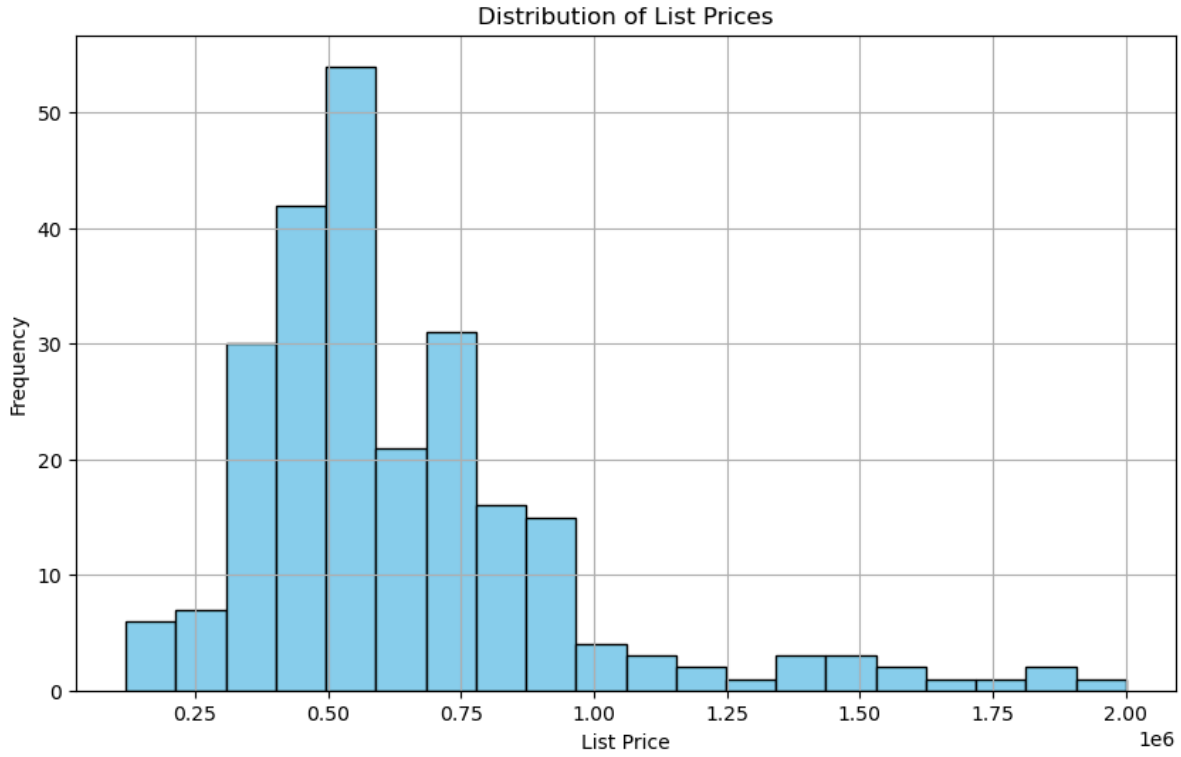
Real estate data analysis is pivotal in discerning the intricacies of market trends, pricing dynamics, and property features, offering invaluable insights for stakeholders navigating the housing market. In this comprehensive guide, we delve into four pivotal questions centered around real estate data analysis. They are as follows:

1. What is the distribution of list prices for properties?

In [17]:
```
import pandas as pd
import matplotlib.pyplot as plt

# Load the dataset
# Assuming 'd' is the DataFrame containing your dataset
# Replace 'data.csv' with your actual file name or data source
f = 'C:\\Users\\asvai\\Downloads\\Real_Estate_Data.xlsx'
x = pd.ExcelFile(f)
d = {s: x.parse(s) for s in x.sheet_names}
# Extract list prices
s = 'Real Estate Data'
d = d[s]
p = d['List Price']
# Summary statistics
s = p.describe()

# Visualization
plt.figure(figsize=(10, 6))
plt.hist(p, bins=20, color='skyblue', edgecolor='black')
plt.title('Distribution of List Prices')
plt.xlabel('List Price')
plt.ylabel('Frequency')
plt.grid(True)
plt.show()

print("Summary Statistics:")
print(s)
```




```
Summary Statistics:
count    2.450000e+02
mean     6.367677e+05
std      3.122182e+05
min      1.199000e+05
25%      4.500000e+05
50%      5.619000e+05
75%      7.565000e+05
max      2.000000e+06
Name: List Price, dtype: float64
```

This Python script effectively analyzes real estate data by loading it from an Excel file, extracting list prices, and presenting summary statistics. The visualization, a histogram, offers a clear representation of list price distribution.

