In [1]:

```
import numpy as np
import pandas as pd
```

Read dataset

In [2]:

```
auto_mpg = pd.read_csv("C:/Users/91920/Desktop/New folder/auto-mpg.csv")
```

View datset

In [3]:

```
auto_mpg.shape
```

Out[3]:

```
(398, 9)
```

In [4]:

```
auto_mpg.head()
```

Out[4]:

	mpg	cylinders	displacement	horsepower	weight	acceleration	model year	origin	car name
0	18.0	8	307.0	130	3504	12.0	70	1	chevrolet chevelle malibu
1	15.0	8	350.0	165	3693	11.5	70	1	buick skylark 320
2	18.0	8	318.0	150	3436	11.0	70	1	plymouth satellite
3	16.0	8	304.0	150	3433	12.0	70	1	amc rebel sst
4	17.0	8	302.0	140	3449	10.5	70	1	ford torino

In [5]:

```
auto_mpg.tail()
```

Out[5]:

	mpg	cylinders	displacement	horsepower	weight	acceleration	model year	origin	car name
393	27.0	4	140.0	86	2790	15.6	82	1	ford mustang gl
394	44.0	4	97.0	52	2130	24.6	82	2	vw pickup
395	32.0	4	135.0	84	2295	11.6	82	1	dodge rampage
396	28.0	4	120.0	79	2625	18.6	82	1	ford ranger
397	31.0	4	119.0	82	2720	19.4	82	1	chevy s-10

https://archive.ics.uci.edu/ml/datasets/auto+mpg (https://archive.ics.uci.edu/ml/datasets/auto+mpg)

Descriptive Statistics

In [6]:

```
auto_mpg.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 398 entries, 0 to 397
Data columns (total 9 columns):
 #   Column        Non-Null Count  Dtype  
---  ------        --------------  -----  
 0   mpg           398 non-null    float64
 1   cylinders     398 non-null    int64  
 2   displacement  398 non-null    float64
 3   horsepower    398 non-null    object 
 4   weight        398 non-null    int64  
 5   acceleration  398 non-null    float64
 6   model year    398 non-null    int64  
 7   origin        398 non-null    int64  
 8   car name      398 non-null    object 
dtypes: float64(3), int64(4), object(2)
memory usage: 28.1+ KB
```

In [7]:

```
auto_mpg.describe()
```

Out[7]:

	mpg	cylinders	displacement	weight	acceleration	model year	origin
count	398.000000	398.000000	398.000000	398.000000	398.000000	398.000000	398.000000
mean	23.514573	5.454774	193.425879	2970.424623	15.568090	76.010050	1.572864
std	7.815984	1.701004	104.269838	846.841774	2.757689	3.697627	0.802055
min	9.000000	3.000000	68.000000	1613.000000	8.000000	70.000000	1.000000
25%	17.500000	4.000000	104.250000	2223.750000	13.825000	73.000000	1.000000
50%	23.000000	4.000000	148.500000	2803.500000	15.500000	76.000000	1.000000
75%	29.000000	8.000000	262.000000	3608.000000	17.175000	79.000000	2.000000
max	46.600000	8.000000	455.000000	5140.000000	24.800000	82.000000	3.000000

pd.set_option('display.max_rows', None)

But then, why is horsepower an object and not a float, the values we saw above were clearly numbers

In [8]:

```
auto_mpg.horsepower.unique()
```

Out[8]:

```
array(['130', '165', '150', '140', '198', '220', '215', '225', '190',
       '170', '160', '95', '97', '85', '88', '46', '87', '90', '113',
       '200', '210', '193', '?', '100', '105', '175', '153', '180', '110',
       '72', '86', '70', '76', '65', '69', '60', '80', '54', '208', '155',
       '112', '92', '145', '137', '158', '167', '94', '107', '230', '49',
       '75', '91', '122', '67', '83', '78', '52', '61', '93', '148',
       '129', '96', '71', '98', '115', '53', '81', '79', '120', '152',
       '102', '108', '68', '58', '149', '89', '63', '48', '66', '139',
       '103', '125', '133', '138', '135', '142', '77', '62', '132', '84',
       '64', '74', '116', '82'], dtype=object)
```

In [9]:

```
auto_mpg.replace(to_replace = '?', value = np.nan , inplace=True)
auto_mpg.style.highlight_null(null_color='red')
```

231	15.500000	8	400.000000	190	4325	12.200000	77	1	chrysler cordoba
232	16.000000	8	351.000000	149	4335	14.500000	77	1	ford thunderbird
233	29.000000	4	97.000000	78	1940	14.500000	77	2	volkswagen rabbit custom
234	24.500000	4	151.000000	88	2740	16.000000	77	1	pontiac sunbird coupe
235	26.000000	4	97.000000	75	2265	18.200000	77	3	toyota corolla liftback
236	25.500000	4	140.000000	89	2755	15.800000	77	1	ford mustang ii 2+2
237	30.500000	4	98.000000	63	2051	17.000000	77	1	chevrolet chevette

auto_mpg.style.highlight_null(null_color='red')

In [10]:

```
auto_mpg.isnull().sum()
```

Out[10]:

```
mpg             0
cylinders       0
displacement    0
horsepower      6
weight          0
acceleration    0
model year      0
origin          0
car name        0
dtype: int64
```

In [11]:

```
auto_mpg[auto_mpg.isna().any(axis=1)]
```

Out[11]:

	mpg	cylinders	displacement	horsepower	weight	acceleration	model year	origin	car name
32	25.0	4	98.0	NaN	2046	19.0	71	1	ford pinto
126	21.0	6	200.0	NaN	2875	17.0	74	1	ford maverick
330	40.9	4	85.0	NaN	1835	17.3	80	2	renault lecar deluxe
336	23.6	4	140.0	NaN	2905	14.3	80	1	ford mustang cobra
354	34.5	4	100.0	NaN	2320	15.8	81	2	renault 18i
374	23.0	4	151.0	NaN	3035	20.5	82	1	amc concord dl

we can fill NaN values or drop it

In [12]:

```
df = auto_mpg.dropna(axis=0)
df
```

Out[12]:

	mpg	cylinders	displacement	horsepower	weight	acceleration	model year	origin	car name
0	18.0	8	307.0	130	3504	12.0	70	1	chevrolet chevelle malibu
1	15.0	8	350.0	165	3693	11.5	70	1	buick skylark 320
2	18.0	8	318.0	150	3436	11.0	70	1	plymouth satellite
3	16.0	8	304.0	150	3433	12.0	70	1	amc rebel sst
4	17.0	8	302.0	140	3449	10.5	70	1	ford torino
...	...	...	...	...	...	...	...	...	...
393	27.0	4	140.0	86	2790	15.6	82	1	ford mustang gl
394	44.0	4	97.0	52	2130	24.6	82	2	vw pickup
395	32.0	4	135.0	84	2295	11.6	82	1	dodge rampage
396	28.0	4	120.0	79	2625	18.6	82	1	ford ranger
397	31.0	4	119.0	82	2720	19.4	82	1	chevy s-10

392 rows × 9 columns

In [13]:

```
df.horsepower = df.horsepower.astype('float')
df.dtypes
```

```
C:\Users\91920\anaconda3\lib\site-packages\pandas\core\generic.py:5303: Sett
ingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/
stable/user_guide/indexing.html#returning-a-view-versus-a-copy (https://pand
as.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-v
ersus-a-copy)
  self[name] = value
```

Out[13]:

```
mpg             float64
cylinders         int64
displacement    float64
horsepower      float64
weight            int64
acceleration    float64
model year        int64
origin            int64
car name         object
dtype: object
```

In [14]:

```
df.describe()
```

Out[14]:

	mpg	cylinders	displacement	horsepower	weight	acceleration	model yea
count	392.000000	392.000000	392.000000	392.000000	392.000000	392.000000	392.00000
mean	23.445918	5.471939	194.411990	104.469388	2977.584184	15.541327	75.97959
std	7.805007	1.705783	104.644004	38.491160	849.402560	2.758864	3.68373
min	9.000000	3.000000	68.000000	46.000000	1613.000000	8.000000	70.00000
25%	17.000000	4.000000	105.000000	75.000000	2225.250000	13.775000	73.00000
50%	22.750000	4.000000	151.000000	93.500000	2803.500000	15.500000	76.00000
75%	29.000000	8.000000	275.750000	126.000000	3614.750000	17.025000	79.00000
max	46.600000	8.000000	455.000000	230.000000	5140.000000	24.800000	82.00000

In [15]:

```
df.corr()
```

Out[15]:

	mpg	cylinders	displacement	horsepower	weight	acceleration	mode yea
mpg	1.000000	-0.777618	-0.805127	-0.778427	-0.832244	0.423329	0.58054
cylinders	-0.777618	1.000000	0.950823	0.842983	0.897527	-0.504683	-0.34564
displacement	-0.805127	0.950823	1.000000	0.897257	0.932994	-0.543800	-0.36985
horsepower	-0.778427	0.842983	0.897257	1.000000	0.864538	-0.689196	-0.41636
weight	-0.832244	0.897527	0.932994	0.864538	1.000000	-0.416839	-0.30912
acceleration	0.423329	-0.504683	-0.543800	-0.689196	-0.416839	1.000000	0.29031
model year	0.580541	-0.345647	-0.369855	-0.416361	-0.309120	0.290316	1.00000
origin	0.565209	-0.568932	-0.614535	-0.455171	-0.585005	0.212746	0.18152

Question 1 Find the most economic car company name

In [16]:

```
df[df.mpg==df['mpg'].max()]
```

Out[16]:

	mpg	cylinders	displacement	horsepower	weight	acceleration	model year	origin	car name
322	46.6	4	86.0	65.0	2110	17.9	80	3	mazda glc

Question 2 Print oldest car

In [17]:

```
df['model year'].min()
```

Out[17]:

```
70
```

In [18]:

```
df[df['model year'] == df['model year'].min()]
```

Out[18]:

	mpg	cylinders	displacement	horsepower	weight	acceleration	model year	origin	car name
0	18.0	8	307.0	130.0	3504	12.0	70	1	chevrolet chevelle malibu
1	15.0	8	350.0	165.0	3693	11.5	70	1	buick skylark 320
2	18.0	8	318.0	150.0	3436	11.0	70	1	plymouth satellite
3	16.0	8	304.0	150.0	3433	12.0	70	1	amc rebel sst
4	17.0	8	302.0	140.0	3449	10.5	70	1	ford torino
5	15.0	8	429.0	198.0	4341	10.0	70	1	ford galaxie 500
6	14.0	8	454.0	220.0	4354	9.0	70	1	chevrolet impala
7	14.0	8	440.0	215.0	4312	8.5	70	1	plymouth fury iii
8	14.0	8	455.0	225.0	4425	10.0	70	1	pontiac catalina
9	15.0	8	390.0	190.0	3850	8.5	70	1	amc ambassador dpl
10	15.0	8	383.0	170.0	3563	10.0	70	1	dodge challenger se
11	14.0	8	340.0	160.0	3609	8.0	70	1	plymouth 'cuda 340
12	15.0	8	400.0	150.0	3761	9.5	70	1	chevrolet monte carlo
13	14.0	8	455.0	225.0	3086	10.0	70	1	buick estate wagon (sw)
14	24.0	4	113.0	95.0	2372	15.0	70	3	toyota corona mark ii
15	22.0	6	198.0	95.0	2833	15.5	70	1	plymouth duster
16	18.0	6	199.0	97.0	2774	15.5	70	1	amc hornet
17	21.0	6	200.0	85.0	2587	16.0	70	1	ford maverick
18	27.0	4	97.0	88.0	2130	14.5	70	3	datsun pl510
19	26.0	4	97.0	46.0	1835	20.5	70	2	volkswagen 1131 deluxe sedan
20	25.0	4	110.0	87.0	2672	17.5	70	2	peugeot 504
21	24.0	4	107.0	90.0	2430	14.5	70	2	audi 100 ls

	mpg	cylinders	displacement	horsepower	weight	acceleration	model year	origin	car name
22	25.0	4	104.0	95.0	2375	17.5	70	2	saab 99e
23	26.0	4	121.0	113.0	2234	12.5	70	2	bmw 2002
24	21.0	6	199.0	90.0	2648	15.0	70	1	amc gremlin
25	10.0	8	360.0	215.0	4615	14.0	70	1	ford f250
26	10.0	8	307.0	200.0	4376	15.0	70	1	chevy c20
27	11.0	8	318.0	210.0	4382	13.5	70	1	dodge d200
28	9.0	8	304.0	193.0	4732	18.5	70	1	hi 1200d

In [19]:

```
df.horsepower.mean()
```

Out[19]:

```
104.46938775510205
```

In [20]:

```
df.horsepower.median()
```

Out[20]:

```
93.5
```

In [21]:

```
df.horsepower.min()
```

Out[21]:

```
46.0
```

In [22]:

```
df.horsepower.max()
```

Out[22]:

```
230.0
```

In [53]:

```
df.horsepower.values
```

Out[53]:

```
array([130., 165., 150., 150., 140., 198., 220., 215., 225., 190., 170.,
       160., 150., 225.,  95.,  95.,  97.,  85.,  88.,  46.,  87.,  90.,
        95., 113.,  90., 215., 200., 210., 193.,  88.,  90.,  95., 100.,
       105., 100.,  88., 100., 165., 175., 153., 150., 180., 170., 175.,
       110.,  72., 100.,  88.,  86.,  90.,  70.,  76.,  65.,  69.,  60.,
        70.,  95.,  80.,  54.,  90.,  86., 165., 175., 150., 153., 150.,
       208., 155., 160., 190.,  97., 150., 130., 140., 150., 112.,  76.,
        87.,  69.,  86.,  92.,  97.,  80.,  88., 175., 150., 145., 137.,
       150., 198., 150., 158., 150., 215., 225., 175., 105., 100., 100.,
        88.,  95.,  46., 150., 167., 170., 180., 100.,  88.,  72.,  94.,
        90.,  85., 107.,  90., 145., 230.,  49.,  75.,  91., 112., 150.,
       110., 122., 180.,  95., 100., 100.,  67.,  80.,  65.,  75., 100.,
       110., 105., 140., 150., 150., 140., 150.,  83.,  67.,  78.,  52.,
        61.,  75.,  75.,  75.,  97.,  93.,  67.,  95., 105.,  72.,  72.,
       170., 145., 150., 148., 110., 105., 110.,  95., 110., 110., 129.,
        75.,  83., 100.,  78.,  96.,  71.,  97.,  97.,  70.,  90.,  95.,
        88.,  98., 115.,  53.,  86.,  81.,  92.,  79.,  83., 140., 150.,
       120., 152., 100., 105.,  81.,  90.,  52.,  60.,  70.,  53., 100.,
        78., 110.,  95.,  71.,  70.,  75.,  72., 102., 150.,  88., 108.,
       120., 180., 145., 130., 150.,  68.,  80.,  58.,  96.,  70., 145.,
       110., 145., 130., 110., 105., 100.,  98., 180., 170., 190., 149.,
        78.,  88.,  75.,  89.,  63.,  83.,  67.,  78.,  97., 110., 110.,
        48.,  66.,  52.,  70.,  60., 110., 140., 139., 105.,  95.,  85.,
        88., 100.,  90., 105.,  85., 110., 120., 145., 165., 139., 140.,
        68.,  95.,  97.,  75.,  95., 105.,  85.,  97., 103., 125., 115.,
       133.,  71.,  68., 115.,  85.,  88.,  90., 110., 130., 129., 138.,
       135., 155., 142., 125., 150.,  71.,  65.,  80.,  80.,  77., 125.,
        71.,  90.,  70.,  70.,  65.,  69.,  90., 115., 115.,  90.,  76.,
        60.,  70.,  65.,  90.,  88.,  90.,  90.,  78.,  90.,  75.,  92.,
        75.,  65., 105.,  65.,  48.,  48.,  67.,  67.,  67.,  67.,  62.,
       132., 100.,  88.,  72.,  84.,  84.,  92., 110.,  84.,  58.,  64.,
        60.,  67.,  65.,  62.,  68.,  63.,  65.,  65.,  74.,  75.,  75.,
       100.,  74.,  80.,  76., 116., 120., 110., 105.,  88.,  85.,  88.,
        88.,  88.,  85.,  84.,  90.,  92.,  74.,  68.,  68.,  63.,  70.,
        88.,  75.,  70.,  67.,  67.,  67., 110.,  85.,  92., 112.,  96.,
        84.,  90.,  86.,  52.,  84.,  79.,  82.])
```

In [54]:

```
pd.set_option('display.max_rows', None)
df.horsepower.value_counts()
```

Out[54]:

```
150.0    22
90.0     20
88.0     19
110.0    18
100.0    17
95.0     14
75.0     14
67.0     12
105.0    12
70.0     12
65.0     10
85.0      9
97.0      9
145.0     7
140.0     7
80.0      7
68.0      6
72.0      6
84.0      6
78.0      6
92.0      6
175.0     5
115.0     5
180.0     5
60.0      5
86.0      5
130.0     5
71.0      5
170.0     5
165.0     4
83.0      4
52.0      4
120.0     4
76.0      4
96.0      3
69.0      3
190.0     3
74.0      3
63.0      3
112.0     3
215.0     3
48.0      3
225.0     3
125.0     3
62.0      2
198.0     2
155.0     2
153.0     2
81.0      2
139.0     2
53.0      2
160.0     2
58.0      2
87.0      2
```

```
46.0      2
129.0     2
98.0      2
79.0      2
210.0     1
200.0     1
113.0     1
220.0     1
193.0     1
82.0      1
54.0      1
102.0     1
64.0      1
132.0     1
77.0      1
142.0     1
135.0     1
138.0     1
133.0     1
103.0     1
66.0      1
89.0      1
149.0     1
108.0     1
152.0     1
208.0     1
148.0     1
93.0      1
61.0      1
122.0     1
91.0      1
49.0      1
230.0     1
116.0     1
94.0      1
167.0     1
158.0     1
137.0     1
107.0     1
Name: horsepower, dtype: int64
```

In [52]:

```
df.horsepower.describe()
```

Out[52]:

```
count    392.000000
mean     104.469388
std       38.491160
min       46.000000
25%       75.000000
50%       93.500000
75%      126.000000
max      230.000000
Name: horsepower, dtype: float64
```

In [ ]:

```
```