Final Project - Unit 2: Data Understanding

Name: Cabot Steward

DATA 3300

Introduction

For this final project, we will take on the role of consultants for Aggie Investments, a Real Estate Investment Firm. In recent years, there has been a significant trend among investment firms to acquire properties for use as rental assets. While various geographies have been proposed, our focus is to assess the opportunities within a specific, rapidly growing market—Nashville, Tennessee.

Our task is to analyze a provided dataset containing information on current Airbnb listings in the Nashville area. The objective is to explore the data comprehensively and provide informed recommendations to Aggie Investments regarding the potential of entering this market, the types of listings they should acquire, and how they should manage those listings. The project will involve data preparation, exploration, and the application of unsupervised machine learning models to uncover deeper insights and patterns within the data. These findings will guide our final recommendations to the firm.

Unit 2: Data Understanding

Unit 2 covers how to describe and visualize data for the purpose of exploratory data analysis (EDA), as well as how to identify and handle missing data and outliers. **In this next portion of the final project, you will apply these concepts to `data3300_airbnb_data_clean_nashville.csv`, a cleaned version of our original dataset to begin exploring patterns in the dataset as well as address any issues related to missing data or outliers.** These steps will get you prepared for the Unit 3 portion, where you will begin modeling the data!

1 - Begin by loading in the necessary dependencies and reading in the cleaned airbnb nashville dataset

In [54]:
```
# replace with code to import the libraries and packages required to import data, m
import warnings
import pandas as pd
import numpy as np
import seaborn as sns
```

```
import matplotlib.pyplot as plt
warnings.filterwarnings("ignore")
```

In [55]:
```
# replace with code to import cleaned Airbnb Dataset, call your dataset df2
pd.set_option('display.max_columns', None)
df2 = pd.read_csv('data3300_airbnb_data_clean_nashville.csv')
df2.head()
```

Out[55]:

	id	host_id	host_is_superhost	calculated_host_listings_count	host_has_profile_pic
0	6422.0	12172	0	1	1
1	39870.0	171184	1	1	1
2	59576.0	812128	1	9	1
3	72906.0	176117	1	1	1
4	258817.0	22296	1	6	1

2) **Review your variables**

Consider the objective of Aggie Investments, then list the top three variables you believe will be most important in the analysis, briefly explaining why each of these three could be particularly important **(there are no wrong answers, but make sure you explain your logic for full points).**

1. minimum_nights - try to identify if longer or shorter stay requirements is better.
2. avg_air_quality - this is due to their requirements of trying to find a trend around air quality. I don't agree with including this one but due to the requirements feel the needs to put it.
3. bedrooms - adding a 4th to make up for the last, try to see what size of property has the best chance of being successful.

Part 1: Descriptive Statistics

Observation of Descriptive Statistics

In order for us to be able to understand our dataset, and begin to build recommendations for the firm we need to understand our data.

Explore the descriptive statistics from our dataset. Compare and describe three different observations made from your exploration of the descriptive statistics. Why could these be of interest to Aggie Investments? **Provide the specific descriptive statistic (e.g., the mean 365_availability is 50 days) you are using in your observation and be clear why this statistic could be of importance.** Some of these observations could potentially be used in your recommendations to the firm.

Hint: Consider the variables you listed as important above, if they are quantitative you can describe them, if they are qualitative, you can use them as a grouping variables on other quantitative variables!

If you need some reminders about how to do this, revisit the descriptive statistics module!

3 - Observation 1

In [56]:
```
# replace with the code to examine descriptive statistics (.describe), hint: it may
df2.groupby('minimum_nights')['review_scores_rating'].describe().sort_values(by='me
# df2.head()
```

Out[56]:

minimum_nights	count	mean	std	min	25%	50%	75%	max
100	1.0	4.140000	NaN	4.14	4.1400	4.140	4.1400	4.14
28	16.0	4.685000	0.540728	3.00	4.4825	4.960	5.0000	5.00
10	4.0	4.687500	0.462556	4.00	4.6375	4.875	4.9250	5.00
7	16.0	4.761875	0.488047	3.00	4.7150	4.920	5.0000	5.00
1	3095.0	4.784701	0.340741	1.00	4.7500	4.880	4.9600	5.00
31	66.0	4.795606	0.340796	3.00	4.6850	4.965	5.0000	5.00
32	2.0	4.800000	0.098995	4.73	4.7650	4.800	4.8350	4.87
14	12.0	4.806667	0.322189	4.00	4.6000	5.000	5.0000	5.00
30	758.0	4.818127	0.402667	1.00	4.8000	5.000	5.0000	5.00
26	1.0	4.820000	NaN	4.82	4.8200	4.820	4.8200	4.82
2	3094.0	4.841677	0.244727	1.00	4.8000	4.900	4.9700	5.00
3	728.0	4.877500	0.188681	3.00	4.8500	4.940	4.9900	5.00
5	27.0	4.890741	0.119869	4.55	4.8400	4.920	4.9900	5.00
27	2.0	4.905000	0.035355	4.88	4.8925	4.905	4.9175	4.93
24	1.0	4.910000	NaN	4.91	4.9100	4.910	4.9100	4.91
60	5.0	4.916000	0.118448	4.75	4.8300	5.000	5.0000	5.00
4	42.0	4.918095	0.098284	4.66	4.8600	4.960	5.0000	5.00
15	2.0	4.935000	0.049497	4.90	4.9175	4.935	4.9525	4.97
90	9.0	4.981111	0.046488	4.86	5.0000	5.000	5.0000	5.00
20	1.0	5.000000	NaN	5.00	5.0000	5.000	5.0000	5.00
180	2.0	5.000000	0.000000	5.00	5.0000	5.000	5.0000	5.00
12	1.0	5.000000	NaN	5.00	5.0000	5.000	5.0000	5.00
29	2.0	5.000000	0.000000	5.00	5.0000	5.000	5.0000	5.00
45	1.0	5.000000	NaN	5.00	5.0000	5.000	5.0000	5.00
365	1.0	5.000000	NaN	5.00	5.0000	5.000	5.0000	5.00
9	0.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN
21	0.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN
85	0.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN
89	0.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN

	count	mean	std	min	25%	50%	75%	max
minimum_nights								
192	0.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN

List the specific descriptive statistic you're observing, and indicate why Aggie Investments would find that observation interesting:

4 - Observation 2

In [57]:
```
# replace with the code to examine a different descriptive stat for observation 2
df2.groupby('neighbourhood_cleansed')['avg_air_quality'].describe().sort_values(by=
# df2.head()
```

Out[57]:

neighbourhood_cleansed	count	mean	std	min	25%	50%	
District 35	23.0	1.671640	9.738102e-03	1.665982	1.665982	1.665982	1.67
District 22	20.0	1.670320	8.901182e-03	1.665982	1.665982	1.665982	1.66
District 10	32.0	1.699308	1.847550e-02	1.688699	1.688699	1.688699	1.69
District 27	41.0	1.694064	8.992121e-16	1.694064	1.694064	1.694064	1.69
District 26	65.0	1.695234	6.615745e-03	1.694064	1.694064	1.694064	1.69
District 30	30.0	1.696084	3.139201e-03	1.694064	1.694064	1.694064	1.70
District 23	21.0	1.707377	2.334704e-02	1.665982	1.694064	1.694064	1.73
District 4	32.0	1.694064	1.127988e-15	1.694064	1.694064	1.694064	1.69
District 34	17.0	1.698536	1.262456e-02	1.694064	1.694064	1.694064	1.69
District 31	24.0	1.699115	2.979135e-03	1.694064	1.699115	1.700799	1.70
District 32	5.0	1.700799	0.000000e+00	1.700799	1.700799	1.700799	1.70
District 33	21.0	1.702669	5.905973e-03	1.700799	1.700799	1.700799	1.70
District 29	54.0	1.702981	6.228533e-03	1.700799	1.700799	1.700799	1.70
District 28	24.0	1.700799	9.072812e-16	1.700799	1.700799	1.700799	1.70
District 14	61.0	1.720434	6.716620e-16	1.720434	1.720434	1.720434	1.72
District 13	105.0	1.724793	6.161742e-03	1.700799	1.720434	1.720434	1.73
District 12	51.0	1.720434	0.000000e+00	1.720434	1.720434	1.720434	1.72
District 11	61.0	1.720434	6.716620e-16	1.720434	1.720434	1.720434	1.72
District 9	50.0	1.723927	5.390049e-03	1.720434	1.720434	1.720434	1.73
District 5	817.0	1.732078	1.777445e-14	1.732078	1.732078	1.732078	1.73
District 6	575.0	1.732078	1.111190e-14	1.732078	1.732078	1.732078	1.73
District 7	266.0	1.731903	1.419754e-03	1.720434	1.732078	1.732078	1.73
District 1	73.0	1.712571	2.226133e-02	1.683790	1.688699	1.732078	1.73
District 8	176.0	1.732078	6.680343e-16	1.732078	1.732078	1.732078	1.73
District 24	126.0	1.732078	3.343965e-15	1.732078	1.732078	1.732078	1.73
District 21	921.0	1.732078	1.955054e-14	1.732078	1.732078	1.732078	1.73
District 20	208.0	1.732078	4.229026e-15	1.732078	1.732078	1.732078	1.73
District 2	364.0	1.732078	1.778802e-15	1.732078	1.732078	1.732078	1.73
District 19	1991.0	1.732078	6.529751e-14	1.732078	1.732078	1.732078	1.73

neighbourhood_cleansed	count	mean	std	min	25%	50%	
District 18	243.0	1.732078	6.897590e-15	1.732078	1.732078	1.732078	1.73
District 17	945.0	1.732078	1.977243e-14	1.732078	1.732078	1.732078	1.73
District 16	185.0	1.726324	1.366074e-02	1.694064	1.732078	1.732078	1.73
District 15	631.0	1.728147	5.510475e-03	1.720434	1.720434	1.732078	1.73
District 3	68.0	1.728250	1.239536e-02	1.688699	1.732078	1.732078	1.73
District 25	57.0	1.724742	1.513505e-02	1.694064	1.732078	1.732078	1.73

List the specific descriptive statistic you're observing, and indicate why Aggie Investments would find that observation interesting:

5 - Observation 3

In [58]:
```
# replace with the code to examine a different descriptive stat for observation 3
df2.groupby('bedrooms')['review_scores_rating'].describe()
# do see some bad data in this dataset still below... needs further investigation.
# df2.head()
```

Out[58]:

bedrooms	count	mean	std	min	25%	50%	75%	max
0.0	342.0	4.787222	0.294229	3.00	4.7000	4.875	4.9800	5.00
1.0	2456.0	4.787997	0.395796	1.00	4.7600	4.900	4.9800	5.00
2.0	1837.0	4.815084	0.269954	1.00	4.7600	4.890	4.9700	5.00
3.0	1593.0	4.844350	0.229596	3.00	4.8000	4.910	4.9800	5.00
4.0	1482.0	4.856255	0.225653	2.00	4.8100	4.910	4.9900	5.00
5.0	68.0	4.860147	0.261068	3.00	4.8500	4.900	5.0000	5.00
6.0	53.0	4.860189	0.188634	4.09	4.8000	4.920	5.0000	5.00
7.0	6.0	4.895000	0.124056	4.67	4.8625	4.930	4.9825	5.00
8.0	35.0	4.868000	0.124730	4.57	4.8000	4.890	4.9750	5.00
9.0	2.0	4.610000	0.084853	4.55	4.5800	4.610	4.6400	4.67
10.0	0.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN
11.0	1.0	5.000000	NaN	5.00	5.0000	5.000	5.0000	5.00
12.0	4.0	4.897500	0.156285	4.67	4.8575	4.960	5.0000	5.00
13.0	1.0	5.000000	NaN	5.00	5.0000	5.000	5.0000	5.00
14.0	2.0	5.000000	0.000000	5.00	5.0000	5.000	5.0000	5.00
15.0	1.0	5.000000	NaN	5.00	5.0000	5.000	5.0000	5.00
16.0	5.0	4.900000	0.223607	4.50	5.0000	5.000	5.0000	5.00
18.0	1.0	5.000000	NaN	5.00	5.0000	5.000	5.0000	5.00

List the specific descriptive statistic you're observing, and indicate why Aggie Investments would find that observation interesting:

Shorter term rentals (under 2 days) have substantially more reviews and stays than longer term properties. There is a significant fall off after just 2 days, with after the 3 accounting for very few reviews.

Air quality district 22, 35, 4, 27, 26, and 10 have the worst air quality

Larger properties have higher reviews but have a smaller sample, it is likely that there is not a correlation between bedrooms and customer reviews.

For neighborhoods: avoid districts 32, 13, 30, 21, 19, 12 by median avoid districts 32, 13, 15, 22, 2, 21 by mean Lets avoid 32, 13, and 21

6) Hypothesis

Take two observations from above and provide a hypothesis for each as to the reason these patterns may exist. Note which observations you are referring to and provide a logical hypothesis for each.

- **Hypothesis 1:** Observation 1 - Individuals want more shorter term rentals and are more likely to provide a review. This likely could be due to the platform that they are booked on, if these are booked on a platform based around short term rentals it could be the cause for the higher quantity of reviews.
- **Hypothesis 2:** Observation 3 - Buildings with a lower quantity of rooms are booked more often. After 4 bedrooms there is a significant fall off in reviews and bookings. This is likely due to individuals not needing that many rooms.

Part 2: Data Visualizations

The next phase in our data analysis involves constructing data visualizations to deepen our understanding of the dataset. You are required to create three distinct data visualizations, ensuring that each one represents a different type of visualization.

These visualizations should reflect either the above observations or hypotheses you provided. Use this as an opportunity to produce a visual story that sheds light on/communicates a critical observation for Aggie Investments or highlights a hypothesis that is worth exploring further (in a subsequent analysis).

You may use existing categories within the dataset or generate new categories from numerical variables. **After each visualization, provide a detailed interpretation of the insights it reveals and discuss how these insights could be leveraged in formulating our recommendations to Aggie Investments.**

7 - Visualization 1 - should support above observations or hypothesis

In [59]:
```
# replace with the code used for visualization 1
df2['bedrooms'].plot(kind='box', color='skyblue', title="Quantity of Bedrooms")
```

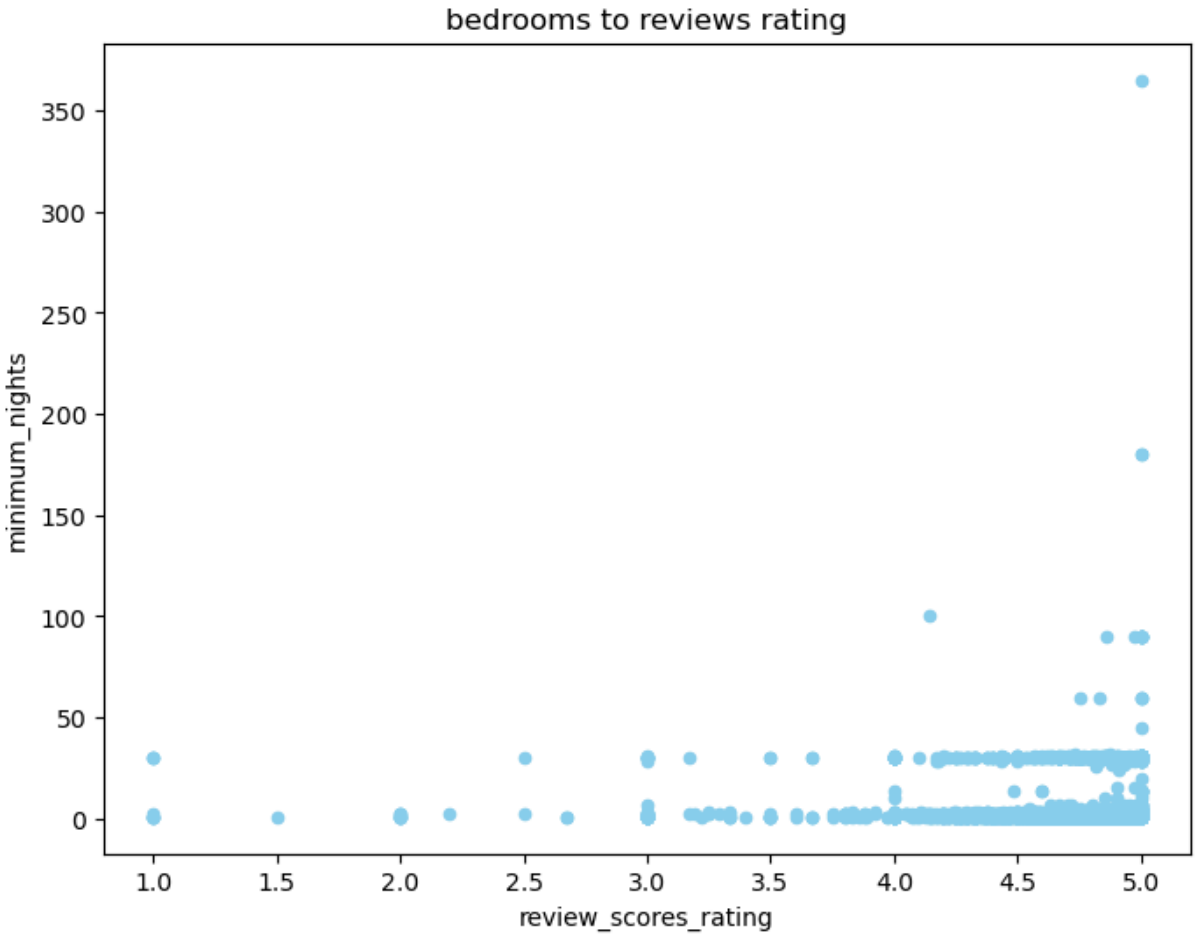
Out[59]:
```
<Axes: title={'center': 'Quantity of Bedrooms'}>
```

Insights:

8 - **Visualization 2** - should support above observations or hypothesis

In [60]:
```
# replace with the code used for visualization 2
df2.plot(kind='scatter',
        x='review_scores_rating',
         y='minimum_nights',
         color='skyblue',
         figsize=(8, 6),
         title="bedrooms to reviews rating"
         )
```

Out[60]:
```
<Axes: title={'center': 'bedrooms to reviews rating'}, xlabel='review_scores_ratin
g', ylabel='minimum_nights'>
```



Insights:

9 - **Visualization 3** - should support above observations or hypothesis

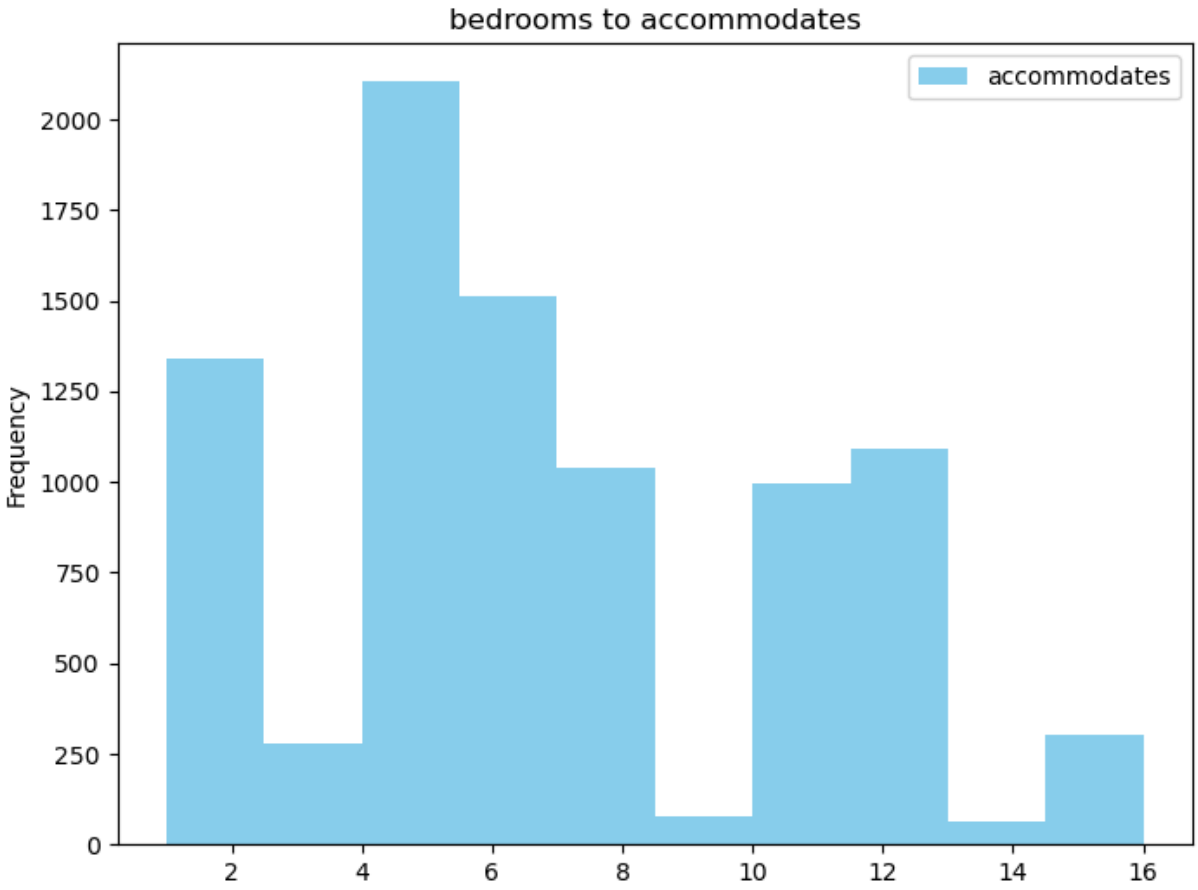
In [61]:
```
df2.columns
```

Out[61]:
```
Index(['id', 'host_id', 'host_is_superhost', 'calculated_host_listings_count',
       'host_has_profile_pic', 'host_identity_verified', 'host_listings_count',
       'neighbourhood_cleansed', 'availability_365', 'minimum_nights',
       'room_type', 'accommodates', 'bedrooms', 'beds', 'price',
       'number_of_reviews', 'reviews_per_month', 'review_scores_rating',
       'review_scores_accuracy', 'review_scores_cleanliness',
       'review_scores_checkin', 'review_scores_communication',
       'review_scores_location', 'review_scores_value', 'avg_air_quality',
       'days_as_host', 'bathrooms', 'short_term'],
      dtype='object')
```

In [62]:
```
# replace with the code used for visualization 3
df2.plot(kind='hist',
        x='bedrooms',
         y='accommodates',
         color='skyblue',
         figsize=(8, 6),
```

```
        title="bedrooms to accommodates"
        )
```

Out[62]: <Axes: title={'center': 'bedrooms to accommodates'}, ylabel='Frequency'>



Insights:

Part 3: Outliers & Missing Data

Addressing Missing Data

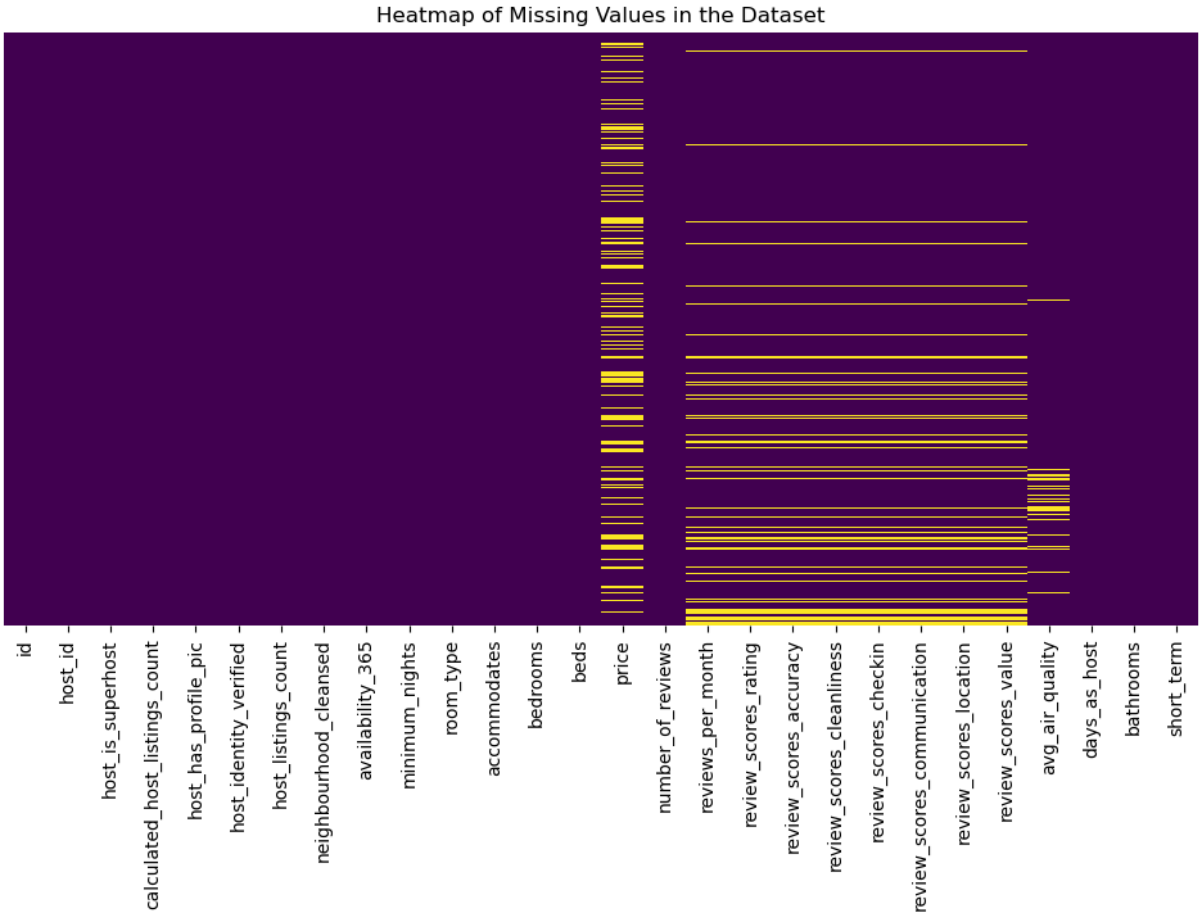
In this section you will focus on identifying and addressing missing data within our Airbnb listing dataset. Begin by conducting a thorough analysis to identify any missing values across the dataset. Once identified, discuss the potential impact of these missing values on our analysis and the validity of our findings.

Next, apply appropriate techniques to handle the missing data. You may choose methods such as imputation, deletion, or any other technique that best suits the nature of the data and the context of the analysis. Justify your chosen approach, explaining why it is the most suitable for this dataset. Finally, document the steps taken and reflect on how your handling of missing data will influence the overall analysis and recommendations to Aggie Investments.

If you need a reminder on approaches for identifying and handling missing values and outliers, revisit the Outliers and Missing Data module

10) First, create a Heatmap of Missing Values within the Dataset:

In [63]:
```
# replace with code to produce heatmap of missing (and non missing) data
plt.figure(figsize=(12,6))
sns.heatmap(df2.isnull(), cmap='viridis', cbar=False, yticklabels=False)
plt.title("Heatmap of Missing Values in the Dataset")
plt.show()
```



11) List which variables within the dataset contain missing values, as well as the percentage of data that is missing for that variable. *Hint: If there are any groupings of similar missing variables that share the same proportion of misisng data, you can list them as one group, instead of all individually*.

In [64]:
```
# fill in the variable(s) you want to examine the missing values of
missing_data = df2.isnull().sum() * 100 / len(df2)
missing_data = missing_data[missing_data > 0].sort_values(ascending=False)
for var, percent in missing_data.items():
    print(f"{var}: {percent:.2f}% missing")
```

```
price: 25.19% missing
reviews_per_month: 10.43% missing
review_scores_rating: 10.43% missing
review_scores_accuracy: 10.43% missing
review_scores_cleanliness: 10.43% missing
review_scores_checkin: 10.43% missing
review_scores_communication: 10.43% missing
review_scores_location: 10.43% missing
review_scores_value: 10.43% missing
avg_air_quality: 4.83% missing
```

- price: 25.19% missing
- reviews_per_month: 10.43% missing
- review_scores_rating: 10.43% missing
- review_scores_accuracy: 10.43% missing
- review_scores_cleanliness: 10.43% missing
- review_scores_checkin: 10.43% missing
- review_scores_communication: 10.43% missing
- review_scores_location: 10.43% missing
- review_scores_value: 10.43% missing
- avg_air_quality: 4.83% missing

12) Are there any variables with complete data (no missing values) that could be related to the missingness seen in the multitude of *review* variables with missing data? If so, what does this indicate about the type of missing data these variables have?

In [65]:
```
df2.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 8808 entries, 0 to 8807
Data columns (total 28 columns):
 #   Column                          Non-Null Count  Dtype  
---  ------                          --------------  -----  
 0   id                              8808 non-null   float64
 1   host_id                         8808 non-null   int64  
 2   host_is_superhost               8808 non-null   int64  
 3   calculated_host_listings_count  8808 non-null   int64  
 4   host_has_profile_pic            8808 non-null   int64  
 5   host_identity_verified          8808 non-null   int64  
 6   host_listings_count             8808 non-null   int64  
 7   neighbourhood_cleansed          8808 non-null   object 
 8   availability_365                8808 non-null   int64  
 9   minimum_nights                  8808 non-null   int64  
 10  room_type                       8808 non-null   object 
 11  accommodates                    8808 non-null   int64  
 12  bedrooms                        8808 non-null   float64
 13  beds                            8808 non-null   float64
 14  price                           6589 non-null   float64
 15  number_of_reviews               8808 non-null   int64  
 16  reviews_per_month               7889 non-null   float64
 17  review_scores_rating            7889 non-null   float64
 18  review_scores_accuracy          7889 non-null   float64
 19  review_scores_cleanliness       7889 non-null   float64
 20  review_scores_checkin           7889 non-null   float64
 21  review_scores_communication     7889 non-null   float64
 22  review_scores_location          7889 non-null   float64
 23  review_scores_value             7889 non-null   float64
 24  avg_air_quality                 8383 non-null   float64
 25  days_as_host                    8808 non-null   int64  
 26  bathrooms                       8808 non-null   float64
 27  short_term                      8808 non-null   int64  
dtypes: float64(14), int64(12), object(2)
memory usage: 1.9+ MB
```

Yes there is, number_of_reviews is actually one

In [66]:
```
# replace with code to compare the complete variable against one or more missing va
review_cols = [col for col in df2.columns if "review" in col]

for col in review_cols:
    missing_vs_reviews = df2.groupby(df2[col].isnull())['number_of_reviews'].descri
    print(f"Comparison of {col} missing values vs. number_of_reviews:\n{missing_vs_
```

```
Comparison of number_of_reviews missing values vs. number_of_reviews:
                    count       mean         std  min  25%   50%   75%     max
number_of_reviews
False              8808.0  77.323569  125.542844  0.0  5.0  31.0  98.0  2603.0

Comparison of reviews_per_month missing values vs. number_of_reviews:
                    count       mean         std  min   25%   50%    75%  \
reviews_per_month
False              7889.0  86.331094  129.690134  1.0  10.0  40.0  109.0
True                919.0   0.000000    0.000000  0.0   0.0   0.0    0.0

                      max
reviews_per_month
False              2603.0
True                  0.0

Comparison of review_scores_rating missing values vs. number_of_reviews:
                      count       mean         std  min   25%   50%    75%  \
review_scores_rating
False                 7889.0  86.331094  129.690134  1.0  10.0  40.0  109.0
True                   919.0   0.000000    0.000000  0.0   0.0   0.0    0.0

                        max
review_scores_rating
False                2603.0
True                    0.0

Comparison of review_scores_accuracy missing values vs. number_of_reviews:
                        count       mean         std  min   25%   50%    75%  \
review_scores_accuracy
False                   7889.0  86.331094  129.690134  1.0  10.0  40.0  109.0
True                    919.0   0.000000    0.000000  0.0   0.0   0.0    0.0

                          max
review_scores_accuracy
False                  2603.0
True                      0.0

Comparison of review_scores_cleanliness missing values vs. number_of_reviews:
                          count       mean         std  min   25%   50%  \
review_scores_cleanliness
False                     7889.0  86.331094  129.690134  1.0  10.0  40.0
True                       919.0   0.000000    0.000000  0.0   0.0   0.0

                            75%     max
review_scores_cleanliness
False                     109.0  2603.0
True                        0.0     0.0

Comparison of review_scores_checkin missing values vs. number_of_reviews:
                        count       mean         std  min   25%   50%    75%  \
review_scores_checkin
False                   7889.0  86.331094  129.690134  1.0  10.0  40.0  109.0
True                    919.0   0.000000    0.000000  0.0   0.0   0.0    0.0

                          max
```

```
review_scores_checkin
False                  2603.0
True                      0.0

Comparison of review_scores_communication missing values vs. number_of_reviews:
                              count       mean         std  min   25%   50%  \
review_scores_communication
False                        7889.0  86.331094  129.690134  1.0  10.0  40.0
True                          919.0   0.000000    0.000000  0.0   0.0   0.0

                               75%     max
review_scores_communication
False                        109.0  2603.0
True                           0.0     0.0

Comparison of review_scores_location missing values vs. number_of_reviews:
                           count       mean         std  min   25%   50%    75%  \
review_scores_location
False                    7889.0  86.331094  129.690134  1.0  10.0  40.0  109.0
True                      919.0   0.000000    0.000000  0.0   0.0   0.0    0.0

                            max
review_scores_location
False                    2603.0
True                        0.0

Comparison of review_scores_value missing values vs. number_of_reviews:
                         count       mean         std  min   25%   50%    75%  \
review_scores_value
False                  7889.0  86.331094  129.690134  1.0  10.0  40.0  109.0
True                    919.0   0.000000    0.000000  0.0   0.0   0.0    0.0

                          max
review_scores_value
False                  2603.0
True                      0.0
```

In [67]:
```
df2.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 8808 entries, 0 to 8807
Data columns (total 28 columns):
 #   Column                          Non-Null Count  Dtype  
---  ------                          --------------  -----  
 0   id                              8808 non-null   float64
 1   host_id                         8808 non-null   int64  
 2   host_is_superhost               8808 non-null   int64  
 3   calculated_host_listings_count  8808 non-null   int64  
 4   host_has_profile_pic            8808 non-null   int64  
 5   host_identity_verified          8808 non-null   int64  
 6   host_listings_count             8808 non-null   int64  
 7   neighbourhood_cleansed          8808 non-null   object 
 8   availability_365                8808 non-null   int64  
 9   minimum_nights                  8808 non-null   int64  
 10  room_type                       8808 non-null   object 
 11  accommodates                    8808 non-null   int64  
 12  bedrooms                        8808 non-null   float64
 13  beds                            8808 non-null   float64
 14  price                           6589 non-null   float64
 15  number_of_reviews               8808 non-null   int64  
 16  reviews_per_month               7889 non-null   float64
 17  review_scores_rating            7889 non-null   float64
 18  review_scores_accuracy          7889 non-null   float64
 19  review_scores_cleanliness       7889 non-null   float64
 20  review_scores_checkin           7889 non-null   float64
 21  review_scores_communication     7889 non-null   float64
 22  review_scores_location          7889 non-null   float64
 23  review_scores_value             7889 non-null   float64
 24  avg_air_quality                 8383 non-null   float64
 25  days_as_host                    8808 non-null   int64  
 26  bathrooms                       8808 non-null   float64
 27  short_term                      8808 non-null   int64  
dtypes: float64(14), int64(12), object(2)
memory usage: 1.9+ MB
```

13) Proposed Type of Missing Data:

Fill in the variables you believe fall under each type of missing data (it's fine if some missing data categories have no variables listed), and then briefly explain below why you think those variables are structurally missing, MCAR, MAR, or MNAR.

- Structurally Missing Variables: reviews_per_month, all review_scores
- Missing Completely at Random (MCAR) Variables: avg_air_quality
- Missing at Random (MAR) Variables: review_scores_rating, review_scores_accuracy, review_scores_cleanliness, review_scores_checkin, review_scores_communication, review_scores_location, review_scores_value
- Missing Not at Random (MNAR) Variables: price

explanation: structully missing is only missing only when there are no reviews, mcar appears to be missing at random. mar's are missing when the site hasn't received a review yet. price missingness shows a potential relationship with host_is_superhost, suggesting that missing

values might be intentional, potentially due to inactive listings, temporary delisting, or strategic price withholding.

14) Replacing Missing Values:

Start with the variables that have structurally missing data. What is an acceptable value (or set of values) to fill in for these observations, given that the data are structurally missing?

15) Based on your explanation above, proceed with filling the missing values for your structurally missing variables below:

In [68]:
```
# replace with code to fill in the missing values in the reviews_per_month column w
df2["reviews_per_month"].fillna(0, inplace=True)
print(df2["reviews_per_month"].isna().sum())
```

```
0
```

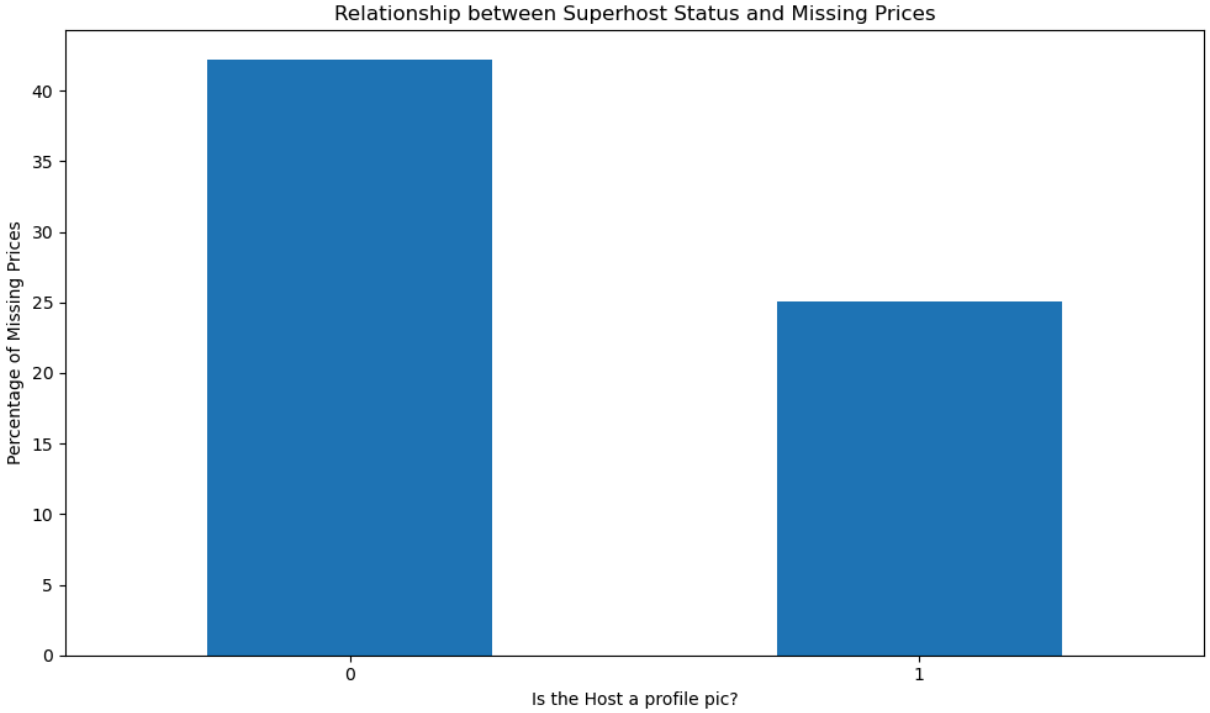
In [69]:
```
# this code can be used to fill in NaNs in all the columns starting with 'review_sc

review_scores_cols = [col for col in df2.columns if col.startswith("review_scores_"
for col in review_scores_cols:
    df2[col].fillna(-1, inplace=True)
```

16) Next, address any variables that have data that are MCAR or MAR. What differentiates variables that are missing *completely* at random (MCAR) from those missing at random?

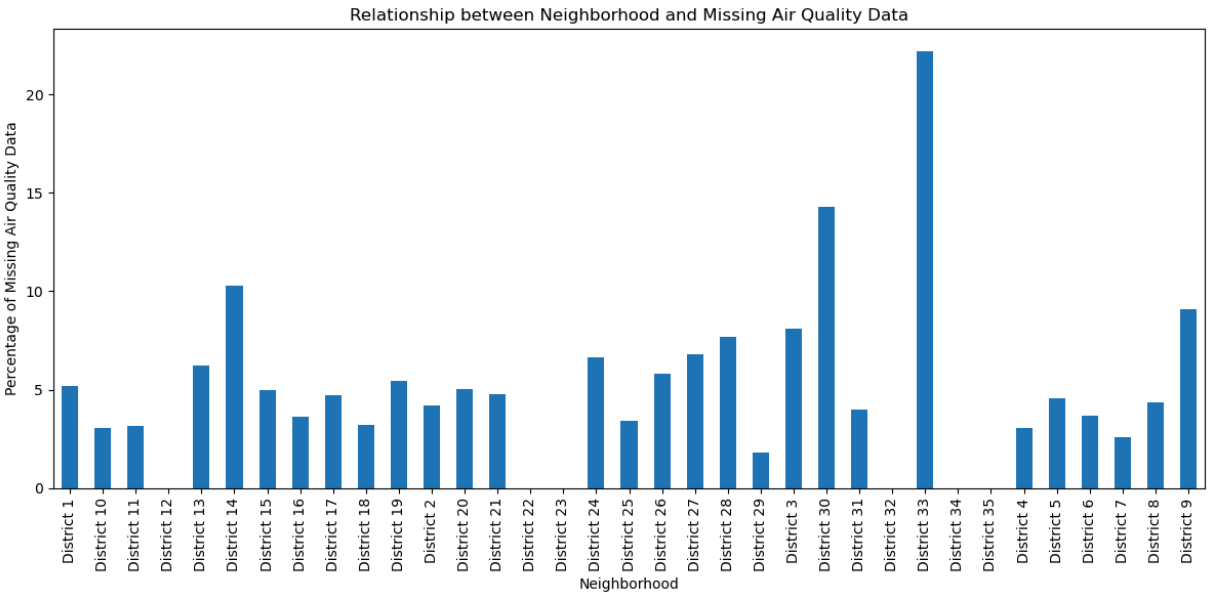
MCAR missingness is entirely random whereas a MAR missing reviews may be morer common for reviews with fewer reviews.

17) Use the code cells below to investigate any patterns of missingness for variables you think are MAR. *Hint: try plotting the relationship between missingness of the given variables against complete variables you think could influence missingness.*

In [70]:
```
"""replace with code used for plotting missingness of one variable against the valu
   Here is one example for `price` against `host_has_profile_pic`, you can copy thi

# Calculate the percentage of missing values in 'price' for each 'host_has_profile_
percentage_missing = df2.groupby('host_has_profile_pic')['price'].apply(lambda x: x

plt.figure(figsize=(10, 6))
percentage_missing.plot(kind='bar')
plt.xlabel('Is the Host a profile pic?')
plt.ylabel('Percentage of Missing Prices')
plt.title('Relationship between Superhost Status and Missing Prices')
plt.xticks(rotation=0)
plt.tight_layout()
plt.show()
```



In [71]:
```
percentage_missing_air_quality = df2.groupby("neighbourhood_cleansed")["avg_air_qua

# Create a bar plot
plt.figure(figsize=(12, 6))
percentage_missing_air_quality.plot(kind='bar')
plt.xlabel("Neighborhood")
plt.ylabel("Percentage of Missing Air Quality Data")
plt.title("Relationship between Neighborhood and Missing Air Quality Data")
plt.xticks(rotation=90)
plt.tight_layout()
plt.show()
```



18) Can you note any patterns to the missingness of these variables that could further indicate they are MAR?

- Notes about Price - related to whether the host is a super host. non-super hosts have a higher percentage of missing data...
- Notes about air quality - certain districts have substantially more missing reviews, such as 33, this could be due to small sample size or a frequent pattern of no avalible data

19) How should MAR observations be handled? How will you proceed based on your findings above?

for air quality we could do - df2["avg_air_quality"] = df2.groupby("neighbourhood_cleansed")["avg_air_quality"].apply(lambda x: x.fillna(x.median())) which takes the median airquality in the same neighborhood and fills in the information

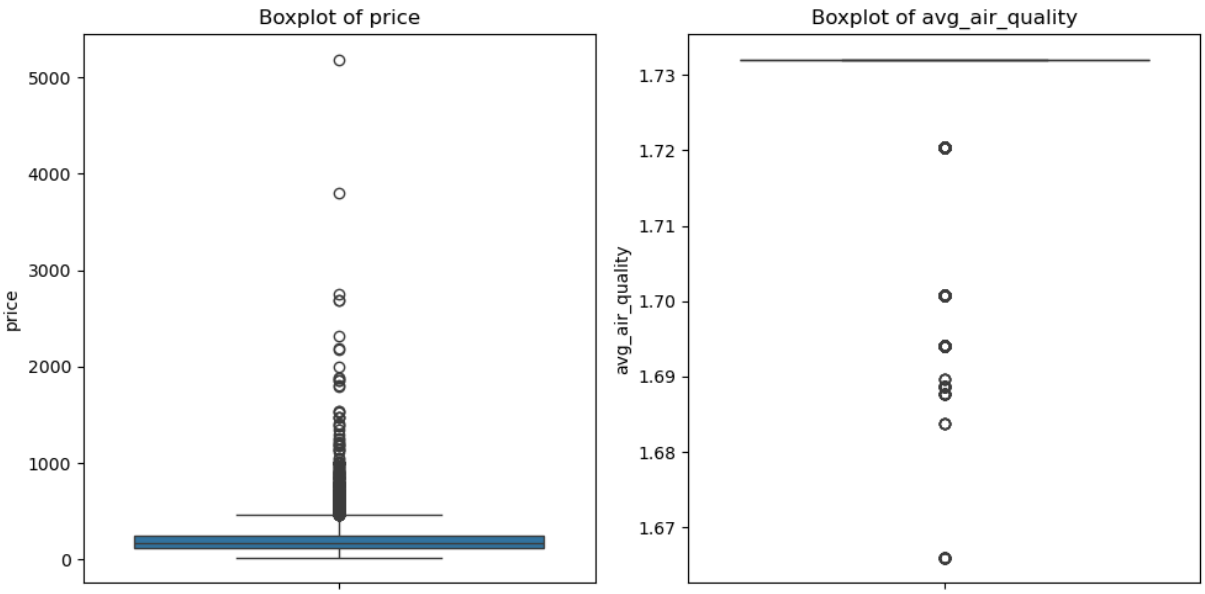
for price we could do - df2["price"] = df2.groupby("host_is_superhost")["price"].apply(lambda x: x.fillna(x.median())) also using the median or mean

Identifying & Addressing Outliers for Handling Missing Data

20) Before we can peform any imputation, let's make sure we don't have outliers first. Note down any variables we will be imputing that also containing outliers below...

- avg_air_quality
- price

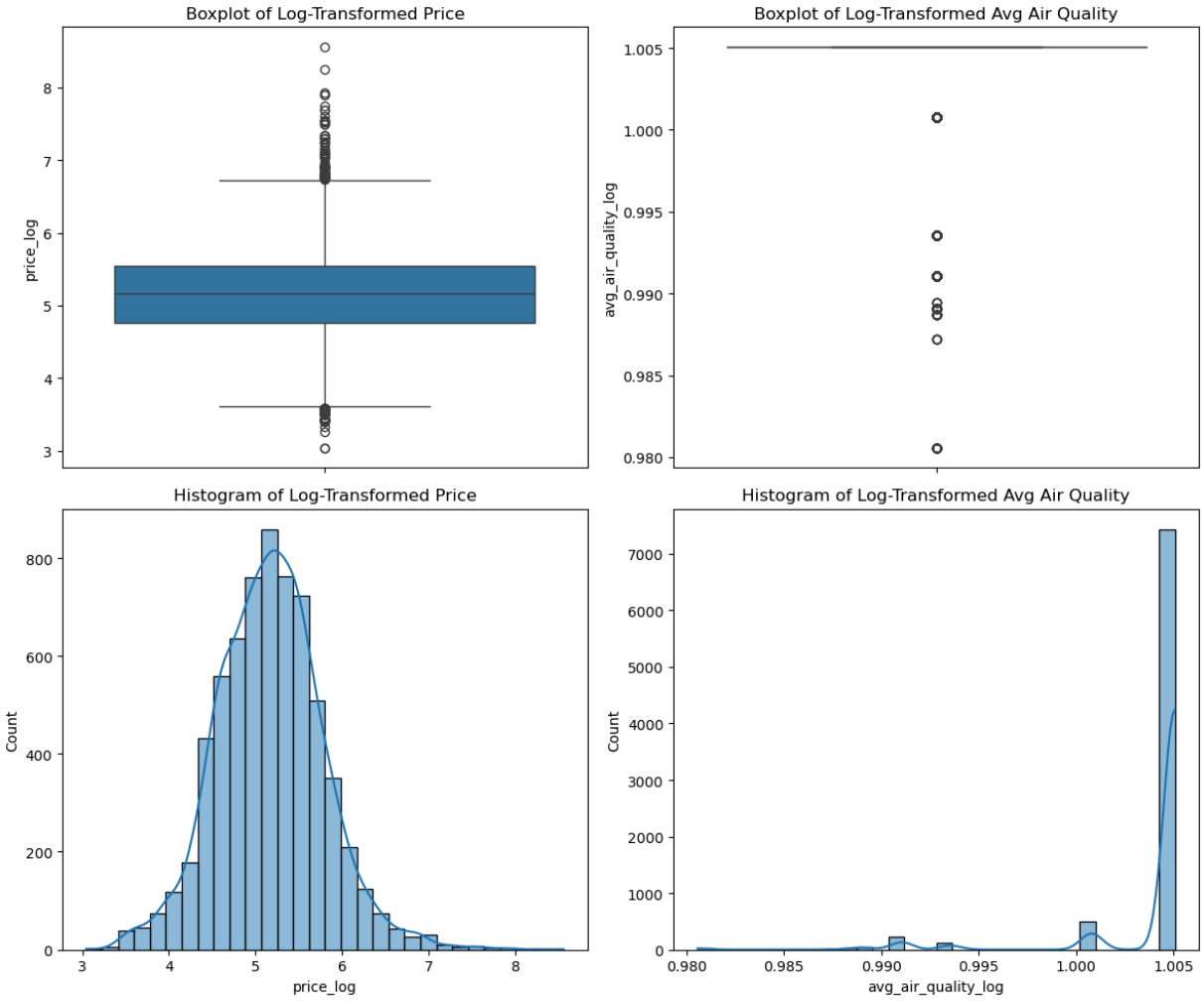
In [72]:
```
# replace with code to check variables for outliers (via a boxplot)
outlier_vars = ["price", "avg_air_quality"]

# Create boxplots for each variable
plt.figure(figsize=(10, 5))
for i, var in enumerate(outlier_vars, 1):
    plt.subplot(1, len(outlier_vars), i)
    sns.boxplot(y=df2[var])
    plt.title(f'Boxplot of {var}')
plt.tight_layout()
plt.show()
```



21) How can you address these outliers before proceeding with handling your missing data values?

just use median

In [73]:
```
# replace with code to log transform outliers for first variable
# replace with code to recheck for outliers via a boxplot
df2["price_log"] = np.log1p(df2["price"])
df2["avg_air_quality_log"] = np.log1p(df2["avg_air_quality"])

# Create a 2x2 grid for subplots
fig, axes = plt.subplots(2, 2, figsize=(12, 10))

# Boxplots
sns.boxplot(y=df2["price_log"], ax=axes[0, 0])
axes[0, 0].set_title("Boxplot of Log-Transformed Price")

sns.boxplot(y=df2["avg_air_quality_log"], ax=axes[0, 1])
axes[0, 1].set_title("Boxplot of Log-Transformed Avg Air Quality")

# Histograms
sns.histplot(df2["price_log"], bins=30, kde=True, ax=axes[1, 0])
axes[1, 0].set_title("Histogram of Log-Transformed Price")

sns.histplot(df2["avg_air_quality_log"], bins=30, kde=True, ax=axes[1, 1])
axes[1, 1].set_title("Histogram of Log-Transformed Avg Air Quality")

plt.tight_layout()
plt.show()
```



22) Did transforming the variable change the distribution? Is it roughly normal (yes or no), and what does that mean in terms of performing a simple imputation?

Yes - we can't check if its roughly normal with a box plot, that is flawed, we have done a histogram as well

23) Perform Imputation (simple or multiple) for the first variable below:

In [74]:
```
# replace with code to remove or impute first variable observations with missing da
df2["price"] = df2.groupby("neighbourhood_cleansed")["price"].transform(lambda x: x
```

24) Perform Imputation (simple or multiple) for the second variable below:

In [75]:
```
# repeat with additional variables, consider what type of imputation is appropriate
# AND patterns of missingness
df2["avg_air_quality"] = df2.groupby("neighbourhood_cleansed")["avg_air_quality"].t
```

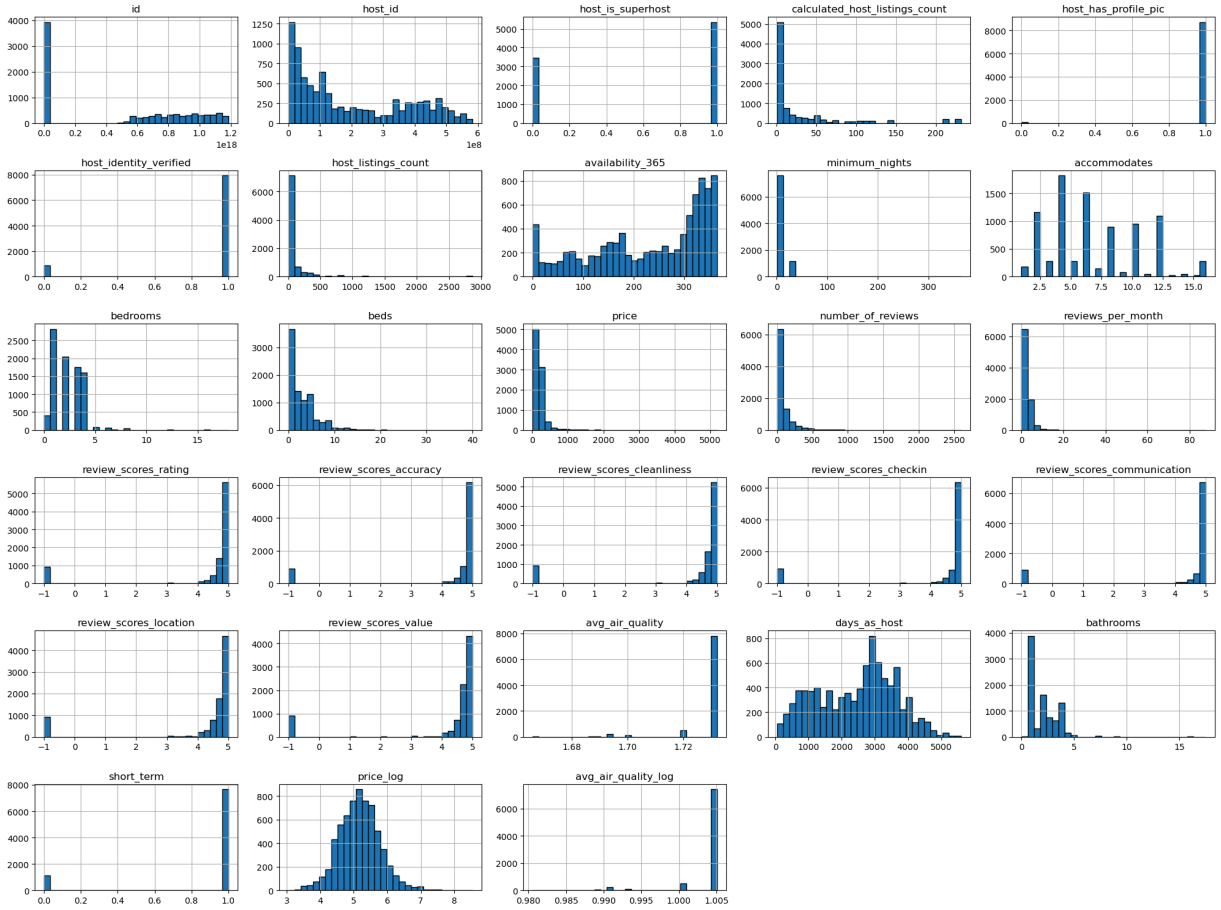
25) Are there any variables that could potentially be MNAR? When data are MNAR, what can be done about it?

air quality potentially

Additional Outliers

Identify any remaining outliers by using a grid of boxplots. Then, choose two variables with outliers to either transform or discretize. For each variable, explain what step you took and whether the step was successful in addressing the presence (of at least extreme) outliers.

In [78]:
```
df2.hist(figsize=(20, 15), bins=30, layout=(6, 5), edgecolor="black")

plt.tight_layout()
plt.show()
```

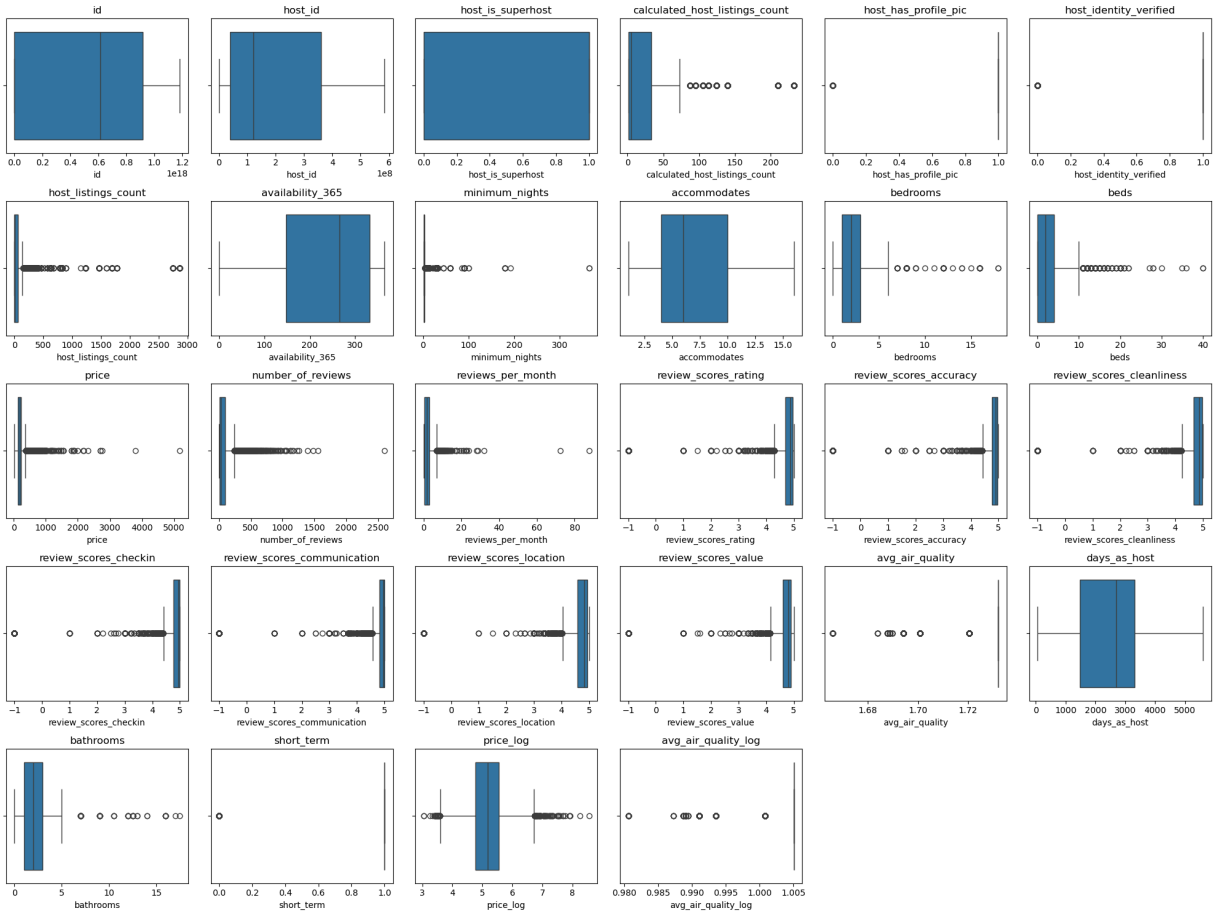


26) List the two variables you identified as having outliers that you want to transform/discretize here:

- review_scores_location
- beds

In [ ]:
```
# this code creates a grid of boxplots for the int and float variables

# Select numerical columns
numerical_cols = df2.select_dtypes(include=['int64', 'float64']).columns

plt.figure(figsize=(20, 15))
```

```
for i, col in enumerate(numerical_cols, 1):
    plt.subplot(5, 6, i)
    sns.boxplot(x=df2[col])
    plt.title(col)

plt.tight_layout()
plt.show()
```

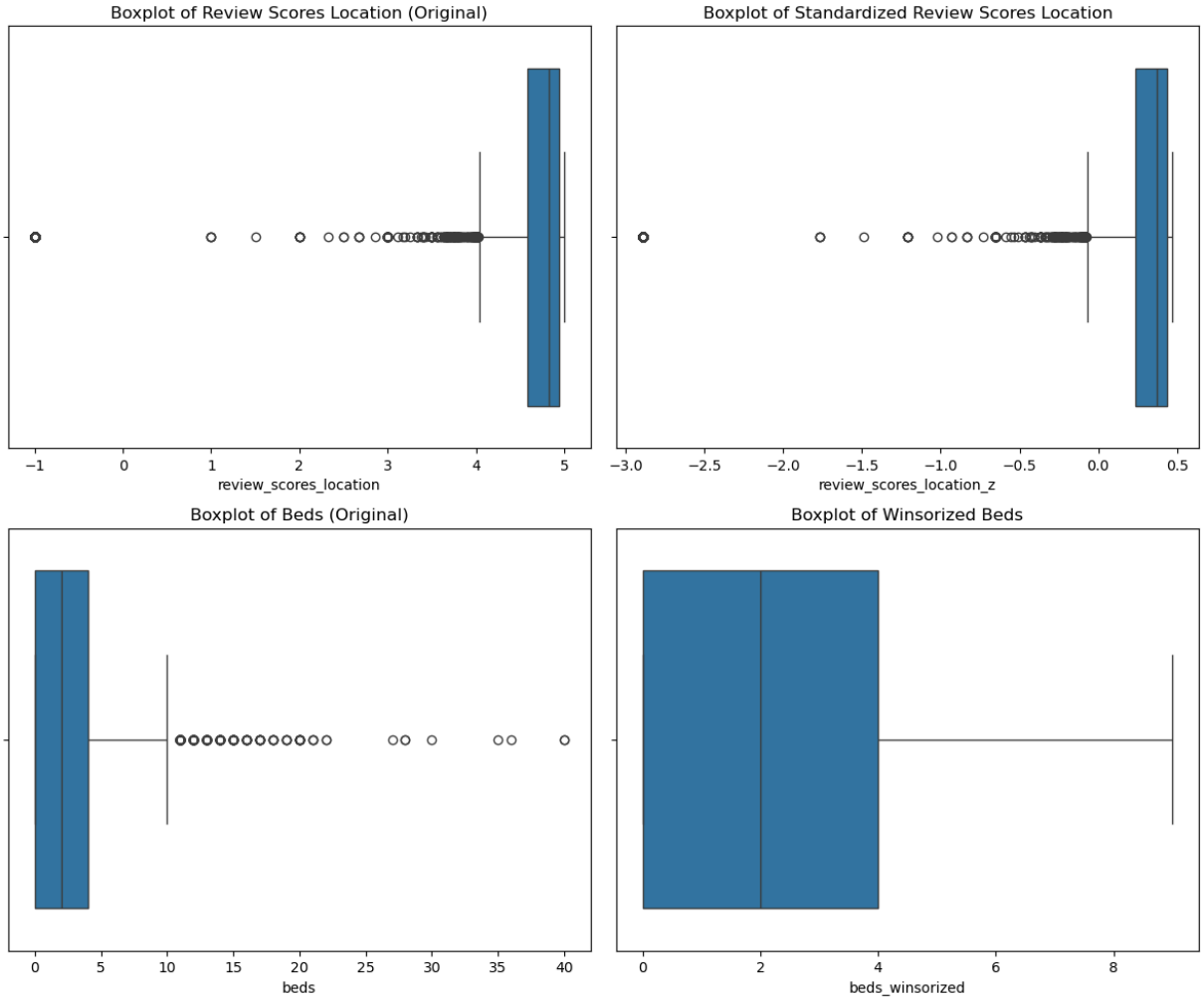


27) Perform a transformation/discretization on your first selected variable below, explain what step you took and whether it has reduced/fixed your outliers for this variable (hint: you can check with a boxplot or histogram to confirm).

In [84]:
```
# replace with code needed for addressing variable 1 with outliers
from scipy.stats import zscore
from scipy.stats.mstats import winsorize

df2["review_scores_location_z"] = zscore(df2["review_scores_location"])
df2["beds_winsorized"] = winsorize(df2["beds"], limits=[0.05, 0.05])
```

In [85]:
```
fig, axes = plt.subplots(2, 2, figsize=(12, 10))

# Boxplots
sns.boxplot(x=df2["review_scores_location"], ax=axes[0, 0])
axes[0, 0].set_title("Boxplot of Review Scores Location (Original)")

sns.boxplot(x=df2["review_scores_location_z"], ax=axes[0, 1])
axes[0, 1].set_title("Boxplot of Standardized Review Scores Location")
```

```
sns.boxplot(x=df2["beds"], ax=axes[1, 0])
axes[1, 0].set_title("Boxplot of Beds (Original)")

sns.boxplot(x=df2["beds_winsorized"], ax=axes[1, 1])
axes[1, 1].set_title("Boxplot of Winsorized Beds")

plt.tight_layout()
plt.show()
```



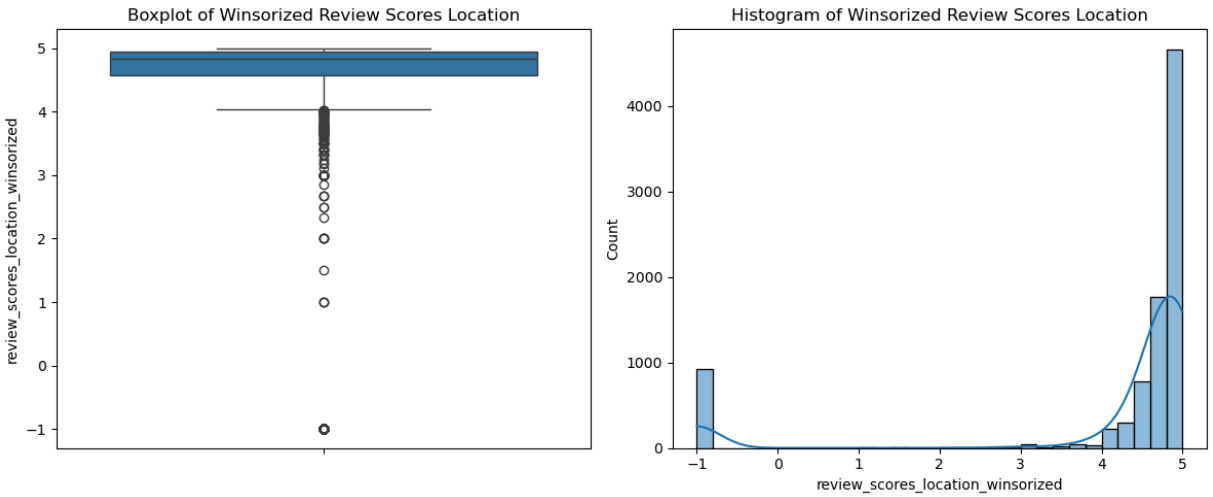
Explanation of steps taken and whether the steps worked: We used z-score standardization to transform review_scores_location, which allows us to identify outliers more easily by converting the values into standard deviations from the mean. However, we did not remove values—we only transformed them to make outliers more detectable.

For beds, we applied Winsorization, which caps extreme outliers at the 5th and 95th percentiles rather than removing them. This ensures that extreme values do not distort the dataset while maintaining a reasonable range.

Now, the data is in a better range, reducing the influence of extreme values while preserving the overall distribution.

28) Perform a transformation or discretization on your second selected variable. Explain what step you took and whether it has reduced/fixed your outliers for this variable (hint: you can check with a boxplot or histogram to confirm).

In [86]:
```
# replace with code needed for addressing variable 2 with outliers
df2["review_scores_location_winsorized"] = winsorize(df2["review_scores_location"],

# Plot boxplot and histogram to confirm outlier adjustment
fig, axes = plt.subplots(1, 2, figsize=(12, 5))

sns.boxplot(y=df2["review_scores_location_winsorized"], ax=axes[0])
axes[0].set_title("Boxplot of Winsorized Review Scores Location")

sns.histplot(df2["review_scores_location_winsorized"], bins=30, kde=True, ax=axes[1
axes[1].set_title("Histogram of Winsorized Review Scores Location")

plt.tight_layout()
plt.show()
```



Explanation of steps taken and whether the steps worked: We performed the same method as we did for the first variable

29) Guiding Next Steps

You are about to pass off this data for its next steps in modeling. 1) What should the data team know about problems you encountered in the dataset, and how you handled these issues? 2) Are there any unresolved issues they should be aware of? 3) Finally, provide a brief explanation from Parts 1 and 2 regarding interesting observations or hypotheses you noted, that they should continue to explore as they begin modeling.

- 1. Reviews can be null
- 2. There are outliers that make sense, such as properties that are really large.

- 3. Seeing what size properties are most common could be extremely important to look into.

In [ ]:
```
# export your finalized dataset and turn in with your submission!
df2.to_csv("cleansed.csv")
```