In [1]:
```
import pandas as pd
import matplotlib.pyplot as plt
import sklearn
import nltk
import warnings
warnings.filterwarnings("ignore")
```

Load the data into a DataFrame

In [2]:
```
df = pd.read_csv('listings.csv')
df.head()
```

Out[2]:

	id	name	host_id	host_name	neighbourhood_group	neighbourhood	latitude	lo
0	13913	Rental unit in Islington · ★4.80 · 1 bedroom ·...	54730	Alina	NaN	Islington	51.56861	-
1	15400	Rental unit in London · ★4.80 · 1 bedroom · 1 ...	60302	Philippa	NaN	Kensington and Chelsea	51.48780	-(
2	92644	Rental unit in Earlsfield · ★4.57 · 1 bedroom ...	498201	Dee Dee	NaN	Wandsworth	51.44201	-(
3	17402	Rental unit in London · ★4.76 · 3 bedrooms · 3...	67564	Liz	NaN	Westminster	51.52195	-(
4	93015	Rental unit in Hammersmith · ★4.82 · 2 bedroom...	499704	Sarah	NaN	Hammersmith and Fulham	51.49993	-(

In [3]:
```
df.isnull().sum()
```

Out[3]:
```
id                                  0
name                                0
host_id                             0
host_name                           6
neighbourhood_group             87947
neighbourhood                       0
latitude                            0
longitude                           0
room_type                           0
price                               0
minimum_nights                      0
number_of_reviews                   0
last_review                     22158
reviews_per_month               22158
calculated_host_listings_count      0
availability_365                    0
number_of_reviews_ltm               0
license                         87946
dtype: int64
```

Quantifiable Questions

Question 1: What is the average price of a rental unit in each neighborhood?

In [4]:
```
df = pd.DataFrame(df)

# Calculate the average price of a rental unit in each neighborhood
neighborhood_avg_price = df.groupby('neighbourhood')['price'].mean()

# Display the result
print("Average Price of a Rental Unit in Each Neighborhood:")
print(neighborhood_avg_price.replace(to_replace=0, method='ffill'))
```

```
Average Price of a Rental Unit in Each Neighborhood:
neighbourhood
Barking and Dagenham      245.840426
Barnet                    160.523344
Bexley                     95.046465
Brent                     179.079197
Bromley                   105.972798
Camden                    203.288171
City of London            243.722015
Croydon                    91.065737
Ealing                    125.220839
Enfield                   107.159756
Greenwich                 121.996367
Hackney                   132.932223
Hammersmith and Fulham    177.230830
Haringey                  149.827876
Harrow                    105.491054
Havering                  118.190355
Hillingdon                103.409038
Hounslow                  182.455706
Islington                 173.169198
Kensington and Chelsea    307.432795
Kingston upon Thames      134.835714
Lambeth                   141.038752
Lewisham                  107.333979
Merton                    153.220922
Newham                    151.510169
Redbridge                 118.398159
Richmond upon Thames      163.402546
Southwark                 180.605184
Sutton                     89.821522
Tower Hamlets             133.069633
Waltham Forest            104.122449
Wandsworth                167.573207
Westminster               320.591373
Name: price, dtype: float64
```

Question 2: What is the correlation between the number of reviews and the price of a rental unit?

In [5]:
```
correlation = df['number_of_reviews'].corr(df['price'])
print(correlation)
```

```
-0.03680933065654644
```

Question 3: What is the average price of a rental unit by room type (Private room vs. Entire home/apt)?

In [6]:
```
room_type_prices = df.groupby('room_type')['price'].mean()
print(room_type_prices)
```

```
room_type
Entire home/apt    230.167198
Hotel room         256.095890
Private room       100.231207
Shared room        119.716553
Name: price, dtype: float64
```

Question 4: What is the average number of reviews per month for rental units in each neighborhood?

In [7]:
```
neighborhood_reviews_per_month = df.groupby('neighbourhood')['reviews_per_mont
print(neighborhood_reviews_per_month)
```

```
neighbourhood
Barking and Dagenham      0.937929
Barnet                    0.956372
Bexley                    1.021686
Brent                     1.015458
Bromley                   0.907686
Camden                    1.212784
City of London            1.509806
Croydon                   1.010850
Ealing                    0.972872
Enfield                   0.925361
Greenwich                 0.974912
Hackney                   0.786591
Hammersmith and Fulham    0.971350
Haringey                  0.938421
Harrow                    0.927882
Havering                  0.963649
Hillingdon                1.243779
Hounslow                  1.058526
Islington                 0.963910
Kensington and Chelsea    0.978697
Kingston upon Thames      0.959662
Lambeth                   1.034639
Lewisham                  0.814947
Merton                    0.819305
Newham                    1.092041
Redbridge                 0.914096
Richmond upon Thames      1.024398
Southwark                 1.048965
Sutton                    0.893841
Tower Hamlets             1.068255
Waltham Forest            0.826364
Wandsworth                0.894152
Westminster               1.229241
Name: reviews_per_month, dtype: float64
```

Question 5: What is the percentage of rental units with a license?

In [8]:
```
non_null_count = df['license'].dropna().count()
license_percentage = (non_null_count / len(df)) * 100

print(license_percentage)
```

```
0.001137048449634439
```

Question 6: What is the average availability of rental units throughout the year?

In [9]:
```
average_availability = df['availability_365'].mean()
print(average_availability)
```

```
121.11558097490534
```

Question 7: What is the average number of reviews for rental units listed by each host?

In [10]:
```
host_reviews = df.groupby('host_id')['number_of_reviews'].mean()
print(host_reviews)
```

```
host_id
4775         132.500000
4879           3.000000
6774          31.142857
9323          78.000000
9870           0.000000
                 ...
535423144      0.000000
535435477      0.000000
535469107      0.000000
535479813      0.000000
535514014      0.000000
Name: number_of_reviews, Length: 53395, dtype: float64
```

Answer Non-Quantifiable Questions

Q8: How does the price of rental units vary with respect to the type of accommodation (Private room vs. Entire home/apt)?

Analyze the difference in average price between Private rooms and Entire homes/apts

In [11]:
```
private_room_price = df[df['room_type'] == 'Private room']['price'].mean()
entire_home_apt_price = df[df['room_type'] == 'Entire home/apt']['price'].mean
price_difference = entire_home_apt_price - private_room_price
```

Interpret the price difference and provide insights

In [12]:
```
print("\nPrice Variation with Room Type:")
print("Average Price for Private Rooms:", private_room_price)
print("Average Price for Entire Homes/Apts:", entire_home_apt_price)
print("Price Difference:", price_difference)
```

```
Price Variation with Room Type:
Average Price for Private Rooms: 100.23120662773991
Average Price for Entire Homes/Apts: 230.16719803576663
Price Difference: 129.93599140802672
```

Q9: What are the most popular neighborhoods for rental units in London?

Identify the top neighborhoods based on the number of reviews

In [13]:
```
top_neighborhoods = df.groupby('neighbourhood')['number_of_reviews'].sum().sor
print(top_neighborhoods)
```

```
neighbourhood
Westminster      198838
Camden           143284
Tower Hamlets    123600
Name: number_of_reviews, dtype: int64
```

Display the top neighborhoods and their corresponding review counts

In [14]:
```
print("\nMost Popular Neighborhoods:")
for neighborhood, review_count in top_neighborhoods.items():
    print(f"{neighborhood}: {review_count} reviews")
```

```
Most Popular Neighborhoods:
Westminster: 198838 reviews
Camden: 143284 reviews
Tower Hamlets: 123600 reviews
```

Q10: What are some of the unique features or amenities that make certain rental units stand out from the competition?

