

In []: *# dataset downloaded from Kaggle*

<https://www.kaggle.com/datasets/shashanknecrothapa/ames-housing-dataset>

```
In [1]: import numpy as np
import pandas as pd
from matplotlib import pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings("ignore")
```

```
In [2]: df = pd.read_csv("AmesHousing.csv")
```

```
In [3]: df.head()
```

Out[3]:

	Order	PID	MS SubClass	MS Zoning	Lot Frontage	Lot Area	Street	Alley	Lot Shape	Land Contour
0	1	526301100	20	RL	141.0	31770	Pave	NaN	IR1	Lvl
1	2	526350040	20	RH	80.0	11622	Pave	NaN	Reg	Lvl
2	3	526351010	20	RL	81.0	14267	Pave	NaN	IR1	Lvl
3	4	526353030	20	RL	93.0	11160	Pave	NaN	Reg	Lvl
4	5	527105010	60	RL	74.0	13830	Pave	NaN	IR1	Lvl

5 rows × 82 columns



```
In [4]: df.dtypes
```

Out[4]:

```
Order          int64
PID            int64
MS SubClass    int64
MS Zoning      object
Lot Frontage   float64
...
Mo Sold        int64
Yr Sold        int64
Sale Type      object
Sale Condition object
SalePrice      int64
Length: 82, dtype: object
```

```
In [5]: df.columns
```

```
Out[5]: Index(['Order', 'PID', 'MS SubClass', 'MS Zoning', 'Lot Frontage', 'Lot Area',  
             'Street', 'Alley', 'Lot Shape', 'Land Contour', 'Utilities',  
             'Lot Config', 'Land Slope', 'Neighborhood', 'Condition 1',  
             'Condition 2', 'Bldg Type', 'House Style', 'Overall Qual',  
             'Overall Cond', 'Year Built', 'Year Remod/Add', 'Roof Style',  
             'Roof Matl', 'Exterior 1st', 'Exterior 2nd', 'Mas Vnr Type',  
             'Mas Vnr Area', 'Exter Qual', 'Exter Cond', 'Foundation', 'Bsmt Qual',  
             'Bsmt Cond', 'Bsmt Exposure', 'BsmtFin Type 1', 'BsmtFin SF 1',  
             'BsmtFin Type 2', 'BsmtFin SF 2', 'Bsmt Unf SF', 'Total Bsmt SF',  
             'Heating', 'Heating QC', 'Central Air', 'Electrical', '1st Flr SF',  
             '2nd Flr SF', 'Low Qual Fin SF', 'Gr Liv Area', 'Bsmt Full Bath',  
             'Bsmt Half Bath', 'Full Bath', 'Half Bath', 'Bedroom AbvGr',  
             'Kitchen AbvGr', 'Kitchen Qual', 'TotRms AbvGrd', 'Functional',  
             'Fireplaces', 'Fireplace Qu', 'Garage Type', 'Garage Yr Blt',  
             'Garage Finish', 'Garage Cars', 'Garage Area', 'Garage Qual',  
             'Garage Cond', 'Paved Drive', 'Wood Deck SF', 'Open Porch SF',  
             'Enclosed Porch', '3Ssn Porch', 'Screen Porch', 'Pool Area', 'Pool QC',  
             'Fence', 'Misc Feature', 'Misc Val', 'Mo Sold', 'Yr Sold', 'Sale Type',  
             'Sale Condition', 'SalePrice'],  
            dtype='object')
```

```
In [7]: df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
```

```
RangeIndex: 2930 entries, 0 to 2929
```

```
Data columns (total 82 columns):
```

