In [114…

```
import pandas as pd
from matplotlib import pyplot as plt
%matplotlib inline
import seaborn as sns
from scipy import stats


import warnings
warnings.filterwarnings('ignore')
```

Q7 Answer

In [115…

```
cars_data = pd.read_csv('Q7.csv')
cars_data
```

Out[115…

	Unnamed: 0	Points	Score	Weigh
0	Mazda RX4	3.90	2.620	16.46
1	Mazda RX4 Wag	3.90	2.875	17.02
2	Datsun 710	3.85	2.320	18.61
3	Hornet 4 Drive	3.08	3.215	19.44
4	Hornet Sportabout	3.15	3.440	17.02
5	Valiant	2.76	3.460	20.22
6	Duster 360	3.21	3.570	15.84
7	Merc 240D	3.69	3.190	20.00
8	Merc 230	3.92	3.150	22.90
9	Merc 280	3.92	3.440	18.30
10	Merc 280C	3.92	3.440	18.90
11	Merc 450SE	3.07	4.070	17.40
12	Merc 450SL	3.07	3.730	17.60
13	Merc 450SLC	3.07	3.780	18.00
14	Cadillac Fleetwood	2.93	5.250	17.98
15	Lincoln Continental	3.00	5.424	17.82
16	Chrysler Imperial	3.23	5.345	17.42
17	Fiat 128	4.08	2.200	19.47
18	Honda Civic	4.93	1.615	18.52
19	Toyota Corolla	4.22	1.835	19.90
20	Toyota Corona	3.70	2.465	20.01
21	Dodge Challenger	2.76	3.520	16.87

	Unnamed: 0	Points	Score	Weigh
22	AMC Javelin	3.15	3.435	17.30
23	Camaro Z28	3.73	3.840	15.41
24	Pontiac Firebird	3.08	3.845	17.05
25	Fiat X1-9	4.08	1.935	18.90
26	Porsche 914-2	4.43	2.140	16.70
27	Lotus Europa	3.77	1.513	16.90
28	Ford Pantera L	4.22	3.170	14.50
29	Ferrari Dino	3.62	2.770	15.50
30	Maserati Bora	3.54	3.570	14.60
31	Volvo 142E	4.11	2.780	18.60

Mean

In [3]:
```
cars_data.mean()
```

Out[3]:
```
Points     3.596563
Score      3.217250
Weigh     17.848750
dtype: float64
```

Medain

In [4]:
```
cars_data.median()
```

Out[4]:
```
Points     3.695
Score      3.325
Weigh     17.710
dtype: float64
```

Mode

In [6]:
```
cars_data["Points"].mode()
```

Out[6]:
```
0    3.07
1    3.92
dtype: float64
```

In [7]:
```
cars_data["Score"].mode()
```

Out[7]:
```
0    3.44
dtype: float64
```

In [8]:
```
cars_data["Weigh"].mode()
```

Out[8]:
```
0    17.02
1    18.90
dtype: float64
```

Variance

In [9]:
```
cars_data.var()
```

Out[9]:
```
Points    0.285881
Score     0.957379
Weigh     3.193166
dtype: float64
```

Standard deviation

In [10]:
```
cars_data.std()
```

Out[10]:
```
Points    0.534679
Score     0.978457
Weigh     1.786943
dtype: float64
```

Range

In [11]:
```
cars_data.describe()
```

Out[11]:

	Points	Score	Weigh
count	32.000000	32.000000	32.000000
mean	3.596563	3.217250	17.848750
std	0.534679	0.978457	1.786943
min	2.760000	1.513000	14.500000
25%	3.080000	2.581250	16.892500
50%	3.695000	3.325000	17.710000
75%	3.920000	3.610000	18.900000
max	4.930000	5.424000	22.900000

In [12]:
```
points_range = cars_data['Points'].max() - cars_data['Points'].min()
points_range
```

Out[12]:
```
2.17
```

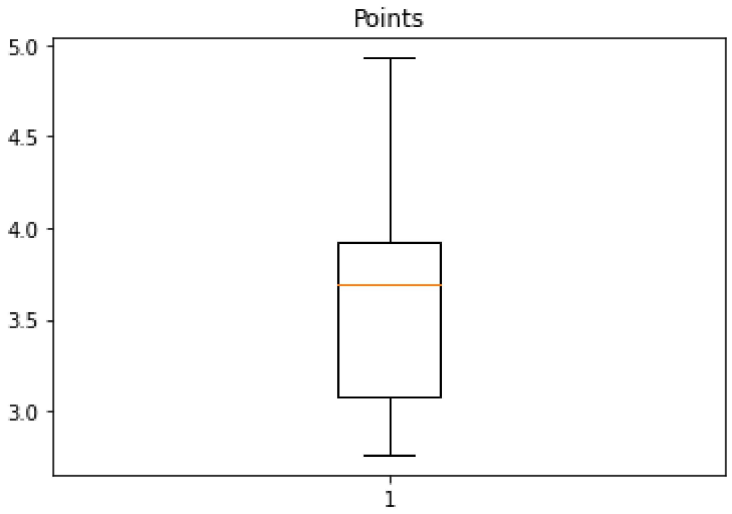
In [14]:
```
score_range = cars_data['Score'].max() - cars_data['Score'].min()
score_range
```

Out[14]:
```
3.9110000000000005
```

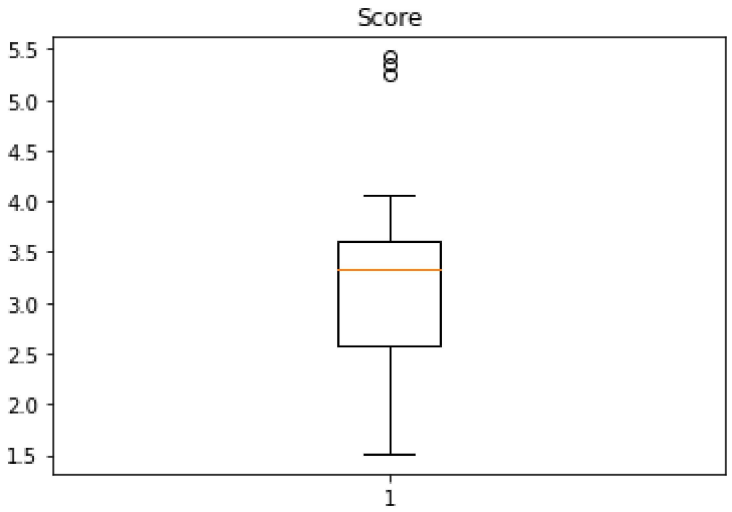
In [16]:
```
weigh_range = cars_data['Weigh'].max() - cars_data['Weigh'].min()
weigh_range
```

Out[16]:
```
8.399999999999999
```

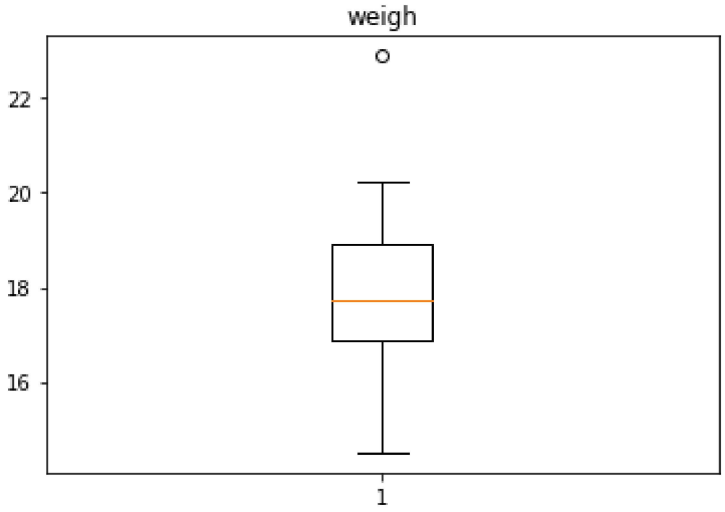
In [20]:
```
plt.boxplot(cars_data['Points'])
plt.title('Points')
plt.show()
```



In [21]:
```
plt.boxplot(cars_data['Score'])
plt.title('Score')
plt.show()
```



In [22]:
```
plt.boxplot(cars_data['Weigh'])
plt.title('weigh')
plt.show()
```



Q8 Answer

In [26]:
```
x = [108,110,123,134,135,145,167,187,199]
x = pd.DataFrame(x)
x.mean()
```

Out[26]:
```
0    145.333333
dtype: float64
```

Q9 Answer

a)

In [37]:
```
dt = pd.read_csv( 'Q9_a.csv')
dt
```

Out[37]:

	Index	speed	dist
0	1	4	2
1	2	4	10
2	3	7	4
3	4	7	22
4	5	8	16
5	6	9	10
6	7	10	18
7	8	10	26
8	9	10	34
9	10	11	17
10	11	11	28
11	12	12	14

	Index	speed	dist
12	13	12	20
13	14	12	24
14	15	12	28
15	16	13	26
16	17	13	34
17	18	13	34
18	19	13	46
19	20	14	26
20	21	14	36
21	22	14	60
22	23	14	80
23	24	15	20
24	25	15	26
25	26	15	54
26	27	16	32
27	28	16	40
28	29	17	32
29	30	17	40
30	31	17	50
31	32	18	42
32	33	18	56
33	34	18	76
34	35	18	84
35	36	19	36
36	37	19	46
37	38	19	68
38	39	20	32
39	40	20	48
40	41	20	52
41	42	20	56
42	43	20	64
43	44	22	66
44	45	23	54

	Index	speed	dist
45	46	24	70
46	47	24	92
47	48	24	93
48	49	24	120
49	50	25	85

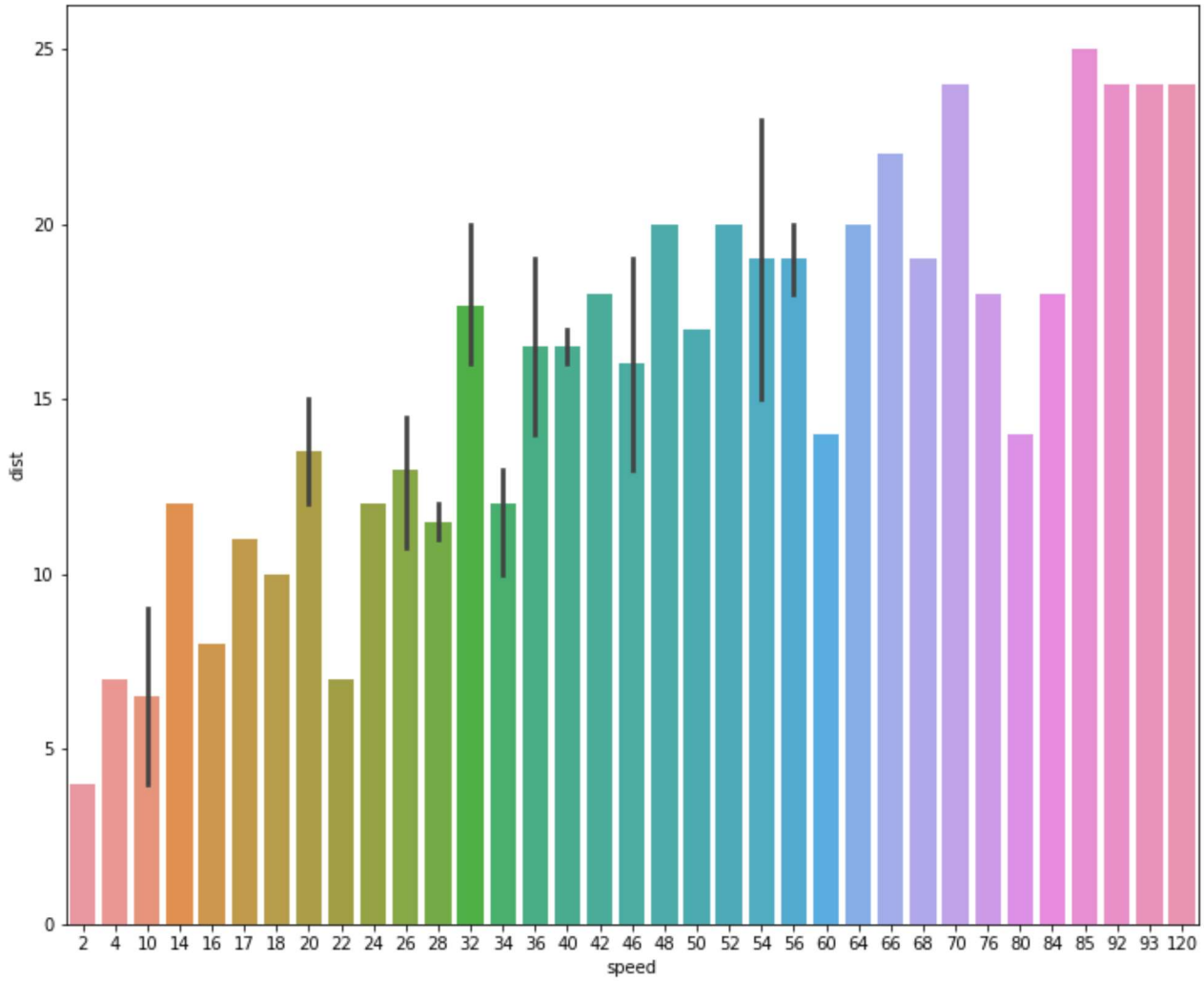
In [38]:
```
dt.skew()
```

Out[38]:
```
Index    0.000000
speed   -0.117510
dist     0.806895
dtype: float64
```

In [39]:
```
dt.kurtosis()
```

Out[39]:
```
Index   -1.200000
speed   -0.508994
dist     0.405053
dtype: float64
```

In [59]:
```
plt.figure(figsize=(12,10))
sns.barplot(x = 'dist',y = 'speed',data=dt)
plt.xlabel('speed')
plt.ylabel('dist')
plt.show()
```



b)

In [85]:
```
sp = pd.read_csv('Q9_b.csv')
sp.round(2)
```

Out[85]:

	Unnamed: 0	SP	WT
0	1	104.19	28.76
1	2	105.46	30.47
2	3	105.46	30.19
3	4	113.46	30.63
4	5	104.46	29.89
...	...	...	...
76	77	169.60	16.13
77	78	150.58	37.92
78	79	151.60	15.77
79	80	167.94	39.42
80	81	139.84	34.95

81 rows × 3 columns

In [86]:
```
del sp['Unnamed: 0']
sp.head()
```

Out[86]:

	SP	WT
0	104.185353	28.762059
1	105.461264	30.466833
2	105.461264	30.193597
3	113.461264	30.632114
4	104.461264	29.889149

In [87]:
```
sp.skew()
```

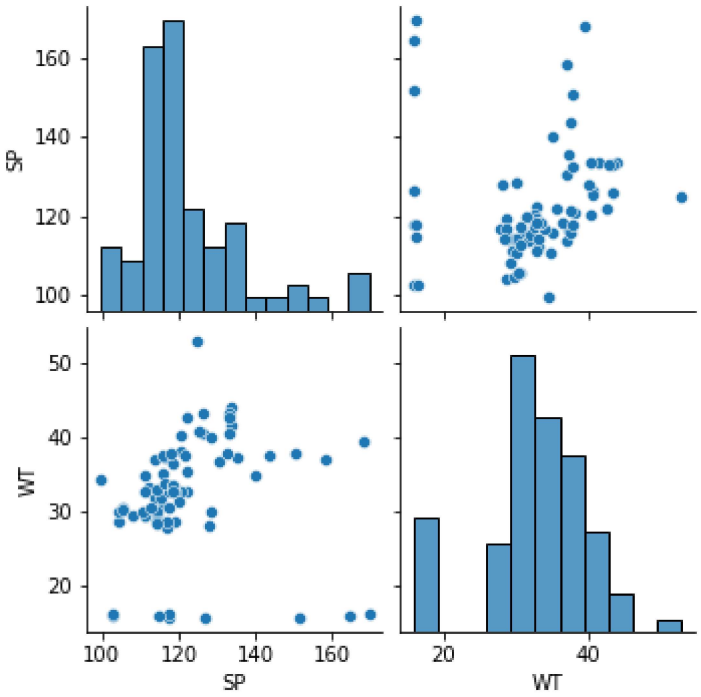
Out[87]:
```
SP    1.611450
WT   -0.614753
dtype: float64
```

In [88]:
```
sp.kurtosis()
```

Out[88]:
```
SP    2.977329
WT    0.950291
dtype: float64
```

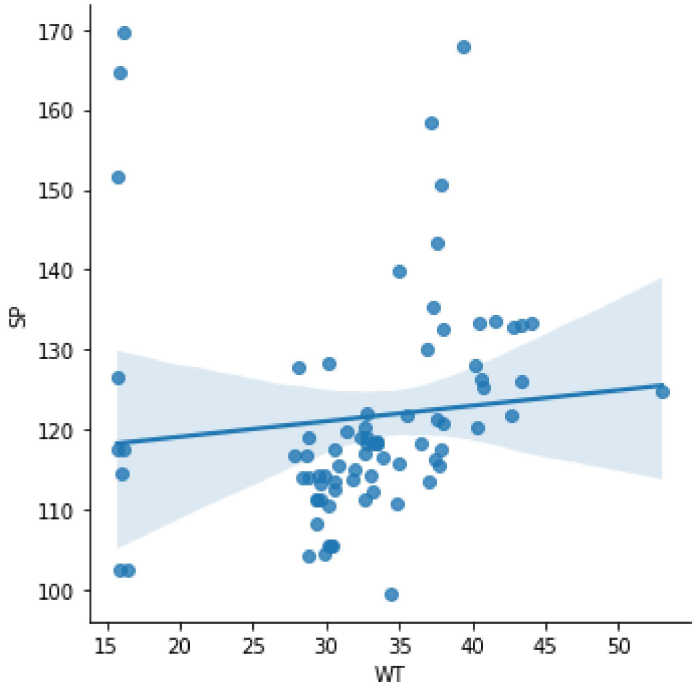
In [91]:
```
plt.figure(figsize=(15,10))
sns.pairplot(sp)
plt.show()
```

```
<Figure size 1080x720 with 0 Axes>
```



In [99]:
```
sns.lmplot(x = 'WT',y = 'SP',data=sp)
```

Out[99]:
```
<seaborn.axisgrid.FacetGrid at 0x1fbdb7f89a0>
```



Q11 Answer

94% of confidence interval

In [104…
```
stats.norm.interval(0.94,200,30/(2000**0.5))
```

Out[104…
```
(198.738325292158, 201.261674707842)
```

98% of confidence interval

In [105…
```
stats.norm.interval(0.98,200,30/(2000**0.5))
```

Out[105…
```
(198.43943840429978, 201.56056159570022)
```

96% of cofidence interval

In [106…
```
stats.norm.interval(0.96,200,30/(2000**0.5))
```

Out[106…
```
(198.62230334813333, 201.37769665186667)
```

Q12 Answer

1)

In [118…
```
scores = [34,36,36,38,38,39,39,40,40,41,41,41,41,42,42,45,49,56]
scores = pd.DataFrame(scores)
```

In [120…
```
## MEAN
scores.mean()
```

Out[120…
```
0    41.0
dtype: float64
```

In [121…
```
## MEDIAN
scores.median()
```

Out[121…
```
0    40.5
dtype: float64
```

In [122…
```
## MODE
scores.mode()
```

Out[122…

	0
0	41

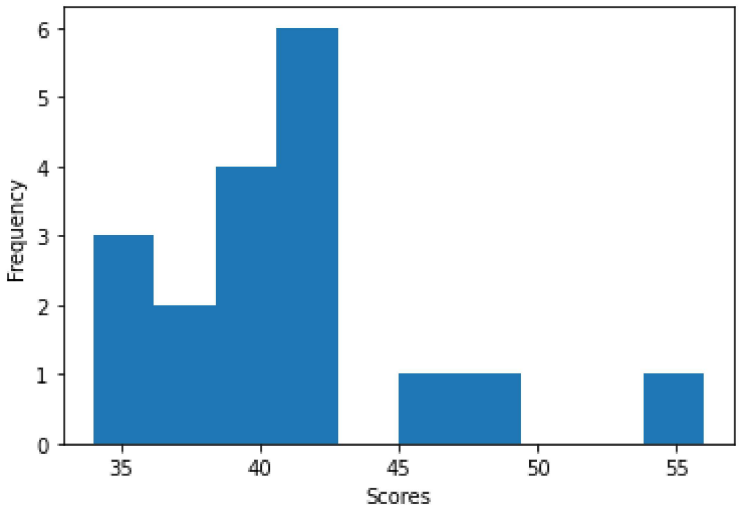
In [123…
```
## VARIANCE
scores.var()
```

Out[123…
```
0    25.529412
dtype: float64
```

In [124…
```
## STANDARD DEVIATION
scores.std()
```

Out[124…
```
0    5.052664
dtype: float64
```

In [126…
```
plt.hist(scores)
plt.xlabel('Scores')
plt.ylabel('Frequency')
plt.show()
```



Q20 Answer

In [129…
```
car_data = pd.read_csv('Cars.csv')
car_data
```

Out[129…

	HP	MPG	VOL	SP	WT
0	49	53.700681	89	104.185353	28.762059
1	55	50.013401	92	105.461264	30.466833
2	55	50.013401	92	105.461264	30.193597
3	70	45.696322	92	113.461264	30.632114
4	53	50.504232	92	104.461264	29.889149
…	…	…	…	…	…
76	322	36.900000	50	169.598513	16.132947
77	238	19.197888	115	150.576579	37.923113
78	263	34.000000	50	151.598513	15.769625
79	295	19.833733	119	167.944460	39.423099
80	236	12.101263	107	139.840817	34.948615

81 rows × 5 columns

In [131…
```
## Mean
round(car_data.MPG.mean(),4)
```

Out[131… 34.4221

In [133…
```
## STD
round(car_data.MPG.std(),4)
```

Out[133… 9.1314

In [134…
```
## P(MPG>38)
1-stats.norm.cdf(x=38,loc=34.42,scale=9.13)
```

Out[134… 0.34748702501304063

In [135…
```
## P(MPG<40)
stats.norm.cdf(40,loc=34.42,scale=9.13)
```

Out[135… 0.7294571279557076

In [136…
```
## P(20<MPG<50)
stats.norm.cdf(50,loc=34.42,scale=9.13) - stats.norm.cdf(20,loc=34.42,scale=9.13)
```

Out[136… 0.8989177824549222

Q21 Answer

a)

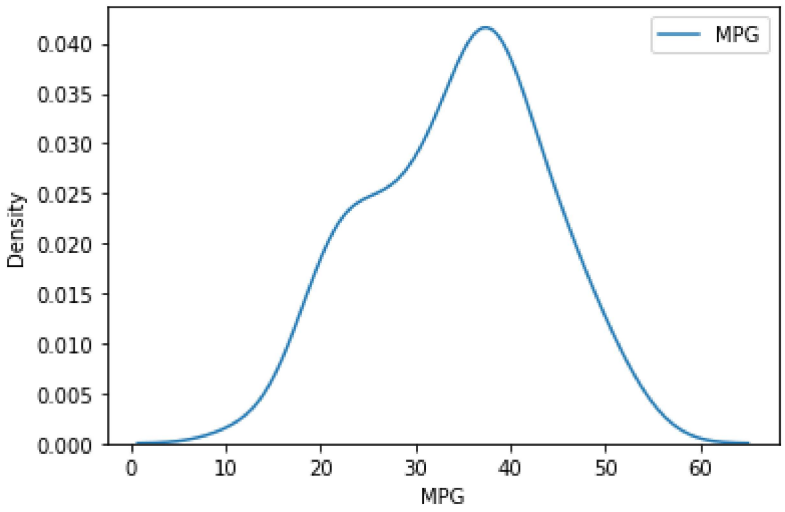
In [137…
```
cd = pd.read_csv('Cars.csv')
cd
```

Out[137…

	HP	MPG	VOL	SP	WT
0	49	53.700681	89	104.185353	28.762059
1	55	50.013401	92	105.461264	30.466833
2	55	50.013401	92	105.461264	30.193597
3	70	45.696322	92	113.461264	30.632114
4	53	50.504232	92	104.461264	29.889149
…	…	…	…	…	…
76	322	36.900000	50	169.598513	16.132947
77	238	19.197888	115	150.576579	37.923113
78	263	34.000000	50	151.598513	15.769625
79	295	19.833733	119	167.944460	39.423099
80	236	12.101263	107	139.840817	34.948615

81 rows × 5 columns

In [152…
```
sns.distplot(cd.MPG,label = 'MPG',hist=False)
plt.legend()
plt.show()
```



In [142…
```
cd.MPG.mean()
```

Out[142...
```
34.422075728024666
```

In [143...
```
cd.MPG.median()
```

Out[143...
```
35.15272697
```

b)

In [144...
```
wc_at = pd.read_csv('wc-at.csv')
```

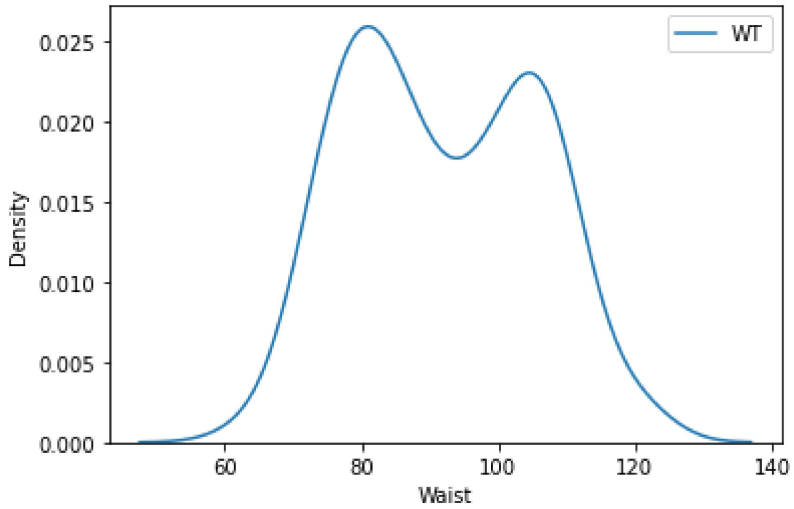
In [145...
```
wc_at
```

Out[145...

	Waist	AT
0	74.75	25.72
1	72.60	25.89
2	81.80	42.60
3	83.95	42.80
4	74.65	29.84
...	...	...
104	100.10	124.00
105	93.30	62.20
106	101.80	133.00
107	107.90	208.00
108	108.50	208.00

109 rows × 2 columns

In [151...
```
sns.distplot(wc_at.Waist,label=('WT'),hist=False)
plt.legend()
plt.show()
```



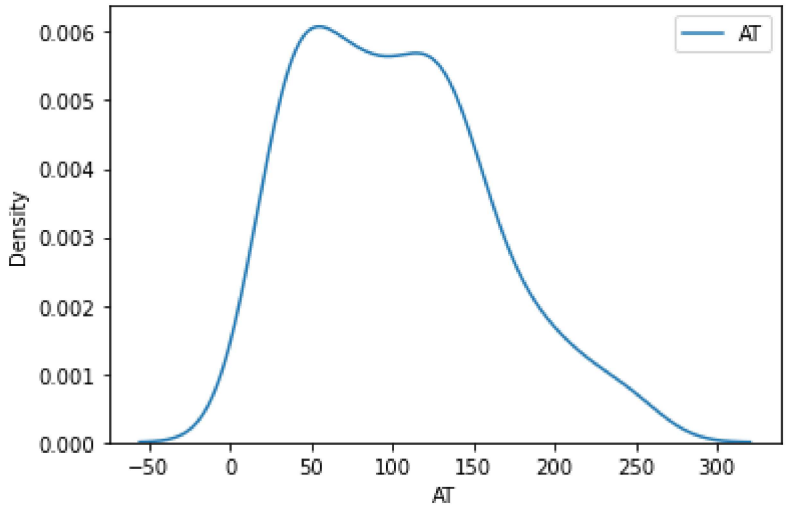
In [147…
```
wc_at.Waist.mean()
```

Out[147…
```
91.90183486238533
```

In [148…
```
wc_at.Waist.median()
```

Out[148…
```
90.8
```

In [155…
```
sns.distplot(wc_at.AT,label='AT',hist=False)
plt.legend()
plt.show()
```



In [156…
```
wc_at.AT.mean()
```

Out[156…
```
101.89403669724771
```

In [157…
```
wc_at.AT.median()
```

Out[157…
```
96.54
```

Q22 Answer

In [159…
```
#zscore of 90% confidence interval
stats.norm.ppf(0.90)
```

Out[159…
```
1.2815515655446004
```

In [160…
```
#zscore of 94% confidence interval
stats.norm.ppf(0.94)
```

Out[160…
```
1.5547735945968535
```

In [162…
```
#zscore of 60% confidence interval
stats.norm.ppf(0.60)
```

Out[162…
```
0.2533471031357997
```

Q23 Answer

In [166…
```
# t score of 95% confidence intervel
stats.t.ppf(0.975,24)
```

Out[166…
```
2.0638985616280205
```

In [167…
```
# t score of 96% confidence intervel
stats.t.ppf(0.98,24)
```

Out[167…
```
2.1715446760080677
```

In [169…
```
# t score of 99% confidence intervel
stats.t.ppf(0.995,24)
```

Out[169…
```
2.796939504772804
```

Q24 Answer

In [173…
```
# t scores at x=260
t = (260-270)/(90/18**0.5)
print('t-score is',t)
```

```
t-score is -0.4714045207910317
```

In [174…
```
# p-value
p = 1-stats.t.cdf(abs(-0.4714),df=17)
print('p-value is',p)
```

```
p-value is 0.32167411684460556
```