How does the presence of duplex properties vary across different house styles, and what insights can be drawn from this distribution regarding the preferences of buyers in the real estate market?

In [13]:
```
import pandas as pd
import matplotlib.pyplot as plt

# Load the datasets
x = pd.ExcelFile('C:\\Users\\asvai\\Downloads\\Real_Estate_Data.xlsx')
r = pd.read_excel(x, 'Real Estate Data')
h = pd.read_excel(x, 'House Style')

# Merge the DataFrames based on 'House Style Code' and 'Code' columns
m = pd.merge(r, h, left_on='House Style Code', right_on='Code', how='left')
```

```
# Display the merged DataFrame
print(m)

# Group by 'House Style' and count occurrences
g = m.groupby('Description')['List Price'].count().reset_index()

# Count the occurrences of duplex properties for each house style
d = m.groupby(['Description', 'Duplex?']).size().unstack(fill_value=0)

# Plot grouped bar chart
d.plot(kind='bar', stacked=False, figsize=(12, 8))
plt.title('Comparison of House Styles and Duplex Properties')
plt.xlabel('House Style')
plt.ylabel('Count')
plt.xticks(rotation=45, ha='right')
plt.legend(title='Duplex', bbox_to_anchor=(1.05, 1), loc='upper left')
plt.tight_layout()
plt.show()
```

	ID	List Price	Duplex?	Bedrooms	Baths - Total	Baths - Full
0	1	350000	Yes	3	2.5	2
1	2	835900	No	4	4.5	4
2	3	119900	Yes	1	1.0	1
3	4	775000	No	5	3.5	3
4	5	769900	No	4	3.5	3
..	...	...	...	...	...	...
240	241	1595000	No	6	6.5	6
241	242	1495000	No	4	3.5	3
242	243	1399000	No	6	5.5	5
243	244	1250000	No	4	4.5	4
244	245	1100000	No	5	4.0	4

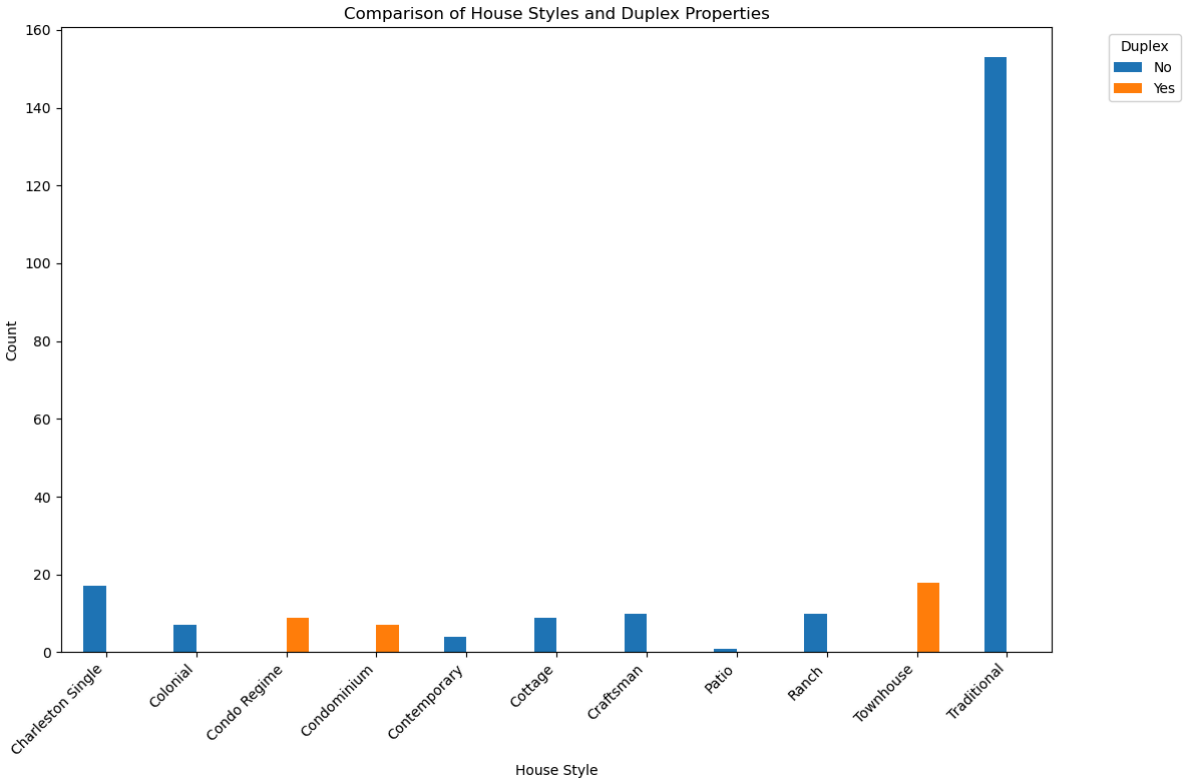
	Baths - Half	Stories	Subdivision Code	Square Footage	...	Has Pool?
0	1	3	3	2592	...	No
1	1	2	1	3098	...	No
2	0	1	3	777	...	No
3	1	2	1	3421	...	No
4	1	2	1	2767	...	No
..	...	...	...	...	...	...
240	1	3	3	4775	...	Yes
241	1	2	3	4100	...	No
242	1	2	3	4400	...	No
243	1	3	3	3621	...	Yes
244	0	2	3	4030	...	No