#	Column	Non-Null Count	Dtype
0	Order	2930 non-null	int64
1	PID	2930 non-null	int64
2	MS SubClass	2930 non-null	int64
3	MS Zoning	2930 non-null	object
4	Lot Frontage	2440 non-null	float64
5	Lot Area	2930 non-null	int64
6	Street	2930 non-null	object
7	Alley	198 non-null	object
8	Lot Shape	2930 non-null	object
9	Land Contour	2930 non-null	object
10	Utilities	2930 non-null	object
11	Lot Config	2930 non-null	object
12	Land Slope	2930 non-null	object
13	Neighborhood	2930 non-null	object
14	Condition 1	2930 non-null	object
15	Condition 2	2930 non-null	object
16	Bldg Type	2930 non-null	object
17	House Style	2930 non-null	object
18	Overall Qual	2930 non-null	int64
19	Overall Cond	2930 non-null	int64
20	Year Built	2930 non-null	int64
21	Year Remod/Add	2930 non-null	int64
22	Roof Style	2930 non-null	object
23	Roof Matl	2930 non-null	object
24	Exterior 1st	2930 non-null	object
25	Exterior 2nd	2930 non-null	object
26	Mas Vnr Type	1155 non-null	object
27	Mas Vnr Area	2907 non-null	float64
28	Exter Qual	2930 non-null	object
29	Exter Cond	2930 non-null	object
30	Foundation	2930 non-null	object
31	Bsmt Qual	2850 non-null	object
32	Bsmt Cond	2850 non-null	object
33	Bsmt Exposure	2847 non-null	object
34	BsmtFin Type 1	2850 non-null	object
35	BsmtFin SF 1	2929 non-null	float64
36	BsmtFin Type 2	2849 non-null	object
37	BsmtFin SF 2	2929 non-null	float64
38	Bsmt Unf SF	2929 non-null	float64
39	Total Bsmt SF	2929 non-null	float64
40	Heating	2930 non-null	object
41	Heating QC	2930 non-null	object
42	Central Air	2930 non-null	object
43	Electrical	2929 non-null	object
44	1st Flr SF	2930 non-null	int64
45	2nd Flr SF	2930 non-null	int64
46	Low Qual Fin SF	2930 non-null	int64
47	Gr Liv Area	2930 non-null	int64
48	Bsmt Full Bath	2928 non-null	float64
49	Bsmt Half Bath	2928 non-null	float64
50	Full Bath	2930 non-null	int64

```

51 Half Bath          2930 non-null  int64
52 Bedroom AbvGr     2930 non-null  int64
53 Kitchen AbvGr     2930 non-null  int64
54 Kitchen Qual      2930 non-null  object
55 TotRms AbvGrd     2930 non-null  int64
56 Functional        2930 non-null  object
57 Fireplaces        2930 non-null  int64
58 Fireplace Qu      1508 non-null  object
59 Garage Type       2773 non-null  object
60 Garage Yr Blt     2771 non-null  float64
61 Garage Finish     2771 non-null  object
62 Garage Cars       2929 non-null  float64
63 Garage Area       2929 non-null  float64
64 Garage Qual      2771 non-null  object
65 Garage Cond       2771 non-null  object
66 Paved Drive       2930 non-null  object
67 Wood Deck SF      2930 non-null  int64
68 Open Porch SF     2930 non-null  int64
69 Enclosed Porch    2930 non-null  int64
70 3Ssn Porch        2930 non-null  int64
71 Screen Porch      2930 non-null  int64
72 Pool Area         2930 non-null  int64
73 Pool QC           13 non-null   object
74 Fence             572 non-null  object
75 Misc Feature      106 non-null  object
76 Misc Val          2930 non-null  int64
77 Mo Sold           2930 non-null  int64
78 Yr Sold           2930 non-null  int64
79 Sale Type         2930 non-null  object
80 Sale Condition    2930 non-null  object
81 SalePrice         2930 non-null  int64
dtypes: float64(11), int64(28), object(43)
memory usage: 1.8+ MB

```

```
In [8]: df.isnull().sum()
```

```

Out[8]: Order          0
        PID            0
        MS SubClass    0
        MS Zoning      0
        Lot Frontage   490
        ...
        Mo Sold        0
        Yr Sold        0
        Sale Type      0
        Sale Condition  0
        SalePrice      0
        Length: 82, dtype: int64

```

```

In [9]: missing_percentage = df.isnull().mean() * 100
        threshold = 30
        columns_to_drop = missing_percentage[missing_percentage > threshold].index
        df = df.drop(columns=columns_to_drop)
        print(f"Columns dropped: {list(columns_to_drop)}")
        print(f"Remaining columns: {df.columns}")

```

Columns dropped: ['Alley', 'Mas Vnr Type', 'Fireplace Qu', 'Pool QC', 'Fence', 'Misc Feature']

