$TASK - 1$

- HOUSE PRICE PREDICTION

- use a dataset that includes information about housing prices and features like sqaure footage, number of bedrooms, etc. to train a model that can predict the price of a new house

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

In [3]:
```
import pandas as pd
house_df=pd.read_csv("C:\\Users\\pedur\\OneDrive\\kc_house_data.csv")
house_df
```

Out[3]:

	id	date	price	bedrooms	bathrooms	sqft_living	sqft_lot	floors	waterfront	view	...	grade	sqft_above	sqft_basement	yr_built	yr_renovated	zipcode
0	7129300520	20141013T000000	221900.0	3	1.00	1180	5650	1.0	0	0	...	7	1180	0	1955	0	98178
1	6414100192	20141209T000000	538000.0	3	2.25	2570	7242	2.0	0	0	...	7	2170	400	1951	1991	98125
2	5631500400	20150225T000000	180000.0	2	1.00	770	10000	1.0	0	0	...	6	770	0	1933	0	98028
3	2487200875	20141209T000000	604000.0	4	3.00	1960	5000	1.0	0	0	...	7	1050	910	1965	0	98136
4	1954400510	20150218T000000	510000.0	3	2.00	1680	8080	1.0	0	0	...	8	1680	0	1987	0	98074
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
21608	263000018	20140521T000000	360000.0	3	2.50	1530	1131	3.0	0	0	...	8	1530	0	2009	0	98103
21609	6600060120	20150223T000000	400000.0	4	2.50	2310	5813	2.0	0	0	...	8	2310	0	2014	0	98146
21610	1523300141	20140623T000000	402101.0	2	0.75	1020	1350	2.0	0	0	...	7	1020	0	2009	0	98144
21611	291310100	20150116T000000	400000.0	3	2.50	1600	2388	2.0	0	0	...	8	1600	0	2004	0	98027
21612	1523300157	20141015T000000	325000.0	2	0.75	1020	1076	2.0	0	0	...	7	1020	0	2008	0	98144

21613 rows × 21 columns

In [33]:
```
house_df.head()
```

Out[33]:

	id	date	price	bedrooms	bathrooms	sqft_living	sqft_lot	floors	waterfront	view	...	grade	sqft_above	sqft_basement	yr_built	yr_renovated	zipcode	
0	7129300520	20141013T000000	221900.0	3	1.00	1180	5650	1.0	0	0	...	7	1180	0	1955	0	98178	47.
1	6414100192	20141209T000000	538000.0	3	2.25	2570	7242	2.0	0	0	...	7	2170	400	1951	1991	98125	47.
2	5631500400	20150225T000000	180000.0	2	1.00	770	10000	1.0	0	0	...	6	770	0	1933	0	98028	47.
3	2487200875	20141209T000000	604000.0	4	3.00	1960	5000	1.0	0	0	...	7	1050	910	1965	0	98136	47.
4	1954400510	20150218T000000	510000.0	3	2.00	1680	8080	1.0	0	0	...	8	1680	0	1987	0	98074	47.

5 rows × 21 columns

In [20]:
```
house_df.tail()
```

Out[20]:

	id	date	price	bedrooms	bathrooms	sqft_living	sqft_lot	floors	waterfront	view	...	grade	sqft_above	sqft_basement	yr_built	yr_renovated	zipcode
21608	263000018	20140521T000000	360000.0	3	2.50	1530	1131	3.0	0	0	...	8	1530	0	2009	0	98103
21609	6600060120	20150223T000000	400000.0	4	2.50	2310	5813	2.0	0	0	...	8	2310	0	2014	0	98146
21610	1523300141	20140623T000000	402101.0	2	0.75	1020	1350	2.0	0	0	...	7	1020	0	2009	0	98144
21611	291310100	20150116T000000	400000.0	3	2.50	1600	2388	2.0	0	0	...	8	1600	0	2004	0	98027
21612	1523300157	20141015T000000	325000.0	2	0.75	1020	1076	2.0	0	0	...	7	1020	0	2008	0	98144

5 rows × 21 columns

In [2]:
```
house_df.columns
```

Out[2]:
```
Index(['id', 'date', 'price', 'bedrooms', 'bathrooms', 'sqft_living',
       'sqft_lot', 'floors', 'waterfront', 'view', 'condition', 'grade',
       'sqft_above', 'sqft_basement', 'yr_built', 'yr_renovated', 'zipcode',
       'lat', 'long', 'sqft_living15', 'sqft_lot15'],
      dtype='object')
```

In [4]:
```
house_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 21613 entries, 0 to 21612
Data columns (total 21 columns):
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   id             21613 non-null  int64
 1   date           21613 non-null  object
 2   price          21613 non-null  float64
 3   bedrooms       21613 non-null  int64
 4   bathrooms      21613 non-null  float64
 5   sqft_living    21613 non-null  int64
 6   sqft_lot       21613 non-null  int64
 7   floors         21613 non-null  float64
 8   waterfront     21613 non-null  int64
 9   view           21613 non-null  int64
 10  condition      21613 non-null  int64
 11  grade          21613 non-null  int64
 12  sqft_above     21613 non-null  int64
 13  sqft_basement  21613 non-null  int64
 14  yr_built       21613 non-null  int64
 15  yr_renovated   21613 non-null  int64
 16  zipcode        21613 non-null  int64
 17  lat            21613 non-null  float64
 18  long           21613 non-null  float64
 19  sqft_living15  21613 non-null  int64
 20  sqft_lot15     21613 non-null  int64
dtypes: float64(5), int64(15), object(1)
memory usage: 3.5+ MB
```

In [5]:
```
house_df.isnull().sum()
```

Out[5]:
```
id               0
date             0
price            0
bedrooms         0
bathrooms        0
sqft_living      0
sqft_lot         0
floors           0
waterfront       0
view             0
condition        0
grade            0
sqft_above       0
sqft_basement    0
yr_built         0
yr_renovated     0
zipcode          0
lat              0
long             0
sqft_living15    0
sqft_lot15       0
dtype: int64
```

In [6]:
```
house_df.dtypes
```

Out[6]:
```
id                 int64
date              object
price            float64
bedrooms           int64
bathrooms        float64
sqft_living        int64
sqft_lot           int64
floors           float64
waterfront         int64
view               int64
condition          int64
grade              int64
sqft_above         int64
sqft_basement      int64
yr_built           int64
yr_renovated       int64
zipcode            int64
lat              float64
long             float64
sqft_living15      int64
sqft_lot15         int64
dtype: object
```

