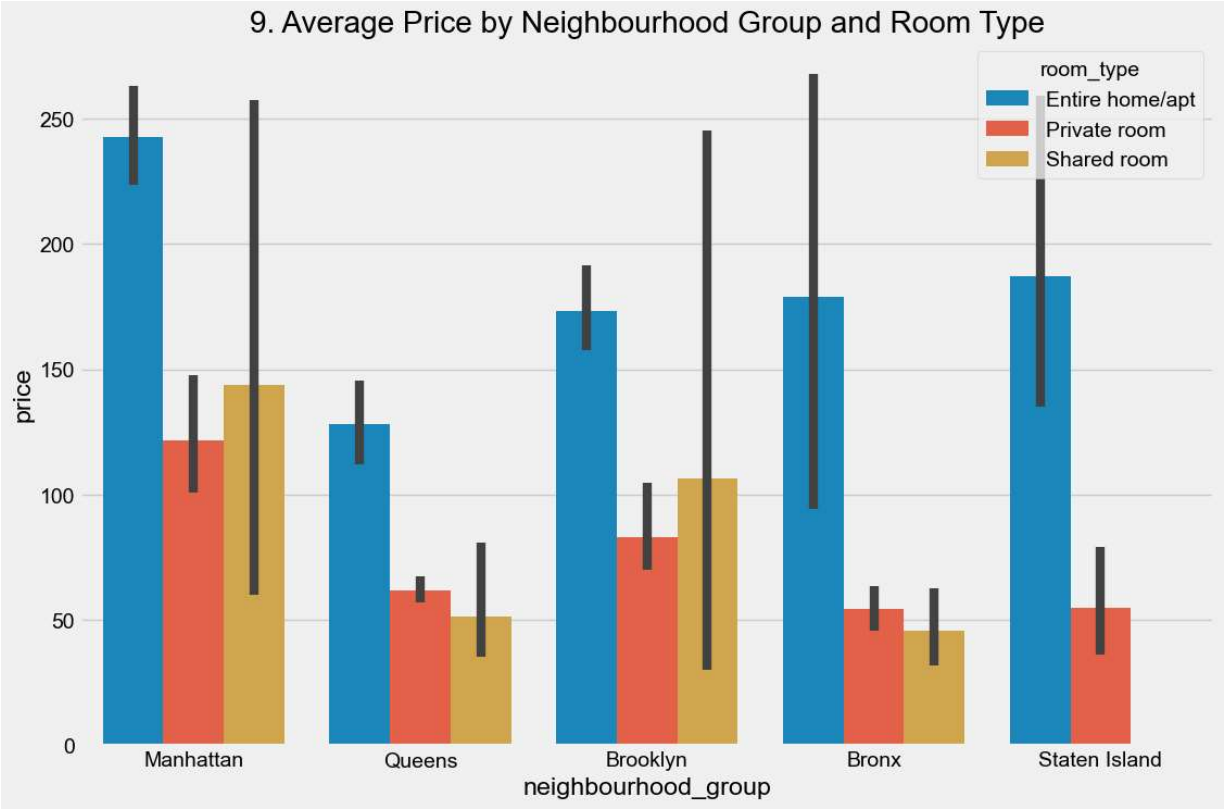


```
3. Average availability by room type:
room_type
Entire home/apt    112.892393
Private room       107.883817
Shared room        143.878788
Name: availability_365, dtype: float64
4. Hosts with Multiple Listings:
     host_id  num_listings
0  219517861             6
1  107434423             5
2    5162192             3
3  137358866             3
4   22541573             3
```



```
6. Top 10 Popular Neighbourhoods:
neighbourhood
Williamsburg          92
Bedford-Stuyvesant    76
Bushwick              53
Upper West Side       49
Hell's Kitchen        46
Harlem                43
East Village          36
Midtown               36
Lower East Side       31
Crown Heights         29
Name: count, dtype: int64
7. Average Price by Neighbourhood:
neighbourhood
Civic Center         950.0
Tribeca              750.0
Greenwich Village    395.5
Eastchester          387.0
NoHo                 330.0
                     ...
Woodlawn              34.0
Arrochar              32.0
Port Morris           28.0
Olinville             25.0
St. Albans            25.0
Name: price, Length: 123, dtype: float64
```




9. Average Price by Neighbourhood Group and Room Type
room_type
Entire home/apt
Private room
Shared room
price
250
200
150
100
50
0
Manhattan
Queens
Brooklyn
Bronx
Staten Island
neighbourhood_group

```
10. Hosts with Longest Minimum Nights:
          id                                                name  host_id  \
7    10267242  Cinque Terre Room.  Clean and Quiet Queen Bedroom     2787
559    246916                 Quality Cozy Studio Next to Subway     3647
618  19532789  NEWLY Renovated Private 2 Bed Apt on 1 st Floor..    15192
184   9603803                                   Brooklyn Charm    23193
712    127387                  Luxe, Spacious 2BR 2BA Nr Trains    23276

     host_name neighbourhood_group       neighbourhood  latitude  longitude  \
7         John            Brooklyn           Gravesend  40.60810  -73.97541
559     Rafael              Queens            Elmhurst  40.73470  -73.88066
618      Oscar            Brooklyn  Bedford-Stuyvesant  40.68268  -73.94548
184      Paris            Brooklyn  Bedford-Stuyvesant  40.68777  -73.93870
712  Katharine            Brooklyn             Gowanus  40.66862  -73.99260

           room_type  price  minimum_nights  number_of_reviews last_review  \
7       Private room    149               1                 24  11-05-2019
559  Entire home/apt     79               4                 60  25-06-2019
618  Entire home/apt     95               4                 33  22-06-2019
184     Private room     40               2                  5  08-04-2019
712  Entire home/apt    260              30                  3  04-08-2014

     reviews_per_month  calculated_host_listings_count  availability_365
7                 0.64                               6               180
559               0.64                               2               297
618               1.40                               1               199
184               0.12                               4                87
712               0.03                               1               316
   Hosts with Shortest Minimum Nights:
          id                                                name  host_id  \
7    10267242  Cinque Terre Room.  Clean and Quiet Queen Bedroom     2787
559    246916                 Quality Cozy Studio Next to Subway     3647
618  19532789  NEWLY Renovated Private 2 Bed Apt on 1 st Floor..    15192
184   9603803                                   Brooklyn Charm    23193
712    127387                  Luxe, Spacious 2BR 2BA Nr Trains    23276

     host_name neighbourhood_group       neighbourhood  latitude  longitude  \
7         John            Brooklyn           Gravesend  40.60810  -73.97541
559     Rafael              Queens            Elmhurst  40.73470  -73.88066
618      Oscar            Brooklyn  Bedford-Stuyvesant  40.68268  -73.94548
184      Paris            Brooklyn  Bedford-Stuyvesant  40.68777  -73.93870
712  Katharine            Brooklyn             Gowanus  40.66862  -73.99260

           room_type  price  minimum_nights  number_of_reviews last_review  \
7       Private room    149               1                 24  11-05-2019
559  Entire home/apt     79               4                 60  25-06-2019
618  Entire home/apt     95               4                 33  22-06-2019
184     Private room     40               2                  5  08-04-2019
712  Entire home/apt    260              30                  3  04-08-2014

     reviews_per_month  calculated_host_listings_count  availability_365
7                 0.64                               6               180
559               0.64                               2               297
618               1.40                               1               199
184               0.12                               4                87
712               0.03                               1               316
```

3. Data Preparation

Check for defects in the data. Perform necessary actions to 'fix' these defects (5 Marks)

Some pointers which would help you, but don't be limited by these

a. Do variables have missing/null values?

b. Do variables have outliers?

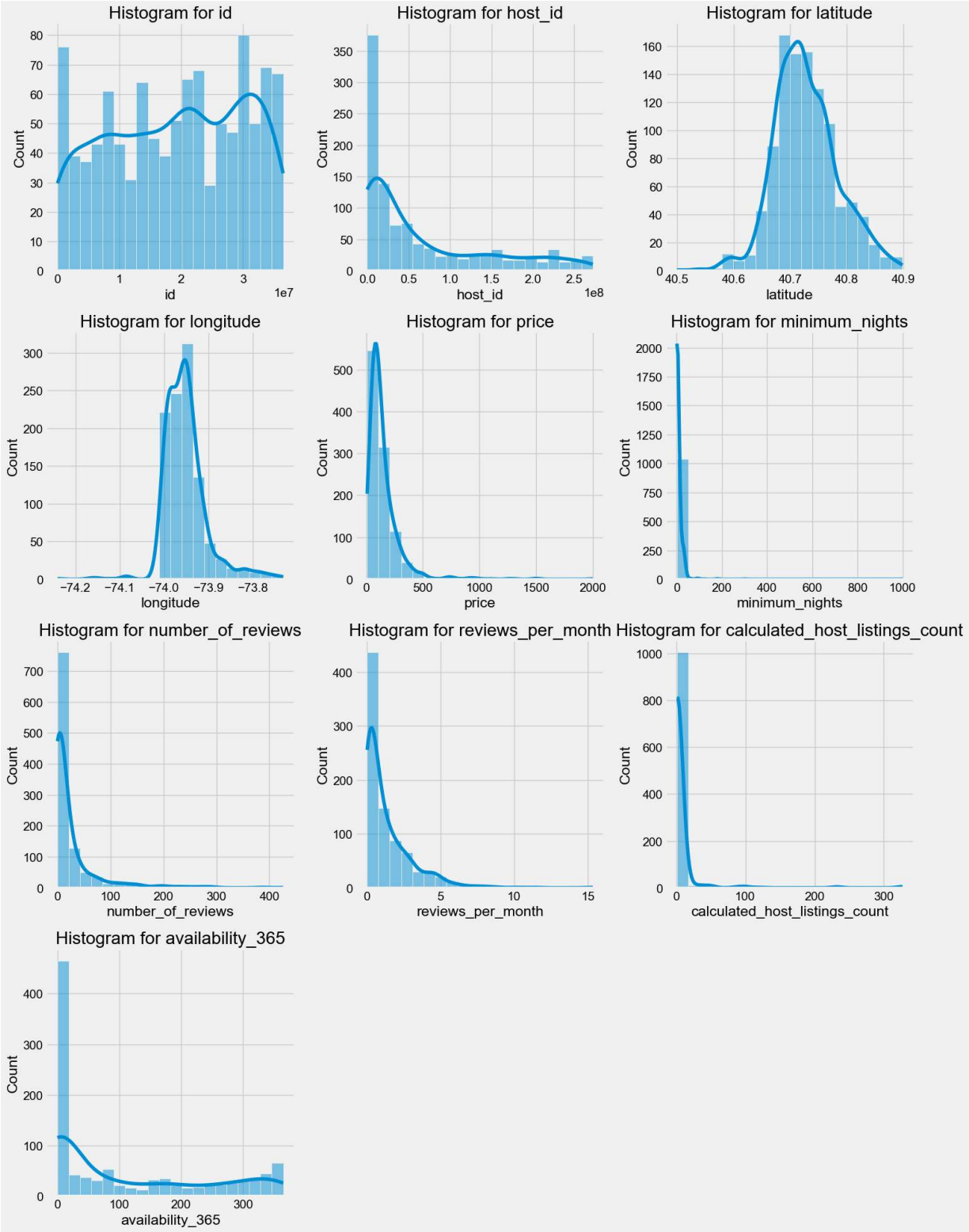
c. Is the data normally distributed? Is it a defect? Why or why not?

In [4]:
```
# Check for missing/null values
missing_values = airbnb_data.isnull().sum()
print("a. Variables with Missing/Null Values:")
print(missing_values[missing_values > 0])

# Handling missing values (Example: Drop rows with missing values)
airbnb_data_cleaned = airbnb_data.dropna()

# Handling outliers (Example: Cap values based on z-score)
from scipy.stats import zscore

z_scores = zscore(airbnb_data.select_dtypes(include=['float64', 'int64']))
airbnb_data_no_outliers = airbnb_data[(abs(z_scores) < 3).all(axis=1)]

# Checking for normal distribution using histograms
numerical_columns = airbnb_data.select_dtypes(include=['float64', 'int64']).columns

num_plots = len(numerical_columns)
num_cols = 3  # Adjust the number of columns as needed
num_rows = (num_plots - 1) // num_cols + 1

plt.figure(figsize=(15, 5 * num_rows))
for i, column in enumerate(numerical_columns, 1):
    plt.subplot(num_rows, num_cols, i)
    sns.histplot(airbnb_data[column], bins=20, kde=True)
    plt.title(f'Histogram for {column}')

plt.tight_layout()
plt.show()


# Addressing non-normal distribution (Example: Applying log transformation)
airbnb_data_log_transformed = airbnb_data.copy()

# Use numpy vectorized operations to apply log transformation
airbnb_data_log_transformed[numerical_columns] = airbnb_data_log_transformed[numeri



# Check for duplicates
```

```
duplicate_rows = airbnb_data.duplicated().sum()
print("\nDuplicate Rows:", duplicate_rows)

# Remove duplicate rows
airbnb_data_no_duplicates = airbnb_data.drop_duplicates()


# Check for unique values in categorical columns
categorical_columns = airbnb_data.select_dtypes(include='object').columns
for column in categorical_columns:
    unique_values = airbnb_data[column].unique()
    print(f"\nUnique values in {column}:\n{unique_values}")

# Check for unrealistic values or outliers in non-numeric columns
non_numeric_columns = airbnb_data.select_dtypes(exclude=['float64', 'int64']).colum
for column in non_numeric_columns:
    unique_values = airbnb_data[column].unique()
    print(f"\nUnique values in {column}:\n{unique_values}")

# Check for unexpected patterns or inconsistencies
# Example: Check if 'availability_365' is greater than 365
unexpected_patterns = airbnb_data[airbnb_data['availability_365'] > 365]
print("\nUnexpected Patterns in 'availability_365':")
print(unexpected_patterns)

# Check for date-related issues
# Example: Ensure 'last_review' is in a valid date format
airbnb_data['last_review'] = pd.to_datetime(airbnb_data['last_review'], errors='coe

# Check for data distributions
# Example: Plot histograms for numerical variables
airbnb_data.hist(figsize=(12, 10), bins=20)
plt.suptitle('Histograms for Numerical Variables', y=0.92)
plt.show()

# Handling Unique Values in Categorical Columns
# Example: If too many unique values, consider grouping or transforming
categorical_columns = airbnb_data.select_dtypes(include='object').columns
for column in categorical_columns:
    unique_values_count = airbnb_data[column].nunique()
    if unique_values_count > 10:
        # Example: Grouping values with low frequency into 'Other'
        value_counts = airbnb_data[column].value_counts()
        low_frequency_values = value_counts[value_counts < 10].index
        airbnb_data[column] = airbnb_data[column].replace(low_frequency_values, 'Ot

# Handling Unrealistic or Out-of-Range Values
# Example: Replace negative values in 'minimum_nights' with their absolute values
airbnb_data['minimum_nights'] = airbnb_data['minimum_nights'].abs()

# Handling Unexpected Patterns
# Example: If 'availability_365' is greater than 365, replace with 365
airbnb_data['availability_365'] = airbnb_data['availability_365'].apply(lambda x: 3

# Handling Date-Related Issues
# Example: Convert 'last_review' to datetime format with errors='coerce'
```

```
# Explicitly specify the date format
airbnb_data['last_review'] = pd.to_datetime(airbnb_data['last_review'], errors='coe


# Handling Data Distributions
# Example: Apply log transformation to 'price' to address skewness
airbnb_data_log_transformed = airbnb_data.copy()
def log_transform(x):
    if pd.notna(x) and x > 0:
        return np.log1p(x)
    else:
        return 0

airbnb_data_log_transformed[numerical_columns] = airbnb_data_log_transformed[numeri



# Checking the changes
print("Updated Dataset:")
print(airbnb_data.head())
```

```
a. Variables with Missing/Null Values:
name                   1
last_review          210
reviews_per_month    210
dtype: int64
```




```
C:\Users\vigne\AppData\Local\Temp\ipykernel_10468\2683071354.py:36: FutureWarning: D
ataFrame.applymap has been deprecated. Use DataFrame.map instead.
  airbnb_data_log_transformed[numerical_columns] = airbnb_data_log_transformed[numer
ical_columns].applymap(np.log1p)
C:\Users\vigne\AppData\Roaming\Python\Python312\site-packages\pandas\core\algorithm
s.py:1814: RuntimeWarning: invalid value encountered in log1p
  return lib.map_infer(values, mapper, convert=convert)
```

```
Duplicate Rows: 0

Unique values in name:
['Spacious 4bdrm next to CentralPk'
 'Great room in amazing Little Italy location!'
 'Clean Cozy Room, Queens-5 min walk to subway (R/M)' ...
 'Nice and warm room with a Queen Size Bed!!' 'Violet Dynasty'
 'Flex 2BR Loft ( 1br +Sleep Loft +Sofa Bed )!']

Unique values in host_name:
['Amy' 'Nathan' 'Bei' 'Ryan' 'Derek' 'Kathy' 'Luis Alejandro' 'John'
 'Joseph' 'Niki' 'Ken' 'Randy' 'Ame' 'Victor' 'J' 'Roderick' 'Raanan'
 'Olivier' 'Oxana' 'Raymond' 'Dadrine' 'Zeleke' 'Kay' 'Mark & Will' 'Eric'
 'Chris And Jamie' 'Noel' 'Ofer' 'Brene' 'Swati' 'Erik' 'Teddy' 'Marilyn'
 'Sandra' 'Andy' 'Sarah' 'Guy' 'Leslie' 'Août' 'Regina' 'Blueground'
 'Thomas' 'Janae' 'Gen' 'Nati' 'Linda' 'David' 'Nicole' 'Sebastian'
 'Hannah' 'Josue' 'Yael & Graham' 'Dennis' 'Bobbi' 'Evan' 'Fmny' 'Moe'
 'Ange' 'Jonathan' 'Tyler' 'Verena' 'Joe' 'Sergii' 'Lilian' 'Alice'
 'Matthew' 'Lisa' 'Ute' 'Edgar' 'Kestrel' 'Yvette' 'William M.' 'Melissa'
 'Alexes' 'Martin' 'Belinda' 'Al' 'Bryan' 'Ekin' 'Elena' 'Carl' 'Rhian'
 'Joanne' 'Diana' 'Vinny' 'Jennifer' 'Alexandre' 'Shane' 'Sonder' 'Karin'
 'Tanya' 'Will AND Kelly' 'Conor' 'Larissa' "Shully'S" 'Shahzad'
 'Amarjit S' 'Deborah' 'Nelle' 'Marc' 'Vian' 'Juliana' 'Carlo' 'Abir'
 'Susan' 'Habib' 'Miguel' 'Madison' 'Chris' 'Savannah' 'Zhanhong' 'Denise'
 'The Bowery House' 'Alex' 'Shogo' 'Andre' 'Arthur' 'Abraham' 'Ruchi'
 'Chloé' 'Morgan' 'Jacqueline' 'Nimmi' 'Priscilla' 'The Box House Hotel'
 'Mikyla' 'Ana' 'Rebecca' 'Paul' 'Liza' 'Yuchen' 'Hudson River Hotel'
 'Chanel' 'Pablo' 'Elva' 'Crystal' 'Mansi' 'Diane & Brice' 'Shanna'
 'Amanda' 'Tom' 'Dylan' 'Erin' 'Zac' 'Evgenia' 'Ursula' 'Andrea' 'Godfrey'
 'George' 'Antina' 'Yuval' 'Eddie&Vlad' 'Tony' 'Juice' 'Jotham' 'Jj'
 'Saquib' 'Stella' 'Chadanut' 'Dee' 'Julio' 'Sylvia' 'Kat' 'Christina'
 'Tristan' 'Host' 'Helen' 'Rosine' 'Rony' 'Megan' 'Nadine'
 'Lory And Cindy' 'Newyorkroomwithaview' 'Sayed' 'Red Awning' 'Paris'
 'Charles' 'Carlos' 'Julia' 'Kyle' 'Max' 'Haley' 'Ashley' 'Drica'
 'Benjamin' 'Michael' 'Kristin' 'Sara' 'A.J.' 'Vanessa' 'Tatiana' 'Jack'
 'Zahavah Cara' 'Antonieta' 'Vishnu' 'Kris' 'Anissa' 'Rasmus'
 'Christine & Philip' 'Mikael' 'MaElena' 'Nataly' 'Katie' 'Monica' 'Anna'
 'Duryea' 'Eli And Maggie' 'Anthony' 'Nkosi' 'Madeleine' 'Geri'
 'Anastasia' 'Antony' 'Abena' 'Andria' 'Edwin' 'Anna & Frank' 'Jonny'
 'Lexy' 'Ally' 'Garret' 'Jun Jun' 'Dov' 'Mark' 'Brock' 'Kyla' 'Giselle'
 'Stephen' 'Tess' 'Inna' 'Daria' 'Matei' 'Stanley' 'Bobby' 'Saad' 'Lina'
 'Athena' 'Karen' 'Luisa' 'Jordan' 'Andrew' 'Mandi' 'Idemudia' 'H\u2006Ai'
 'Yair' 'Simranjeet' 'Mona' 'K' 'Haijiao' 'Josandra' 'Luigi' 'Isaac'
 'Rupi' 'Rachel' 'Valeria' 'Shaina' 'Siyu' 'Lali' 'Karma' 'Robert' 'Jeff'
 'Angela' 'Didi' 'Bernardo & Andressa' 'Gaia&Pietro' 'Lena' 'Mercedes'
 'Cody' 'Yin' 'Claudia' 'Paige' 'Filippo' 'Matilda' 'Jaqui And Mark'
 'Anna Laura' 'Emilia' 'Amaya' 'Owen' 'Misha' 'Prosper' 'Cem' 'Ishmael'
 'Yaakov' 'Raj' 'Paola' 'Jeremy & Laura' 'Carmen'
 'LuxuryApartmentsByAmber' 'Danilo' 'Syeda' 'Alexandra' 'Yaacov' 'Annette'
 'Laine' 'Carrie' 'Eileen' 'Justin' 'Maria' 'Joy' 'Babajide' 'Park Lane'
 'Kara' 'Armando' 'Nancis' 'Vlad' 'Harry & Morgan' 'Phillip' 'Jay' 'Janet'
 'Inieta' 'Vie' 'Freda' 'Patrick' 'Jing' 'Brian' 'Siwen' 'Elizabeth'
 'Jolene And Ryan' 'Antonia' 'Rl' 'Will' 'JuVita' 'Auset' 'Cindy' 'Mischa'
 'Royalton Park Avenue' 'Katherine' 'Roberto' 'Leah' 'Aryuna' 'Analia'
 'Gwendolyn' 'Alix' 'Kellie' 'Arsenio' 'Elvira' 'Austin' 'Aaron' 'Mike'
 'Alesandra' 'Michele' 'Kristina' 'Georgia' 'Andreas Per Daniel'
```

```
'Caroline' 'Mas' 'Mateo And Anna' 'E. Adam' 'Diego' 'Sally' 'Laura'
'Lorraine' 'Raquel' 'Jessica' 'Tammie' 'Ulrika' 'Meng' 'Ale' 'Katarina'
'Spice' 'Patricia' 'Koji & H' 'Lakshmee' 'Oliver' 'Xenios' 'Omar' 'D'
'Dominique' 'Lia' 'Loli' 'Tak' 'Layla' 'Kevin' 'Lolita' 'Frank' 'Olena'
'Kaya' 'Dmitry' 'Ben' 'Harvey' 'Lea' 'Geo' 'Matt' 'Philippe'
'Sonder (NYC)' 'Jérôme' 'Pamilitta' 'Anne' 'Charlotte & Brad' 'Mostafa'
'Jeffer' 'Yotel' 'Cassia' 'Spencer' 'Heath' 'Jason' 'Easton' 'Maureen'
'Domonique' 'Pj' 'James' 'Adrian' 'K. Naomi' 'Renato' 'Neil' 'Julie'
'Ali' 'Brigitte' 'Ramon' 'Sade' 'Phyl' 'Liz And Cesar' 'Liset' 'Kaylan'
'Rory' 'Dawn' 'Daniel' 'Jfk' 'Joica' 'Judith' 'Marlon' 'Louie' 'Kiki'
'Delia' 'Kynneth' 'Yara' 'Jastine' 'Justin & Lilo' 'Jamie And John'
'Simone' 'Marina' 'Lila & Paul' 'Abe' 'Craig' 'Alexa' 'Hayes' 'Cherie'
'Sophia' 'Cihan' 'Dai' 'Donna' 'Hugo' 'Claudina' 'Chin-Feng' 'Luo'
'Carlyn' 'Fernando' 'Bedly Bushwick' 'Zack' 'Abdul' 'Kimberlee' 'Nichole'
'Tara' 'Smidty' 'Richard' 'Courtney' 'Ari' 'Lauren' 'Fone' 'DeLex' 'Oh'
'Rafael' 'Stephanie' 'Charis' 'Tommy' 'George Steven' 'Caitlin' 'Alicia'
'Roger' 'Shahar' 'Shannon' 'J.L.' 'Shen' 'Sonia' 'Sherry  W  X'
'Zora & Chris' 'Taylor' 'Assaf' 'Ross' 'Hiroki' 'Tessa' 'Jess' 'Lucyna'
'Dana' 'Horatio' 'Eva' 'Brandon' 'Xi' 'LeAnne And Keith' 'Jade'
'The Manhattan Club' 'Ian' 'Shany' 'Zahir' 'Michelle' 'Virginia' 'Sonya'
'Peter' 'Bruce & Cyrina' 'Mili' 'Red' 'Dan And Lily' 'Janina' 'Kristy'
'Oscar' 'Marwa' 'Izi' 'Juan' 'Molly' 'Nick' 'Sam' 'Nina' 'Pooja' 'Gus'
'Sheila' 'Kristopher' 'Emmanuelle' 'Alexis' 'Alina' 'Johnny' 'Nadia'
'Ethan' 'Rated' 'Joel' 'Jessy & Grant' 'Mary' 'Dan' 'Diane'
'Weston & Gabby' 'Lauren & Chelsea' 'Olivia' 'Scott' 'Yogi' 'Victoria'
'Marco' 'Ismael' 'Orit' 'Shelby' 'Nikki' 'Zandelle' 'Artem' 'Anabella'
'Hongye' 'Nataliya' 'Mj' 'R' 'Russ' 'Alfonso' 'Moran' 'Allie' 'Kdn'
'Alicia And Zach' 'Grant' 'Shelley' 'Em' 'Pam' 'Lindsay' 'Felipe'
'Ticemen' 'Nate' 'Caspar' 'Katharine' 'Esteem' 'Oz' 'Carol' 'Jenny'
'Jimmy' 'Christos' 'Kadeen' 'Isiah' 'Shera' 'Robbin' 'Ed' 'Chloe'
'Chaydha' 'Paulette' 'Gonzalo' 'Zakaria' 'Cameron' 'Imri' 'Estie' 'Roni'
'Adam' 'Piero' 'Steven And John' 'Debra' 'Michael And Brooke' 'Cassy'
'Rocco' 'Cara' 'Kartik' 'Kazuya' 'Виль' 'Michka' 'Neal' 'Christopher'
'June' 'Keith' 'Pasha' 'Dario' 'Amia' 'Afrah' 'Daniel And Elsa' 'Giulia'
'Tonya' 'Joshua' 'Mac' 'Tonie' 'Roy' 'Erina' 'Beth' 'Oded'
'Timothy + Gina' 'Lukasz' 'Khamis' 'Xue' 'Melanie' 'Angie' 'Pol' 'Corrie'
'Vicki' 'Gabriel' 'Suzan' 'Sasha' 'The James New York - SoHo'
'Mary And Ryan' 'Emily' 'Septina' 'Kathryn' 'Clinton' 'Brad & Rachel'
'Niko' 'Neena' 'Mo' 'Naz' 'Samera' 'Jana' 'Isabel' 'Liz' 'Ira' 'Lenur'
'Leecia' 'Jeniffer' 'Pedro Pablo' 'Dellasie' 'Sofiya' 'Halima' 'Elana'
'Caroline & Georg' 'Karina' 'Hernan' 'Rhonda' 'Chantelle' 'Edmarine'
'Fatou' 'Tim' 'Jeremy' 'Sheri' 'Vincent' 'Mick' 'Steve' 'Wesly/Jessica'
'Jen' 'Isa' 'Audrey' 'Rahul' 'Marino' 'Alan' 'Shimin' 'Billy' 'Rachad'
'Wilson' 'Clara' 'Rémy' 'Benedetta' 'Kevin + Casey' 'Parnell' 'Judy'
'Luiz' 'Heidi' 'Angely' 'Bradley' 'Cathy' 'Amelia' 'Xiaoxia' 'Ruth'
'Ronnie' 'Mohammad' 'Brett' 'Renette' 'Heejun' 'Chun' 'Mel' 'Liliana'
'Margaret And Orville' 'Antonio' 'Zarrar' 'Graham And Ben' 'Cristina'
'Philip' 'Mauro' 'Del' 'Aura Angelica' 'Tiiu' 'Fredrik' 'Chimme'
'Jason & Mary' 'Sharmilli' 'Jamie' 'Walter' 'Joao' 'Gabrielle & Malcolm'
'Roxy' 'Jon' 'Julien' 'Yulia' 'Jeffrey' 'Roxanne' 'R.Henry' 'Thaddeus'
'Sonam' 'Elin' 'Francesca' 'Christine' 'Shawn' 'Eli' 'Jada'
'Brenda And Tim' 'Ash' 'Wendy' 'Ioana' 'Billye And Joe' 'Yohana'
'Enrique' 'Gal' 'Taz' 'Yvonne' 'Dina' 'Betty' 'Lola' 'Hector' 'Deni'
'Daniella' 'Dez' 'Evelyn' 'Cassandra' 'Lorenzo' 'Tarirai' 'Agnes' 'Vered'
'Andrzej' 'Lian' 'Chy' 'Dustin' 'Nyesha' 'Yoav' 'Filipe'
'Lio & Kim And Yotam' 'Shana' 'Dearna' 'Haruhisa' 'Kana' 'Egon'
```

```
 'Kathleen' 'Kip' 'Cannon' 'T' 'Joao Lucas' 'Baktiar' 'Sara & Kevin'
 'Iryna' 'Rayquila' 'Steven' 'Danny' 'Ikrame' 'Victorine' 'Esther' 'Shean'
 'Vida']

Unique values in neighbourhood_group:
['Manhattan' 'Queens' 'Brooklyn' 'Bronx' 'Staten Island']

Unique values in neighbourhood:
['Upper West Side' 'Little Italy' 'Rego Park' 'Woodside'
 'Bedford-Stuyvesant' 'East Village' 'Gravesend' 'Glendale' 'Midtown'
 'Chelsea' 'Park Slope' 'Harlem' 'SoHo' 'Bushwick' 'Sheepshead Bay'