Remaining columns: Index(['Order', 'PID', 'MS SubClass', 'MS Zoning', 'Lot Frontage', 'Lot Area',

'Street', 'Lot Shape', 'Land Contour', 'Utilities', 'Lot Config', 'Land Slope', 'Neighborhood', 'Condition 1', 'Condition 2', 'Bldg Type', 'House Style', 'Overall Qual', 'Overall Cond', 'Year Built', 'Year Remod/Add', 'Roof Style', 'Roof Matl', 'Exterior 1st', 'Exterior 2nd', 'Mas Vnr Area', 'Exter Qual', 'Exter Cond', 'Foundation', 'Bsmt Qual', 'Bsmt Cond', 'Bsmt Exposure', 'BsmtFin Type 1', 'BsmtFin SF 1', 'BsmtFin Type 2', 'BsmtFin SF 2', 'Bsmt Unf SF', 'Total Bsmt SF', 'Heating', 'Heating QC', 'Central Air', 'Electrical', '1st Flr SF', '2nd Flr SF', 'Low Qual Fin SF', 'Gr Liv Area', 'Bsmt Full Bath', 'Bsmt Half Bath', 'Full Bath', 'Half Bath', 'Bedroom AbvGr', 'Kitchen AbvGr', 'Kitchen Qual', 'TotRms AbvGrd', 'Functional', 'Fireplaces', 'Garage Type', 'Garage Yr Blt', 'Garage Finish', 'Garage Cars', 'Garage Area', 'Garage Qual', 'Garage Cond', 'Paved Drive', 'Wood Deck SF', 'Open Porch SF', 'Enclosed Porch', '3Ssn Porch', 'Screen Porch', 'Pool Area', 'Misc Val', 'Mo Sold', 'Yr Sold', 'Sale Type', 'Sale Condition', 'SalePrice'], dtype='object')

```
In [10]: from sklearn.preprocessing import LabelEncoder
label_encoder = LabelEncoder()

for column in df.select_dtypes(include=['object']).columns:
    df[column] = label_encoder.fit_transform(df[column])

df.head()
```

Out[10]:

	Order	PID	MS SubClass	MS Zoning	Lot Frontage	Lot Area	Street	Lot Shape	Land Contour	Utilities
0	1	526301100	20	5	141.0	31770	1	0	3	0
1	2	526350040	20	4	80.0	11622	1	3	3	0
2	3	526351010	20	5	81.0	14267	1	0	3	0
3	4	526353030	20	5	93.0	11160	1	3	3	0
4	5	527105010	60	5	74.0	13830	1	0	3	0

5 rows × 76 columns



```
In [11]: for column in df.select_dtypes(include=['float64']).columns:
    mean_value = df[column].mean()
    df[column] = df[column].fillna(mean_value)

for column in df.select_dtypes(include=['object']).columns:
    mode_value = df[column].mode()[0]
    df[column] = df[column].fillna(mode_value)
```

```
df.isnull().sum()
```

Out[11]:

Order	0
PID	0
MS SubClass	0
MS Zoning	0
Lot Frontage	0
..	
Mo Sold	0
Yr Sold	0
Sale Type	0
Sale Condition	0
SalePrice	0

Length: 76, dtype: int64

```
In [12]: data = df.copy()

def detect_outliers_zscore(column, threshold=3):
    mean = np.mean(column)
    std_dev = np.std(column)
    z_scores = (column - mean) / std_dev
    return np.abs(z_scores) > threshold

for col in data.columns:
    data[f'{col}_Outlier'] = detect_outliers_zscore(data[col])

outliers_only = data[data.filter(like='_Outlier').any(axis=1)]
outliers_only
```

Out[12]:

	Order	PID	MS SubClass	MS Zoning	Lot Frontage	Lot Area	Street	Lot Shape	Land Contour	Uti
0	1	526301100	20	5	141.00000	31770	1	0	3	
2	3	526351010	20	5	81.00000	14267	1	0	3	
11	12	527165230	20	5	69.22459	7980	1	0	3	
14	15	527182190	120	5	69.22459	6820	1	0	3	
15	16	527216070	60	5	47.00000	53504	1	1	1	
...	...	...	...	...	...	...	...	...	...	...
2924	2925	923251180	20	5	160.00000	20000	1	3	3	
2926	2927	923276100	20	5	69.22459	8885	1	0	2	
2927	2928	923400125	85	5	62.00000	10441	1	3	3	
2928	2929	924100070	20	5	77.00000	10010	1	3	3	
2929	2930	924151050	60	5	74.00000	9627	1	3	3	

1740 rows × 152 columns



```
In [13]: data = df.copy()

def replace_outliers_with_nan(column):
    Q1 = column.quantile(0.25)
    Q3 = column.quantile(0.75)
    IQR = Q3 - Q1
    lower_bound = Q1 - 1.5 * IQR
    upper_bound = Q3 + 1.5 * IQR
    return column.where((column >= lower_bound) & (column <= upper_bound), np.nan)

for col in data.columns:
    data[col] = replace_outliers_with_nan(data[col])

null_counts = data.isnull().sum()
print("Sum of null values in each column:")
print(null_counts)
```

Sum of null values in each column:

Order	0
PID	0
MS SubClass	208
MS Zoning	657
Lot Frontage	261
...	
Mo Sold	0
Yr Sold	0
Sale Type	394
Sale Condition	517
SalePrice	137
Length: 76, dtype: int64	

```
In [14]: from sklearn.impute import KNNImputer

imputer = KNNImputer(n_neighbors=2)
data_imputed_knn = pd.DataFrame(imputer.fit_transform(data), columns=data.columns)
data_imputed_knn.isnull().sum()
```

```
Out[14]: Order      0
PID      0
MS SubClass  0
MS Zoning  0
Lot Frontage  0
..
Mo Sold      0
Yr Sold      0
Sale Type    0
Sale Condition  0
SalePrice    0
Length: 76, dtype: int64
```

```
In [15]: def remove_outliers_iqr(data):
    Q1 = data.quantile(0.25)
    Q3 = data.quantile(0.75)
    IQR = Q3 - Q1
    lower_bound = Q1 - 1.5 * IQR
    upper_bound = Q3 + 1.5 * IQR
```

```

data_cleaned = data[~((data < lower_bound) | (data > upper_bound)).any(axis=1)]
return data_cleaned

cleaned_data = remove_outliers_iqr(df)

print("Original Data:")
print(len(df))
print("\nCleaned Data (without outliers):")
print(len(cleaned_data))

```

Original Data:

2930

Cleaned Data (without outliers):

301

```

In [16]: def remove_outliers_zscore(data, threshold=3):
outliers = pd.DataFrame(False, index=df.index, columns=df.columns)
for column in data.columns:
    if data[column].dtype in ['float64', 'int64']:
        mean = data[column].mean()
        std_dev = data[column].std()
        z_scores = (data[column] - mean) / std_dev
        outliers[column] = abs(z_scores) > threshold
rows_to_keep = ~outliers.any(axis=1)
return data[rows_to_keep]

cleaned_data = remove_outliers_zscore(df, threshold=3)

print("Original Data:")
print(len(df))
print("\nCleaned Data (without outliers):")
print(len(cleaned_data))

```

Original Data:

2930

Cleaned Data (without outliers):