In [21]:
```
house_df.describe(include='all').T
```

Out[21]:

	count	unique	top	freq	mean	std	min	25%	50%	75%	max
id	21613.0	NaN	NaN	NaN	4580301520.864988	2876565571.312049	1000102.0	2123049194.0	3904930410.0	7308900445.0	9900000190.0
date	21613	372	20140623T000000	142	NaN	NaN	NaN	NaN	NaN	NaN	NaN
price	21613.0	NaN	NaN	NaN	540088.141767	367127.196483	75000.0	321950.0	450000.0	645000.0	7700000.0
bedrooms	21613.0	NaN	NaN	NaN	3.370842	0.930062	0.0	3.0	3.0	4.0	33.0
bathrooms	21613.0	NaN	NaN	NaN	2.114757	0.770163	0.0	1.75	2.25	2.5	8.0
sqft_living	21613.0	NaN	NaN	NaN	2079.899736	918.440897	290.0	1427.0	1910.0	2550.0	13540.0
sqft_lot	21613.0	NaN	NaN	NaN	15106.967566	41420.511515	520.0	5040.0	7618.0	10688.0	1651359.0
floors	21613.0	NaN	NaN	NaN	1.494309	0.539989	1.0	1.0	1.5	2.0	3.5
waterfront	21613.0	NaN	NaN	NaN	0.007542	0.086517	0.0	0.0	0.0	0.0	1.0
view	21613.0	NaN	NaN	NaN	0.234303	0.766318	0.0	0.0	0.0	0.0	4.0
condition	21613.0	NaN	NaN	NaN	3.40943	0.650743	1.0	3.0	3.0	4.0	5.0
grade	21613.0	NaN	NaN	NaN	7.656873	1.175459	1.0	7.0	7.0	8.0	13.0
sqft_above	21613.0	NaN	NaN	NaN	1788.390691	828.090978	290.0	1190.0	1560.0	2210.0	9410.0
sqft_basement	21613.0	NaN	NaN	NaN	291.509045	442.575043	0.0	0.0	0.0	560.0	4820.0
yr_built	21613.0	NaN	NaN	NaN	1971.005136	29.373411	1900.0	1951.0	1975.0	1997.0	2015.0
yr_renovated	21613.0	NaN	NaN	NaN	84.402258	401.67924	0.0	0.0	0.0	0.0	2015.0
zipcode	21613.0	NaN	NaN	NaN	98077.939805	53.505026	98001.0	98033.0	98065.0	98118.0	98199.0
lat	21613.0	NaN	NaN	NaN	47.560053	0.138564	47.1559	47.471	47.5718	47.678	47.7776
long	21613.0	NaN	NaN	NaN	-122.213896	0.140828	-122.519	-122.328	-122.23	-122.125	-121.315
sqft_living15	21613.0	NaN	NaN	NaN	1986.552492	685.391304	399.0	1490.0	1840.0	2360.0	6210.0
sqft_lot15	21613.0	NaN	NaN	NaN	12768.455652	27304.179631	651.0	5100.0	7620.0	10083.0	871200.0

In [23]:
```
house_df.drop('id',axis=1,inplace=True)
house_df.head()
```

Out[23]:

	date	price	bedrooms	bathrooms	sqft_living	sqft_lot	floors	waterfront	view	condition	grade	sqft_above	sqft_basement	yr_built	yr_renovated	zipcode	lat
0	20141013T000000	221900.0	3	1.00	1180	5650	1.0	0	0	3	7	1180	0	1955	0	98178	47.5112
1	20141209T000000	538000.0	3	2.25	2570	7242	2.0	0	0	3	7	2170	400	1951	1991	98125	47.7210
2	20150225T000000	180000.0	2	1.00	770	10000	1.0	0	0	3	6	770	0	1933	0	98028	47.7379
3	20141209T000000	604000.0	4	3.00	1960	5000	1.0	0	0	5	7	1050	910	1965	0	98136	47.5208
4	20150218T000000	510000.0	3	2.00	1680	8080	1.0	0	0	3	8	1680	0	1987	0	98074	47.6168

In [24]:
```
from sklearn.preprocessing import LabelEncoder
le=LabelEncoder()
house_df['date']=le.fit_transform(house_df['date'])
house_df['date'].dtype
```

Out[24]:
```
dtype('int32')
```

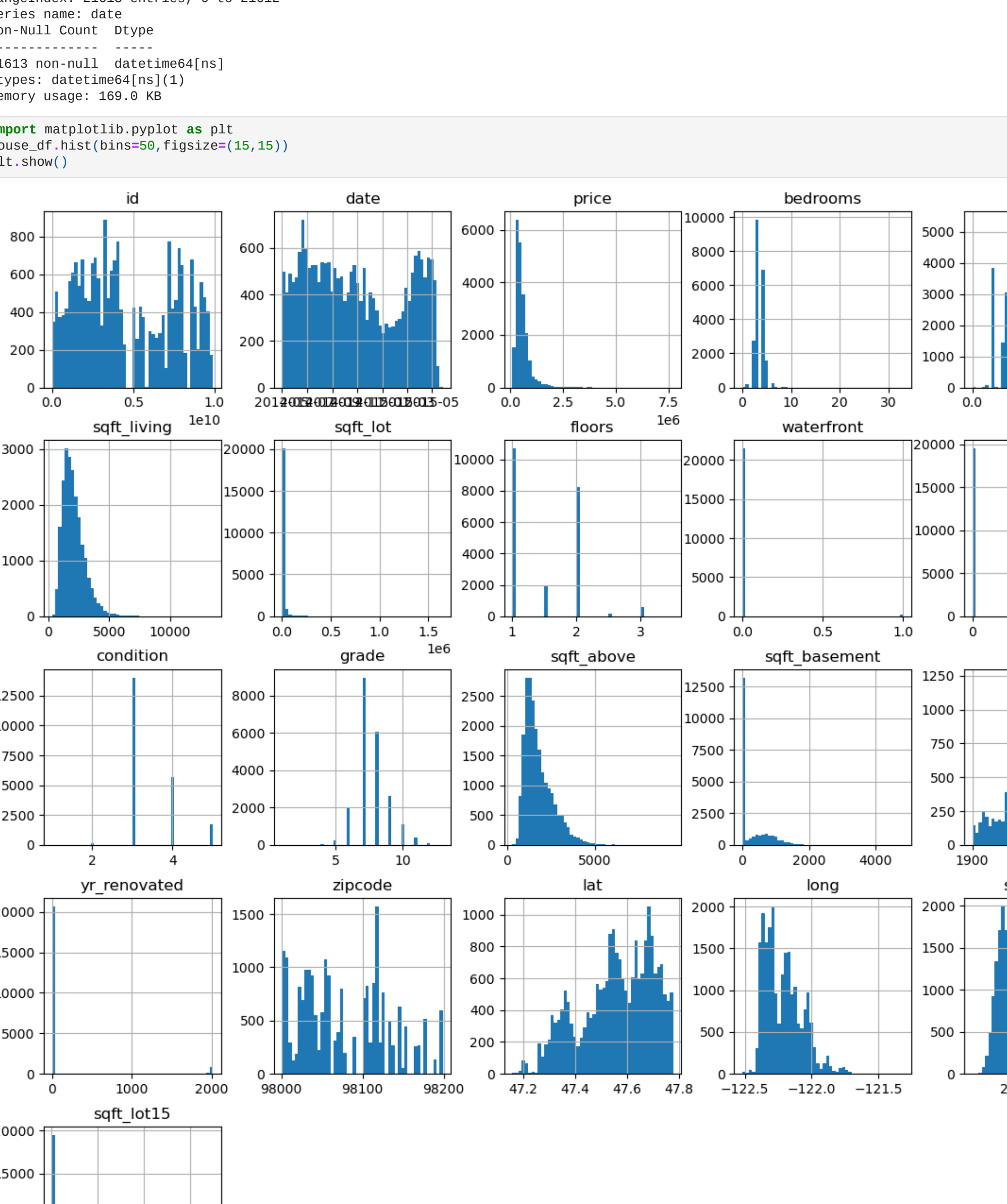
In [25]:
```
house_df['floors'].value_counts()
```

Out[25]:
```
floors
1.0    10680
2.0     8241
1.5     1910
3.0      613
2.5      161
3.5        8
Name: count, dtype: int64
```

In [7]:
```
house_df["date"]=pd.to_datetime(house_df["date"])
house_df["date"].info()
```

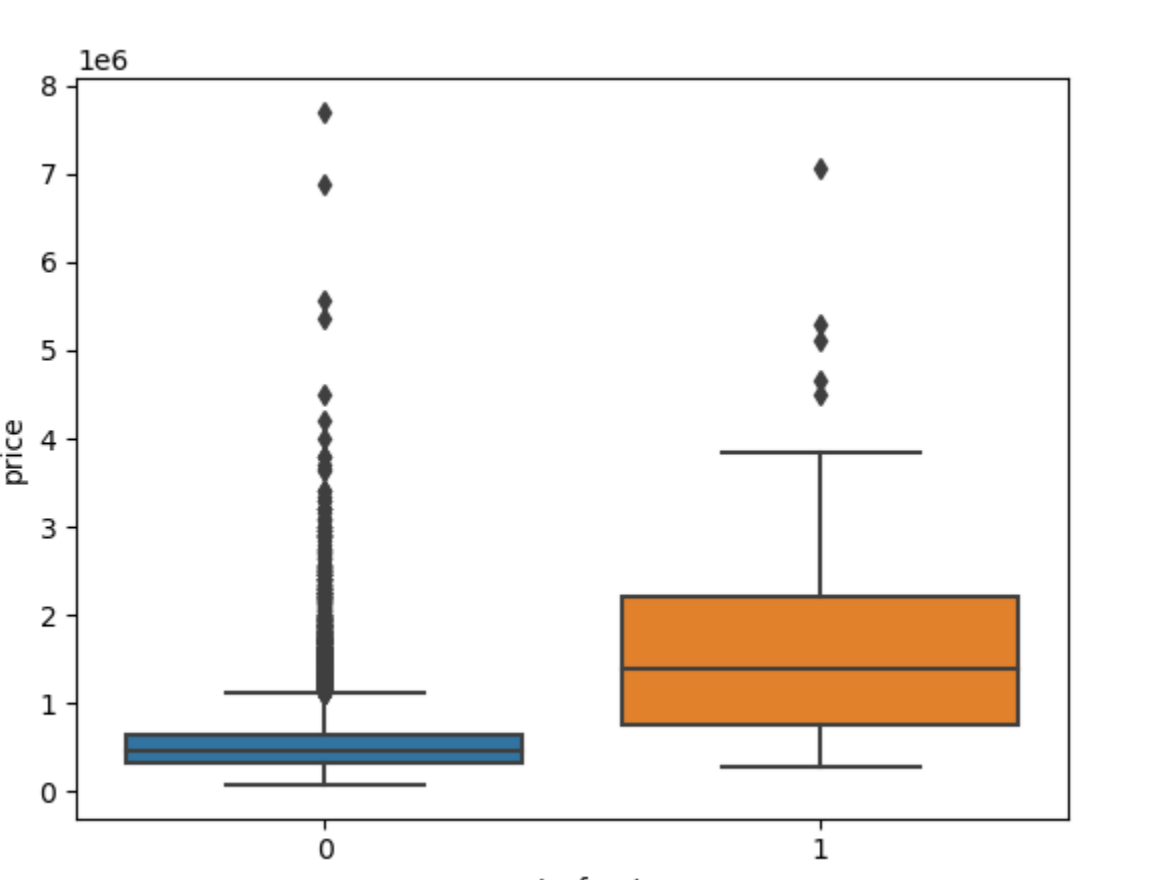
```
<class 'pandas.core.series.Series'>
RangeIndex: 21613 entries, 0 to 21612
Series name: date
Non-Null Count  Dtype
--------------  -----
21613 non-null  datetime64[ns]
dtypes: datetime64[ns](1)
memory usage: 169.0 KB
```

In [9]:
```
import matplotlib.pyplot as plt
house_df.hist(bins=50,figsize=(15,15))
plt.show()
```



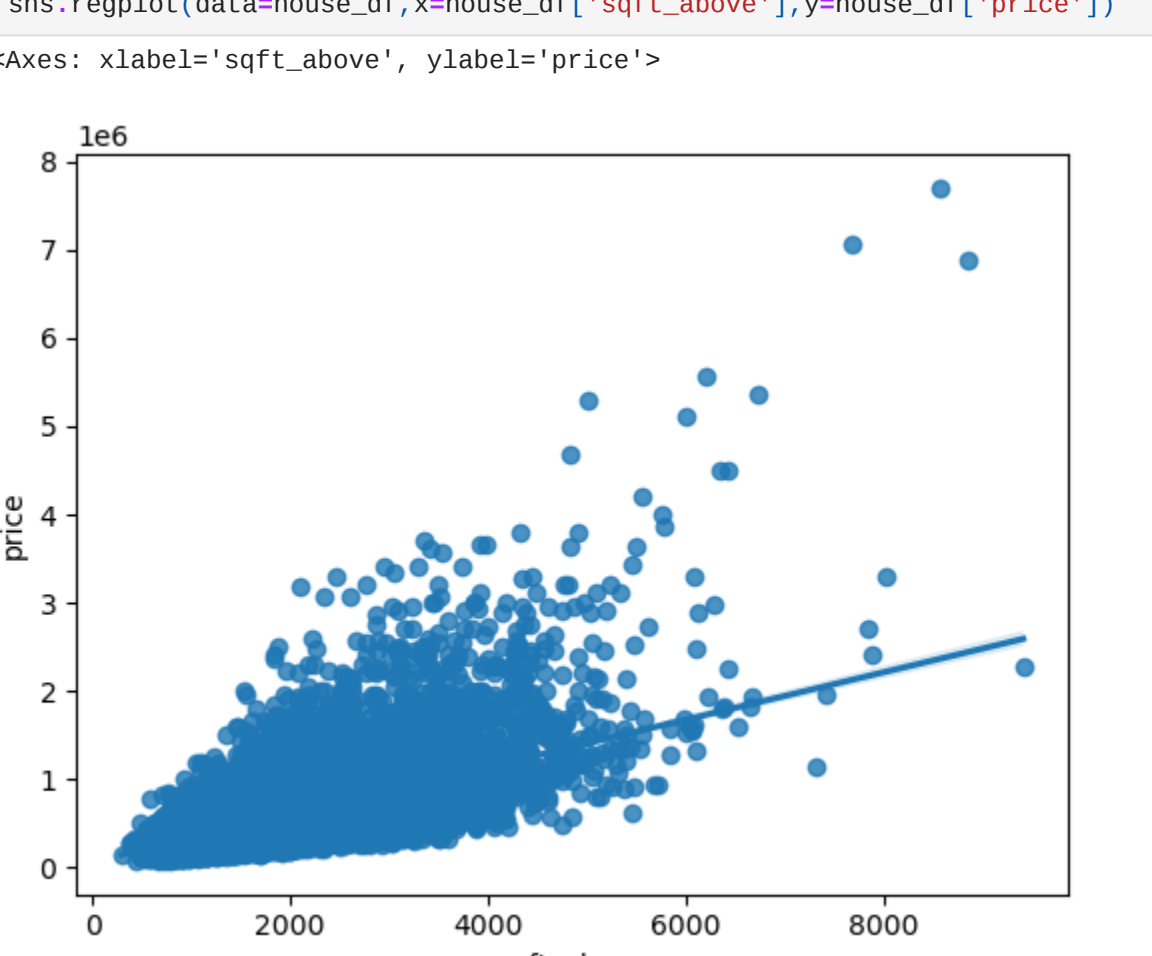
In [5]:
```
import seaborn as sns
sns.boxplot(data=house_df,x=house_df['waterfront'],y=house_df['price'])
```

Out[5]:
```
<Axes: xlabel='waterfront', ylabel='price'>
```



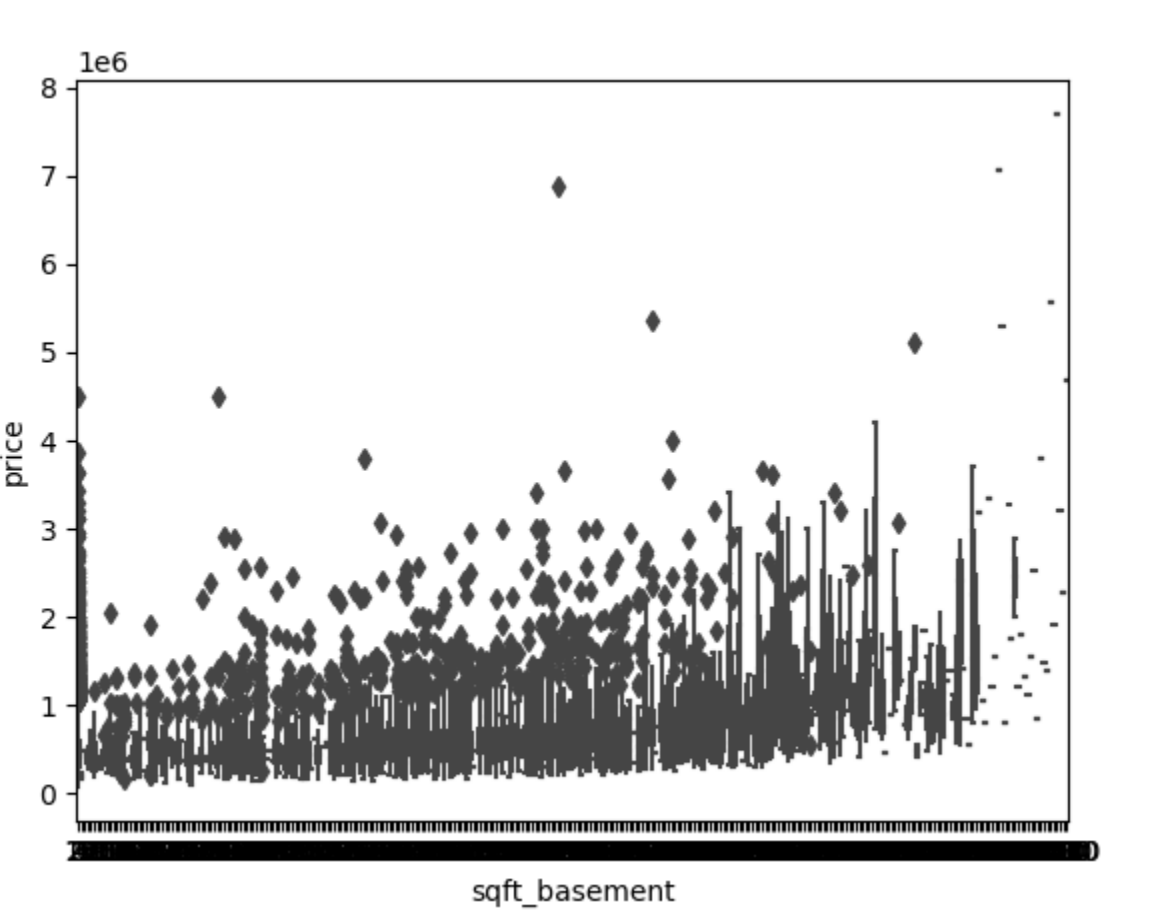
In [8]:
```
sns.regplot(data=house_df,x=house_df['sqft_above'],y=house_df['price'])
```

Out[8]:
```
<Axes: xlabel='sqft_above', ylabel='price'>
```



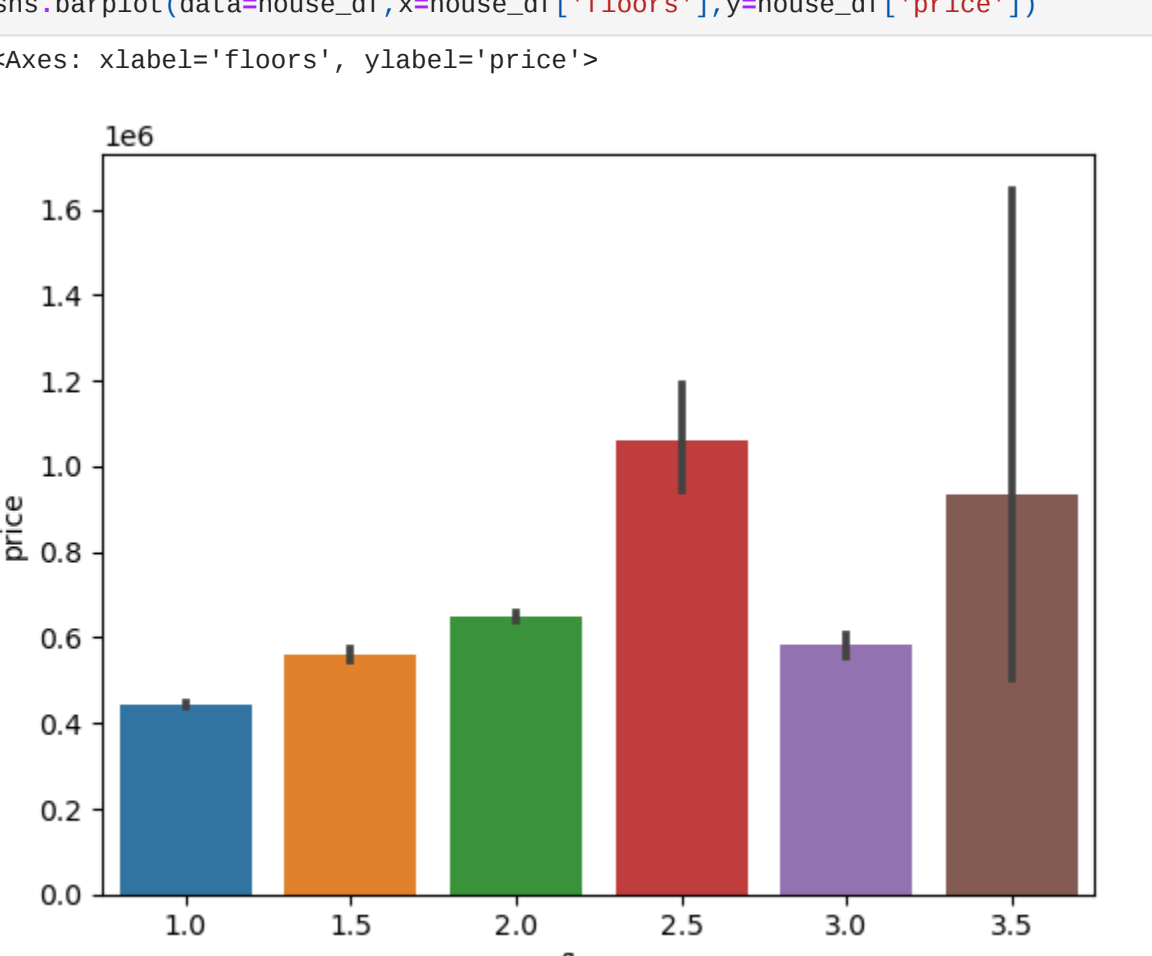
In [9]:
```
sns.boxplot(data=house_df,x=house_df['sqft_basement'],y=house_df['price'])
```

Out[9]:
```
<Axes: xlabel='sqft_basement', ylabel='price'>
```



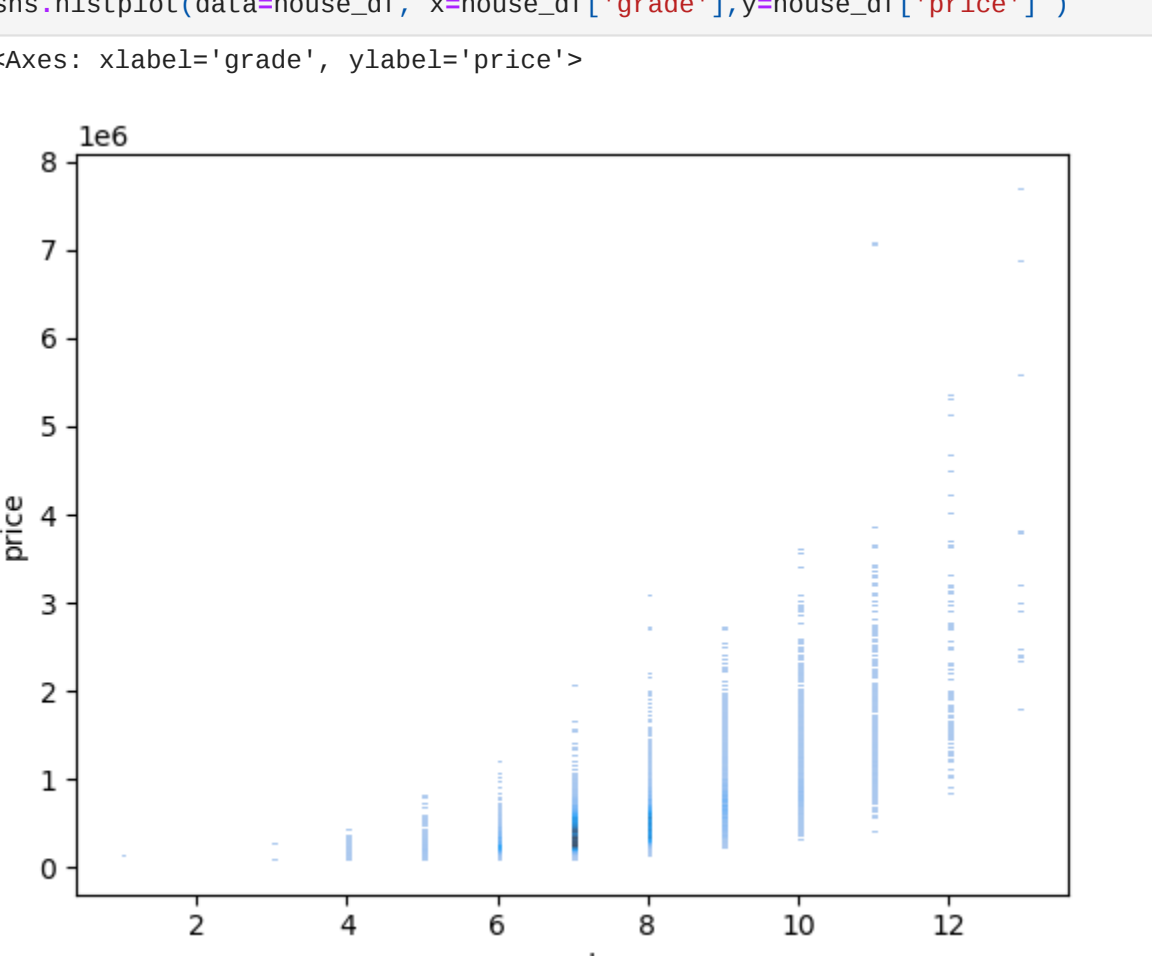
In [10]:
```
sns.barplot(data=house_df,x=house_df['floors'],y=house_df['price'])
```

Out[10]:
```
<Axes: xlabel='floors', ylabel='price'>
```



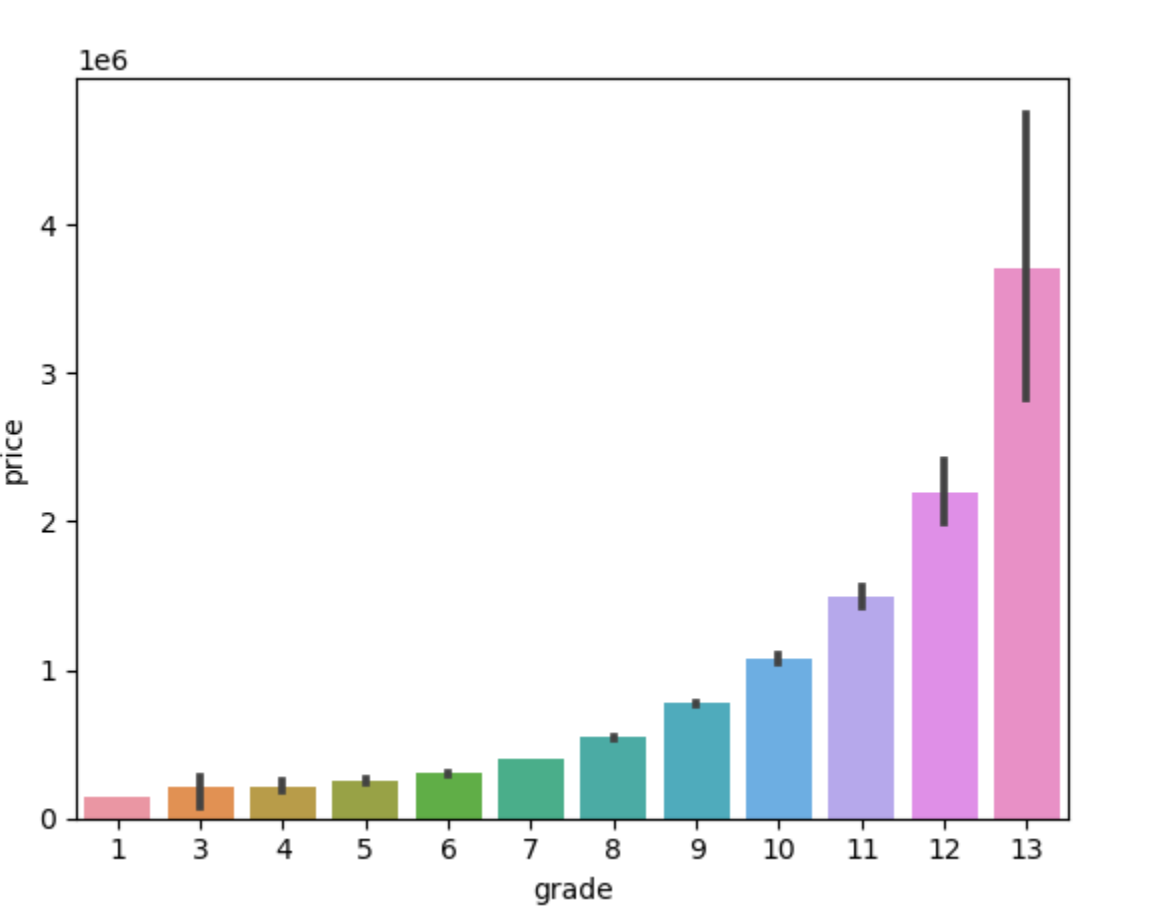
In [11]:
```
sns.histplot(data=house_df, x=house_df['grade'],y=house_df['price'] )
```

Out[11]:
```
<Axes: xlabel='grade', ylabel='price'>
```



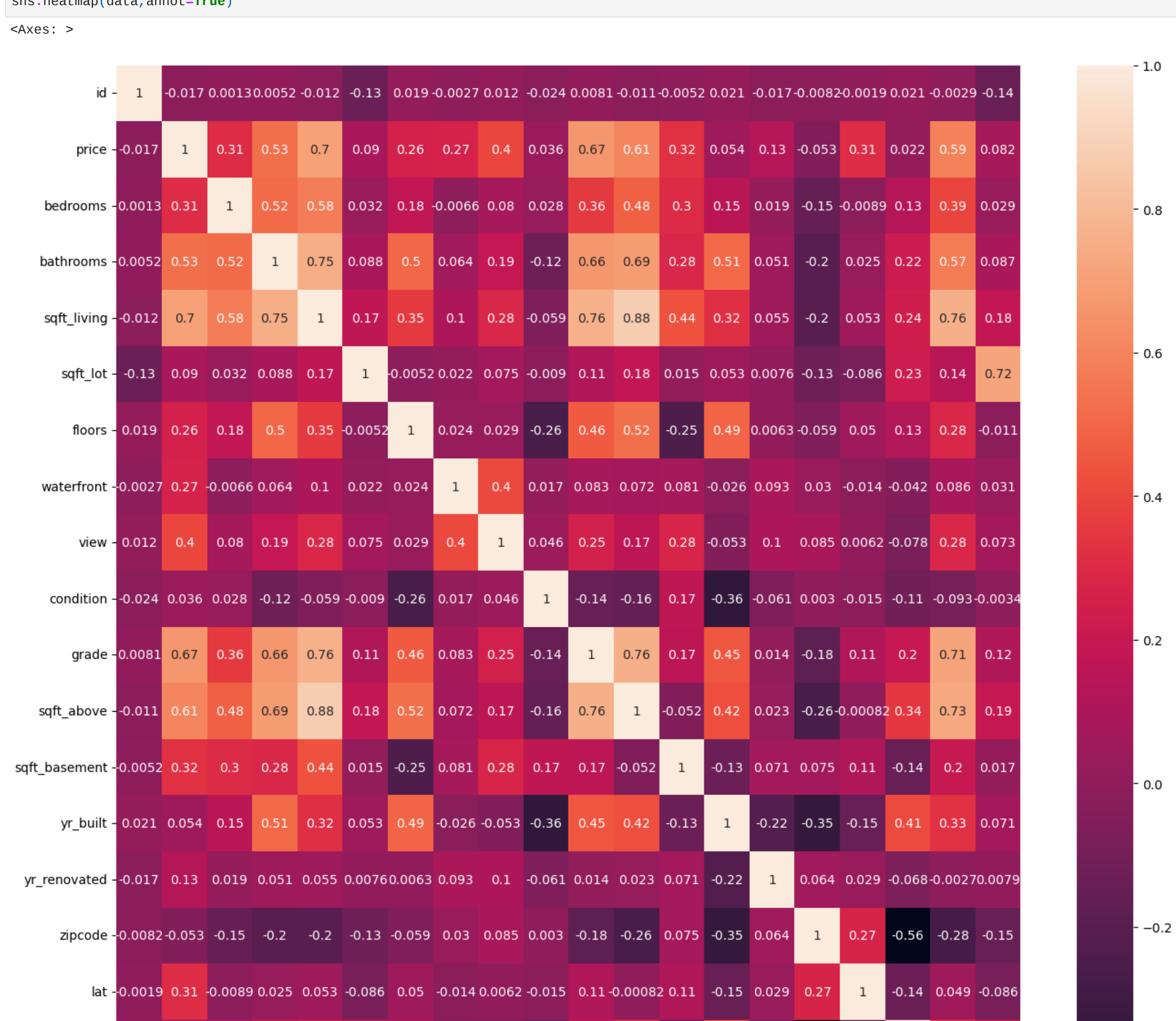
In [12]:
```
sns.barplot(data=house_df,x=house_df['grade'],y=house_df['price'])
```

Out[12]:
```
<Axes: xlabel='grade', ylabel='price'>
```



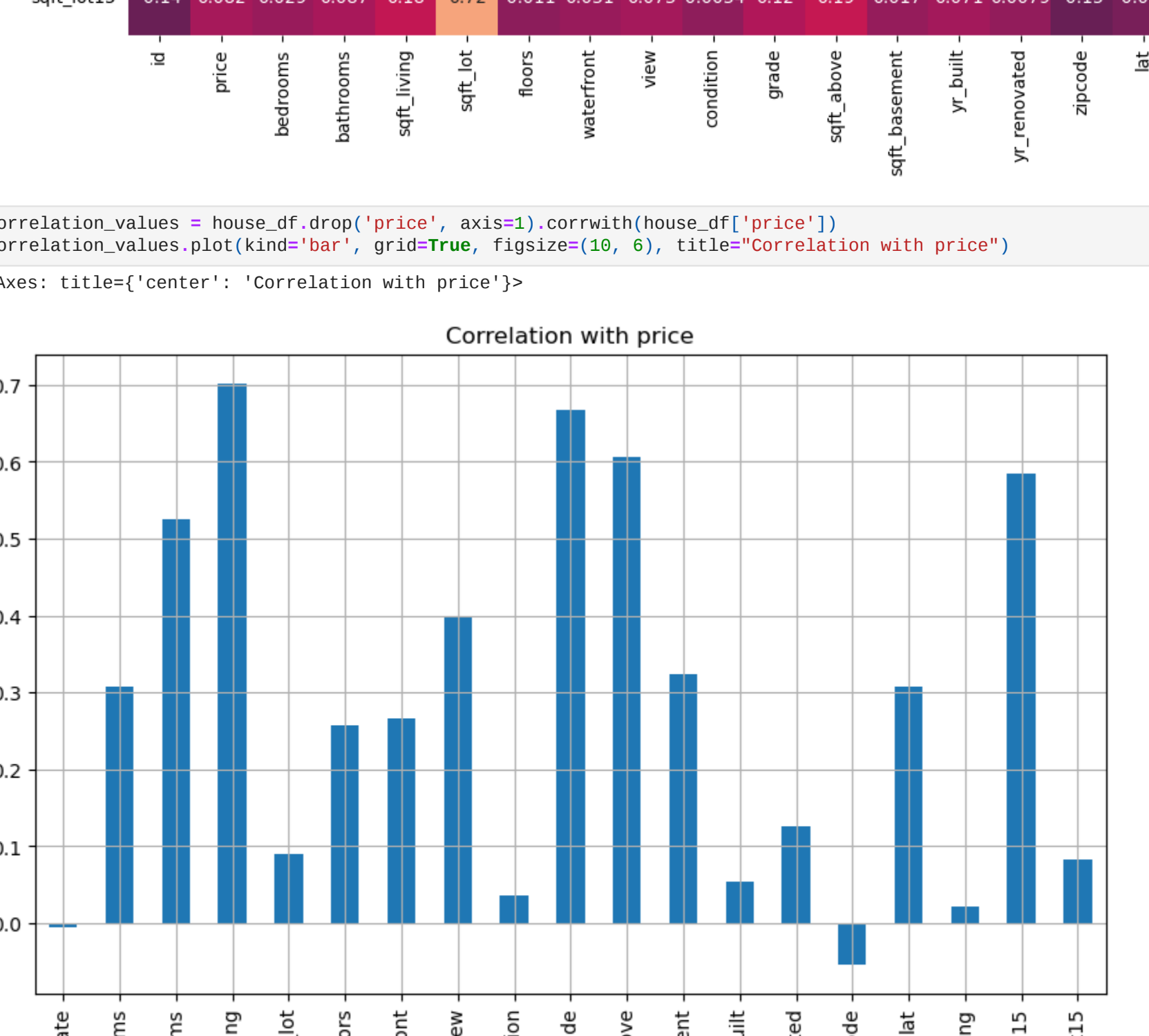
In [5]:
```
import matplotlib.pyplot as plt
import seaborn as sns
plt.figure(figsize=(16,16))
data=house_df.corr(numeric_only=True)
sns.heatmap(data,annot=True)
```

Out[5]:
```
<Axes: >
```



In [26]:
```
correlation_values = house_df.drop('price', axis=1).corrwith(house_df['price'])
correlation_values.plot(kind='bar', grid=True, figsize=(10, 6), title="Correlation with price")
```

Out[26]:
```
<Axes: title={'center': 'Correlation with price'}>
```



In [27]:
```
house_df.skew()
```

Out[27]:
```
date              0.147286
price             4.024069
bedrooms          1.974300
bathrooms         0.511108
sqft_living       1.471555
sqft_lot         13.060019
floors            0.616177
waterfront       11.385108
view              3.395750
condition         1.032805
grade             0.771103
sqft_above        1.446664
sqft_basement     1.577965
yr_built         -0.469805
yr_renovated      4.549493
zipcode           0.405661
lat              -0.485270
long              0.885053
sqft_living15     1.108181
sqft_lot15        9.506743
dtype: float64
```

In [29]:
```
import numpy as np
x=np.array(house_df.drop(columns='price'))
y=np.array(house_df.drop(columns='price'))
price=house_df['price']
space=house_df['sqft_living']
```

In [30]:
```
from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test=train_test_split(x,
                                                  y,
                                                  test_size=0.2,
                                                  random_state=42)
print(f'the shape of x_train is  {x_train.shape}')
print(f'the shape of x_test is {x_test.shape}')
print(f'the shape of y_train is {y_train.shape}')
print(f'the shape of y_test is {y_test.shape}')
```

```
the shape of x_train is  (17290, 19)
the shape of x_test is (4323, 19)
the shape of y_train is (17290, 19)
the shape of y_test is (4323, 19)
```

In [31]:
```
from sklearn.linear_model import LinearRegression
LR=LinearRegression()
LR.fit(x_train,y_train)
LR
```

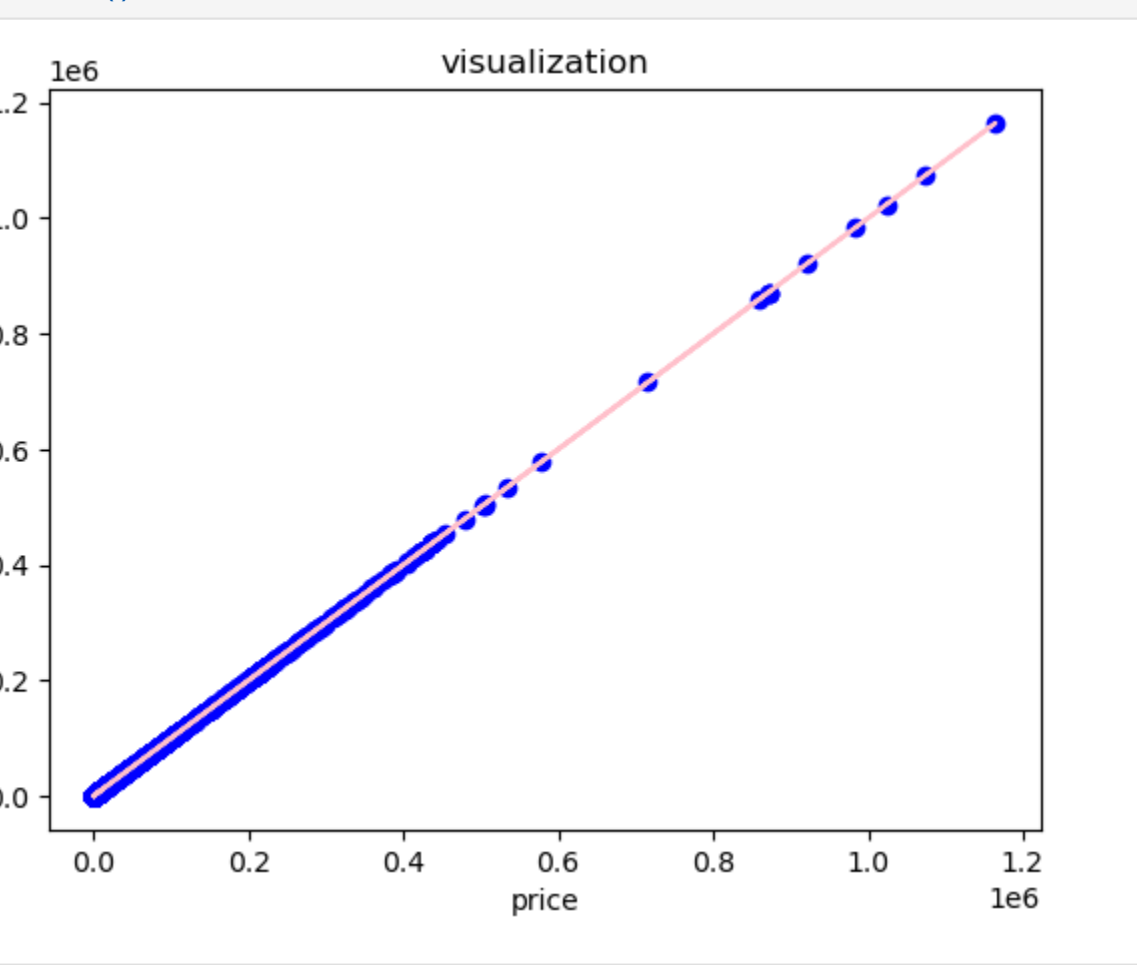
Out[31]:
```
▾ LinearRegression
LinearRegression()
```

In [32]:
```
y_predictions=LR.predict(x_test)
```

In [35]:
```
from sklearn.metrics import r2_score,mean_absolute_error
print(f'mean score is : {mean_absolute_error(y_test,y_predictions)}')
print(f'R2 score is : {r2_score(y_test,y_predictions)}')
```

```
mean score is : 2.48517617728855 63e-12
R2 score is : 1.0
```

In [36]:
```
plt.scatter(x_train,y_train,color='blue')
plt.plot(x_train,y_train,color='pink')
plt.title('visualization')
plt.xlabel('space')
plt.xlabel('price')
plt.show()
```



In [ ]: