Qus.No.7

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

In [2]:
```
Que_data=pd.read_csv('Q7.csv')
Que_data
```

Out[2]:

	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
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

In [3]:
```
Que_data.shape
```

Out[3]: (32, 4)

In [4]:
```
1 Que_data.describe(include='all')
```

Out[4]:

	Unnamed: 0	Points	Score	Weigh
count	32	32.000000	32.000000	32.000000
unique	32	NaN	NaN	NaN
top	Mazda RX4	NaN	NaN	NaN
freq	1	NaN	NaN	NaN
mean	NaN	3.596563	3.217250	17.848750
std	NaN	0.534679	0.978457	1.786943
min	NaN	2.760000	1.513000	14.500000
25%	NaN	3.080000	2.581250	16.892500
50%	NaN	3.695000	3.325000	17.710000
75%	NaN	3.920000	3.610000	18.900000
max	NaN	4.930000	5.424000	22.900000

In [5]:
```
1 import warnings
2 warnings.filterwarnings('ignore')
3
4 Que_data.var()
```

Out[5]:
```
Points    0.285881
Score     0.957379
Weigh     3.193166
dtype: float64
```

In [6]:
```
1 Que_data.mean()
```

Out[6]:
```
Points     3.596563
Score      3.217250
Weigh     17.848750
dtype: float64
```

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

Out[7]:
```
Points     3.695
Score      3.325
Weigh     17.710
dtype: float64
```

In [8]:
```
1 Que_data.mode()
```

Out[8]:

	Unnamed: 0	Points	Score	Weigh
0	AMC Javelin	3.07	3.44	17.02
1	Cadillac Fleetwood	3.92	NaN	18.90
2	Camaro Z28	NaN	NaN	NaN
3	Chrysler Imperial	NaN	NaN	NaN
4	Datsun 710	NaN	NaN	NaN
5	Dodge Challenger	NaN	NaN	NaN
6	Duster 360	NaN	NaN	NaN
7	Ferrari Dino	NaN	NaN	NaN
8	Fiat 128	NaN	NaN	NaN
9	Fiat X1-9	NaN	NaN	NaN
10	Ford Pantera L	NaN	NaN	NaN
11	Honda Civic	NaN	NaN	NaN
12	Hornet 4 Drive	NaN	NaN	NaN
13	Hornet Sportabout	NaN	NaN	NaN
14	Lincoln Continental	NaN	NaN	NaN
15	Lotus Europa	NaN	NaN	NaN
16	Maserati Bora	NaN	NaN	NaN
17	Mazda RX4	NaN	NaN	NaN
18	Mazda RX4 Wag	NaN	NaN	NaN
19	Merc 230	NaN	NaN	NaN
20	Merc 240D	NaN	NaN	NaN
21	Merc 280	NaN	NaN	NaN
22	Merc 280C	NaN	NaN	NaN
23	Merc 450SE	NaN	NaN	NaN
24	Merc 450SL	NaN	NaN	NaN
25	Merc 450SLC	NaN	NaN	NaN
26	Pontiac Firebird	NaN	NaN	NaN
27	Porsche 914-2	NaN	NaN	NaN
28	Toyota Corolla	NaN	NaN	NaN
29	Toyota Corona	NaN	NaN	NaN
30	Valiant	NaN	NaN	NaN
31	Volvo 142E	NaN	NaN	NaN

In [9]:
```
1 Que_data.std()
```

Out[9]:
```
Points    0.534679
Score     0.978457
Weigh     1.786943
dtype: float64
```

Qus.No.9

In [10]:
```
cars_data=pd.read_csv('Q7.csv')
cars_data
```

Out[10]:

	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
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

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

Out[11]:

	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

In [12]:
```
cars_data.skew()
```

Out[12]:
```
Points    0.292780
Score     0.465916
Weigh     0.406347
dtype: float64
```

In [13]:
```
cars_data.kurtosis()
```

Out[13]:
```
Points   -0.450432
Score     0.416595
Weigh     0.864931
dtype: float64
```

In [14]:
```
cars_data_2=pd.read_csv('Cars.csv')
cars_data_2
```

Out[14]:

	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 [15]:
```
cars_data_2.head()
```

Out[15]:

	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

In [16]:
```
cars_data_2.skew()
```

Out[16]:
```
HP     1.716216
MPG   -0.177947
VOL   -0.590197
SP     1.611450
WT    -0.614753
dtype: float64
```

In [17]:
```
cars_data_2.kurtosis()
```

Out[17]:
```
HP     2.960025
MPG   -0.611679
VOL    0.920229
SP     2.977329
WT     0.950291
dtype: float64
```

Qusno.11

In [18]:
```
from scipy import stats
import numpy as np
conf_94=stats.t.interval(alpha= 0.97,df= 1999,loc= 200,scale= 30/np.sqrt(2000))
print(np.round(conf_94,0))
print(conf_94)
```

```
[199. 201.]
(198.54321897505338, 201.45678102494662)
```

In [19]:
```
conf_98=stats.t.interval(alpha= 0.98,df= 1999,loc= 200,scale= 30/np.sqrt(2000))
print(np.round(conf_98,0))
print(conf_98)
```

```
[198. 202.]
(198.4381860483216, 201.5618139516784)
```

In [20]:
```
conf_96=stats.t.interval(alpha= 0.96,df= 1999,loc= 200,scale= 30/np.sqrt(2000))
print(np.round(conf_96,0))
print(conf_96)
```

```
[199. 201.]
(198.6214037429732, 201.3785962570268)
```

In [21]:
```
conf_z_94=stats.norm.interval(0.94,loc =200,scale =30/np.sqrt(2000))
np.round(conf_z_94,0)
```

Out[21]:
```
array([199., 201.])
```

In [22]:
```
conf_z_96=stats.norm.interval(0.96,loc =200,scale =30/np.sqrt(2000))
np.round(conf_z_96,0)
```

Out[22]:
```
array([199., 201.])
```

In [23]:
```
conf_z_98=stats.norm.interval(0.98,loc =200,scale =30/np.sqrt(2000))
np.round(conf_z_98,0)
```

Out[23]:
```
array([198., 202.])
```

Qus.no.20

In [24]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from scipy import stats
from scipy.stats import norm
```

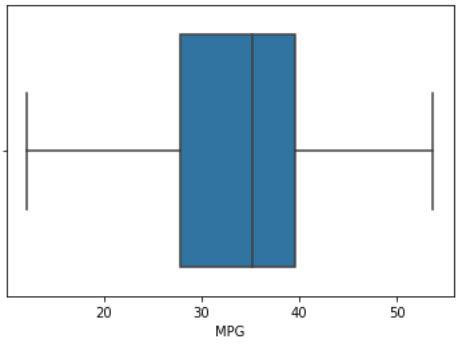
In [25]:
```
cars= pd.read_csv('Cars.csv')
cars
```

Out[25]:

	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

In [26]:
```
sns.boxplot(cars.MPG)
```

Out[26]:
```
<AxesSubplot:xlabel='MPG'>
```



In [27]:
```
# P(MPG>38)
1-stats.norm.cdf(38,cars.MPG.mean(),cars.MPG.std())
```

Out[27]:
```
0.3475939251582705
```

In [28]:
```
#P(MPG<40)
stats.norm.cdf(40,cars.MPG.mean(),cars.MPG.std())
```

Out[28]:
```
0.7293498762151616
```

In [29]:
```
#P(20<MPG<50)
stats.norm.cdf(0.50,cars.MPG.mean(),cars.MPG.std())-stats.norm.cdf(0.20,cars.MPG.mean(),cars.MPG.std())
```

Out[29]: 1.2430968797327613e-05

Que.no.21 (a)

In [30]:
```
import pandas as pd
import numpy as np
import matplotlib as mpl
import matplotlib.pyplot as plt
import seaborn as sns
from scipy import stats
from scipy.stats import norm
```

In [31]:
```
cars= pd.read_csv('Cars.csv')
cars.head()
```

Out[31]:

	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

In [32]:
```
cars['MPG'].mean()
```

Out[32]: 34.422075728024666

In [33]:
```
cars['MPG'].median()
```

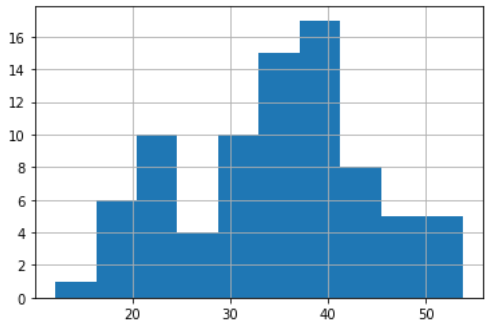
Out[33]: 35.15272697

In [34]:
```
cars['MPG'].mode()
```

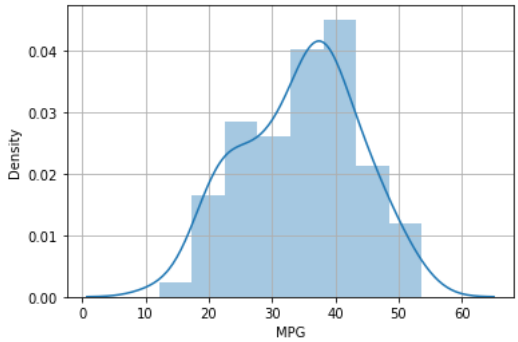
Out[34]:
```
0    29.629936
dtype: float64
```

In [35]:
```
cars['MPG'].hist()
```

Out[35]: <AxesSubplot:>



In [36]:
```
import warnings
warnings.filterwarnings("ignore")
sns.distplot(cars['MPG'])
plt.grid(True)
plt.show()
```



In [37]:
```
cars['MPG'].skew()
```

Out[37]: -0.17794674747025727

In [38]:
```
cars['MPG'].kurt()
```

Out[38]: -0.6116786559430913

Qus.No.21 (b)

In [39]:
```
df=pd.read_csv('wc-at.csv')
df.head()
```

Out[39]:

	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

In [40]:
```
df.mean()
```

Out[40]:
```
Waist     91.901835
AT       101.894037
dtype: float64
```

In [41]:
```
df.median()
```

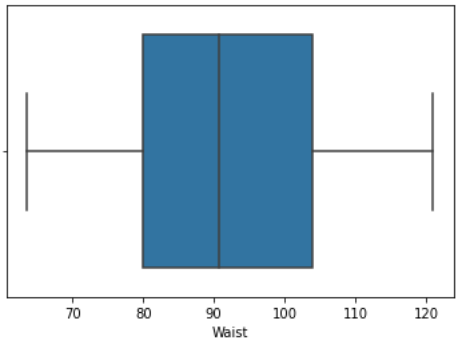
Out[41]:
```
Waist    90.80
AT       96.54
dtype: float64
```

In [42]:
```
df.mode()
```

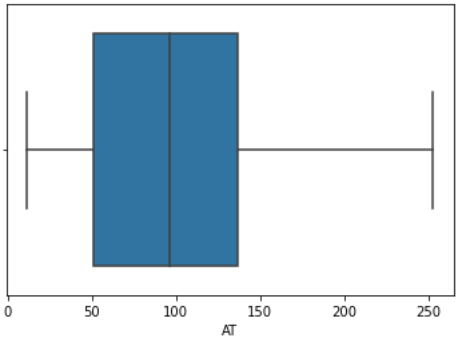
Out[42]:

	Waist	AT
0	94.5	121.0
1	106.0	123.0
2	108.5	NaN

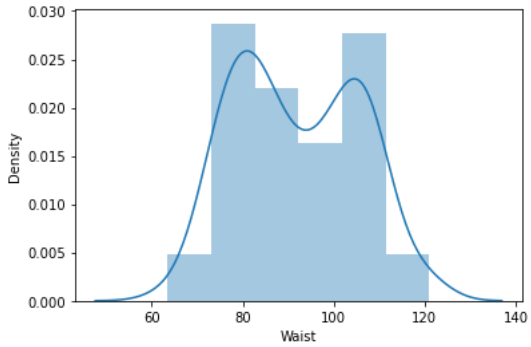
In [43]:
```
sns.boxplot(df['Waist'])
plt.show()
```



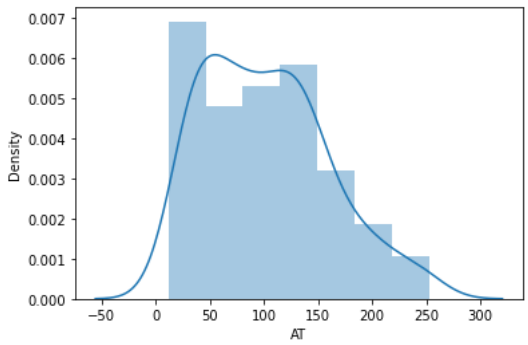
In [44]:
```
sns.boxplot(df['AT'])
plt.show()
```



In [45]:
```
import warnings
warnings.filterwarnings("ignore")
sns.distplot(df['Waist'])
plt.show()
```



In [46]:
```
import warnings
warnings.filterwarnings("ignore")
sns.distplot(df['AT'])
plt.show()
```



Qus.no.22

In [47]:
```
from scipy import stats
from scipy.stats import norm
```

In [48]:
```
# z-score of 90% confidence interval
stats.norm.ppf(0.95)
```

Out[48]: 1.6448536269514722

In [49]:
```
# z-score of 94% confidence interval
stats.norm.ppf(0.97)
```

Out[49]: 1.8807936081512509

In [50]:
```
# z-score of 60% confidence interval
stats.norm.ppf(0.8)
```

Out[50]: 0.8416212335729143

Qus.no. 23

In [51]:
```
from scipy import stats
from scipy.stats import norm
```

In [52]:
```
# t scores of 95% confidence interval for sample size of 25
stats.t.ppf(0.975,24)  # df = n-1 =24
```

Out[52]: 2.0638985616280205

In [53]:
```
# t scores of 96% confidence interval for sample size of 25
stats.t.ppf(0.98,24)
```

Out[53]: 2.1715446760080677

In [54]:
```
# t scores of 99% confidence interval for sample size of 25
stats.t.ppf(0.995,24)  # df = n-1 =24
```

Out[54]: 2.796939504772804

Qus.no.24

In [55]:
```
from scipy import stats
from scipy.stats import norm
```

In [56]:
```
# Assume Null Hypothesis is: Ho = Avg life of Bulb >= 260 days
# Alternate Hypothesis is: Ha = Avg life of Bulb <= 260 days
```

In [57]:
```
# Find t-scores at x=260; t=(s_mean-p_mean)/(s_SD)
t=(260-270)/(90/18**0.5)
t
```

Out[57]: -0.4714045207910317

In [58]:
```
# Find P(X>=260) for null hypothesis
```

In [59]:
```
#p_value= 1-stats.t.cdf(abs(t-scores),df = n-1).....using cdf function
p_value = 1-stats.t.cdf(abs(-0.4714),df=17)
p_value
```

Out[59]: 0.32167411684460556

In [60]:
```
# OR p_value= 1-stats.t.cdf(abs(t-scores),df = n-1).....using cdf function
p_value = 1-stats.t.cdf(abs(-0.4714),df=17)
p_value
```

Out[60]: 0.32167411684460556

In [ ]:
```

```