1190

```

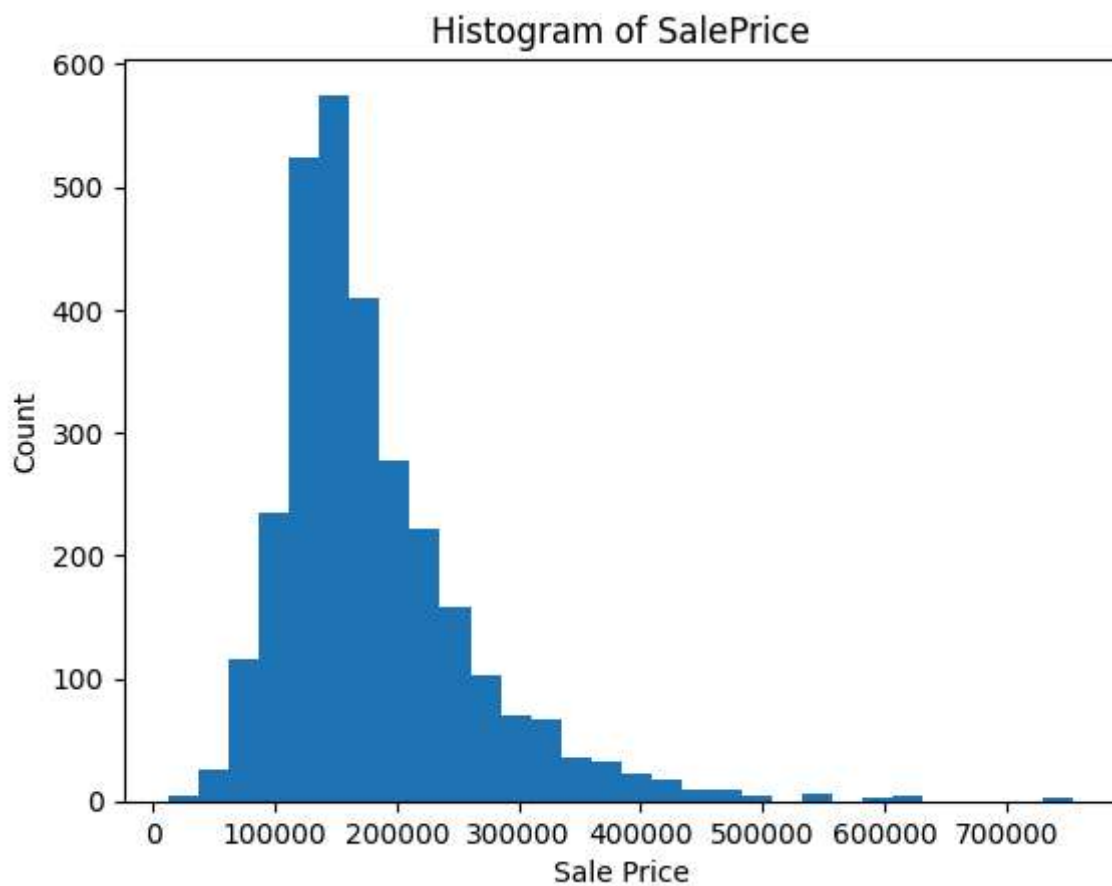
In [17]: import numpy as np
import pandas as pd
from matplotlib import pyplot as plt
import seaborn as sns

```

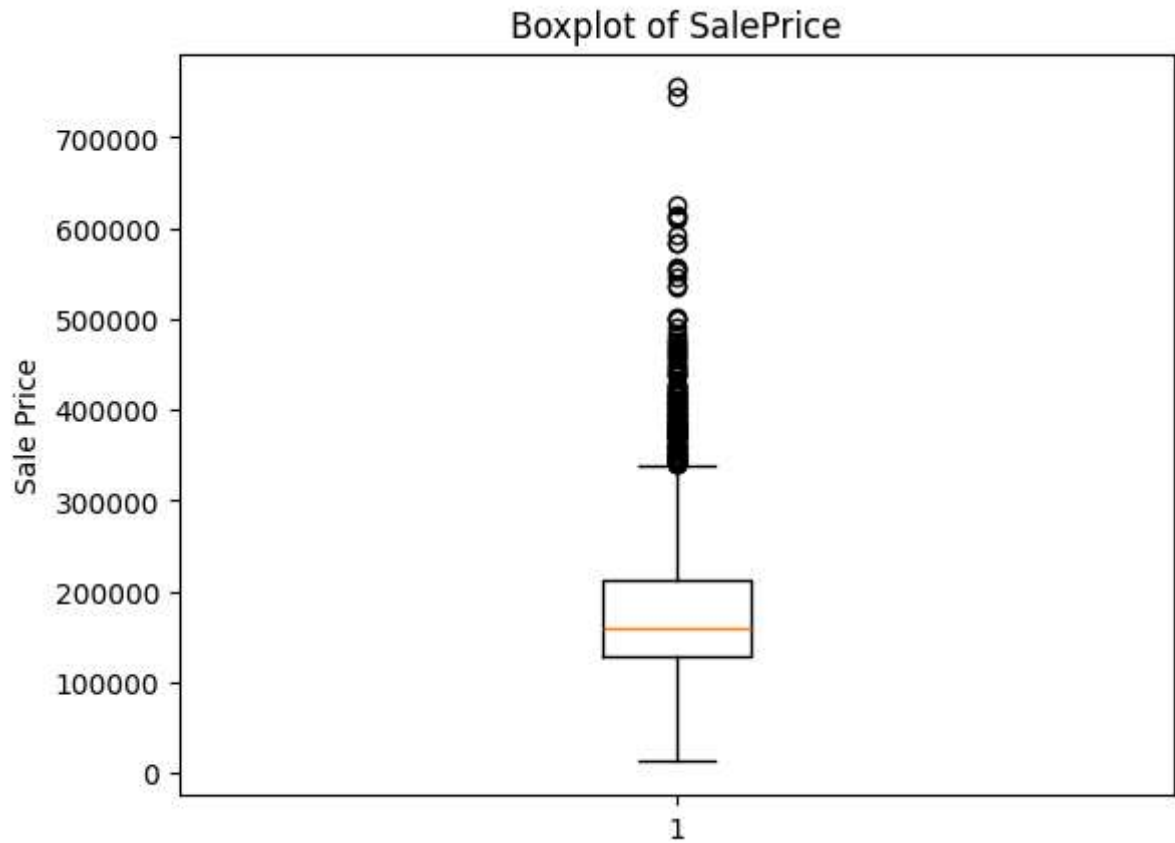
```

In [18]: plt.hist(df['SalePrice'], bins=30)
plt.title("Histogram of SalePrice")
plt.xlabel("Sale Price")
plt.ylabel("Count")
plt.show()

```



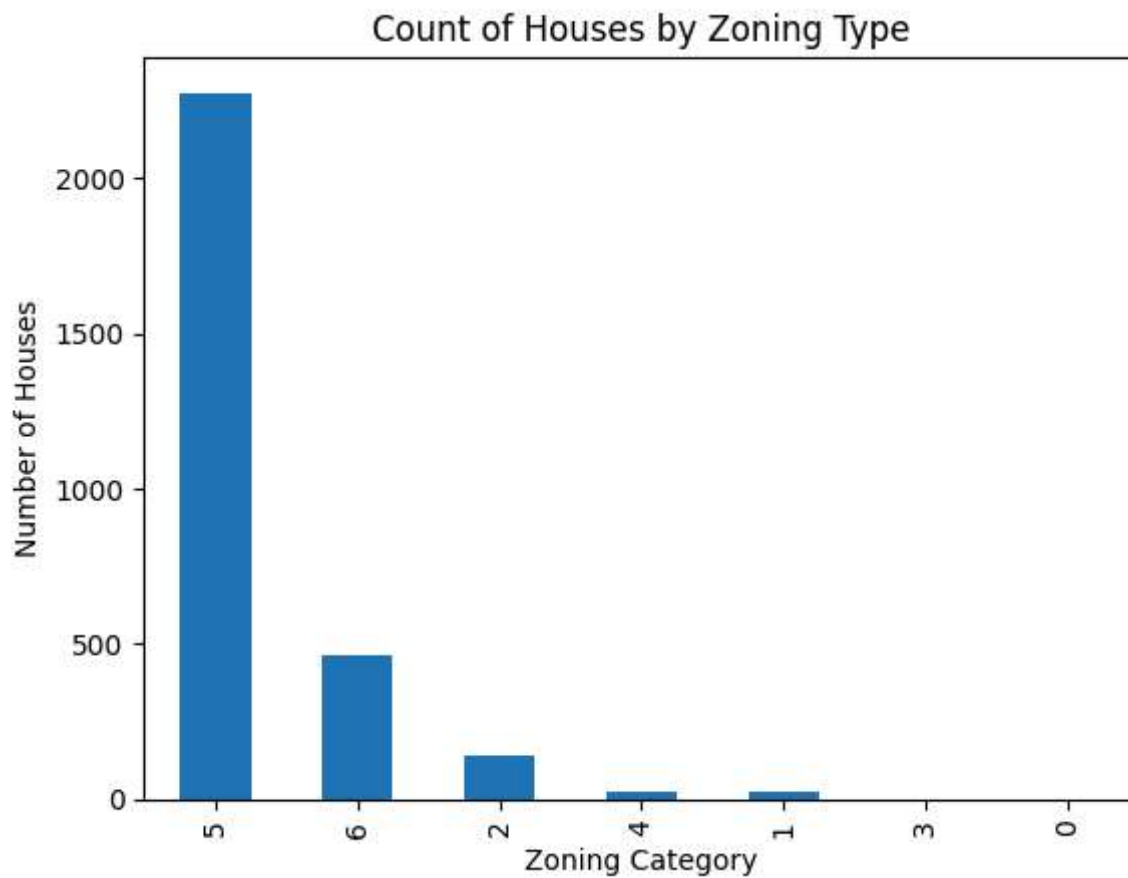
```
In [19]: plt.boxplot(df['SalePrice'])  
plt.title("Boxplot of SalePrice")  
plt.ylabel("Sale Price")  
plt.show()
```



```
In [20]: plt.scatter(df['Gr Liv Area'], df['SalePrice'])  
plt.title("Living Area vs Sale Price")  
plt.xlabel("Ground Living Area (sqft)")  
plt.ylabel("Sale Price")  
plt.show()
```



```
In [21]: df['MS Zoning'].value_counts().plot(kind='bar')
plt.title("Count of Houses by Zoning Type")
plt.xlabel("Zoning Category")
plt.ylabel("Number of Houses")
plt.show()
```

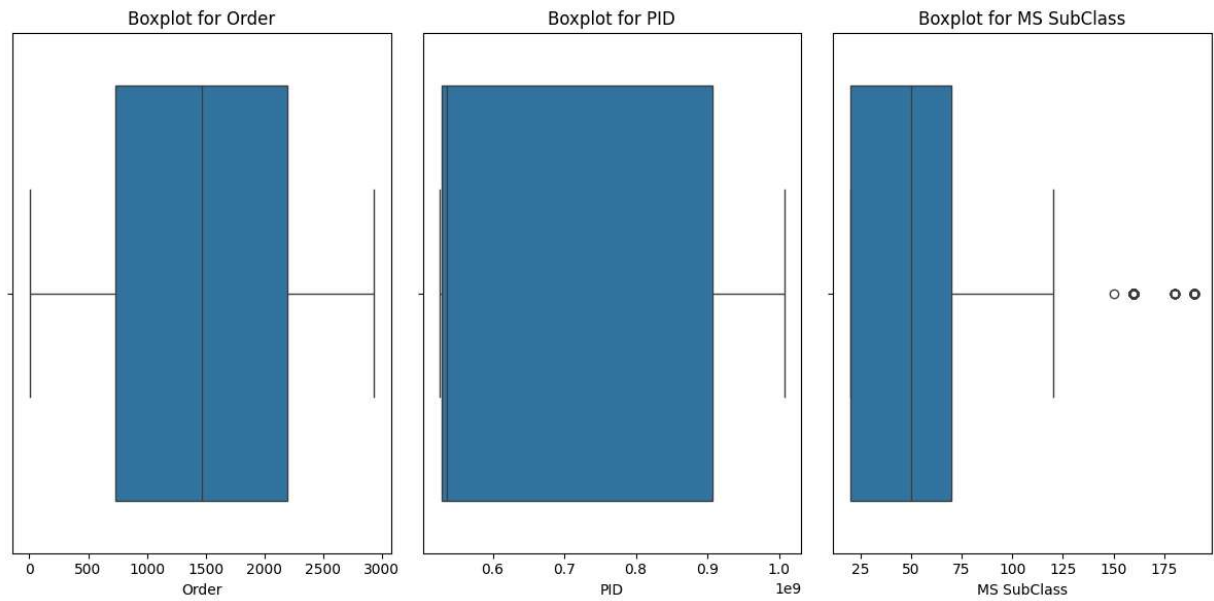


```
In [24]: import matplotlib.pyplot as plt
import seaborn as sns

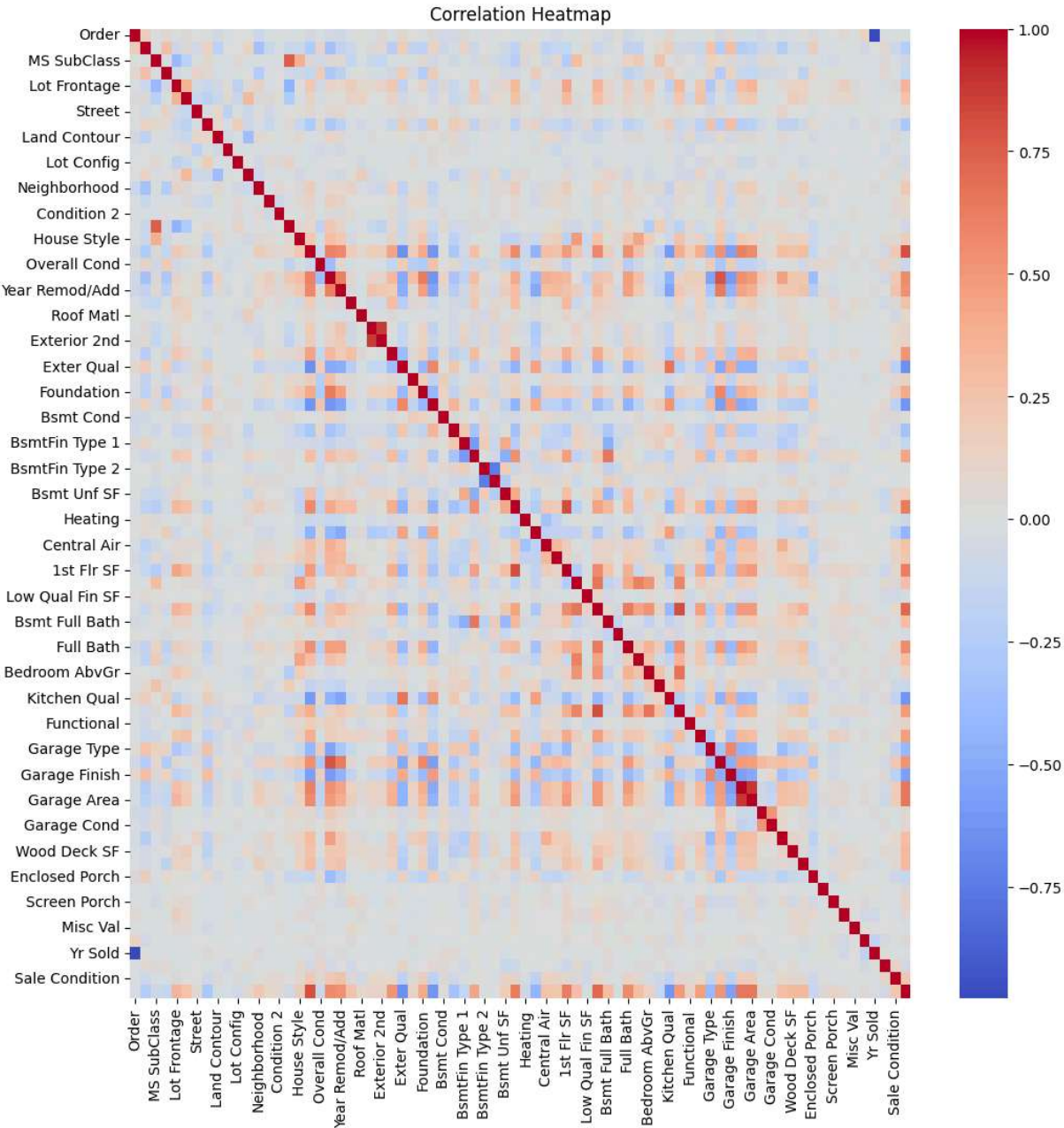
plt.figure(figsize=(12, 6))

for i, col in enumerate(df.columns[:3]):
    plt.subplot(1, 3, i+1)
    sns.boxplot(x=df[col])
    plt.title(f"Boxplot for {col}")

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



```
In [29]: plt.figure(figsize=(12,12))
sns.heatmap(df.corr(), annot=False , cmap='coolwarm')
plt.title("Correlation Heatmap")
plt.show()
```



In []: