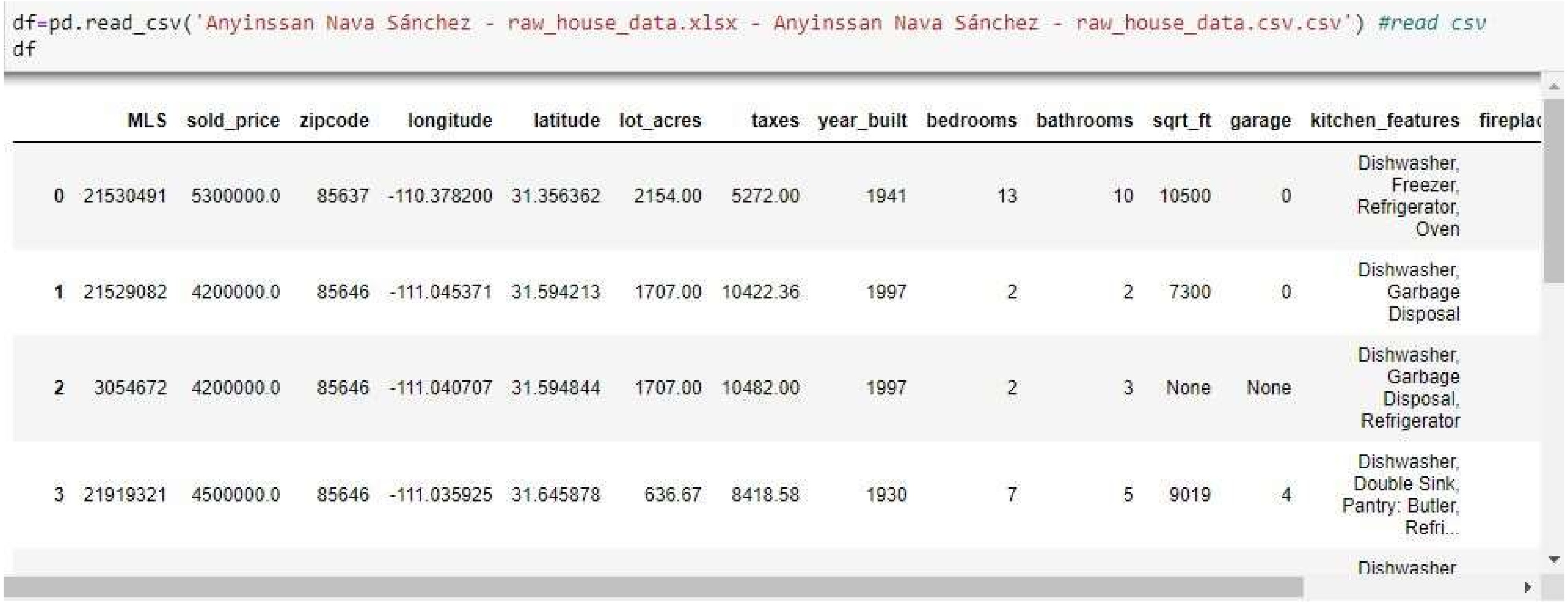
Data cleaning

First we import the necessary libraries that we will use for the exploratory analysis and data cleaning.

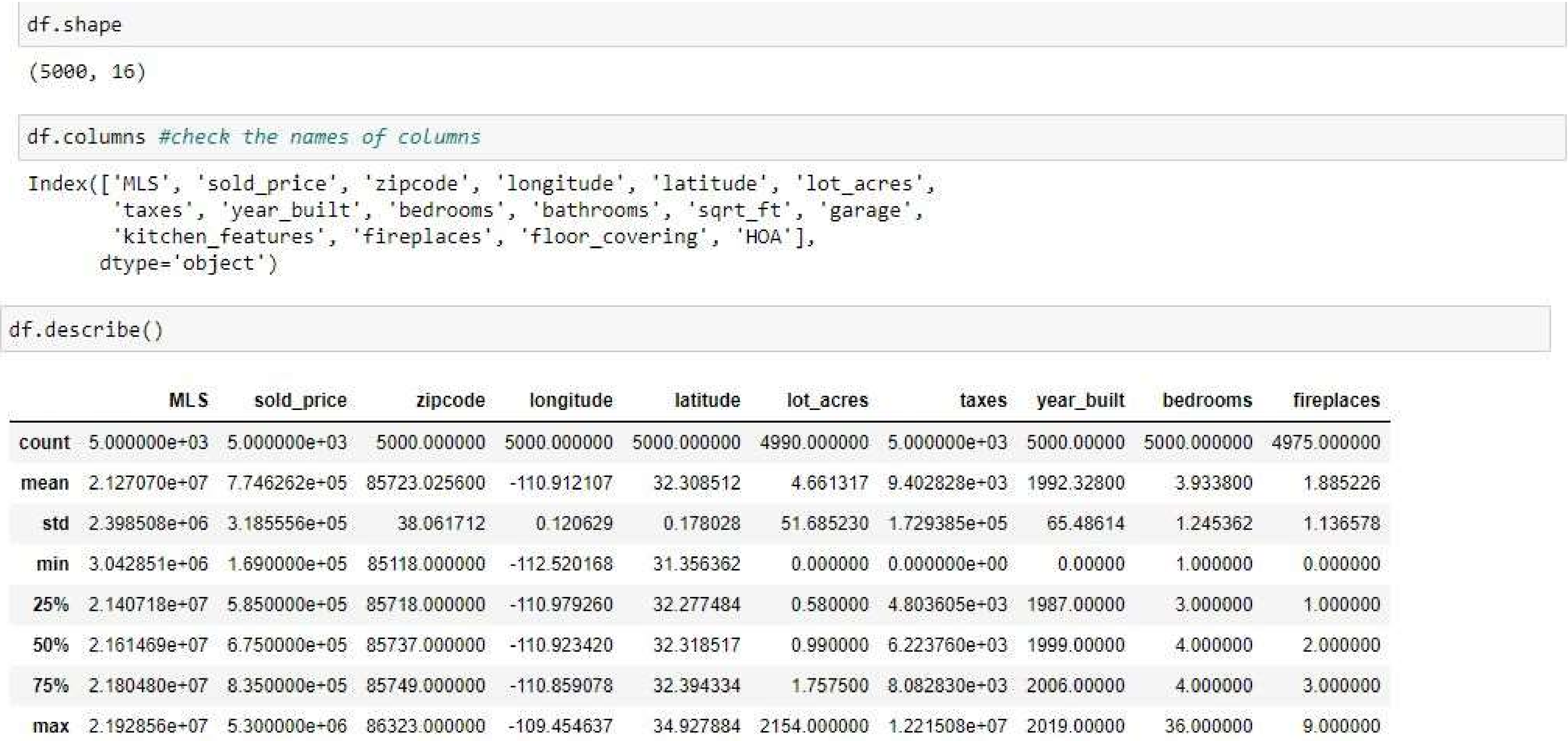
```
#Import libraries
import os
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
```

We import the data and print it



```
df=pd.read_csv('Anyinssan Nava Sánchez - raw_house_data.xlsx - Anyinssan Nava Sánchez - raw_house_data.csv.csv') #read csv
df
```

	MLS	sold_price	zipcode	longitude	latitude	lot_acres	taxes	year_built	bedrooms	bathrooms	sqrt_ft	garage	kitchen_features	fireplac
0	21530491	5300000.0	85637	-110.378200	31.356362	2154.00	5272.00	1941	13	10	10500	0	Dishwasher, Freezer, Refrigerator, Oven	
1	21529082	4200000.0	85646	-111.045371	31.594213	1707.00	10422.36	1997	2	2	7300	0	Dishwasher, Garbage Disposal	
2	3054672	4200000.0	85646	-111.040707	31.594844	1707.00	10482.00	1997	2	3	None	None	Dishwasher, Garbage Disposal, Refrigerator	
3	21919321	4500000.0	85646	-111.035925	31.645878	636.67	8418.58	1930	7	5	9019	4	Dishwasher, Double Sink, Pantry: Butler, Refri...	
													Dishwasher	



```
df.shape
```

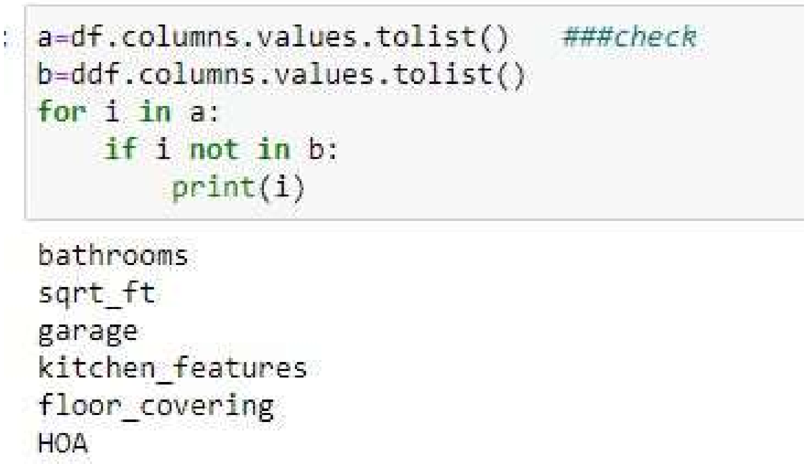
```
(5000, 16)
```

```
df.columns #check the names of columns
```

```
Index(['MLS', 'sold_price', 'zipcode', 'longitude', 'latitude', 'lot_acres',
       'taxes', 'year_built', 'bedrooms', 'bathrooms', 'sqrt_ft', 'garage',
       'kitchen_features', 'fireplaces', 'floor_covering', 'HOA'],
      dtype='object')
```

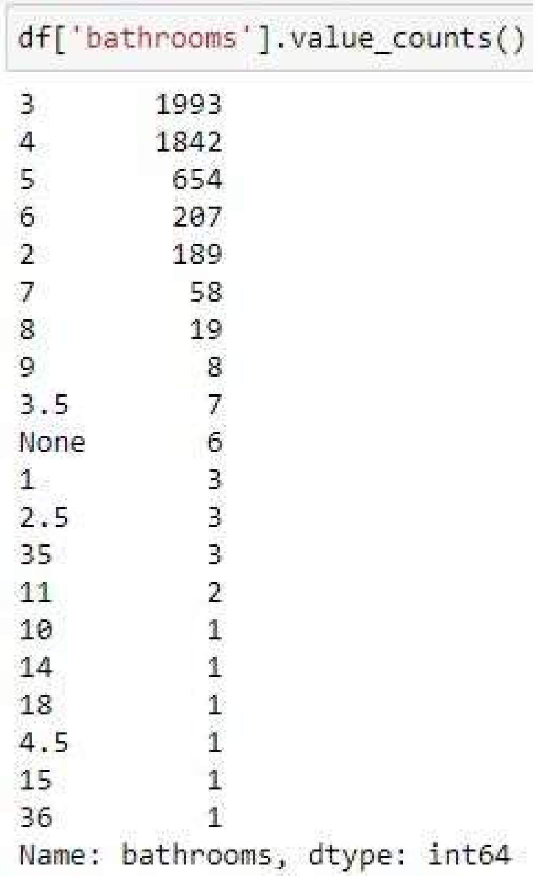
```
df.describe()
```

	MLS	sold_price	zipcode	longitude	latitude	lot_acres	taxes	year_built	bedrooms	fireplaces
count	5.000000e+03	5.000000e+03	5000.000000	5000.000000	5000.000000	4990.000000	5.000000e+03	5000.00000	5000.000000	4975.000000
mean	2.127070e+07	7.746262e+05	85723.025600	-110.912107	32.308512	4.661317	9.402828e+03	1992.32800	3.933800	1.885226
std	2.398508e+06	3.185556e+05	38.061712	0.120629	0.178028	51.685230	1.729385e+05	65.48614	1.245362	1.136578
min	3.042851e+06	1.690000e+05	85118.000000	-112.520168	31.356362	0.000000	0.000000e+00	0.00000	1.000000	0.000000
25%	2.140718e+07	5.850000e+05	85718.000000	-110.979260	32.277484	0.580000	4.803605e+03	1987.00000	3.000000	1.000000
50%	2.161469e+07	6.750000e+05	85737.000000	-110.923420	32.318517	0.990000	6.223760e+03	1999.00000	4.000000	2.000000
75%	2.180480e+07	8.350000e+05	85749.000000	-110.859078	32.394334	1.757500	8.082830e+03	2006.00000	4.000000	3.000000
max	2.192856e+07	5.300000e+06	86323.000000	-109.454637	34.927884	2154.000000	1.221508e+07	2019.00000	36.000000	9.000000



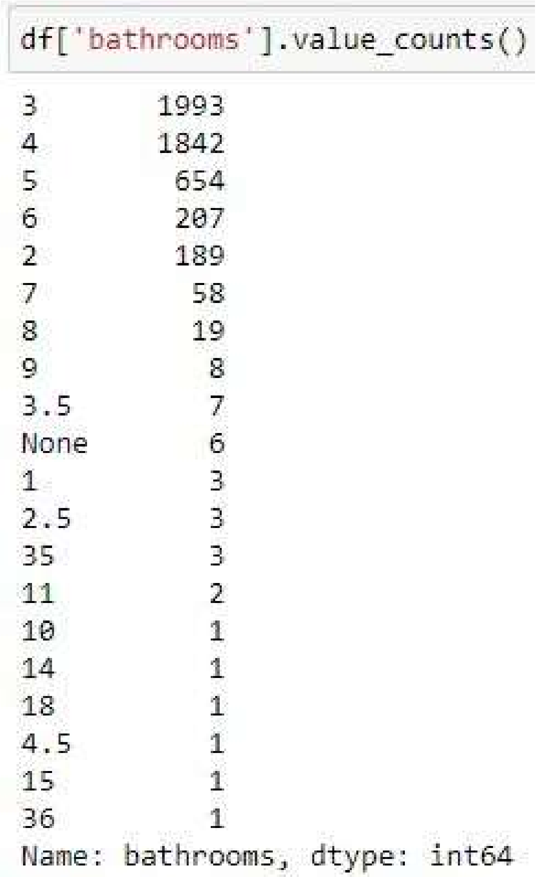
```
a=df.columns.values.tolist()   ###check
b=ddf.columns.values.tolist()
for i in a:
    if i not in b:
        print(i)
```

```
bathrooms
sqrt_ft
garage
kitchen_features
floor_covering
HOA
```



