HOUSE PRICE PREDICTION

importing dataset

In [1]:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
df= pd.read_csv(r'D:\data_science\kc_house_data.csv')
df
```

Out[1]:

	id	date	price	bedrooms	bathrooms	sqft_living	sqft_lot	flo
0	7129300520	20141013T000000	221900.0	3	1.00	1180	5650	
1	6414100192	20141209T000000	538000.0	3	2.25	2570	7242	
2	5631500400	20150225T000000	180000.0	2	1.00	770	10000	
3	2487200875	20141209T000000	604000.0	4	3.00	1960	5000	
4	1954400510	20150218T000000	510000.0	3	2.00	1680	8080	
...	...	...	...	...	...	...	...	
21608	263000018	20140521T000000	360000.0	3	2.50	1530	1131	
21609	6600060120	20150223T000000	400000.0	4	2.50	2310	5813	
21610	1523300141	20140623T000000	402101.0	2	0.75	1020	1350	
21611	291310100	20150116T000000	400000.0	3	2.50	1600	2388	
21612	1523300157	20141015T000000	325000.0	2	0.75	1020	1076	

21613 rows × 21 columns

Exploratory Data Analysis(EDA)

In [2]:

```
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     21611 non-null  float64
 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(6), int64(14), object(1)
memory usage: 3.5+ MB
```

In [3]:

```
sum(df.duplicated())
```

Out[3]:

```
0
```

In [4]:

```
df.shape
```

Out[4]:

```
(21613, 21)
```

In [5]:

```
df.describe().transpose()
```

Out[5]:

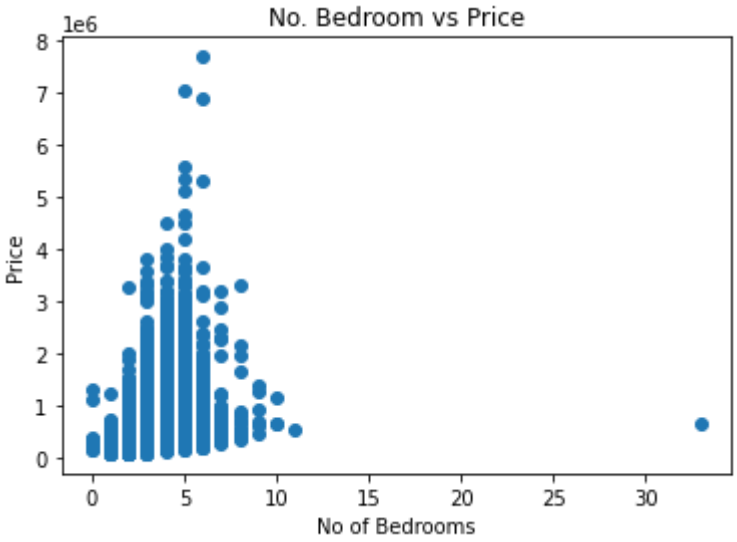
	count	mean	std	min	25%	5(
id	21613.0	4.580302e+09	2.876566e+09	1.000102e+06	2.123049e+09	3.904930e+
price	21613.0	5.400881e+05	3.671272e+05	7.500000e+04	3.219500e+05	4.500000e+
bedrooms	21613.0	3.370842e+00	9.300618e-01	0.000000e+00	3.000000e+00	3.000000e+
bathrooms	21613.0	2.114757e+00	7.701632e-01	0.000000e+00	1.750000e+00	2.250000e+
sqft_living	21613.0	2.079900e+03	9.184409e+02	2.900000e+02	1.427000e+03	1.910000e+
sqft_lot	21613.0	1.510697e+04	4.142051e+04	5.200000e+02	5.040000e+03	7.618000e+
floors	21613.0	1.494309e+00	5.399889e-01	1.000000e+00	1.000000e+00	1.500000e+
waterfront	21613.0	7.541757e-03	8.651720e-02	0.000000e+00	0.000000e+00	0.000000e+
view	21613.0	2.343034e-01	7.663176e-01	0.000000e+00	0.000000e+00	0.000000e+
condition	21613.0	3.409430e+00	6.507430e-01	1.000000e+00	3.000000e+00	3.000000e+
grade	21613.0	7.656873e+00	1.175459e+00	1.000000e+00	7.000000e+00	7.000000e+
sqft_above	21611.0	1.788396e+03	8.281282e+02	2.900000e+02	1.190000e+03	1.560000e+
sqft_basement	21613.0	2.915090e+02	4.425750e+02	0.000000e+00	0.000000e+00	0.000000e+
yr_built	21613.0	1.971005e+03	2.937341e+01	1.900000e+03	1.951000e+03	1.975000e+
yr_renovated	21613.0	8.440226e+01	4.016792e+02	0.000000e+00	0.000000e+00	0.000000e+
zipcode	21613.0	9.807794e+04	5.350503e+01	9.800100e+04	9.803300e+04	9.806500e+
lat	21613.0	4.756005e+01	1.385637e-01	4.715590e+01	4.747100e+01	4.757180e+
long	21613.0	-1.222139e+02	1.408283e-01	-1.225190e+02	-1.223280e+02	-1.222300e+
sqft_living15	21613.0	1.986552e+03	6.853913e+02	3.990000e+02	1.490000e+03	1.840000e+
sqft_lot15	21613.0	1.276846e+04	2.730418e+04	6.510000e+02	5.100000e+03	7.620000e+

In [6]:

```
#Treating Outliers
bed=df['bedrooms']
pr=df['price']
plt.title("No. Bedroom vs Price")
plt.xlabel("No of Bedrooms")
plt.ylabel("Price")
plt.scatter(bed,pr)
 #no of bedroom=33 is an outlier which can be removed
```

Out[6]:

```
<matplotlib.collections.PathCollection at 0x1e996df6e80>
```



Data preparation

In [7]:

```
# check for the extreme values shown in the describe.
df=df[df['bedrooms']<33]
print(df['bedrooms'])
```

```
0        3
1        3
2        2
3        4
4        3
        ..
21608    3
21609    4
21610    2
21611    3
21612    2
Name: bedrooms, Length: 21612, dtype: int64
```

In [8]:

```
df.isnull().sum() #2 records in sqft_above is null
```

Out[8]:

```
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       2
sqft_basement    0
yr_built         0
yr_renovated     0
zipcode          0
lat              0
long             0
sqft_living15    0
sqft_lot15       0
dtype: int64
```

In [9]:

```
#treating null values
df=df.replace(np.nan,0)
```

In [10]:

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

Out[10]:

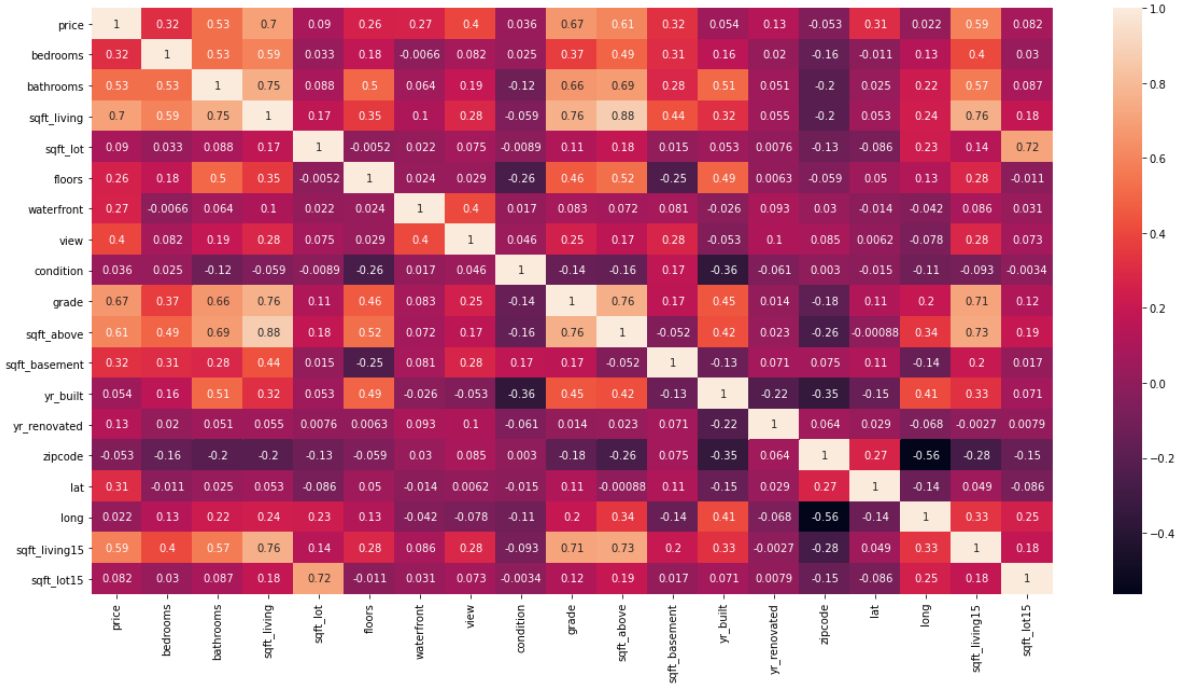
```
id               0
date             0
price            0
bedrooms         0
bathrooms        0
sqft_living      0
sqft_lot         0
floors           0
waterfront       0
view             0
condition        0
grade            0
sqft_above       0
sqft_basement    0
yr_built         0
yr_renovated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

In [11]:

```
plt.figure(figsize=(20, 10))
sns.heatmap(df.drop('id', axis=1).corr(), annot=True, )
```

Out[11]:

```
<AxesSubplot:>
```



Model Training

MULTIPLE

In [12]:

```
x=df[['bedrooms','bathrooms','sqft_lot','sqft_living','yr_built']]
y=df[['price']]
print(x)
```

```
       bedrooms  bathrooms  sqft_lot  sqft_living  yr_built
0             3       1.00      5650         1180      1955
1             3       2.25      7242         2570      1951
2             2       1.00     10000          770      1933
3             4       3.00      5000         1960      1965
4             3       2.00      8080         1680      1987
...         ...        ...       ...          ...       ...
21608         3       2.50      1131         1530      2009
21609         4       2.50      5813         2310      2014
21610         2       0.75      1350         1020      2009
21611         3       2.50      2388         1600      2004
21612         2       0.75      1076         1020      2008

[21612 rows x 5 columns]
```

In [13]:

```
#!pip install scikit-learn
from sklearn.model_selection import train_test_split
x_train, x_test , y_train , y_test = train_test_split(x,y,test_size=0.20,random_state=101)
```

In [14]:

```
from sklearn.linear_model import LinearRegression
from sklearn.metrics import r2_score
lr=LinearRegression()
```

In [15]:

```
lr.fit(x_train,y_train)
```

Out[15]:

```
LinearRegression()
```

In [16]:

```
y_pred=lr.predict(x_test)
r2_score(y_test,y_pred)
```

Out[16]:

```
0.5746585054942905
```

POLYNOMIAL

In [17]:

```
from sklearn.preprocessing import PolynomialFeatures
poly_reg = PolynomialFeatures(degree =2)
x_poly = poly_reg.fit_transform(x)
lin_reg2 = LinearRegression()
lin_reg2.fit(x_poly,y)


print(f'R² score: {r2_score(y, lin_reg2.predict(poly_reg.fit_transform(x)))*100}')
```

```
R² score: 60.22231360110801
```

RANDOMFOREST

In [18]:

```
from sklearn.ensemble import RandomForestRegressor
rf_reg=RandomForestRegressor(n_estimators = 10,random_state=0)
rf_reg.fit(x,y)

print(f'R² score: {r2_score(y, rf_reg.predict(x))*100}')
```

```
<ipython-input-18-6ef2ce8ae430>:3: DataConversionWarning: A column-vector y
was passed when a 1d array was expected. Please change the shape of y to (n_
samples,), for example using ravel().
  rf_reg.fit(x,y)
```

```
R² score: 92.24570347793016
```

Model prediction

In [19]:

```
new_house=[[3,2,2200,1982,1989]]
print('predicted value is:',int(rf_reg.predict(new_house)))
```

```
predicted value is: 549421
```

In [21]:

```
new_house=[[4,1,0.53,3030,1926]]
print('predicted value is:',int(rf_reg.predict(new_house)))
```

```
predicted value is: 918100
```

In [ ]:

```

```