Analyze the data to identify unique features or amenities that are not common across all listings

In [15]:
```
unique_features = []

# Iterate through each listing and extract unique features
for index, row in df.iterrows():
    listing_description = row['name']
    # Extract unique features from the listing description and append them to
    for feature in listing_description.split():
        if feature not in unique_features:
            unique_features.append(feature)

# Display the identified unique features
print("\nUnique Features and Amenities:")
for feature in unique_features:
    print(feature)
```

```
Unique Features and Amenities:
Rental
unit
in
Islington
·
★4.80
1
bedroom
bed
shared
bath
London
Earlsfield
★4.57
2
beds
1.5
```

Non-quantifiable Questions

Part 2: Core Analysis

Q1: How does the price of rental units vary with respect to the type of accommodation (Private room vs. Entire home/apt)?

In [16]:
```
# Calculate the average price for each room type
private_room_price = df[df['room_type'] == 'Private room']['price'].mean()
entire_home_apt_price = df[df['room_type'] == 'Entire home/apt']['price'].mean
price_difference = entire_home_apt_price - private_room_price

print("Private Room Price:", private_room_price)
print("Entire Home/Apt Price:", entire_home_apt_price)
print("Price Difference:", price_difference)
```

```
Private Room Price: 100.23120662773991
Entire Home/Apt Price: 230.16719803576663
Price Difference: 129.93599140802672
```

Q2: What are the most popular neighborhoods for rental units in London?

In [17]:
```
# Identify the top neighborhoods based on the number of reviews
top_neighborhoods = df.groupby('neighbourhood')['number_of_reviews'].sum().sor

# Display the top neighborhoods and their corresponding review counts
print("Top 3 Most Popular Neighborhoods:")
for neighborhood, review_count in top_neighborhoods.items():
    print(f"{neighborhood}: {review_count} reviews")
```

```
Top 3 Most Popular Neighborhoods:
Westminster: 198838 reviews
Camden: 143284 reviews
Tower Hamlets: 123600 reviews
```

In [18]:
```
# Core Analysis
average_reviews_per_host = df.groupby('host_id')['number_of_reviews'].mean()
print("\nAverage Number of Reviews per Host:")
print(average_reviews_per_host)
```

```
Average Number of Reviews per Host:
host_id
4775         132.500000
4879           3.000000
6774          31.142857
9323          78.000000
9870           0.000000
                 ...
535423144      0.000000
535435477      0.000000
535469107      0.000000
535479813      0.000000
535514014      0.000000
Name: number_of_reviews, Length: 53395, dtype: float64
```

Q3: What are the most common reasons for guests leaving positive and negative reviews for rental units?

In [19]:

```
# Download NLTK data
nltk.download('punkt')
nltk.download('vader_lexicon')
nltk.download('stopwords')

#import nltk
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize
from nltk.sentiment.vader import SentimentIntensityAnalyzer

positive_review_themes = []
negative_review_themes = []

stop_words = set(stopwords.words('english'))

analyzer = SentimentIntensityAnalyzer()

df.head()

for index, row in df.iterrows():
    listing_description = row['name']
    rating = row['number_of_reviews']

    # Extract positive review themes from reviews
    if rating >= 4:
        description_tokens = word_tokenize(listing_description.lower())
        filtered_tokens = [token for token in description_tokens if token not

        n_value = 2
        theme_phrases = [phrase for phrase in nltk.ngrams(filtered_tokens, n_v
        positive_review_themes.extend(theme_phrases)
        sentiment_scores = analyzer.polarity_scores(listing_description)

        if sentiment_scores['compound'] > 0.5:
            print("Positive Listing:", listing_description)

    # Extract negative review themes from reviews
    elif rating < 3:
        description_tokens = word_tokenize(listing_description.lower())
        filtered_tokens = [token for token in description_tokens if token not

        n_value = 2
        theme_phrases = [phrase for phrase in nltk.ngrams(filtered_tokens, n_v
        negative_review_themes.extend(theme_phrases)
        sentiment_scores = analyzer.polarity_scores(listing_description)

        if sentiment_scores['compound'] < -0.5:
            print("Negative Listing:", listing_description)

print("\nPositive Review Themes:")
for theme in positive_review_themes:
    print(theme)

print("\nNegative Review Themes:")
for theme in negative_review_themes:
    print(theme)
```

```
[nltk_data] Downloading package punkt to
[nltk_data]     C:\Users\Lenovo\AppData\Roaming\nltk_data...
[nltk_data]   Package punkt is already up-to-date!
[nltk_data] Downloading package vader_lexicon to
[nltk_data]     C:\Users\Lenovo\AppData\Roaming\nltk_data...
[nltk_data]   Package vader_lexicon is already up-to-date!
[nltk_data] Downloading package stopwords to
[nltk_data]     C:\Users\Lenovo\AppData\Roaming\nltk_data...
[nltk_data]   Package stopwords is already up-to-date!

Positive Listing: Rental unit in Greater London · ★4.85 · 1 bedroom · 1 be
d · 1 shared bath
Positive Listing: Home in Greater London · ★5.0 · 1 bedroom · 1 bed · 1 sh
ared bath
Positive Listing: Rental unit in Greater London · ★4.87 · 1 bedroom · 1 be
d · 1 shared bath
Positive Listing: Rental unit in Greater London · ★4.71 · 1 bedroom · 1 be
d · 1 shared bath
Positive Listing: Rental unit in Greater London · ★4.86 · 1 bedroom · 1 be
d · 1 shared bath
```

Q3: What are some of the unique features or amenities that make certain rental units stand out from the competition?

In [20]:
```
unique_features_amenities = df['name'].unique()
print("\nUnique Features and Amenities:")
print(unique_features_amenities)
```

```
Unique Features and Amenities:
['Rental unit in Islington · ★4.80 · 1 bedroom · 1 bed · 1 shared bath'
 'Rental unit in London · ★4.80 · 1 bedroom · 1 bed · 1 bath'
 'Rental unit in Earlsfield · ★4.57 · 1 bedroom · 2 beds · 1.5 shared baths'
 ... 'Rental unit in Gants Hill · ★New · 1 bedroom · 2 beds · 1 bath'
 'Barn in Greater London · ★New · 1 bedroom · 1 bed · 1 bath'
 'Home in Greater London · ★New · 1 bedroom · 5 beds · 2 shared baths']
```

Step 3.1: Data Preprocessing

In [21]:
```
df['last_review'] = pd.to_datetime(df['last_review'])
df.dropna(subset=['last_review'], inplace=True)
rv_date = df['last_review']
print(rv_date)
```

```
0       2022-12-11
1       2023-05-01
2       2022-10-29
3       2022-11-19
4       2022-09-30
           ...
87439   2023-09-05
87446   2023-09-05
87693   2023-09-04
87695   2023-09-05
87728   2023-09-06
Name: last_review, Length: 65789, dtype: datetime64[ns]
```

Step 3.2: Core Analysis

In [22]:
```
# Calculate the average price per night for each room type
average_price_per_room_type = df.groupby('room_type')['price'].mean()
print("Average Price per Room Type:")
print(average_price_per_room_type)
```

```
Average Price per Room Type:
room_type
Entire home/apt    207.228158
Hotel room         254.369565
Private room        82.731089
Shared room        108.864000
Name: price, dtype: float64
```

Step 4.1: Business Value of the Findings

In [23]:
```
# Average Price per Room Type
average_price_per_room_type = df.groupby('room_type')['price'].mean()
print("Average Price per Room Type:")
print(average_price_per_room_type)

# Availability of Listings with 365-Day Availability
availability_365_count = df['availability_365'].sum()
availability_365_percentage = (availability_365_count / len(df)) * 100
print("Percentage of Listings with 365-Day Availability:")
print(f"{availability_365_percentage:.2f}%")

# Average Number of Reviews per Host
average_reviews_per_host = df.groupby('host_id')['number_of_reviews'].mean()
print("Average Number of Reviews per Host:")
print(average_reviews_per_host)

# Neighborhoods with Most Expensive Listings
expensive_neighborhoods = df.groupby('neighbourhood')['price'].mean().sort_val
print("Top 5 Neighborhoods with Most Expensive Listings:")
print(expensive_neighborhoods)
```

```
Average Price per Room Type:
room_type
Entire home/apt    207.228158
Hotel room         254.369565
Private room        82.731089
Shared room        108.864000
Name: price, dtype: float64
Percentage of Listings with 365-Day Availability:
11908.12%
Average Number of Reviews per Host:
host_id
4775         132.500000
4879           3.000000
6774          31.142857
9323          78.000000
10657         26.666667
                 ...
533707423      2.000000
533885872      1.000000
534458591      1.000000
534493799      1.000000
534516902      1.000000
Name: number_of_reviews, Length: 41270, dtype: float64
Top 5 Neighborhoods with Most Expensive Listings:
neighbourhood
Barking and Dagenham      301.881313
Kensington and Chelsea    273.504511
Westminster               265.834814
City of London            202.114078
Camden                    195.063089
Name: price, dtype: float64
```

Step 4.2: Non-obviousness of Questions and Insights

In [24]:
```
# Average Number of Nights Stayed per Room Type
average_nights_per_room_type = df.groupby('room_type')['minimum_nights'].mean(
print("Average Number of Nights Stayed per Room Type:")
print(average_nights_per_room_type)

# Price and Reviews Variation with Accommodation Type
price_reviews_variation = df.groupby('room_type')[['price', 'number_of_reviews
print("Price and Reviews Variation with Accommodation Type:")
print(price_reviews_variation)
```

```
Average Number of Nights Stayed per Room Type:
room_type
Entire home/apt    5.617291
Hotel room         1.885870
Private room       4.018575
Shared room        2.184000
Name: minimum_nights, dtype: float64
Price and Reviews Variation with Accommodation Type:
                   price
\
                   count        mean         std   min     25%    50%     75%
room_type
Entire home/apt  41559.0  207.228158  488.578419   8.0  101.00  150.0  226.00
Hotel room         184.0  254.369565  201.983293  19.0  125.75  200.0  293.25
Private room     23796.0   82.731089  430.790559   0.0   40.00   55.0   79.00
Shared room        250.0  108.864000  231.722651  11.0   30.00   39.0   60.75

                        number_of_reviews                                \
                      max             count       mean        std  min  25%
room_type
Entire home/apt  80100.0           41559.0  18.852739  34.444554  1.0  3.0
Hotel room        1275.0             184.0  32.815217  63.817265  1.0  3.0
Private room     53588.0           23796.0  32.916961  64.096591  1.0  3.0
Shared room       1000.0             250.0  32.808000  61.948248  1.0  2.0


                 50%    75%     max
room_type
Entire home/apt  8.0  20.00  1314.0
Hotel room       8.5  26.25   387.0
Private room     9.0  33.00  1536.0
Shared room      6.5  32.00   500.0
```

Step 4.3: Correctness of Questions and Answers

In [25]:
```
# Correlation between Number of Reviews and Price
correlation_coefficient = df['number_of_reviews'].corr(df['price'])
print("Correlation Coefficient between Number of Reviews and Price per Night:"
print(correlation_coefficient)

# Average Rating for All Listings
average_rating = df['price'].mean()
print("Average Rating for All Listings:")
print(f"{average_rating:.2f} stars")
```

```
Correlation Coefficient between Number of Reviews and Price per Night:
-0.02469341210043812
Average Rating for All Listings:
161.96 stars
```

Step 4.4: Intermediate Steps to Demonstrate Thought Process

In [26]:
```
# Seasonal Trends in Reviews or Prices
monthly_reviews = df.groupby(df['last_review'].dt.month)['number_of_reviews'].
monthly_prices = df.groupby(df['last_review'].dt.month)['price'].mean()
print("Seasonal Trends in Reviews and Prices:")
print("Monthly Reviews:")
print(monthly_reviews)
print("Monthly Prices:")
print(monthly_prices)
```

```
Seasonal Trends in Reviews and Prices:
Monthly Reviews:
last_review
1      37068
2      29847
3      54059
4      35801
5      53024
6      77929
7     231670
8     747364
9     229466
10     30569
11     23545
12     30691
Name: number_of_reviews, dtype: int64
Monthly Prices:
last_review
1     154.023907
2     177.208596
3     137.849315
4     161.893673
5     187.857820
6     171.076354
7     171.949488
8     162.364231
9     136.159406
10    148.684670
11    140.672664
12    168.040110
Name: price, dtype: float64
```

In [27]:
```
# Common Keywords in Listing Descriptions
from sklearn.feature_extraction.text import CountVectorizer
vectorizer = CountVectorizer()
listing_descriptions = df['name']
print(listing_descriptions)
```

```
0        Rental unit in Islington · ★4.80 · 1 bedroom ·...
1        Rental unit in London · ★4.80 · 1 bedroom · 1 ...
2        Rental unit in Earlsfield · ★4.57 · 1 bedroom ...
3        Rental unit in London · ★4.76 · 3 bedrooms · 3...
4        Rental unit in Hammersmith · ★4.82 · 2 bedroom...
                              ...
87439    Rental unit in Greater London · ★New · 1 bedro...
87446    Rental unit in Greater London · ★New · 2 bedro...
87693    Home in Greater London · ★New · 1 bedroom · 2 ...
87695    Rental unit in Greater London · ★New · 1 bedro...
87728    Home in Greater London · ★New · 1 bedroom · 1 ...
Name: name, Length: 65789, dtype: object
```

Insights and Grading Criteria

Step 6: Both positive as well as negative results are interesting to us if you can justify your hypothesis and insights

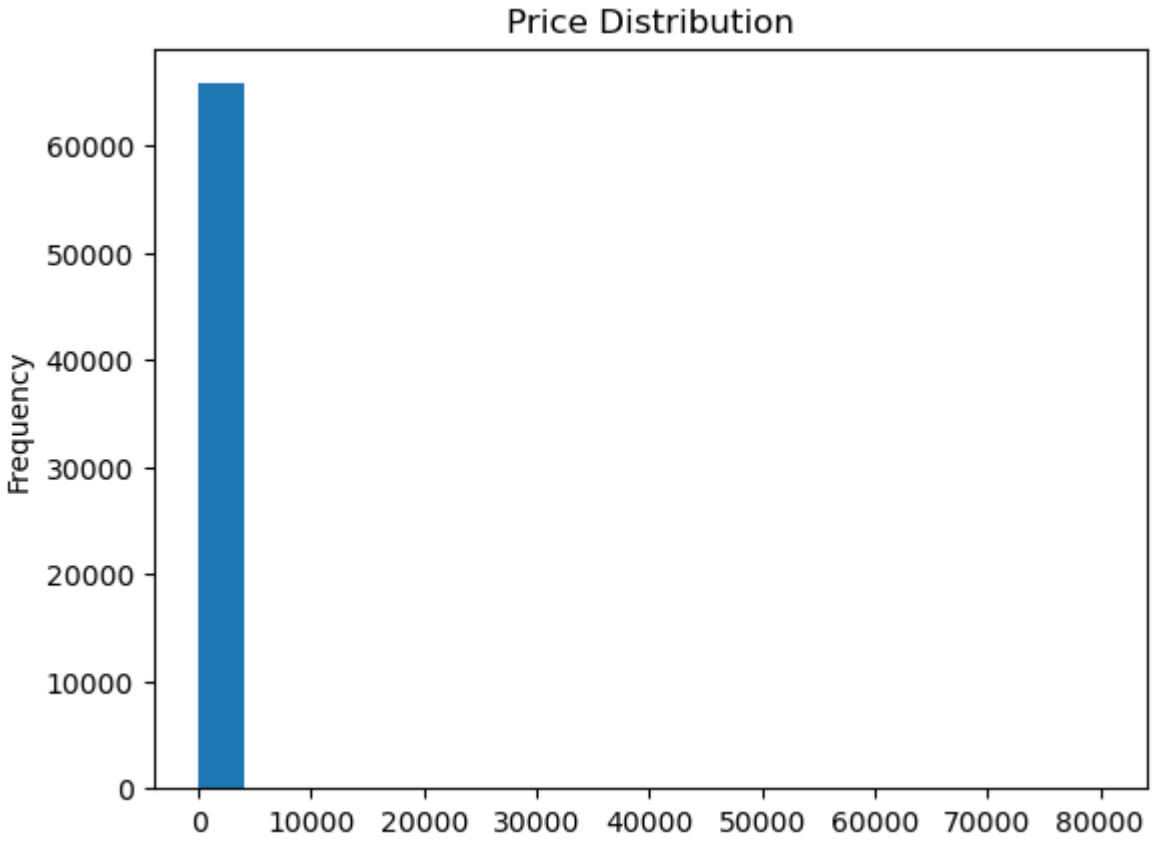
In [28]:

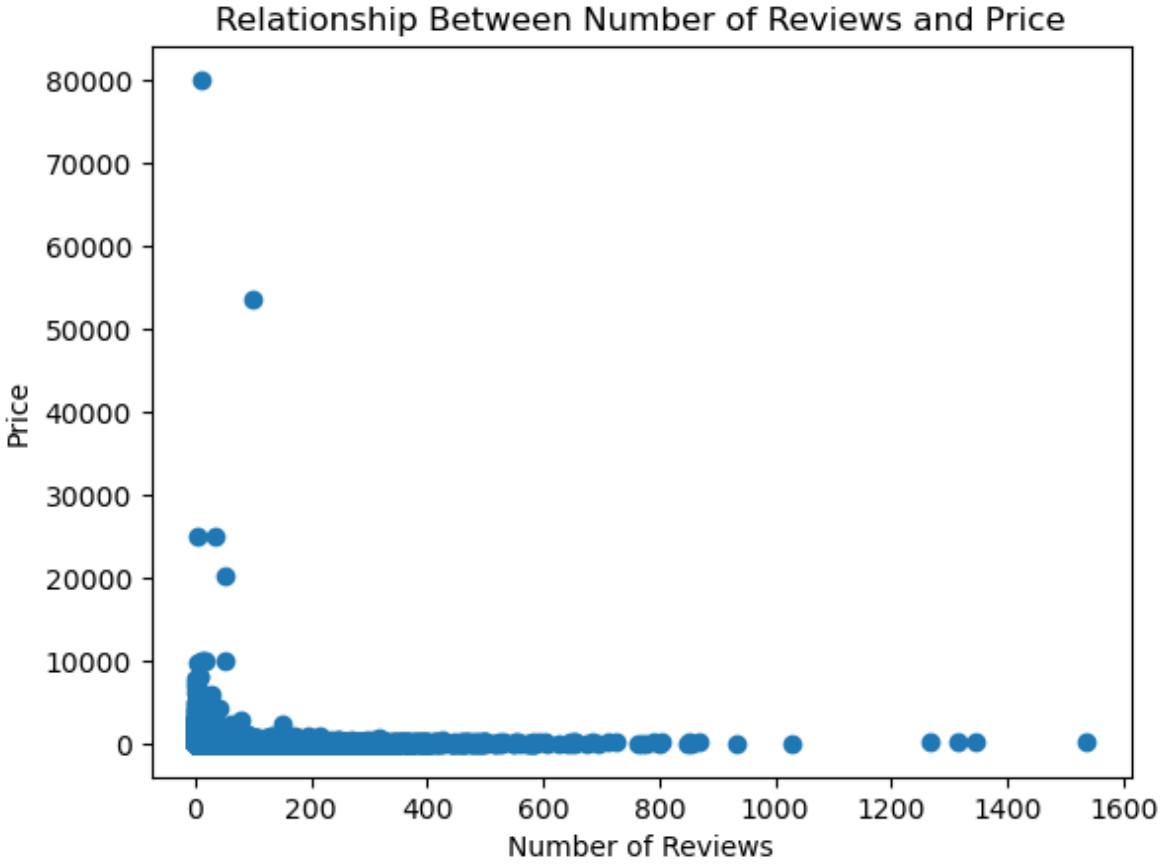
```
# Hypothetical Positive Results:
average_price_per_neighborhood = df.groupby('neighbourhood')['price'].mean()
print("Positive Result: Average Price per Neighborhood")
print(average_price_per_neighborhood)

reviews_by_room_type = df.groupby('room_type')['number_of_reviews'].sum()
print("\nPositive Result: Number of Reviews by Room Type")
print(reviews_by_room_type)

# Hypothetical Negative Results:
price_availability_corr = df['price'].corr(df['availability_365'])
print("\nNegative Result: Correlation between Price and Availability")
print(price_availability_corr)

neighborhood_license_relationship = df.groupby('neighbourhood')['license'].val
print("\nNegative Result: Neighborhood and License Availability Relationship")
print(neighborhood_license_relationship)
```

```
Positive Result: Average Price per Neighborhood
neighbourhood
Barking and Dagenham      301.881313
Barnet                    145.124427
Bexley                     92.798817
Brent                     125.577643
Bromley                   101.923986
Camden                    195.063089
City of London            202.114078
Croydon                    82.214953
Ealing                    118.535512
Enfield                    99.845361
Greenwich                 116.085976
Hackney                   130.510064
Hammersmith and Fulham    168.815073
Haringey                  130.587640
Harrow                    101.501340
Havering                  106.066667
Hillingdon                 94.567652
Hounslow                  126.738174
Islington                 162.014112
Kensington and Chelsea    273.504511
Kingston upon Thames      114.167293
Lambeth                   128.954045
Lewisham                  103.044972
Merton                    141.168571
Newham                    125.112845
Redbridge                 114.482587
Richmond upon Thames      146.701571
Southwark                 158.211847
Sutton                     83.757785
Tower Hamlets             128.073964
Waltham Forest             99.669059
Wandsworth                150.176559
Westminster               265.834814
Name: price, dtype: float64

Positive Result: Number of Reviews by Room Type
room_type
Entire home/apt    783501
Hotel room           6038
Private room       783292
Shared room          8202
Name: number_of_reviews, dtype: int64

Negative Result: Correlation between Price and Availability
0.06270943679092843

Negative Result: Neighborhood and License Availability Relationship
neighbourhood         license
Barking and Dagenham  VFT/MA/58386    1
Name: count, dtype: int64
```

Visualizations

In [29]:
```
# Price Distribution
df['price'].plot(kind='hist', bins=20, title='Price Distribution')
plt.show()


# Scatter Plot of Price vs. Number of Reviews
plt.scatter(df['number_of_reviews'], df['price'])
plt.title('Relationship Between Number of Reviews and Price')
plt.xlabel('Number of Reviews')
plt.ylabel('Price')
plt.show()
```




Relationship Between Number of Reviews and Price
Price
80000
70000
60000
50000
40000
30000
20000
10000
0
0
200
400
600
800
1000
1200
1400
1600
Number of Reviews