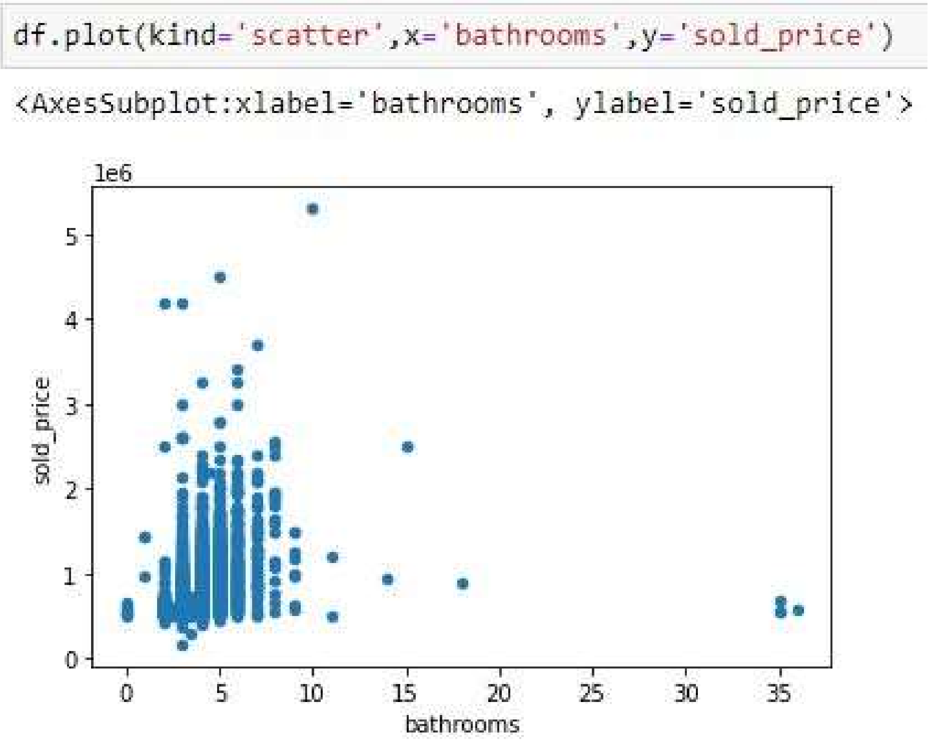
```
df['bathrooms'].value_counts()
```

```
3       1993
4       1842
5        654
6        207
2        189
7         58
8         19
9          8
3.5        7
None       6
1          3
2.5        3
35         3
11         2
10         1
14         1
18         1
4.5        1
15         1
36         1
Name: bathrooms, dtype: int64
```



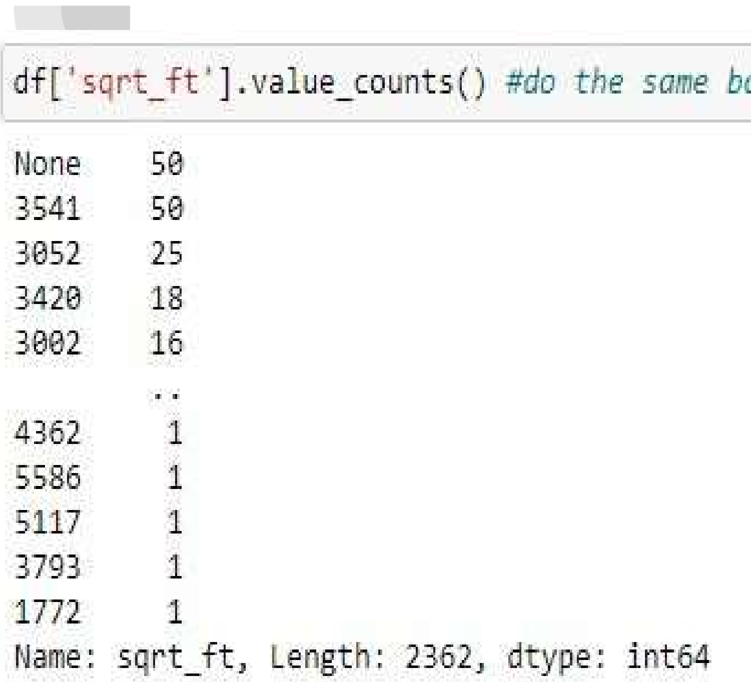
```
df['bathrooms'].value_counts()
```

```
3       1993
4       1842
5        654
6        207
2        189
7         58
8         19
9          8
3.5        7
None       6
1          3
2.5        3
35         3
11         2
10         1
14         1
18         1
4.5        1
15         1
36         1
Name: bathrooms, dtype: int64
```



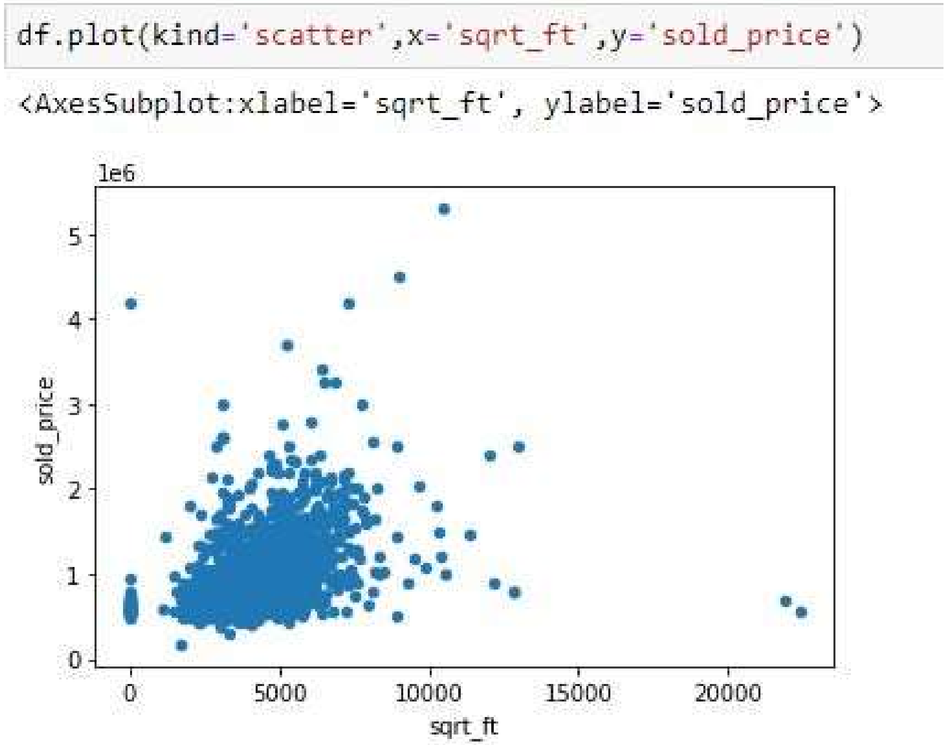
```
df.plot(kind='scatter',x='bathrooms',y='sold_price')
```

```
<AxesSubplot:xlabel='bathrooms', ylabel='sold_price'>
```



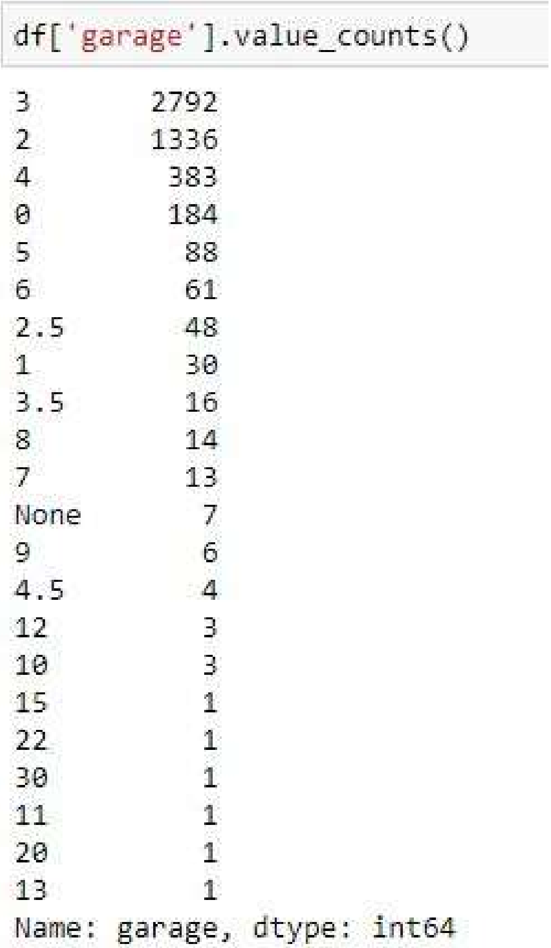
```
df['sqrt_ft'].value_counts() #do the same bo
```

```
None    50
3541    50
3052    25
3420    18
3002    16
        ..
4362     1
5586     1
5117     1
3793     1
1772     1
Name: sqrt_ft, Length: 2362, dtype: int64
```



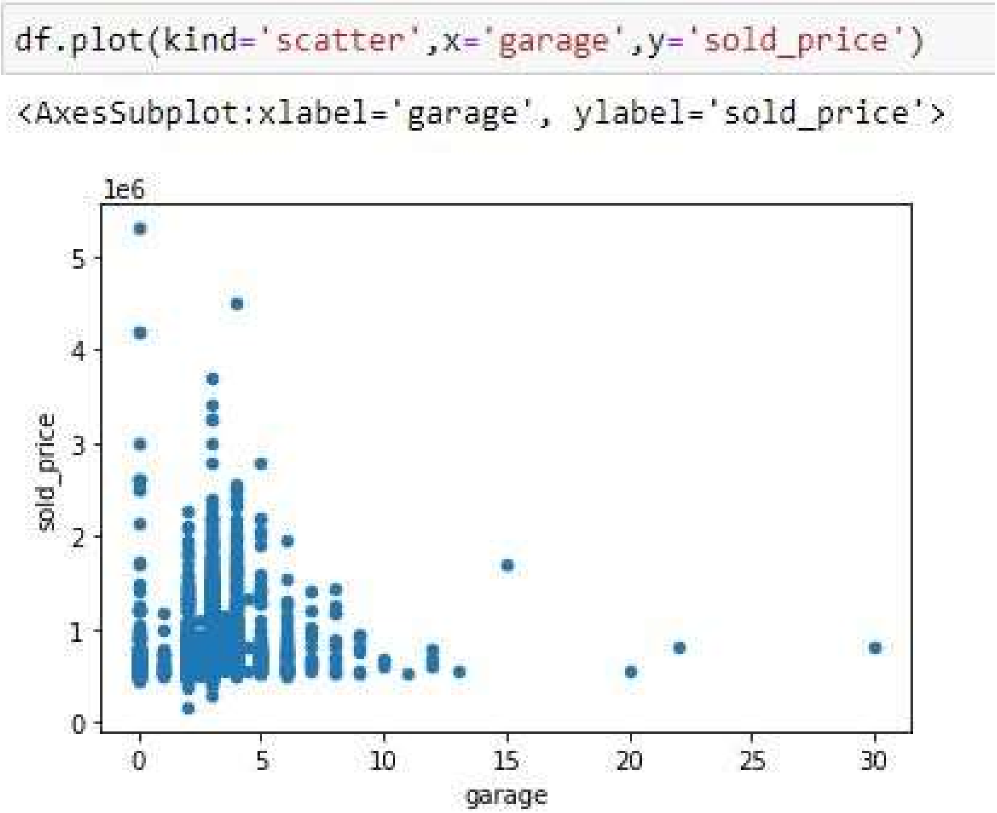
```
df.plot(kind='scatter',x='sqrt_ft',y='sold_price')
```

```
<AxesSubplot:xlabel='sqrt_ft', ylabel='sold_price'>
```



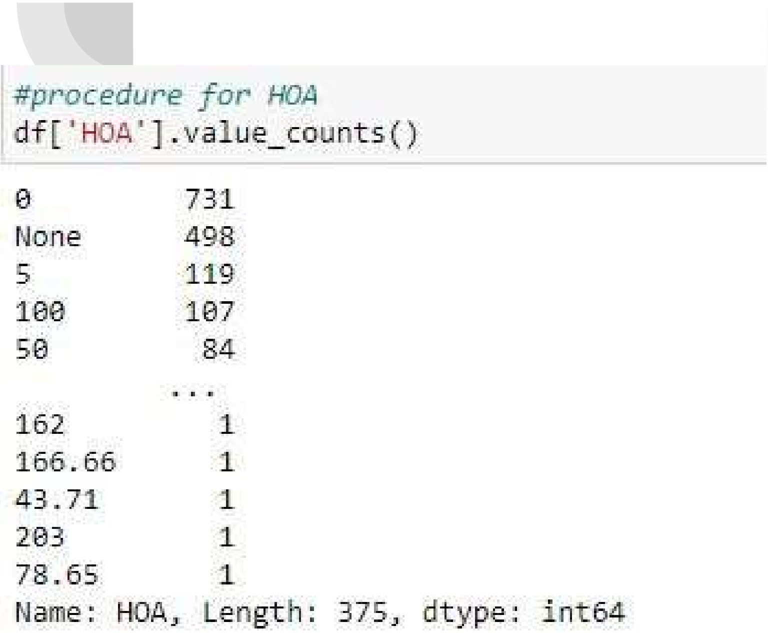
```
df['garage'].value_counts()
```

```
3       2792
2       1336
4        383
0        184
5         88
6         61
2.5       48
1         30
3.5       16
8         14
7         13
None       7
9          6
4.5        4
12         3
10         3
15         1
22         1
30         1
11         1
20         1
13         1
Name: garage, dtype: int64
```



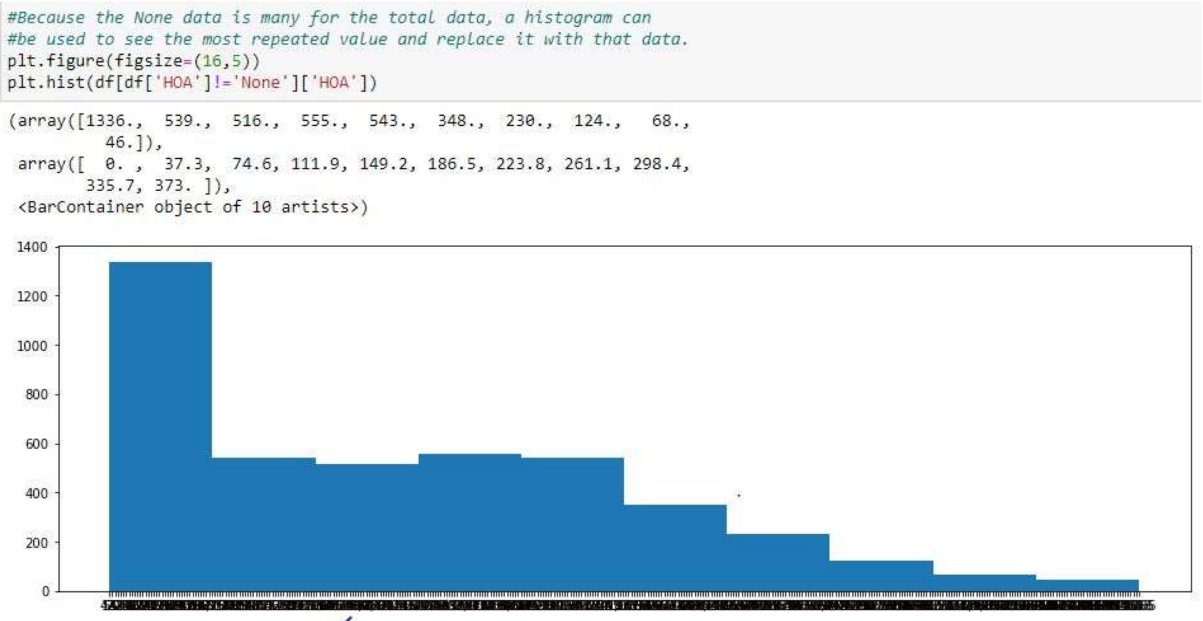
```
df.plot(kind='scatter',x='garage',y='sold_price')
```

```
<AxesSubplot:xlabel='garage', ylabel='sold_price'>
```



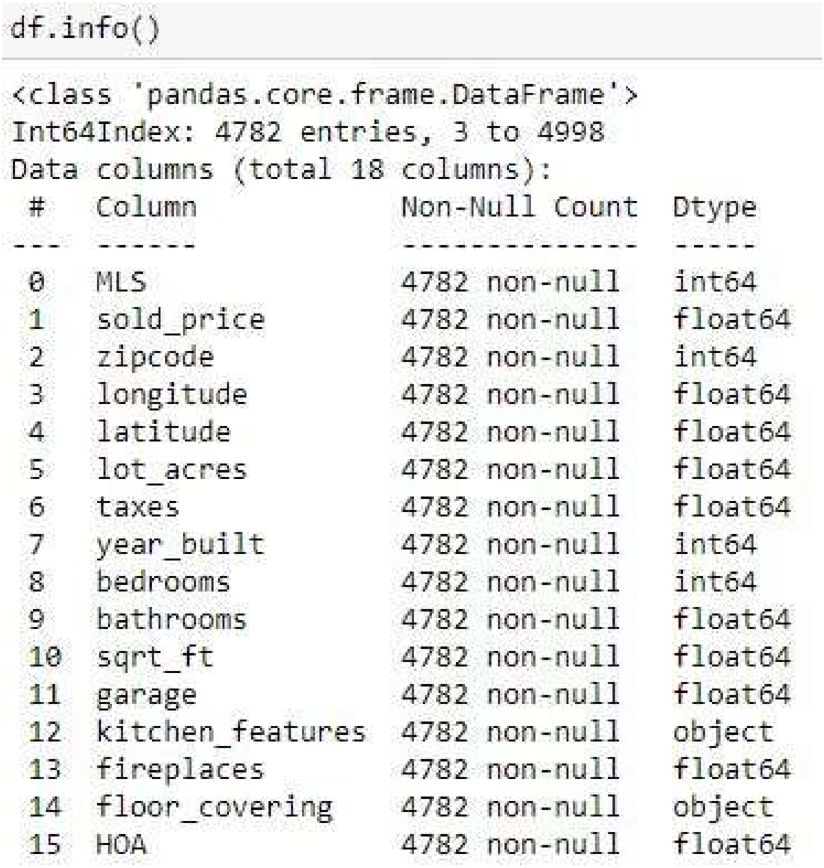
```
#procedure for HOA
df['HOA'].value_counts()
```

```
0          731
None       498
5          119
100        107
50          84
          ... 
162          1
166.66       1
43.71        1
203          1
78.65        1
Name: HOA, Length: 375, dtype: int64
```



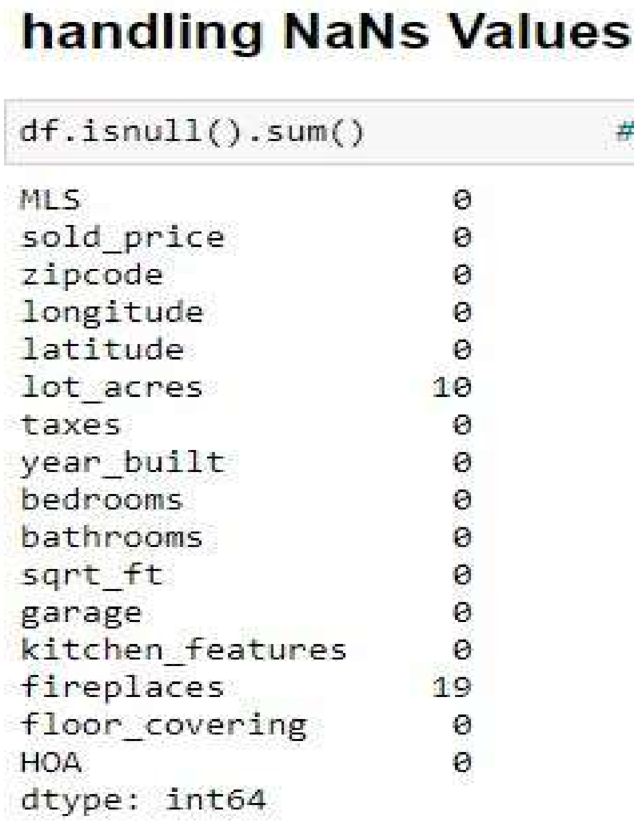
```
#Because the None data is many for the total data, a histogram can
#be used to see the most repeated value and replace it with that data.
plt.figure(figsize=(16,5))
plt.hist(df[df['HOA']!='None']['HOA'])
```

```
(array([1336.,  539.,  516.,  555.,  543.,  348.,  230.,  124.,   68.,
          46.]),
 array([  0. ,  37.3,  74.6, 111.9, 149.2, 186.5, 223.8, 261.1, 298.4,
        335.7, 373. ]),
 <BarContainer object of 10 artists>)
```



```
df.info()
```

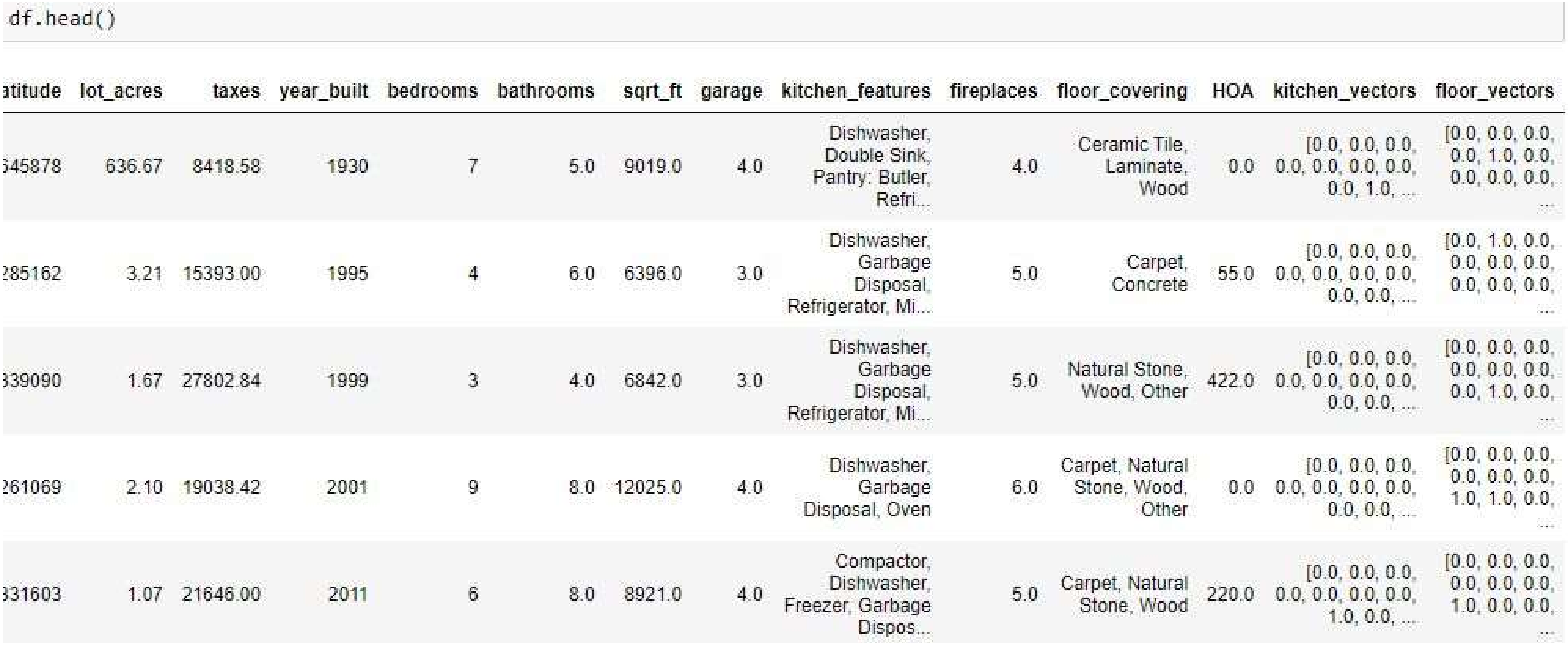
```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 4782 entries, 3 to 4998
Data columns (total 18 columns):
 #   Column            Non-Null Count  Dtype  
---  ------            --------------  -----  
 0   MLS               4782 non-null   int64  
 1   sold_price        4782 non-null   float64
 2   zipcode           4782 non-null   int64  
 3   longitude         4782 non-null   float64
 4   latitude          4782 non-null   float64
 5   lot_acres         4782 non-null   float64
 6   taxes             4782 non-null   float64
 7   year_built        4782 non-null   int64  
 8   bedrooms          4782 non-null   int64  
 9   bathrooms         4782 non-null   float64
 10  sqrt_ft           4782 non-null   float64
 11  garage            4782 non-null   float64
 12  kitchen_features  4782 non-null   object 
 13  fireplaces        4782 non-null   float64
 14  floor_covering    4782 non-null   object 
 15  HOA               4782 non-null   float64
```



handling NaNs Values

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

```
MLS                  0
sold_price           0
zipcode              0
longitude            0
latitude             0
lot_acres           10
taxes                0
year_built           0
bedrooms             0
bathrooms            0
sqrt_ft              0
garage               0
kitchen_features     0
fireplaces          19
floor_covering       0
HOA                  0
dtype: int64
```



```
df.head()
```

atitude	lot_acres	taxes	year_built	bedrooms	bathrooms	sqrt_ft	garage	kitchen_features	fireplaces	floor_covering	HOA	kitchen_vectors	floor_vectors
545878	636.67	8418.58	1930	7	5.0	9019.0	4.0	Dishwasher, Double Sink, Pantry: Butler, Refri...	4.0	Ceramic Tile, Laminate, Wood	0.0	[0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0, ...	[0.0, 0.0, 0.0, 0.0, 1.0, 0.0, 0.0, 0.0, 0.0, ...
285162	3.21	15393.00	1995	4	6.0	6396.0	3.0	Dishwasher, Garbage Disposal, Refrigerator, Mi...	5.0	Carpet, Concrete	55.0	[0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, ...	[0.0, 1.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, ...
339090	1.67	27802.84	1999	3	4.0	6842.0	3.0	Dishwasher, Garbage Disposal, Refrigerator, Mi...	5.0	Natural Stone, Wood, Other	422.0	[0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, ...	[0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0, 0.0, ...
261069	2.10	19038.42	2001	9	8.0	12025.0	4.0	Dishwasher, Garbage Disposal, Oven	6.0	Carpet, Natural Stone, Wood, Other	0.0	[0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, ...	[0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0, 1.0, 0.0, ...
331603	1.07	21646.00	2011	6	8.0	8921.0	4.0	Compactor, Dishwasher, Freezer, Garbage Dispos...	5.0	Carpet, Natural Stone, Wood	220.0	[0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0, 0.0, ...	[0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0, 0.0, 0.0, ...