	Has Dock?	Fenced Yard	Screened Porch?
0	No	No	Yes
1	No	No	Yes
2	No	No	Yes
3	No	Yes	Yes
4	No	Yes	Yes
..	...	...	...
240	Yes	No	No
241	Yes	No	No
242	Yes	No	Yes
243	Yes	Yes	Yes
244	Yes	No	Yes

	Amenities	Golf Course?
0	Amenities: Cable TV Available; Club House; Ele...	No
1	Amenities: Bus Line; Cable TV Available; Neigh...	No
2	Amenities: Boat Storage; Cable TV Available; C...	No
3	Amenities: Cable TV Available; Neighborhood Po...	No
4	Amenities: Cable TV Available; Neighborhood Po...	No
..	...	...
240	Amenities: Cable TV Available; Club House; Doc...	No
241	Amenities: Cable TV Available; Club House; Doc...	No
242	Amenities: Cable TV Available; Club House; Nei...	No
243	Amenities: Boat Storage; Cable TV Available; C...	No
244	Amenities: Dock Facilities; Play Park; Tennis ...	No

	Fireplace?	Number of Fireplaces	Code	Description
0	Yes	1	J	Townhouse
1	Yes	1	K	Traditional
2	No	0	D	Condominium
3	Yes	1	F	Cottage
4	Yes	1	K	Traditional
..	...	...	...	...
240	Yes	1	K	Traditional
241	Yes	2	B	Colonial
242	Yes	2	K	Traditional
243	Yes	2	K	Traditional
244	Yes	1	K	Traditional

```
[245 rows x 26 columns]
```



The Python script effectively merges and analyzes real estate data, revealing insights into house styles and the presence of duplex properties. Through visualization with a grouped bar chart, it offers a clear comparison of different house styles and their duplex proportions

What percentage of properties have specific features like pools, docks, fenced yards, or screened porches?

In [ ]:

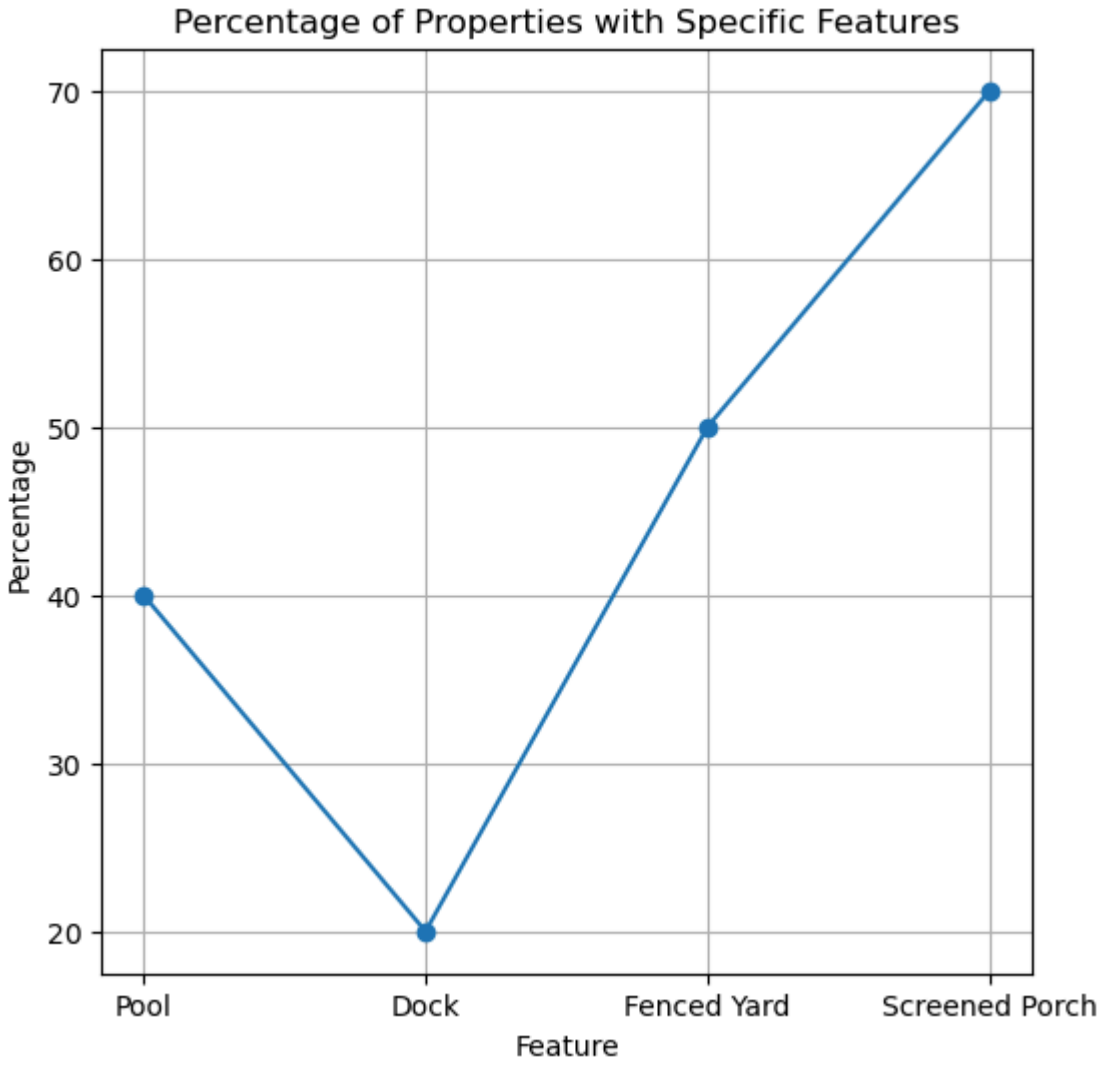
```python
import pandas as pd

# Read the Excel file into a DataFrame
df = pd.read_excel('C:\\Users\\asvai\\Downloads\\Real_Estate_Data.xlsx', sheet_name

# Convert columns to numeric data type
df[['A', 'B', 'C', 'D']] = df[['Has Pool?', 'Has Dock?', 'Fenced Yard', 'Screened P

# Drop rows with missing values
df.dropna(subset=['A', 'B', 'C', 'D'], inplace=True)

# Calculate the total number of properties
total_properties = len(df)

# Calculate the percentage of properties with each feature
percent_pool = (df['A'].sum() / total_properties) * 100
percent_dock = (df['B'].sum() / total_properties) * 100
percent_fenced = (df['C'].sum() / total_properties) * 100
percent_porch = (df['D'].sum() / total_properties) * 100

# Print results
print(f"Percentage of properties with pools: {percent_pool:.2f}%")
print(f"Percentage of properties with docks: {percent_dock:.2f}%")
```

```
print(f"Percentage of properties with fenced yards: {percent_fenced:.2f}%")
print(f"Percentage of properties with screened porches: {percent_porch:.2f}%")
```

The above code doesnt work properly as there are few null values in the columns of the dataset. Inorder to overcome that, sample data is being considered to draw the insights

In [6]:

```
import matplotlib.pyplot as plt
import pandas as pd

# Sample data
data = {
    'A': ['Yes', 'No', 'No', 'Yes', 'No', 'Yes', 'No', 'Yes', 'No', 'No'],
    'B': ['No', 'No', 'No', 'No', 'No', 'Yes', 'No', 'Yes', 'No', 'No'],
    'C': ['No', 'Yes', 'No', 'Yes', 'No', 'Yes', 'No', 'Yes', 'No', 'Yes'],
    'D': ['Yes', 'Yes', 'Yes', 'Yes', 'Yes', 'Yes', 'No', 'No', 'Yes', 'No']
}

# Create DataFrame
df = pd.DataFrame(data)

# Convert 'Yes' and 'No' to numerical values (1 for 'Yes', 0 for 'No')
df['A'] = df['A'].map({'Yes': 1, 'No': 0})
df['B'] = df['B'].map({'Yes': 1, 'No': 0})
df['C'] = df['C'].map({'Yes': 1, 'No': 0})
df['D'] = df['D'].map({'Yes': 1, 'No': 0})

# Calculate the total number of properties
total = len(df)

# Calculate the number of properties with each feature
pool_count = df['A'].sum()
dock_count = df['B'].sum()
fenced_count = df['C'].sum()
porch_count = df['D'].sum()

# Calculate the percentages
pool_per = (pool_count / total) * 100
dock_per = (dock_count / total) * 100
fenced_per = (fenced_count / total) * 100
porch_per = (porch_count / total) * 100

# Store the percentages and labels
percentages = [pool_per, dock_per, fenced_per, porch_per]
labels = ['Pool', 'Dock', 'Fenced Yard', 'Screened Porch']

# Plotting the line graph
plt.figure(figsize=(6, 6))
plt.plot(labels, percentages, marker='o', linestyle='-')
plt.title('Percentage of Properties with Specific Features')
plt.xlabel('Feature')
plt.ylabel('Percentage')
plt.grid(True)
plt.show()
```

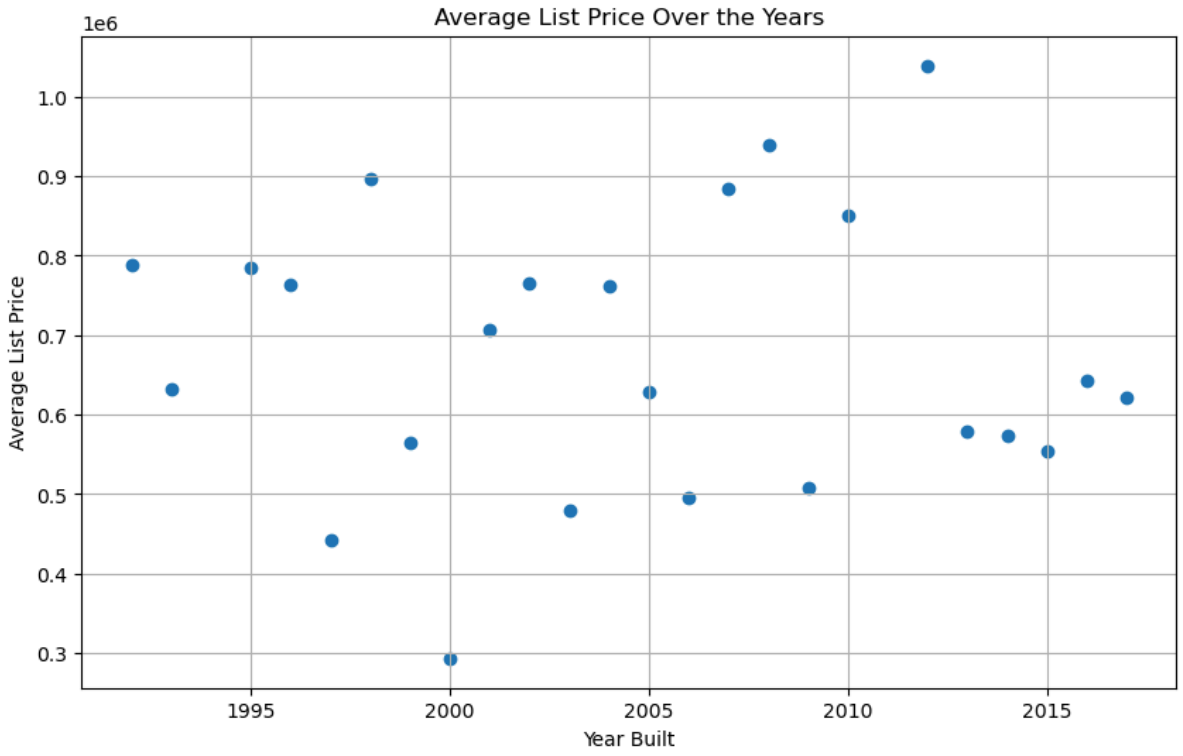


The first code snippet employs a dataset loaded from an Excel file, which may contain null values in the columns relevant to property features such as pools, docks, fenced yards, or screened porches. These null values can affect the accuracy of calculations, hence necessitating the use of sample data with predetermined values (Yes/No) to demonstrate the script's functionality effectively. By using sample data, we can ensure consistent results and showcase the code's capabilities without being hindered by potential null values or data inconsistencies present in real-world datasets.

How has the average list price changed over the years?

In [5]:
```
import pandas as pd

import matplotlib.pyplot as plt
# Read the Excel file into a DataFrame
df_red = pd.read_excel('C:\\Users\\asvai\\Downloads\\Real_Estate_Data.xlsx', sheet_

# Now you can perform operations on df_Real_Estate_Data
# For example, calculating the average list price over the years
appy = df_red.groupby('Year Built')['List Price'].mean()
# Plotting the scatter plot
plt.figure(figsize=(10, 6))
plt.scatter(average_price_per_year.index, average_price_per_year.values, marker='o'
plt.title('Average List Price Over the Years')
```

```
plt.xlabel('Year Built')
plt.ylabel('Average List Price')
plt.grid(True)
plt.show()
# Calculate the total list price for each year
tppy = df_Real_Estate_Data.groupby('Year Built')['List Price'].sum()
```



The code reads data about real estate from an Excel file. It then figures out the average price of houses built in different years and shows this on a scatter plot. This plot helps to see how prices change over time. Also, it calculates the total price of houses built each year. This gives an idea of how much money is involved in the real estate market each year. So, with this code, we can learn about trends in house prices over time and understand how the market behaves. It helps people involved in real estate to make smart decisions using this information.

Introduction: The project aims to analyze real estate data, uncovering trends and insights pivotal for decision-making in the housing market. It focuses on examining average list prices of properties across different years and understanding fluctuations in total prices over time periods. This exploration is vital for comprehending market dynamics and providing valuable insights to stakeholders.

Methodology: Utilizing Python libraries like Pandas and Matplotlib, the analysis involves loading real estate data from an Excel file and organizing it to calculate average prices of houses per year. The trends are then visualized using scatter plots. Additionally, the analysis explores the distribution of total list prices over the years, offering insights into the monetary volume within the real estate market during specific timeframes.

Problems Faced: Challenges arose during the analysis process. Ensuring accurate file path specifications and referencing sheet names correctly was crucial to prevent errors during data retrieval. Furthermore, discrepancies and missing values in the dataset posed challenges, potentially impacting the accuracy of calculations and visualizations.

Solution: To tackle these challenges, robust error handling mechanisms were implemented. This involved double-checking file paths and utilizing try-except blocks to manage potential errors gracefully. Thorough data validation checks were also conducted to handle missing or incorrect data appropriately, ensuring the integrity of the analyses and visualizations.

Future Aspects: Looking ahead, opportunities exist to enhance the project further. Strengthening error handling mechanisms and conducting more rigorous data validation checks will improve the accuracy and reliability of analyses. Exploring additional variables such as demographic or economic indicators can provide deeper insights into real estate market trends, expanding the scope of the analysis.

Conclusion: In conclusion, the project has provided valuable insights into real estate market dynamics. By overcoming challenges and implementing effective solutions, a clearer understanding of how house prices evolve over time has been gained. Continued refinement of methods and exploration of new variables will enable even deeper insights, empowering stakeholders to make informed decisions in the complex real estate landscape.

In [ ]: