In [1]:
```
import pandas as pd
import numpy as np
```

In [2]:
```
df=pd.read_csv(r"C:\Users\kaush\Downloads\housing.csv")
```

In [3]:
```
df
```

Out[3]:

	RM	LSTAT	PTRATIO	MEDV
0	6.575	4.98	15.3	504000.0
1	6.421	9.14	17.8	453600.0
2	7.185	4.03	17.8	728700.0
3	6.998	2.94	18.7	701400.0
4	7.147	5.33	18.7	760200.0
...	...	...	...	...
484	6.593	9.67	21.0	470400.0
485	6.120	9.08	21.0	432600.0
486	6.976	5.64	21.0	501900.0
487	6.794	6.48	21.0	462000.0
488	6.030	7.88	21.0	249900.0

489 rows × 4 columns

In [4]:
```
df.mean()
```

Out[4]:
```
RM              6.240288
LSTAT          12.939632
PTRATIO        18.516564
MEDV       454342.944785
dtype: float64
```

In [5]:
```
df.loc[:,'RM'].mean()
```

Out[5]:
```
6.240288343558283
```

In [6]:
```
df.mean(axis=1)[0:4]
```

Out[6]:
```
0    126006.71375
1    113408.34025
2    182182.25375
3    175357.15950
dtype: float64
```

In [7]:
```
1 df.median()
```

Out[7]:
```
RM              6.185
LSTAT          11.690
PTRATIO        19.100
MEDV       438900.000
dtype: float64
```

In [8]:
```
1 df.loc[:,'RM'].median()
```

Out[8]:
```
6.185
```

In [9]:
```
1 df.median(axis=1)[0:4]
```

Out[9]:
```
0    10.9375
1    13.4700
2    12.4925
3    12.8490
dtype: float64
```

In [10]:
```
1 df.mode()
```

Out[10]:

	RM	LSTAT	PTRATIO	MEDV
0	5.713	6.36	20.2	525000.0
1	6.127	7.79	NaN	NaN
2	6.167	8.05	NaN	NaN
3	6.229	14.10	NaN	NaN
4	6.405	18.13	NaN	NaN
5	6.417	NaN	NaN	NaN

In [11]:
```
1 df.loc[:,'RM'].mode()
```

Out[11]:
```
0    5.713
1    6.127
2    6.167
3    6.229
4    6.405
5    6.417
Name: RM, dtype: float64
```

In [12]:
```
1 df.min()
```

Out[12]:
```
RM              3.561
LSTAT           1.980
PTRATIO        12.600
MEDV       105000.000
dtype: float64
```

In [13]:
```
1 df.loc[:,'RM'].min(skipna=False)
```

Out[13]:
```
3.561
```

In [14]:
```
1 df.max()
```

Out[14]:
```
RM               8.398
LSTAT           37.970
PTRATIO         22.000
MEDV       1024800.000
dtype: float64
```

In [16]:
```
1 df.loc[:,'RM'].max(skipna=False)
```

Out[16]:
```
8.398
```

In [17]:
```
1 df.std()
```

Out[17]:
```
RM              0.643650
LSTAT           7.081990
PTRATIO         2.111268
MEDV       165340.277653
dtype: float64
```

In [18]:
```
1 df.loc[:,'RM'].std()
```

Out[18]:
```
0.6436497627572431
```

In [19]:
```
1 df.std(axis=1)[0:4]
```

Out[19]:
```
0    251995.524207
1    226794.439885
2    364345.164214
3    350695.227064
dtype: float64
```

In [20]:
```
1 df.groupby(['LSTAT'])['RM'].mean()
```

Out[20]:
```
LSTAT
1.98     7.024
2.47     8.337
2.87     7.178
2.94     6.998
2.98     6.854
         ...
34.37    4.628
34.41    5.019
34.77    4.906
36.98    4.519
37.97    4.138
Name: RM, Length: 442, dtype: float64
```

In [22]:
```
from scipy import stats
df=pd.read_csv(r"C:\Users\kaush\Downloads\housing.csv")
z= np.abs(stats.zscore(df['RM']))
print(z)
```

```
0      0.520554
1      0.281048
2      1.469245
3      1.178417
4      1.410146
         ...
484    0.548548
485    0.187076
486    1.144202
487    0.861150
488    0.327047
Name: RM, Length: 489, dtype: float64
```

In [23]:
```
threshold=0.18
sample_outliers=np.where(z<threshold)
sample_outliers
```

Out[23]:
```
(array([  7,  22,  42,  43,  58,  65,  73,  74,  75,  76,  77,  78,  82,
         83,  93,  94,  96, 103, 104, 107, 109, 110, 114, 116, 131, 135,
        139, 140, 145, 154, 155, 160, 164, 173, 177, 181, 195, 199, 208,
        214, 239, 261, 275, 276, 277, 282, 287, 302, 315, 318, 319, 322,
        323, 345, 347, 350, 362, 376, 384, 410, 411, 413, 417, 420, 425,
        429, 430, 431, 435, 441, 445, 447, 453, 454, 458, 461, 462, 463,
        468], dtype=int64),)
```

In [24]:
```
sorted_rscore= sorted(df['RM'])
sorted_rscore
q1 =np.percentile(sorted_rscore,6)
q3 =np.percentile(sorted_rscore,9)
print(q1,q3)
```

```
5.392519999999999 5.5580799999999995
```

In [25]:
```
IQR = q3-q1
lwr_bound =q1-(1.5*IQR)
upr_bound =q3+(1.5*IQR)
print(lwr_bound, upr_bound)
```

```
5.144179999999999 5.806419999999999
```

In [26]:
```
r_outliers =[]
for i in sorted_rscore:
    if(i<lwr_bound or i>upr_bound):
        r_outliers.append(i)
print(r_outliers)
```

```
[3.561, 3.863, 4.138, 4.138, 4.368, 4.519, 4.628, 4.652, 4.88, 4.903, 4.906,
4.926, 4.963, 4.973, 5.0, 5.012, 5.019, 5.036, 5.093, 5.807, 5.813, 5.813, 5.
818, 5.822, 5.834, 5.836, 5.837, 5.841, 5.85, 5.851, 5.852, 5.854, 5.854, 5.8
56, 5.856, 5.857, 5.859, 5.868, 5.869, 5.87, 5.871, 5.872, 5.874, 5.875, 5.87
6, 5.877, 5.878, 5.879, 5.88, 5.884, 5.885, 5.887, 5.888, 5.888, 5.889, 5.89
1, 5.895, 5.896, 5.898, 5.905, 5.913, 5.914, 5.92, 5.924, 5.926, 5.926, 5.92
7, 5.928, 5.933, 5.935, 5.935, 5.936, 5.936, 5.942, 5.949, 5.95, 5.951, 5.95
2, 5.957, 5.96, 5.961, 5.961, 5.963, 5.965, 5.966, 5.966, 5.968, 5.972, 5.97
6, 5.981, 5.983, 5.983, 5.985, 5.986, 5.987, 5.99, 5.998, 6.003, 6.004, 6.00
4, 6.006, 6.009, 6.009, 6.012, 6.014, 6.015, 6.019, 6.02, 6.021, 6.023, 6.02
7, 6.03, 6.03, 6.031, 6.037, 6.041, 6.047, 6.051, 6.059, 6.064, 6.065, 6.066,
6.069, 6.072, 6.081, 6.083, 6.086, 6.092, 6.095, 6.096, 6.096, 6.101, 6.103,
6.108, 6.108, 6.112, 6.113, 6.114, 6.115, 6.12, 6.121, 6.122, 6.122, 6.127,
6.127, 6.127, 6.129, 6.13, 6.137, 6.14, 6.142, 6.144, 6.144, 6.145, 6.151, 6.
152, 6.152, 6.153, 6.162, 6.162, 6.163, 6.164, 6.167, 6.167, 6.167, 6.169, 6.
172, 6.174, 6.176, 6.182, 6.185, 6.185, 6.193, 6.193, 6.195, 6.202, 6.208, 6.
209, 6.209, 6.211, 6.211, 6.212, 6.219, 6.223, 6.226, 6.229, 6.229, 6.229, 6.
23, 6.232, 6.24, 6.242, 6.245, 6.249, 6.25, 6.251, 6.251, 6.254, 6.266, 6.27
3, 6.279, 6.286, 6.29, 6.297, 6.301, 6.302, 6.31, 6.312, 6.312, 6.315, 6.315,
6.316, 6.317, 6.319, 6.326, 6.326, 6.333, 6.335, 6.341, 6.343, 6.345, 6.348,
6.358, 6.372, 6.373, 6.375, 6.376, 6.376, 6.377, 6.38, 6.38, 6.382, 6.383, 6.
389, 6.393, 6.395, 6.398, 6.402, 6.404, 6.405, 6.405, 6.405, 6.406, 6.411, 6.
415, 6.416, 6.417, 6.417, 6.417, 6.421, 6.425, 6.426, 6.43, 6.431, 6.431, 6.4
33, 6.434, 6.436, 6.437, 6.438, 6.442, 6.453, 6.454, 6.456, 6.458, 6.459, 6.4
61, 6.471, 6.474, 6.481, 6.482, 6.484, 6.485, 6.487, 6.49, 6.495, 6.495, 6.5
1, 6.511, 6.513, 6.516, 6.525, 6.538, 6.54, 6.545, 6.546, 6.549, 6.552, 6.55
6, 6.563, 6.565, 6.567, 6.575, 6.579, 6.59, 6.593, 6.595, 6.604, 6.606, 6.61
6, 6.618, 6.619, 6.625, 6.629, 6.63, 6.63, 6.631, 6.635, 6.635, 6.642, 6.649,
6.655, 6.657, 6.674, 6.678, 6.696, 6.701, 6.715, 6.718, 6.726, 6.727, 6.727,
6.728, 6.728, 6.739, 6.749, 6.75, 6.758, 6.762, 6.77, 6.781, 6.782, 6.782, 6.
794, 6.794, 6.8, 6.812, 6.816, 6.824, 6.826, 6.833, 6.842, 6.849, 6.852, 6.85
4, 6.86, 6.861, 6.871, 6.874, 6.879, 6.897, 6.939, 6.943, 6.951, 6.951, 6.95
7, 6.968, 6.968, 6.975, 6.976, 6.98, 6.98, 6.982, 6.998, 7.007, 7.014, 7.024,
7.041, 7.061, 7.079, 7.088, 7.104, 7.107, 7.135, 7.147, 7.148, 7.155, 7.163,
7.178, 7.185, 7.185, 7.203, 7.206, 7.236, 7.241, 7.249, 7.267, 7.274, 7.287,
7.313, 7.327, 7.333, 7.358, 7.393, 7.412, 7.416, 7.42, 7.454, 7.47, 7.52, 7.6
1, 7.645, 7.686, 7.691, 7.765, 7.82, 7.82, 7.853, 8.04, 8.069, 8.247, 8.259,
8.266, 8.337, 8.398]
```