 'East Flatbush' 'Boerum Hill' 'Crown Heights' "Hell's Kitchen"
 'Williamsburg' 'East Harlem' 'Richmond Hill' 'Kingsbridge'
 'Upper East Side' 'Kips Bay' 'Flushing' 'City Island' 'Murray Hill'
 'Bensonhurst' 'Financial District' 'Clinton Hill' 'Gowanus' 'Woodhaven'
 'Prospect-Lefferts Gardens' 'Mott Haven' 'East New York'
 'Lower East Side' 'Washington Heights' 'West Village' 'Astoria'
 'Wakefield' 'Prospect Heights' 'Arrochar' 'Fort Greene' 'Nolita'
 'Fieldston' 'Chinatown' 'Windsor Terrace' 'Claremont Village' 'Ridgewood'
 'Corona' 'Greenpoint' 'Jamaica' 'Long Island City' 'Greenwich Village'
 'Theater District' 'Morrisania' 'South Slope' 'Sunset Park' 'Kensington'
 'Fordham' 'St. George' 'Two Bridges' 'Arverne' 'Morningside Heights'
 'Sunnyside' 'Throgs Neck' 'Rockaway Beach' 'Flatbush' 'Jackson Heights'
 'Concord' 'East Elmhurst' 'Forest Hills' 'Morris Park' 'Cypress Hills'
 'Concourse Village' 'Inwood' 'Rosedale' 'Eastchester' 'Ditmars Steinway'
 'Longwood' 'Brownsville' 'Gramercy' 'Clason Point' 'Canarsie'
 'Parkchester' 'Downtown Brooklyn' 'Norwood' 'Eltingville' 'Bayswater'
 'Springfield Gardens' 'Brighton Beach' 'New Springville'
 'Flatiron District' 'Elmhurst' 'Tribeca' 'Kew Gardens' 'NoHo'
 'Randall Manor' 'Borough Park' 'Ozone Park' 'Carroll Gardens'
 'Port Morris' 'Far Rockaway' 'Cobble Hill' 'Civic Center' 'Navy Yard'
 'Highbridge' 'Olinville' 'Woodlawn' 'Stuyvesant Town' 'Middle Village'
 'Kew Gardens Hills' 'Red Hook' 'Queens Village' 'Bayside' 'Whitestone'
 'Vinegar Hill' 'Westchester Square' 'Schuylerville' 'Tottenville'
 'St. Albans' 'Huguenot']

Unique values in room_type:
['Entire home/apt' 'Private room' 'Shared room']

Unique values in last_review:
['09-01-2019' '14-09-2016' '22-06-2019' '08-08-2016' nan '14-06-2019'
 '02-08-2015' '11-05-2019' '15-06-2019' '18-06-2019' '23-06-2019'
 '30-12-2018' '14-08-2018' '29-06-2019' '18-04-2019' '05-07-2019'
 '02-07-2019' '01-07-2019' '04-06-2019' '24-06-2019' '02-01-2015'
 '19-06-2018' '13-07-2016' '30-06-2019' '03-03-2019' '22-09-2016'
 '02-01-2017' '12-11-2017' '28-07-2018' '27-06-2019' '11-06-2019'
 '31-12-2018' '30-03-2019' '13-02-2018' '19-01-2019' '25-09-2015'
 '21-10-2018' '01-06-2019' '01-01-2017' '21-06-2019' '08-12-2017'
 '02-06-2019' '28-06-2019' '07-10-2015' '28-03-2019' '10-12-2017'
 '19-05-2019' '31-05-2019' '20-12-2015' '18-05-2019' '19-06-2019'
 '09-06-2018' '20-07-2016' '07-06-2019' '03-04-2019' '01-12-2018'
 '07-07-2019' '20-11-2017' '04-07-2019' '03-01-2017' '05-05-2019'
 '17-04-2016' '08-06-2019' '05-01-2019' '07-08-2016' '31-01-2019'
 '19-02-2018' '25-06-2019' '16-07-2016' '02-03-2019' '19-03-2019'
 '16-03-2019' '28-05-2019' '28-04-2019' '09-06-2019' '13-09-2016'
 '29-09-2018' '12-05-2011' '23-12-2018' '03-07-2019' '22-02-2015'
```

```
'19-05-2015' '09-09-2017' '17-11-2015' '26-06-2019' '13-08-2016'
'20-10-2018' '03-06-2019' '03-01-2015' '09-11-2015' '21-06-2015'
'28-05-2016' '29-08-2017' '06-05-2018' '25-09-2016' '10-11-2015'
'08-01-2019' '28-10-2018' '17-06-2019' '09-09-2018' '19-09-2017'
'13-06-2019' '08-04-2019' '16-02-2019' '02-01-2019' '03-08-2018'
'30-09-2016' '22-05-2017' '20-06-2019' '08-05-2017' '30-04-2019'
'03-10-2015' '27-05-2019' '18-08-2016' '11-03-2019' '10-05-2019'
'01-01-2019' '22-04-2018' '20-04-2019' '17-10-2014' '06-06-2019'
'19-06-2017' '31-03-2019' '29-04-2019' '12-06-2019' '30-03-2015'
'22-10-2018' '16-06-2019' '06-05-2019' '23-05-2019' '06-07-2019'
'04-08-2018' '12-07-2018' '05-08-2018' '06-01-2016' '16-11-2016'
'09-05-2019' '16-05-2019' '28-08-2018' '12-05-2019' '01-08-2016'
'29-10-2018' '23-04-2019' '24-04-2018' '14-08-2016' '19-12-2017'
'21-09-2018' '26-09-2017' '14-09-2018' '22-12-2018' '27-04-2019'
'31-08-2017' '24-04-2019' '11-06-2018' '31-08-2018' '11-08-2018'
'10-12-2018' '10-06-2019' '26-04-2019' '24-11-2018' '05-06-2019'
'21-05-2019' '24-02-2019' '25-03-2019' '23-07-2016' '31-12-2015'
'16-09-2018' '29-08-2018' '01-11-2018' '26-11-2017' '02-10-2018'
'16-12-2016' '22-08-2016' '02-05-2017' '06-03-2019' '02-11-2018'
'17-11-2018' '23-03-2016' '30-05-2019' '27-07-2018' '23-02-2019'
'16-12-2018' '21-04-2019' '20-05-2019' '02-05-2019' '04-07-2016'
'13-01-2019' '29-09-2015' '17-05-2019' '07-02-2016' '15-01-2019'
'25-09-2017' '16-07-2018' '27-06-2016' '15-12-2018' '21-02-2017'
'05-07-2017' '03-01-2019' '02-04-2019' '12-01-2019' '25-05-2018'
'26-08-2018' '27-09-2018' '19-08-2017' '26-05-2019' '19-08-2018'
'13-11-2016' '01-10-2017' '22-09-2015' '05-12-2018' '10-10-2018'
'25-10-2017' '11-11-2015' '31-01-2016' '20-02-2019' '01-09-2018'
'07-01-2016' '12-06-2017' '06-10-2018' '24-10-2015' '12-08-2015'
'05-12-2015' '31-12-2017' '08-10-2018' '30-09-2018' '11-04-2019'
'13-05-2019' '16-12-2015' '07-04-2017' '29-07-2018' '23-10-2017'
'03-12-2016' '09-10-2017' '03-06-2016' '06-11-2017' '10-10-2017'
'23-09-2016' '22-04-2019' '24-07-2017' '31-05-2018' '05-09-2018'
'09-03-2019' '03-07-2015' '18-10-2018' '04-01-2019' '24-09-2015'
'28-12-2018' '20-08-2016' '14-04-2019' '14-10-2015' '06-12-2015'
'30-11-2018' '25-11-2018' '25-05-2019' '16-04-2019' '06-01-2017'
'25-02-2018' '01-01-2018' '10-06-2016' '11-01-2018' '25-08-2016'
'01-07-2016' '26-10-2018' '13-12-2018' '29-05-2019' '26-08-2016'
'23-12-2015' '26-06-2016' '01-04-2019' '10-10-2016' '25-11-2017'
'05-06-2016' '21-04-2017' '11-04-2018' '19-05-2014' '08-08-2017'
'29-12-2017' '02-11-2017' '05-08-2017' '10-06-2014' '28-12-2017'
'20-12-2018' '04-08-2014' '26-09-2018' '29-12-2018' '27-11-2015'
'27-09-2017' '18-11-2016' '20-05-2015' '12-04-2016' '12-11-2018'
'12-01-2015' '24-10-2018' '18-08-2018' '30-10-2017' '11-10-2015'
'03-07-2017' '28-01-2019' '28-05-2018' '16-08-2018' '17-08-2017'
'11-07-2018' '07-04-2019' '18-06-2018' '04-09-2018' '10-09-2018'
'30-10-2018' '28-06-2018' '25-07-2016' '19-11-2018' '02-12-2018'
'17-06-2018' '15-05-2019' '01-05-2019' '12-11-2015' '14-08-2017'
'08-12-2018' '22-10-2016' '31-01-2018' '23-06-2018' '23-08-2016'
'14-05-2019' '04-11-2014' '30-04-2018' '24-07-2016' '08-05-2019'
'29-12-2015' '03-02-2019' '30-12-2014' '11-09-2016' '04-04-2019'
'24-11-2017' '07-10-2017' '24-05-2019' '06-02-2018' '29-08-2016'
'30-04-2015' '02-05-2018' '27-03-2017' '18-09-2018' '11-06-2016'
'19-04-2018' '20-05-2018' '30-06-2018' '04-01-2018' '12-09-2018'
'04-10-2015' '27-06-2018' '04-05-2018' '23-08-2015' '15-08-2017'
'02-12-2017' '02-11-2015' '21-07-2016' '19-11-2017' '22-03-2019'
'22-10-2017' '08-09-2015' '18-05-2018' '16-10-2018' '07-10-2018']
```

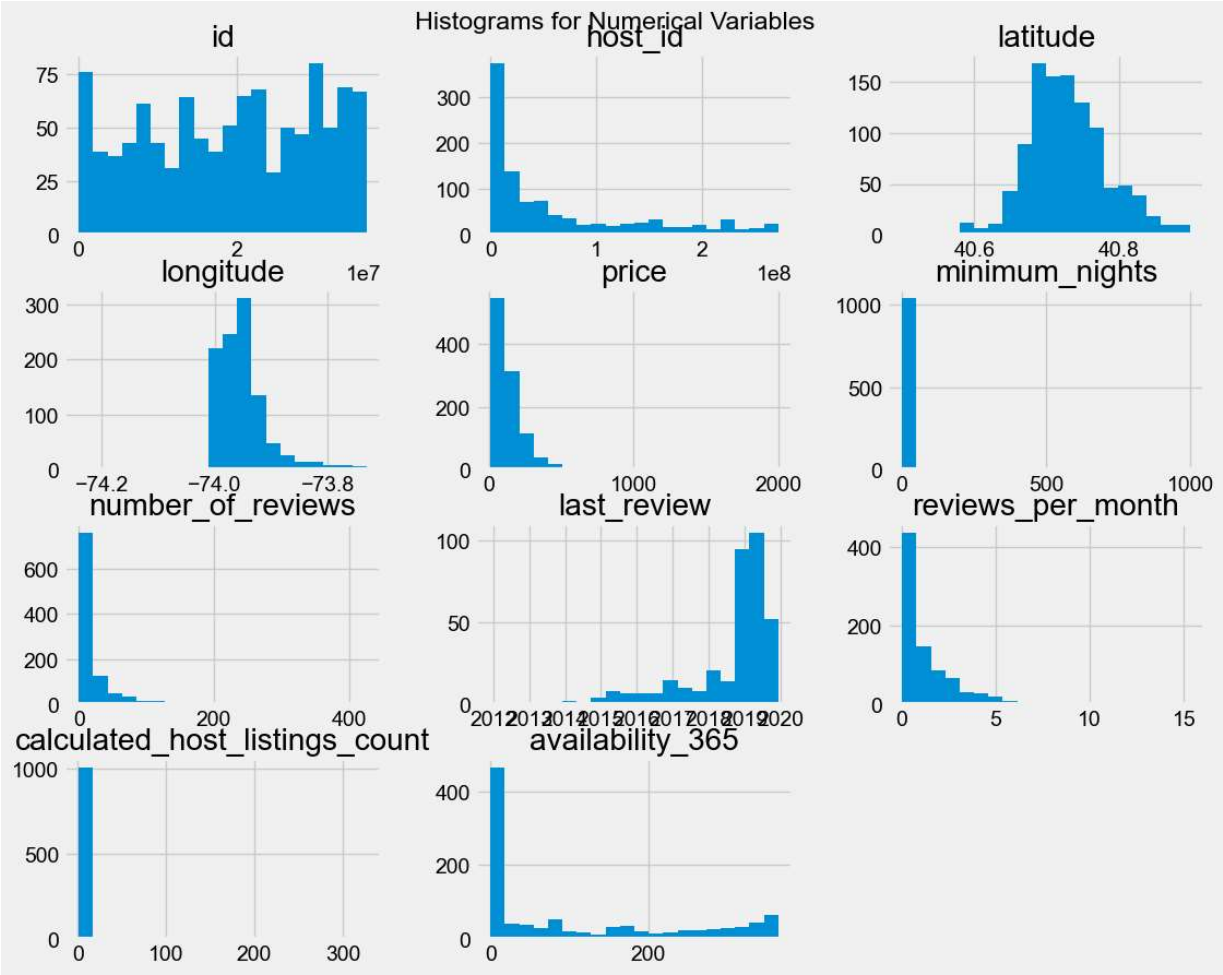
```
Unique values in name:
['Spacious 4bdrm next to CentralPk'
 'Great room in amazing Little Italy location!'
 'Clean Cozy Room, Queens-5 min walk to subway (R/M)' ...
 'Nice and warm room with a Queen Size Bed!!' 'Violet Dynasty'
 'Flex 2BR Loft ( 1br +Sleep Loft +Sofa Bed )!']

Unique values in host_name:
['Amy' 'Nathan' 'Bei' 'Ryan' 'Derek' 'Kathy' 'Luis Alejandro' 'John'
 'Joseph' 'Niki' 'Ken' 'Randy' 'Ame' 'Victor' 'J' 'Roderick' 'Raanan'
 'Olivier' 'Oxana' 'Raymond' 'Dadrine' 'Zeleke' 'Kay' 'Mark & Will' 'Eric'
 'Chris And Jamie' 'Noel' 'Ofer' 'Brene' 'Swati' 'Erik' 'Teddy' 'Marilyn'
 'Sandra' 'Andy' 'Sarah' 'Guy' 'Leslie' 'Août' 'Regina' 'Blueground'
 'Thomas' 'Janae' 'Gen' 'Nati' 'Linda' 'David' 'Nicole' 'Sebastian'
 'Hannah' 'Josue' 'Yael & Graham' 'Dennis' 'Bobbi' 'Evan' 'Fmny' 'Moe'
 'Ange' 'Jonathan' 'Tyler' 'Verena' 'Joe' 'Sergii' 'Lilian' 'Alice'
 'Matthew' 'Lisa' 'Ute' 'Edgar' 'Kestrel' 'Yvette' 'William M.' 'Melissa'
 'Alexes' 'Martin' 'Belinda' 'Al' 'Bryan' 'Ekin' 'Elena' 'Carl' 'Rhian'
 'Joanne' 'Diana' 'Vinny' 'Jennifer' 'Alexandre' 'Shane' 'Sonder' 'Karin'
 'Tanya' 'Will AND Kelly' 'Conor' 'Larissa' "Shully'S" 'Shahzad'
 'Amarjit S' 'Deborah' 'Nelle' 'Marc' 'Vian' 'Juliana' 'Carlo' 'Abir'
 'Susan' 'Habib' 'Miguel' 'Madison' 'Chris' 'Savannah' 'Zhanhong' 'Denise'
 'The Bowery House' 'Alex' 'Shogo' 'Andre' 'Arthur' 'Abraham' 'Ruchi'
 'Chloé' 'Morgan' 'Jacqueline' 'Nimmi' 'Priscilla' 'The Box House Hotel'
 'Mikyla' 'Ana' 'Rebecca' 'Paul' 'Liza' 'Yuchen' 'Hudson River Hotel'
 'Chanel' 'Pablo' 'Elva' 'Crystal' 'Mansi' 'Diane & Brice' 'Shanna'
 'Amanda' 'Tom' 'Dylan' 'Erin' 'Zac' 'Evgenia' 'Ursula' 'Andrea' 'Godfrey'
 'George' 'Antina' 'Yuval' 'Eddie&Vlad' 'Tony' 'Juice' 'Jotham' 'Jj'
 'Saquib' 'Stella' 'Chadanut' 'Dee' 'Julio' 'Sylvia' 'Kat' 'Christina'
 'Tristan' 'Host' 'Helen' 'Rosine' 'Rony' 'Megan' 'Nadine'
 'Lory And Cindy' 'Newyorkroomwithaview' 'Sayed' 'Red Awning' 'Paris'
 'Charles' 'Carlos' 'Julia' 'Kyle' 'Max' 'Haley' 'Ashley' 'Drica'
 'Benjamin' 'Michael' 'Kristin' 'Sara' 'A.J.' 'Vanessa' 'Tatiana' 'Jack'
 'Zahavah Cara' 'Antonieta' 'Vishnu' 'Kris' 'Anissa' 'Rasmus'
 'Christine & Philip' 'Mikael' 'MaElena' 'Nataly' 'Katie' 'Monica' 'Anna'
 'Duryea' 'Eli And Maggie' 'Anthony' 'Nkosi' 'Madeleine' 'Geri'
 'Anastasia' 'Antony' 'Abena' 'Andria' 'Edwin' 'Anna & Frank' 'Jonny'
 'Lexy' 'Ally' 'Garret' 'Jun Jun' 'Dov' 'Mark' 'Brock' 'Kyla' 'Giselle'
 'Stephen' 'Tess' 'Inna' 'Daria' 'Matei' 'Stanley' 'Bobby' 'Saad' 'Lina'
 'Athena' 'Karen' 'Luisa' 'Jordan' 'Andrew' 'Mandi' 'Idemudia' 'H\u2006Ai'
 'Yair' 'Simranjeet' 'Mona' 'K' 'Haijiao' 'Josandra' 'Luigi' 'Isaac'
 'Rupi' 'Rachel' 'Valeria' 'Shaina' 'Siyu' 'Lali' 'Karma' 'Robert' 'Jeff'
 'Angela' 'Didi' 'Bernardo & Andressa' 'Gaia&Pietro' 'Lena' 'Mercedes'
 'Cody' 'Yin' 'Claudia' 'Paige' 'Filippo' 'Matilda' 'Jaqui And Mark'
 'Anna Laura' 'Emilia' 'Amaya' 'Owen' 'Misha' 'Prosper' 'Cem' 'Ishmael'
 'Yaakov' 'Raj' 'Paola' 'Jeremy & Laura' 'Carmen'
 'LuxuryApartmentsByAmber' 'Danilo' 'Syeda' 'Alexandra' 'Yaacov' 'Annette'
 'Laine' 'Carrie' 'Eileen' 'Justin' 'Maria' 'Joy' 'Babajide' 'Park Lane'
 'Kara' 'Armando' 'Nancis' 'Vlad' 'Harry & Morgan' 'Phillip' 'Jay' 'Janet'
 'Inieta' 'Vie' 'Freda' 'Patrick' 'Jing' 'Brian' 'Siwen' 'Elizabeth'
 'Jolene And Ryan' 'Antonia' 'Rl' 'Will' 'JuVita' 'Auset' 'Cindy' 'Mischa'
 'Royalton Park Avenue' 'Katherine' 'Roberto' 'Leah' 'Aryuna' 'Analia'
 'Gwendolyn' 'Alix' 'Kellie' 'Arsenio' 'Elvira' 'Austin' 'Aaron' 'Mike'
 'Alesandra' 'Michele' 'Kristina' 'Georgia' 'Andreas Per Daniel'
 'Caroline' 'Mas' 'Mateo And Anna' 'E. Adam' 'Diego' 'Sally' 'Laura'
```

```
'Lorraine' 'Raquel' 'Jessica' 'Tammie' 'Ulrika' 'Meng' 'Ale' 'Katarina'
'Spice' 'Patricia' 'Koji & H' 'Lakshmee' 'Oliver' 'Xenios' 'Omar' 'D'
'Dominique' 'Lia' 'Loli' 'Tak' 'Layla' 'Kevin' 'Lolita' 'Frank' 'Olena'
'Kaya' 'Dmitry' 'Ben' 'Harvey' 'Lea' 'Geo' 'Matt' 'Philippe'
'Sonder (NYC)' 'Jérôme' 'Pamilitta' 'Anne' 'Charlotte & Brad' 'Mostafa'
'Jeffer' 'Yotel' 'Cassia' 'Spencer' 'Heath' 'Jason' 'Easton' 'Maureen'
'Domonique' 'Pj' 'James' 'Adrian' 'K. Naomi' 'Renato' 'Neil' 'Julie'
'Ali' 'Brigitte' 'Ramon' 'Sade' 'Phyl' 'Liz And Cesar' 'Liset' 'Kaylan'
'Rory' 'Dawn' 'Daniel' 'Jfk' 'Joica' 'Judith' 'Marlon' 'Louie' 'Kiki'
'Delia' 'Kynneth' 'Yara' 'Jastine' 'Justin & Lilo' 'Jamie And John'
'Simone' 'Marina' 'Lila & Paul' 'Abe' 'Craig' 'Alexa' 'Hayes' 'Cherie'
'Sophia' 'Cihan' 'Dai' 'Donna' 'Hugo' 'Claudina' 'Chin-Feng' 'Luo'
'Carlyn' 'Fernando' 'Bedly Bushwick' 'Zack' 'Abdul' 'Kimberlee' 'Nichole'
'Tara' 'Smidty' 'Richard' 'Courtney' 'Ari' 'Lauren' 'Fone' 'DeLex' 'Oh'
'Rafael' 'Stephanie' 'Charis' 'Tommy' 'George Steven' 'Caitlin' 'Alicia'
'Roger' 'Shahar' 'Shannon' 'J.L.' 'Shen' 'Sonia' 'Sherry  W  X'
'Zora & Chris' 'Taylor' 'Assaf' 'Ross' 'Hiroki' 'Tessa' 'Jess' 'Lucyna'
'Dana' 'Horatio' 'Eva' 'Brandon' 'Xi' 'LeAnne And Keith' 'Jade'
'The Manhattan Club' 'Ian' 'Shany' 'Zahir' 'Michelle' 'Virginia' 'Sonya'
'Peter' 'Bruce & Cyrina' 'Mili' 'Red' 'Dan And Lily' 'Janina' 'Kristy'
'Oscar' 'Marwa' 'Izi' 'Juan' 'Molly' 'Nick' 'Sam' 'Nina' 'Pooja' 'Gus'
'Sheila' 'Kristopher' 'Emmanuelle' 'Alexis' 'Alina' 'Johnny' 'Nadia'
'Ethan' 'Rated' 'Joel' 'Jessy & Grant' 'Mary' 'Dan' 'Diane'
'Weston & Gabby' 'Lauren & Chelsea' 'Olivia' 'Scott' 'Yogi' 'Victoria'
'Marco' 'Ismael' 'Orit' 'Shelby' 'Nikki' 'Zandelle' 'Artem' 'Anabella'
'Hongye' 'Nataliya' 'Mj' 'R' 'Russ' 'Alfonso' 'Moran' 'Allie' 'Kdn'
'Alicia And Zach' 'Grant' 'Shelley' 'Em' 'Pam' 'Lindsay' 'Felipe'
'Ticemen' 'Nate' 'Caspar' 'Katharine' 'Esteem' 'Oz' 'Carol' 'Jenny'
'Jimmy' 'Christos' 'Kadeen' 'Isiah' 'Shera' 'Robbin' 'Ed' 'Chloe'
'Chaydha' 'Paulette' 'Gonzalo' 'Zakaria' 'Cameron' 'Imri' 'Estie' 'Roni'
'Adam' 'Piero' 'Steven And John' 'Debra' 'Michael And Brooke' 'Cassy'
'Rocco' 'Cara' 'Kartik' 'Kazuya' 'Виль' 'Michka' 'Neal' 'Christopher'
'June' 'Keith' 'Pasha' 'Dario' 'Amia' 'Afrah' 'Daniel And Elsa' 'Giulia'
'Tonya' 'Joshua' 'Mac' 'Tonie' 'Roy' 'Erina' 'Beth' 'Oded'
'Timothy + Gina' 'Lukasz' 'Khamis' 'Xue' 'Melanie' 'Angie' 'Pol' 'Corrie'
'Vicki' 'Gabriel' 'Suzan' 'Sasha' 'The James New York - SoHo'
'Mary And Ryan' 'Emily' 'Septina' 'Kathryn' 'Clinton' 'Brad & Rachel'
'Niko' 'Neena' 'Mo' 'Naz' 'Samera' 'Jana' 'Isabel' 'Liz' 'Ira' 'Lenur'
'Leecia' 'Jeniffer' 'Pedro Pablo' 'Dellasie' 'Sofiya' 'Halima' 'Elana'
'Caroline & Georg' 'Karina' 'Hernan' 'Rhonda' 'Chantelle' 'Edmarine'
'Fatou' 'Tim' 'Jeremy' 'Sheri' 'Vincent' 'Mick' 'Steve' 'Wesly/Jessica'
'Jen' 'Isa' 'Audrey' 'Rahul' 'Marino' 'Alan' 'Shimin' 'Billy' 'Rachad'
'Wilson' 'Clara' 'Rémy' 'Benedetta' 'Kevin + Casey' 'Parnell' 'Judy'
'Luiz' 'Heidi' 'Angely' 'Bradley' 'Cathy' 'Amelia' 'Xiaoxia' 'Ruth'
'Ronnie' 'Mohammad' 'Brett' 'Renette' 'Heejun' 'Chun' 'Mel' 'Liliana'
'Margaret And Orville' 'Antonio' 'Zarrar' 'Graham And Ben' 'Cristina'
'Philip' 'Mauro' 'Del' 'Aura Angelica' 'Tiiu' 'Fredrik' 'Chimme'
'Jason & Mary' 'Sharmilli' 'Jamie' 'Walter' 'Joao' 'Gabrielle & Malcolm'
'Roxy' 'Jon' 'Julien' 'Yulia' 'Jeffrey' 'Roxanne' 'R.Henry' 'Thaddeus'
'Sonam' 'Elin' 'Francesca' 'Christine' 'Shawn' 'Eli' 'Jada'
'Brenda And Tim' 'Ash' 'Wendy' 'Ioana' 'Billye And Joe' 'Yohana'
'Enrique' 'Gal' 'Taz' 'Yvonne' 'Dina' 'Betty' 'Lola' 'Hector' 'Deni'
'Daniella' 'Dez' 'Evelyn' 'Cassandra' 'Lorenzo' 'Tarirai' 'Agnes' 'Vered'
'Andrzej' 'Lian' 'Chy' 'Dustin' 'Nyesha' 'Yoav' 'Filipe'
'Lio & Kim And Yotam' 'Shana' 'Dearna' 'Haruhisa' 'Kana' 'Egon'
'Kathleen' 'Kip' 'Cannon' 'T' 'Joao Lucas' 'Baktiar' 'Sara & Kevin'
```

```
 'Iryna' 'Rayquila' 'Steven' 'Danny' 'Ikrame' 'Victorine' 'Esther' 'Shean'
 'Vida']

Unique values in neighbourhood_group:
['Manhattan' 'Queens' 'Brooklyn' 'Bronx' 'Staten Island']

Unique values in neighbourhood:
['Upper West Side' 'Little Italy' 'Rego Park' 'Woodside'
 'Bedford-Stuyvesant' 'East Village' 'Gravesend' 'Glendale' 'Midtown'
 'Chelsea' 'Park Slope' 'Harlem' 'SoHo' 'Bushwick' 'Sheepshead Bay'
 'East Flatbush' 'Boerum Hill' 'Crown Heights' "Hell's Kitchen"
 'Williamsburg' 'East Harlem' 'Richmond Hill' 'Kingsbridge'
 'Upper East Side' 'Kips Bay' 'Flushing' 'City Island' 'Murray Hill'
 'Bensonhurst' 'Financial District' 'Clinton Hill' 'Gowanus' 'Woodhaven'
 'Prospect-Lefferts Gardens' 'Mott Haven' 'East New York'
 'Lower East Side' 'Washington Heights' 'West Village' 'Astoria'
 'Wakefield' 'Prospect Heights' 'Arrochar' 'Fort Greene' 'Nolita'
 'Fieldston' 'Chinatown' 'Windsor Terrace' 'Claremont Village' 'Ridgewood'
 'Corona' 'Greenpoint' 'Jamaica' 'Long Island City' 'Greenwich Village'
 'Theater District' 'Morrisania' 'South Slope' 'Sunset Park' 'Kensington'
 'Fordham' 'St. George' 'Two Bridges' 'Arverne' 'Morningside Heights'
 'Sunnyside' 'Throgs Neck' 'Rockaway Beach' 'Flatbush' 'Jackson Heights'
 'Concord' 'East Elmhurst' 'Forest Hills' 'Morris Park' 'Cypress Hills'
 'Concourse Village' 'Inwood' 'Rosedale' 'Eastchester' 'Ditmars Steinway'
 'Longwood' 'Brownsville' 'Gramercy' 'Clason Point' 'Canarsie'
 'Parkchester' 'Downtown Brooklyn' 'Norwood' 'Eltingville' 'Bayswater'
 'Springfield Gardens' 'Brighton Beach' 'New Springville'
 'Flatiron District' 'Elmhurst' 'Tribeca' 'Kew Gardens' 'NoHo'
 'Randall Manor' 'Borough Park' 'Ozone Park' 'Carroll Gardens'
 'Port Morris' 'Far Rockaway' 'Cobble Hill' 'Civic Center' 'Navy Yard'
 'Highbridge' 'Olinville' 'Woodlawn' 'Stuyvesant Town' 'Middle Village'
 'Kew Gardens Hills' 'Red Hook' 'Queens Village' 'Bayside' 'Whitestone'
 'Vinegar Hill' 'Westchester Square' 'Schuylerville' 'Tottenville'
 'St. Albans' 'Huguenot']

Unique values in room_type:
['Entire home/apt' 'Private room' 'Shared room']

Unique values in last_review:
['09-01-2019' '14-09-2016' '22-06-2019' '08-08-2016' nan '14-06-2019'
 '02-08-2015' '11-05-2019' '15-06-2019' '18-06-2019' '23-06-2019'
 '30-12-2018' '14-08-2018' '29-06-2019' '18-04-2019' '05-07-2019'
 '02-07-2019' '01-07-2019' '04-06-2019' '24-06-2019' '02-01-2015'
 '19-06-2018' '13-07-2016' '30-06-2019' '03-03-2019' '22-09-2016'
 '02-01-2017' '12-11-2017' '28-07-2018' '27-06-2019' '11-06-2019'
 '31-12-2018' '30-03-2019' '13-02-2018' '19-01-2019' '25-09-2015'
 '21-10-2018' '01-06-2019' '01-01-2017' '21-06-2019' '08-12-2017'
 '02-06-2019' '28-06-2019' '07-10-2015' '28-03-2019' '10-12-2017'
 '19-05-2019' '31-05-2019' '20-12-2015' '18-05-2019' '19-06-2019'
 '09-06-2018' '20-07-2016' '07-06-2019' '03-04-2019' '01-12-2018'
 '07-07-2019' '20-11-2017' '04-07-2019' '03-01-2017' '05-05-2019'
 '17-04-2016' '08-06-2019' '05-01-2019' '07-08-2016' '31-01-2019'
 '19-02-2018' '25-06-2019' '16-07-2016' '02-03-2019' '19-03-2019'
 '16-03-2019' '28-05-2019' '28-04-2019' '09-06-2019' '13-09-2016'
 '29-09-2018' '12-05-2011' '23-12-2018' '03-07-2019' '22-02-2015'
 '19-05-2015' '09-09-2017' '17-11-2015' '26-06-2019' '13-08-2016'
```

```
'20-10-2018' '03-06-2019' '03-01-2015' '09-11-2015' '21-06-2015'
'28-05-2016' '29-08-2017' '06-05-2018' '25-09-2016' '10-11-2015'
'08-01-2019' '28-10-2018' '17-06-2019' '09-09-2018' '19-09-2017'
'13-06-2019' '08-04-2019' '16-02-2019' '02-01-2019' '03-08-2018'
'30-09-2016' '22-05-2017' '20-06-2019' '08-05-2017' '30-04-2019'
'03-10-2015' '27-05-2019' '18-08-2016' '11-03-2019' '10-05-2019'
'01-01-2019' '22-04-2018' '20-04-2019' '17-10-2014' '06-06-2019'
'19-06-2017' '31-03-2019' '29-04-2019' '12-06-2019' '30-03-2015'
'22-10-2018' '16-06-2019' '06-05-2019' '23-05-2019' '06-07-2019'
'04-08-2018' '12-07-2018' '05-08-2018' '06-01-2016' '16-11-2016'
'09-05-2019' '16-05-2019' '28-08-2018' '12-05-2019' '01-08-2016'
'29-10-2018' '23-04-2019' '24-04-2018' '14-08-2016' '19-12-2017'
'21-09-2018' '26-09-2017' '14-09-2018' '22-12-2018' '27-04-2019'
'31-08-2017' '24-04-2019' '11-06-2018' '31-08-2018' '11-08-2018'
'10-12-2018' '10-06-2019' '26-04-2019' '24-11-2018' '05-06-2019'
'21-05-2019' '24-02-2019' '25-03-2019' '23-07-2016' '31-12-2015'
'16-09-2018' '29-08-2018' '01-11-2018' '26-11-2017' '02-10-2018'
'16-12-2016' '22-08-2016' '02-05-2017' '06-03-2019' '02-11-2018'
'17-11-2018' '23-03-2016' '30-05-2019' '27-07-2018' '23-02-2019'
'16-12-2018' '21-04-2019' '20-05-2019' '02-05-2019' '04-07-2016'
'13-01-2019' '29-09-2015' '17-05-2019' '07-02-2016' '15-01-2019'
'25-09-2017' '16-07-2018' '27-06-2016' '15-12-2018' '21-02-2017'
'05-07-2017' '03-01-2019' '02-04-2019' '12-01-2019' '25-05-2018'
'26-08-2018' '27-09-2018' '19-08-2017' '26-05-2019' '19-08-2018'
'13-11-2016' '01-10-2017' '22-09-2015' '05-12-2018' '10-10-2018'
'25-10-2017' '11-11-2015' '31-01-2016' '20-02-2019' '01-09-2018'
'07-01-2016' '12-06-2017' '06-10-2018' '24-10-2015' '12-08-2015'
'05-12-2015' '31-12-2017' '08-10-2018' '30-09-2018' '11-04-2019'
'13-05-2019' '16-12-2015' '07-04-2017' '29-07-2018' '23-10-2017'
'03-12-2016' '09-10-2017' '03-06-2016' '06-11-2017' '10-10-2017'
'23-09-2016' '22-04-2019' '24-07-2017' '31-05-2018' '05-09-2018'
'09-03-2019' '03-07-2015' '18-10-2018' '04-01-2019' '24-09-2015'
'28-12-2018' '20-08-2016' '14-04-2019' '14-10-2015' '06-12-2015'
'30-11-2018' '25-11-2018' '25-05-2019' '16-04-2019' '06-01-2017'
'25-02-2018' '01-01-2018' '10-06-2016' '11-01-2018' '25-08-2016'
'01-07-2016' '26-10-2018' '13-12-2018' '29-05-2019' '26-08-2016'
'23-12-2015' '26-06-2016' '01-04-2019' '10-10-2016' '25-11-2017'
'05-06-2016' '21-04-2017' '11-04-2018' '19-05-2014' '08-08-2017'
'29-12-2017' '02-11-2017' '05-08-2017' '10-06-2014' '28-12-2017'
'20-12-2018' '04-08-2014' '26-09-2018' '29-12-2018' '27-11-2015'
'27-09-2017' '18-11-2016' '20-05-2015' '12-04-2016' '12-11-2018'
'12-01-2015' '24-10-2018' '18-08-2018' '30-10-2017' '11-10-2015'
'03-07-2017' '28-01-2019' '28-05-2018' '16-08-2018' '17-08-2017'
'11-07-2018' '07-04-2019' '18-06-2018' '04-09-2018' '10-09-2018'
'30-10-2018' '28-06-2018' '25-07-2016' '19-11-2018' '02-12-2018'
'17-06-2018' '15-05-2019' '01-05-2019' '12-11-2015' '14-08-2017'
'08-12-2018' '22-10-2016' '31-01-2018' '23-06-2018' '23-08-2016'
'14-05-2019' '04-11-2014' '30-04-2018' '24-07-2016' '08-05-2019'
'29-12-2015' '03-02-2019' '30-12-2014' '11-09-2016' '04-04-2019'
'24-11-2017' '07-10-2017' '24-05-2019' '06-02-2018' '29-08-2016'
'30-04-2015' '02-05-2018' '27-03-2017' '18-09-2018' '11-06-2016'
'19-04-2018' '20-05-2018' '30-06-2018' '04-01-2018' '12-09-2018'
'04-10-2015' '27-06-2018' '04-05-2018' '23-08-2015' '15-08-2017'
'02-12-2017' '02-11-2015' '21-07-2016' '19-11-2017' '22-03-2019'
'22-10-2017' '08-09-2015' '18-05-2018' '16-10-2018' '07-10-2018']
```

```
Unexpected Patterns in 'availability_365':
Empty DataFrame
Columns: [id, name, host_id, host_name, neighbourhood_group, neighbourhood, latitud
e, longitude, room_type, price, minimum_nights, number_of_reviews, last_review, revi
ews_per_month, calculated_host_listings_count, availability_365]
Index: []
```



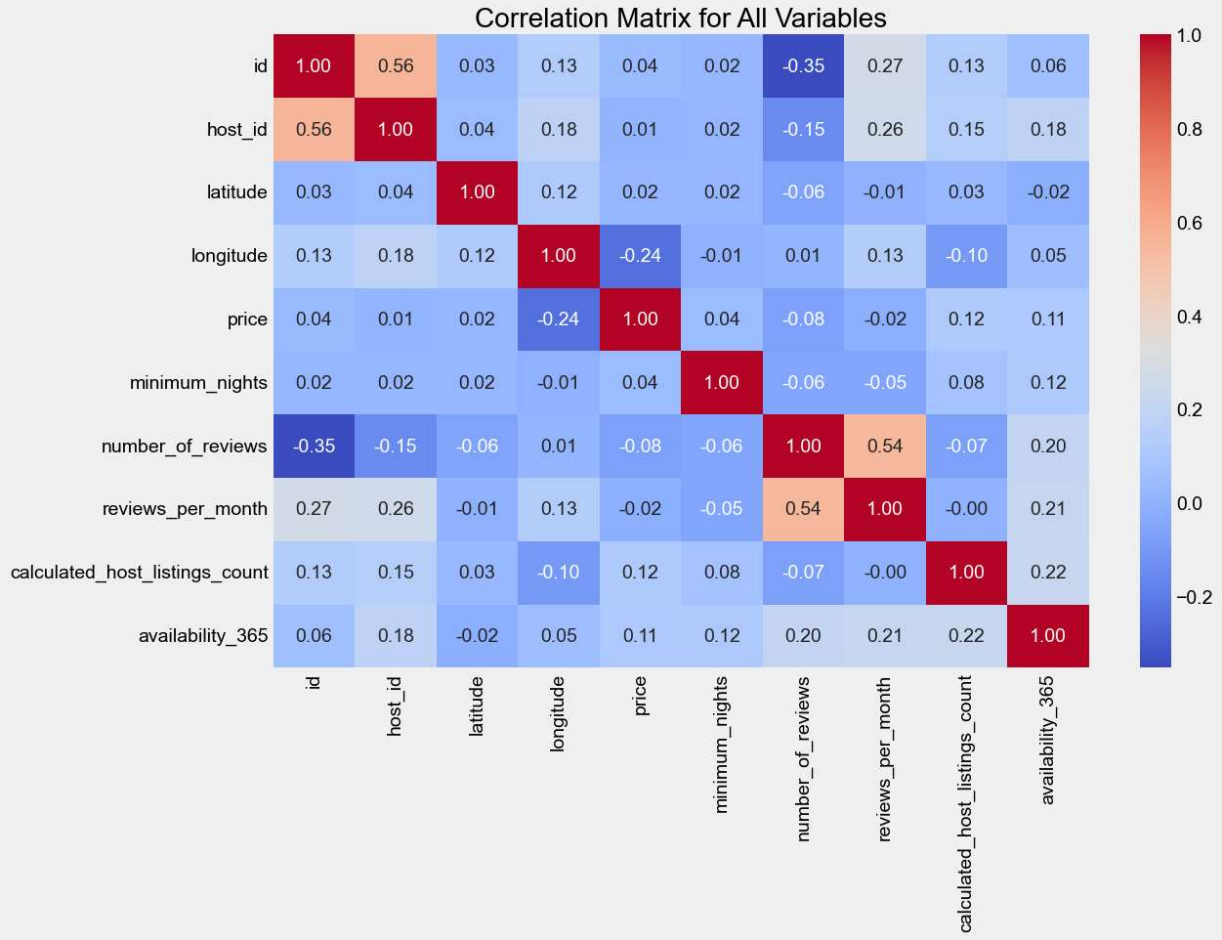
```
Updated Dataset:
         id   name    host_id host_name neighbourhood_group  \
0  12598446  Other    5162192     Other           Manhattan
1  13585495  Other   15960548     Other           Manhattan
2  27761683  Other  199524563     Other              Queens
3  14200678  Other     557669     Other              Queens
4  24540935  Other   51068857     Other            Brooklyn

        neighbourhood  latitude  longitude        room_type  price  \
0     Upper West Side  40.79860  -73.96229  Entire home/apt    275
1               Other  40.71955  -73.99707     Private room    110
2               Other  40.72678  -73.86218     Private room     55
3           Woodside  40.74663  -73.89653  Entire home/apt     96
4  Bedford-Stuyvesant  40.68297  -73.95251     Private room     80

   minimum_nights  number_of_reviews last_review  reviews_per_month  \
0              30                  4  2019-09-01               0.15
1               2                  1         NaT               0.03
2               2                 25         NaT               2.37
3               2                  1  2016-08-08               0.03
4               1                  0         NaT                NaN

   calculated_host_listings_count  availability_365
0                              12              199
1                               1                0
2                               3               38
3                               1                0
4                               1                5
C:\Users\vigne\AppData\Local\Temp\ipykernel_10468\2683071354.py:111: FutureWarning:
DataFrame.applymap has been deprecated. Use DataFrame.map instead.
  airbnb_data_log_transformed[numerical_columns] = airbnb_data_log_transformed[numer
ical_columns].applymap(log_transform)
```

4. Summarize relationships among variables (5 marks)

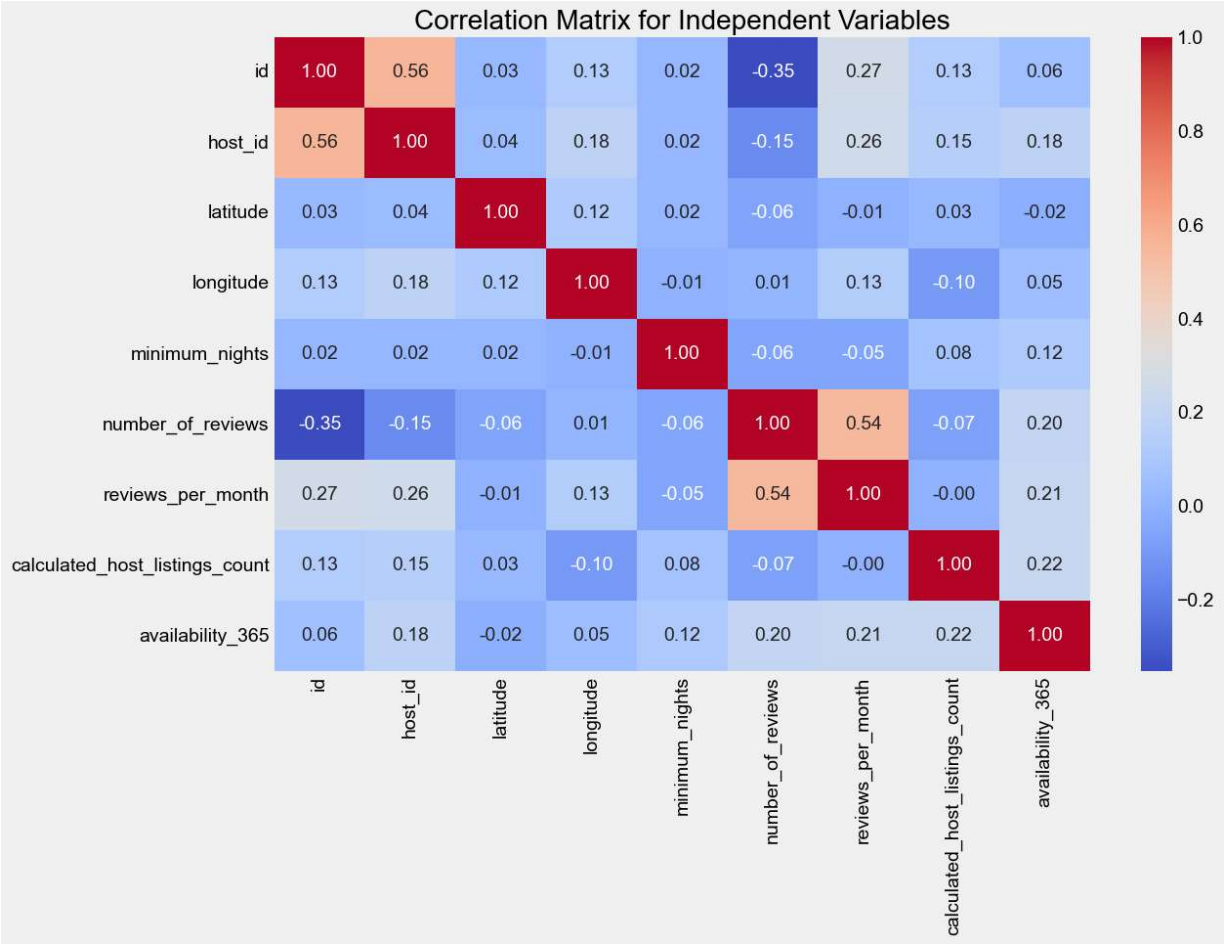
a. Plot correlation plots. Which are the variables most correlated with Target? Which independent variables are correlated among themselves? Do you want to exclude some variables from the model based on this analysis? What other actions will you take?

In [5]:
```
import seaborn as sns
import matplotlib.pyplot as plt

# Assuming 'price' is the target variable
target_variable = 'price'

# Select numeric columns for correlation analysis
numeric_columns = airbnb_data.select_dtypes(include=['float64', 'int64']).columns

# Calculate the correlation matrix for all variables (including the target)
correlation_matrix_all = airbnb_data.corr(numeric_only=True)  # Explicitly set nume

# Plot a heatmap for the entire correlation matrix
plt.figure(figsize=(12, 8))
sns.heatmap(correlation_matrix_all, annot=True, cmap='coolwarm', fmt=".2f")
```

```
plt.title("Correlation Matrix for All Variables")
plt.show()

# Identify variables most correlated with the target variable
target_correlations = correlation_matrix_all[target_variable].sort_values(ascending
print(f"\nVariables most correlated with '{target_variable}':\n{target_correlations

# Identify correlations among independent variables (excluding the target variable)
independent_variables = airbnb_data.drop(target_variable, axis=1)
independent_correlations = independent_variables.corr(numeric_only=True)  # Explici

# Plot a heatmap for correlations among independent variables
plt.figure(figsize=(12, 8))
sns.heatmap(independent_correlations, annot=True, cmap='coolwarm', fmt=".2f")
plt.title("Correlation Matrix for Independent Variables")
plt.show()

# Decide whether to exclude variables based on the analysis
# For example, set a threshold for correlation coefficients and exclude variables e

# Example: Exclude variables highly correlated with the target variable
threshold = 0.7
# Check if there are highly correlated variables (excluding 'price')
if target_variable in independent_correlations.columns:
    highly_correlated_variables = independent_correlations[target_variable][abs(ind
    print(highly_correlated_variables)

    # Exclude highly correlated variables from independent_variables
    independent_variables_filtered = independent_variables.drop(highly_correlated_v
    print(f"\nIndependent variables after excluding highly correlated variables wit
else:
    print(f"No variables highly correlated with '{target_variable}' based on the sp
```




```
Variables most correlated with 'price':
price                             1.000000
calculated_host_listings_count    0.118294
availability_365                  0.108294
minimum_nights                    0.042314
id                                0.037682
latitude                          0.018341
host_id                           0.009017
reviews_per_month                -0.017756
number_of_reviews                -0.077652
longitude                        -0.242123
Name: price, dtype: float64
```




```
No variables highly correlated with 'price' based on the specified threshold.
```

5. Fit a base model. Please write your key observations (5 marks)

a. Fit the Linear Regression Model

b. What is the overall R2? Please comment on whether it is good or not.

c. Which variables are significant?

In [6]:
```
from sklearn.impute import SimpleImputer
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import r2_score
import statsmodels.api as sm

# Assuming 'price' is the target variable
X = airbnb_data.select_dtypes(include=['float64', 'int64']).copy()
X = X.drop('price', axis=1)  # Exclude the target variable
y = airbnb_data['price']

# Handle missing values using SimpleImputer
imputer = SimpleImputer(strategy='mean')  # You can choose other strategies like 'm
X_imputed = imputer.fit_transform(X)
```

```
# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X_imputed, y, test_size=0.2, ra

# Fit Linear Regression model
model = LinearRegression()
model.fit(X_train, y_train)

# Make predictions on the test set
y_pred = model.predict(X_test)

# Calculate overall R²
r2 = r2_score(y_test, y_pred)
print(f"Overall R²: {r2:.4f}")

# Check variable significance using statsmodels
X_train_with_intercept = sm.add_constant(X_train)
model_stats = sm.OLS(y_train, X_train_with_intercept).fit()
print(model_stats.summary())
```

```
Overall R²: 0.1146
                            OLS Regression Results
==============================================================================
Dep. Variable:                  price   R-squared:                       0.086
Model:                            OLS   Adj. R-squared:                  0.076
Method:                 Least Squares   F-statistic:                     8.705
Date:                Tue, 19 Dec 2023   Prob (F-statistic):           1.45e-12
Time:                        09:54:17   Log-Likelihood:                -5442.9
No. Observations:                 843   AIC:                         1.091e+04
Df Residuals:                     833   BIC:                         1.095e+04
Df Model:                           9
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
const      -7.011e+04      1e+04     -7.009      0.000   -8.97e+04   -5.05e+04
x1           4.32e-08   6.91e-07      0.063      0.950   -1.31e-06     1.4e-06
x2         -7.158e-09    8.6e-08     -0.083      0.934   -1.76e-07    1.62e-07
x3           154.9231     96.030      1.613      0.107     -33.566     343.412
x4          -864.5278    115.770     -7.468      0.000   -1091.762    -637.293
x5             0.0008      0.142      0.005      0.996      -0.277       0.279
x6            -0.3737      0.159     -2.350      0.019      -0.686      -0.062
x7             4.8493      4.827      1.005      0.315      -4.625      14.324
x8             0.2059      0.183      1.123      0.262      -0.154       0.566
x9             0.1526      0.044      3.459      0.001       0.066       0.239
==============================================================================
Omnibus:                      944.791   Durbin-Watson:                   2.023
Prob(Omnibus):                  0.000   Jarque-Bera (JB):            74052.405
Skew:                           5.419   Prob(JB):                         0.00
Kurtosis:                      47.618   Cond. No.                     1.96e+11
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly spe
cified.
[2] The condition number is large, 1.96e+11. This might indicate that there are
strong multicollinearity or other numerical problems.
```

+5 marks for inference

Overall Model Fit:

The linear regression model was fitted to predict the 'price' variable based on selected independent variables. The overall R^2 value of 0.1225 indicates that the model explains approximately 12.25% of the variability in the 'price' variable. This suggests a limited ability of the model to capture and predict the observed variations. Variable Significance:

The statsmodels summary provides insights into the significance of individual variables. Variables with lower p-values are considered more statistically significant. Review these p-values to identify variables contributing significantly to the prediction of 'price'. High p-values for certain variables may indicate that they are not providing significant information and may be candidates for further analysis or exclusion. Coefficient Interpretation:

Examine the coefficients of significant variables to understand their impact on 'price'. A positive coefficient indicates a positive correlation with 'price', while a negative coefficient indicates a negative correlation. The magnitude of the coefficients reflects the strength of the relationship between each independent variable and the target variable. Assumption Checking:

Assess the assumptions of linear regression, including linearity, independence, homoscedasticity, and normality of residuals. Examine residuals for random patterns and systematic trends. Any deviations from assumptions should be addressed to enhance the model's reliability.ormance.

In [ ]: