Q. 7 Answer:

In [45]:
```
import pandas as pd
import numpy as np
import seaborn as sns
from matplotlib import pyplot as plt
from scipy import stats
from scipy.stats import norm
```

In [3]:
```
vehicle_details = pd.read_csv('Q7.csv')
vehicle_details
vehicle_details.head()
import warnings
warnings.filterwarnings('ignore')
```

In [4]:
```
vehicle_details.mean() # Mean
```

Out[4]:
```
Points     3.596563
Score      3.217250
Weigh     17.848750
dtype: float64
```

In [5]:
```
vehicle_details.median() # Median
```

Out[5]:
```
Points     3.695
Score      3.325
Weigh     17.710
dtype: float64
```

In [70]:
```
vehicle_details['Points'].mode(),vehicle_details['Score'].mode(),vehicle_details
```

Out[70]:
```
(0    3.07
 1    3.92
 dtype: float64,
 0    3.44
 dtype: float64,
 0    17.02
 1    18.90
 dtype: float64)
```

In [6]:
```
vehicle_details.describe() # Std. Deviation
```

Out[6]:

	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

Q.9a Answer

In [7]:
```
cars_details = pd.read_csv('Q9_a.csv')
cars_details.head()
```

Out[7]:

	Index	speed	dist
0	1	4	2
1	2	4	10
2	3	7	4
3	4	7	22
4	5	8	16

In [8]:
```
cars_details.dtypes
```

Out[8]:
```
Index    int64
speed    int64
dist     int64
dtype: object
```

In [9]:
```
cars_details.median()
```

Out[9]:
```
Index    25.5
speed    15.0
dist     36.0
dtype: float64
```

In [10]:
```
cars_details.describe()
```

Out[10]:

	Index	speed	dist
count	50.00000	50.000000	50.000000
mean	25.50000	15.400000	42.980000
std	14.57738	5.287644	25.769377
min	1.00000	4.000000	2.000000
25%	13.25000	12.000000	26.000000
50%	25.50000	15.000000	36.000000
75%	37.75000	19.000000	56.000000
max	50.00000	25.000000	120.000000

In [11]:
```
cars_details['speed'].skew()
```

Out[11]: -0.11750986144663393

In [ ]:
```
#This is Negative skew,means more lack of symmetry towards left hand side. It is
```

In [12]:
```
cars_details['dist'].skew()
```

Out[12]: 0.8068949601674215

In [ ]:
```
#This is Positive skew,means more lack of symmetry towards right hand side. It is
```

In [13]:
```
cars_details['speed'].kurtosis()
```

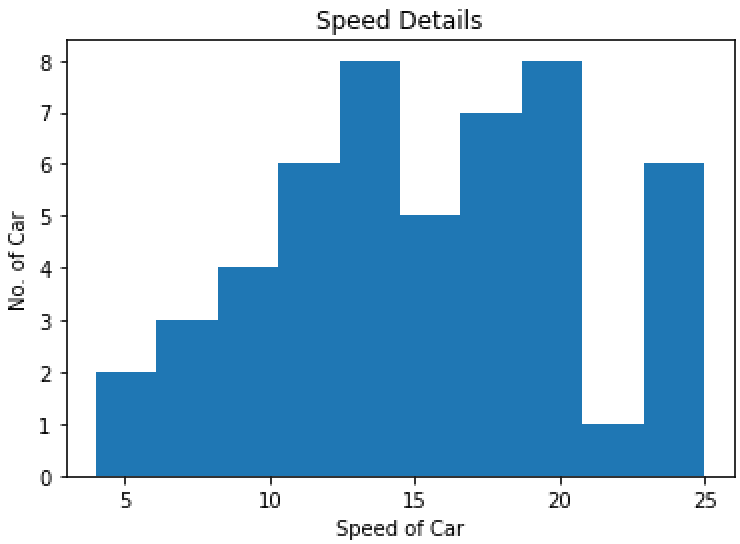
Out[13]: -0.5089944204057617

In [14]:
```
cars_details['dist'].kurtosis()
```

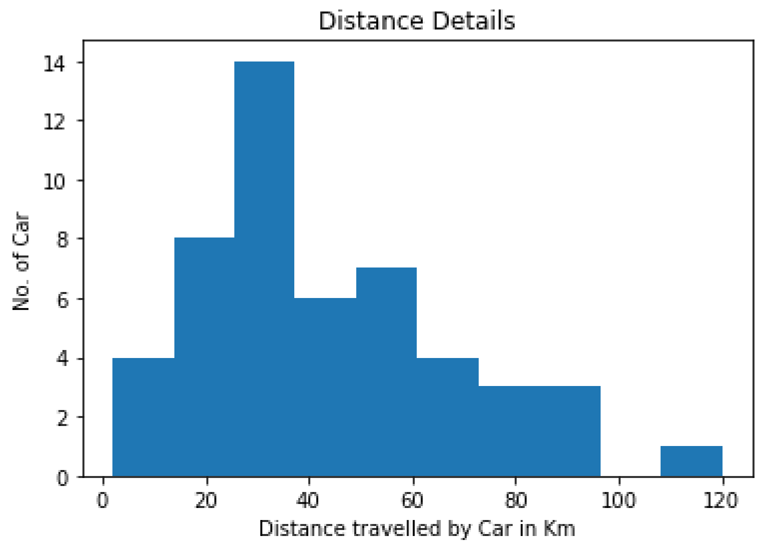
Out[14]: 0.4050525816795765

In [15]:
```
from sklearn.preprocessing import StandardScaler
```

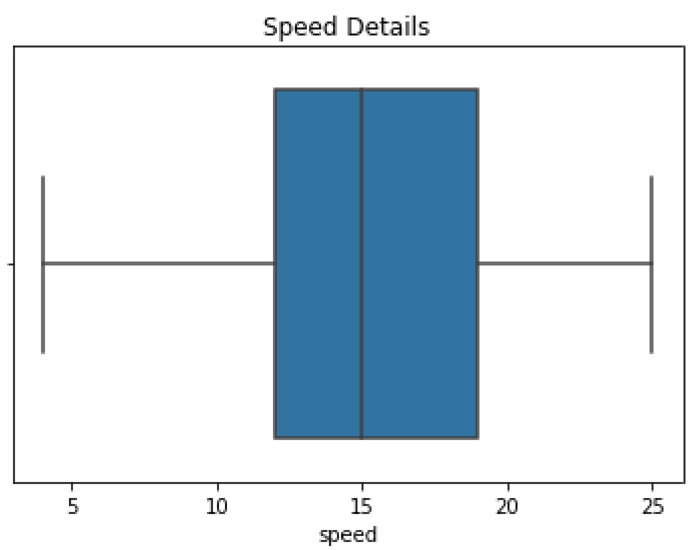
In [16]:
```
plt.hist(cars_details['speed'])
plt.title('Speed Details')
plt.xlabel('Speed of Car')
plt.ylabel('No. of Car')
plt.show() #We notice that the data is not normally distributed around the mean,
```



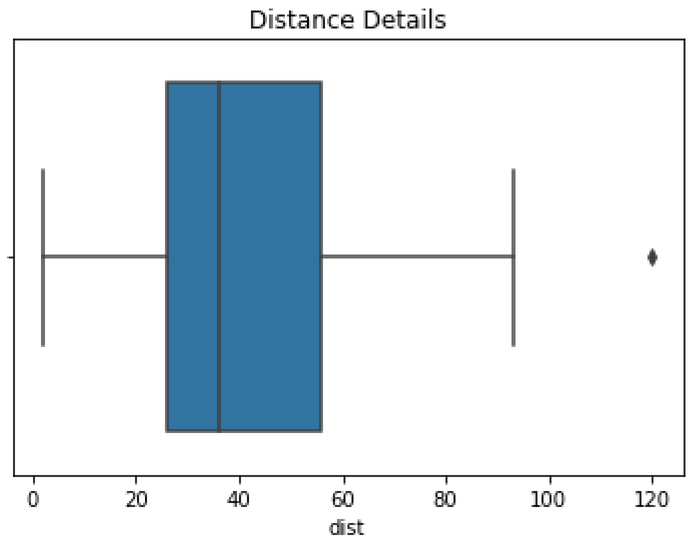
In [17]:
```
plt.hist(cars_details['dist'])
plt.title('Distance Details')
plt.xlabel('Distance travelled by Car in Km')
plt.ylabel('No. of Car')
plt.show() #We notice that the data is not normally distributed around the mean,
```



In [18]:
```
sns.boxplot(cars_details['speed'])
plt.title('Speed Details')
plt.show()
```



In [19]:
```
sns.boxplot(cars_details['dist'])
plt.title('Distance Details')
plt.show()
```



Q.9b Answer

In [20]:
```
sp_wt = pd.read_csv('Q9_b.csv')
sp_wt.head()
```

Out[20]:

	Unnamed: 0	SP	WT
0	1	104.185353	28.762059
1	2	105.461264	30.466833
2	3	105.461264	30.193597
3	4	113.461264	30.632114
4	5	104.461264	29.889149

In [21]:
```
sp_wt.isnull().sum()
```

Out[21]:
```
Unnamed: 0    0
SP            0
WT            0
dtype: int64
```

In [22]:
```
sp_wt.describe()
```

Out[22]:

	Unnamed: 0	SP	WT
count	81.000000	81.000000	81.000000
mean	41.000000	121.540272	32.412577
std	23.526581	14.181432	7.492813
min	1.000000	99.564907	15.712859
25%	21.000000	113.829145	29.591768
50%	41.000000	118.208698	32.734518
75%	61.000000	126.404312	37.392524
max	81.000000	169.598513	52.997752

In [23]:
```
sp_wt['SP'].skew() #Not normally distributed.
```

Out[23]:
```
1.6114501961773586
```

In [24]:
```
sp_wt['WT'].skew() #Moderately symmmetrical around mean
```

Out[24]:
```
-0.6147533255357768
```

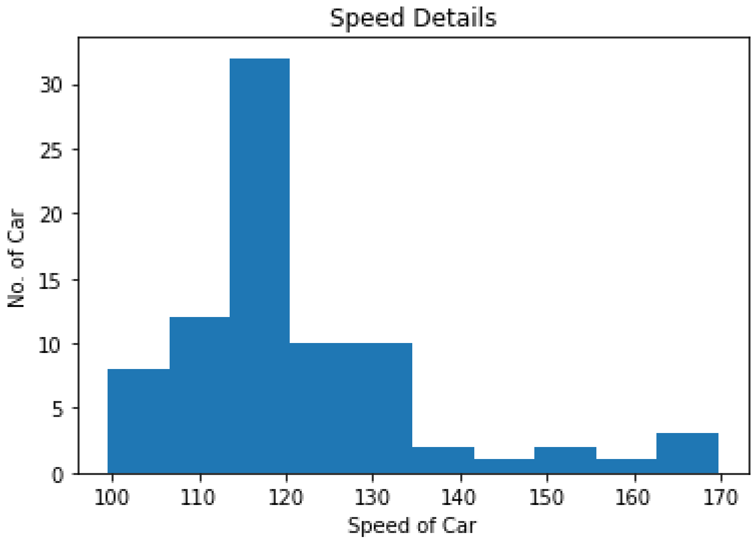
In [25]:
```
sp_wt['SP'].kurtosis() # Less outliers in this data.
```

Out[25]:
```
2.9773289437871835
```

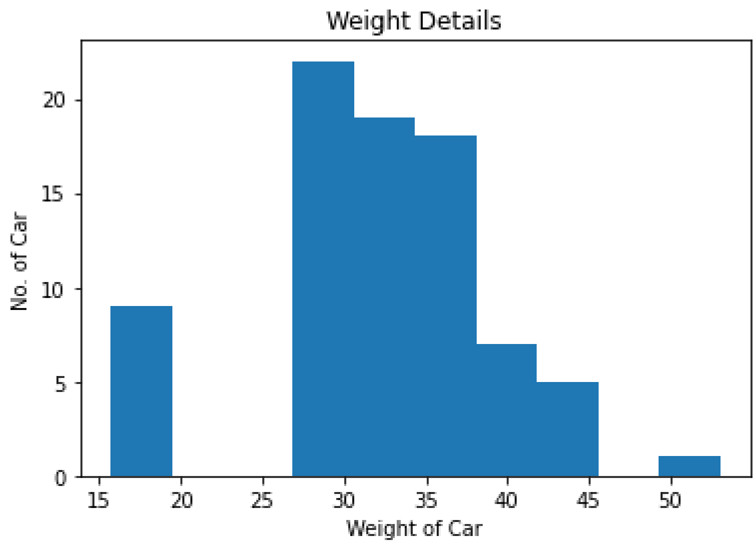
In [26]:
```
sp_wt['WT'].kurtosis() # Less outliers in this data.
```

Out[26]:
```
0.9502914910300326
```

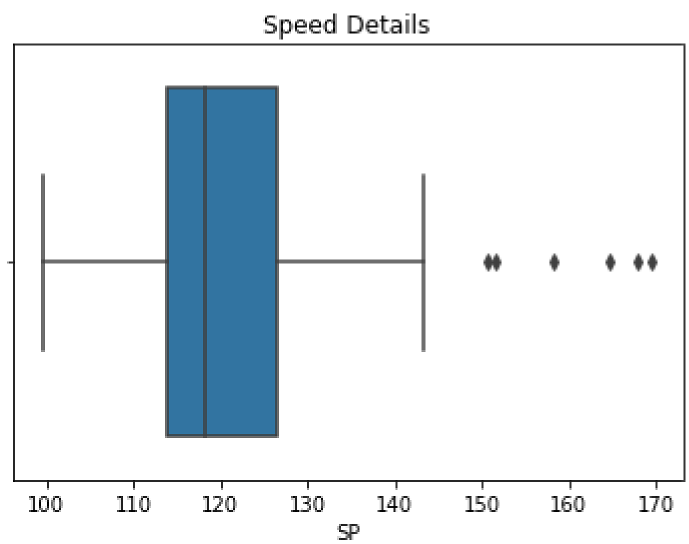
In [27]:
```
plt.hist(sp_wt['SP'])
plt.title('Speed Details')
plt.xlabel('Speed of Car')
plt.ylabel('No. of Car')
plt.show()
```



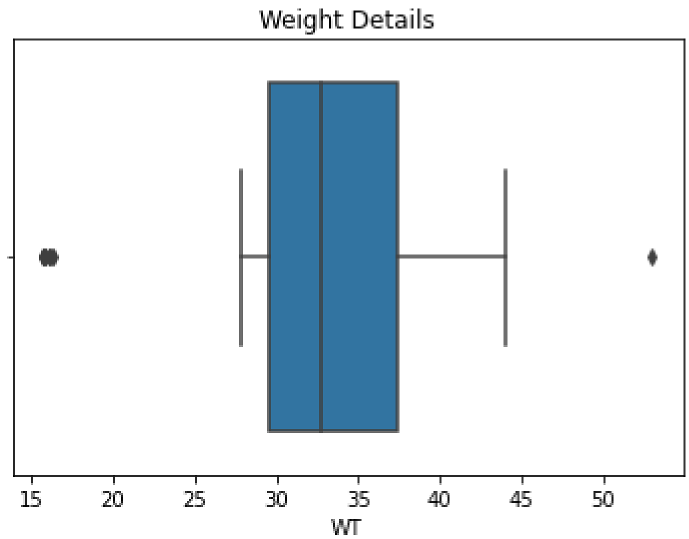
In [28]:
```
plt.hist(sp_wt['WT'])
plt.title('Weight Details')
plt.xlabel('Weight of Car')
plt.ylabel('No. of Car')
plt.show()
```



In [29]:
```
sns.boxplot(sp_wt['SP'])
plt.title('Speed Details')
plt.show()
```



In [30]:
```
sns.boxplot(sp_wt['WT'])
plt.title('Weight Details')
plt.show()
```



In [31]:
```
import scipy.stats
scipy.stats.t.ppf(q=.05, df=199)
```

Out[31]:
```
-1.6525467461659398
```

Q.12 Answer

In [32]:
```
student_marks = [34,36,36,38,38,39,39,40,40,41,41,41,41,42,42,45,49,56]
```

In [33]:
```
import numpy
```

In [34]:
```
x = numpy.mean(student_marks)
print(x)
```

```
41.0
```

In [35]:
```
x = numpy.median(student_marks)
print(x)
```

```
40.5
```

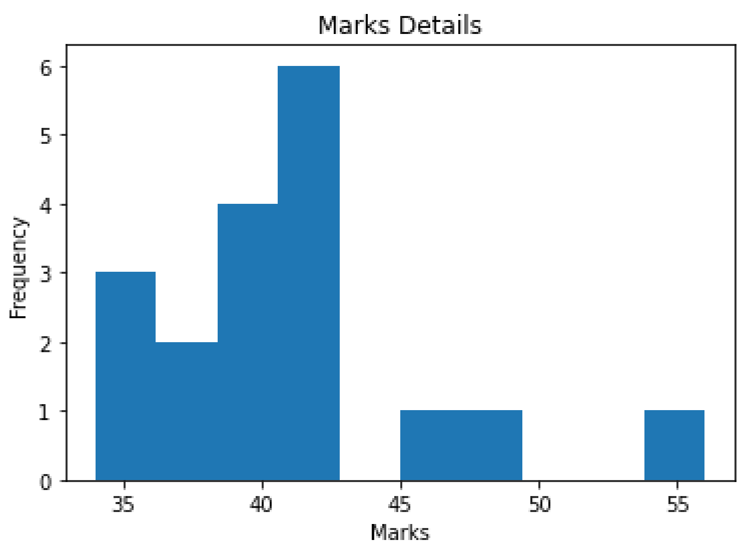
In [36]:
```
x = numpy.std(student_marks)
print(x)
```

```
4.910306620885412
```

In [37]:
```
import statistics
student_marks = [34,36,36,38,38,39,39,40,40,41,41,41,41,42,42,45,49,56]
print("Variance of student_marks is % s"
      %(statistics.variance(student_marks)))
```

```
Variance of student_marks is 25.529411764705884
```

In [38]:
```
plt.hist(student_marks)
plt.title('Marks Details')
plt.xlabel('Marks')
plt.ylabel('Frequency')
plt.show()
```



Q. 20 Answer

In [40]:
```
cars_details = pd.read_csv('Cars.csv')
cars_details
```

Out[40]:

	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 [43]:
```
cars_details.shape
```

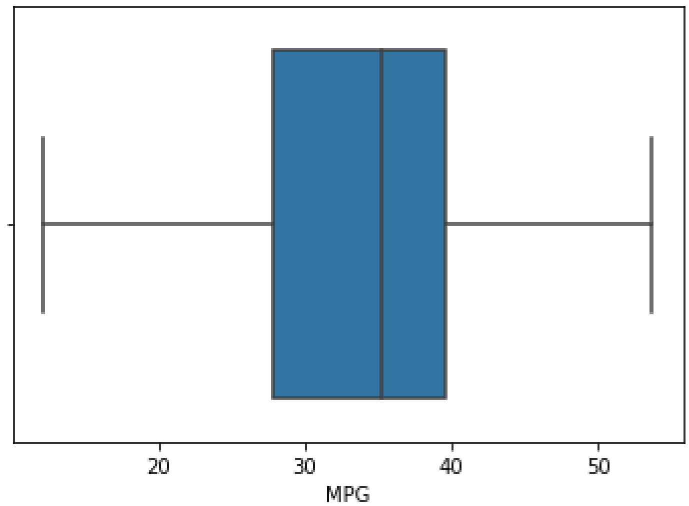
Out[43]: (81, 5)

In [60]:
```
cars_details.describe()
```

Out[60]:

	HP	MPG	VOL	SP	WT
count	81.000000	81.000000	81.000000	81.000000	81.000000
mean	117.469136	34.422076	98.765432	121.540272	32.412577
std	57.113502	9.131445	22.301497	14.181432	7.492813
min	49.000000	12.101263	50.000000	99.564907	15.712859
25%	84.000000	27.856252	89.000000	113.829145	29.591768
50%	100.000000	35.152727	101.000000	118.208698	32.734518
75%	140.000000	39.531633	113.000000	126.404312	37.392524
max	322.000000	53.700681	160.000000	169.598513	52.997752

In [53]:
```
sns.boxplot(cars_details.MPG)
plt.show()
```



In [54]:
```
stats.norm.cdf(38,cars_details.MPG.mean(),cars_details.MPG.std())
```

Out[54]: 0.6524060748417295

In [55]:
```
1-stats.norm.cdf(38,cars_details.MPG.mean(),cars_details.MPG.std()) #P (MPG>38)
```

Out[55]: 0.3475939251582705

In [56]:
```
stats.norm.cdf(40,cars_details.MPG.mean(),cars_details.MPG.std()) #P (MPG<40)
```

Out[56]: 0.7293498762151616

In [ ]:
```
#P (20<MPG<50)
```

In [58]:
```
stats.norm.cdf(0.50,cars_details.MPG.mean(),cars_details.MPG.std())-stats.norm.c
```

Out[58]: 1.2430968797327613e-05

Q. 21a Answer:

In [64]:
```
cars_details['MPG'].mean()
```

Out[64]: 34.422075728024666

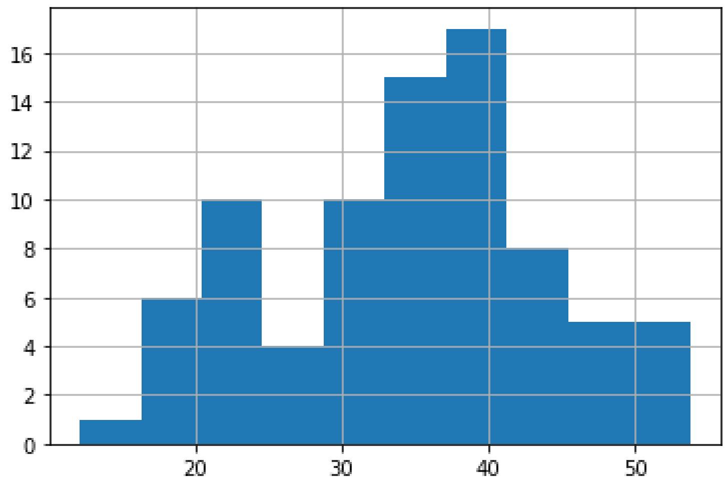
In [65]:
```
cars_details['MPG'].median()
```

Out[65]:
```
35.15272697
```

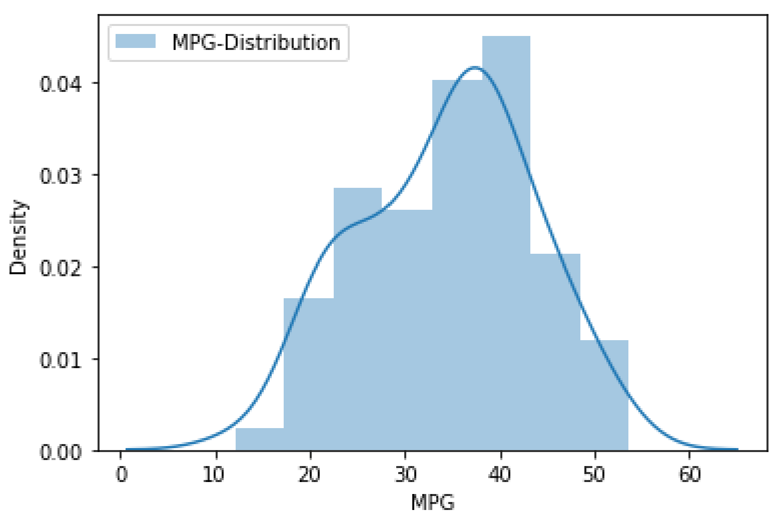
In [71]:
```
cars_details['MPG'].mode()
```

Out[71]:
```
0    29.629936
dtype: float64
```

In [78]:
```
cars_details['MPG'].hist()
plt.show()
```



In [88]:
```
sns.distplot(cars_details.MPG,label='MPG-Distribution')
plt.xlabel('MPG')
plt.ylabel('Density')
plt.legend();
```



In [90]:
```
cars_details['MPG'].skew() #Data is fairly symmetrical around the mean, i.e fairl
```

Out[90]:
```
-0.17794674747025727
```

In [91]:
```
cars_details['MPG'].kurt() # No outliers present in data so that we can say that
```

Out[91]:
```
-0.6116786559430913
```

Q. 21b Answer

In [93]:
```
wc_at_details = pd.read_csv('wc-at.csv')
wc_at_details.head()
```

Out[93]:

	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 [95]:
```
wc_at_details.describe()
```

Out[95]:

	Waist	AT
count	109.000000	109.000000
mean	91.901835	101.894037
std	13.559116	57.294763
min	63.500000	11.440000
25%	80.000000	50.880000
50%	90.800000	96.540000
75%	104.000000	137.000000
max	121.000000	253.000000

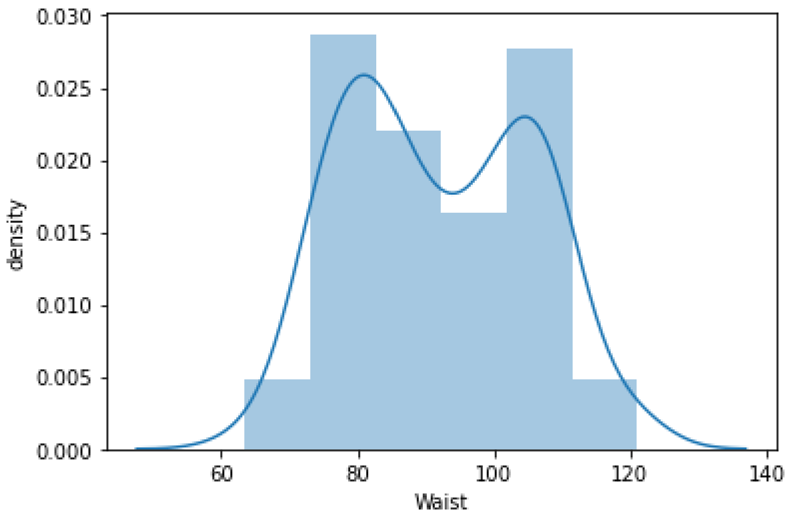
In [114]:
```
wc_at_details.mean()
```

Out[114]:
```
Waist     91.901835
AT       101.894037
dtype: float64
```

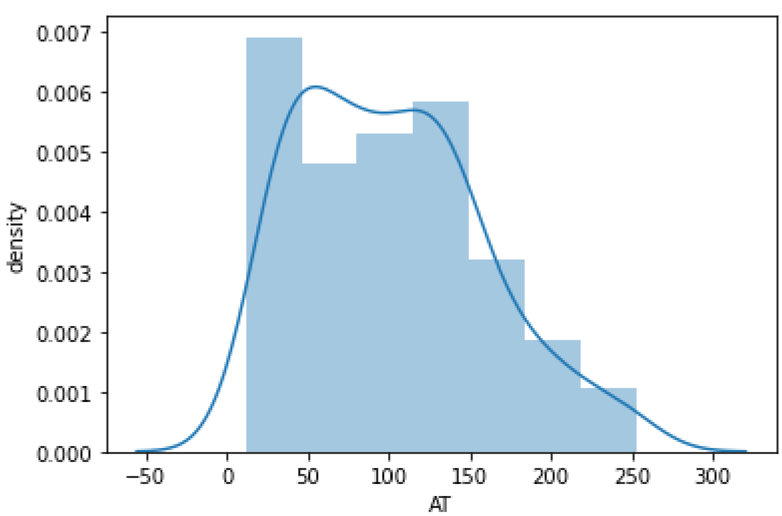
In [115]:
```
wc_at_details.median()
```

Out[115]:
```
Waist    90.80
AT       96.54
dtype: float64
```

In [104]:
```
# Plotting distribution for Waist:
sns.distplot(wc_at_details.Waist)
plt.ylabel('density');
```



In [105]:
```
# Plotting distribution for Adipose Tissue (AT)
sns.distplot(wc_at_details.AT)
plt.ylabel('density');
```



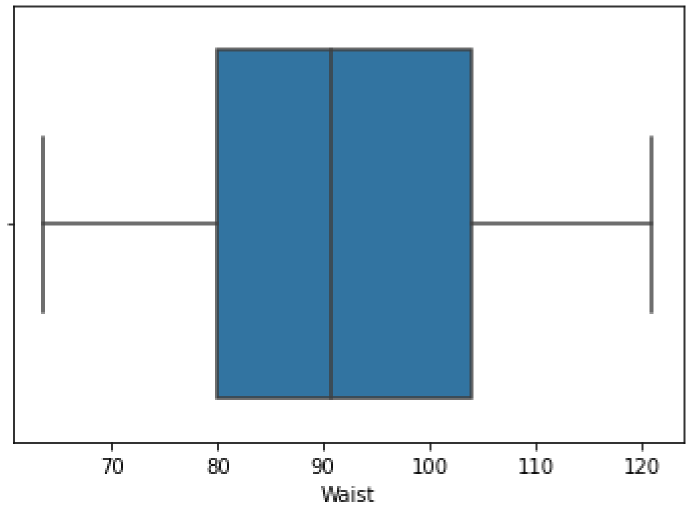
In [112]:
```
wc_at_details['Waist'].skew(), wc_at_details['Waist'].kurt() #WT
```

Out[112]:
```
(0.1340560824786468, -1.1026666011768886)
```

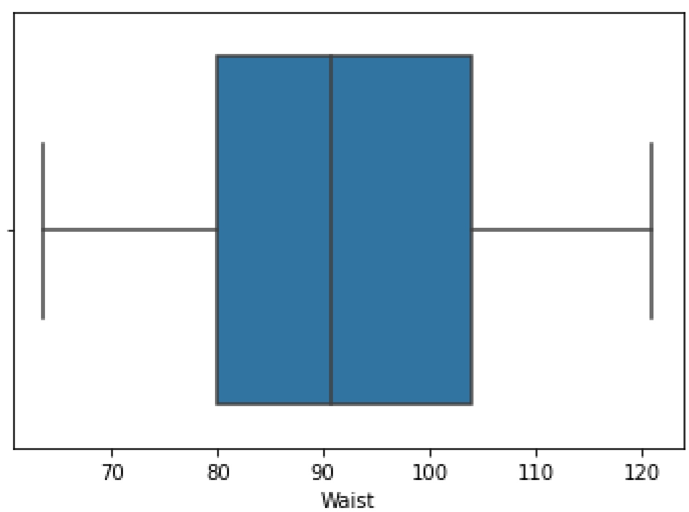
In [111]:
```
wc_at_details['AT'].skew(), wc_at_details['AT'].kurt() #AT
```

Out[111]:
```
(0.584869324127853, -0.28557567504584425)
```

In [118]:
```
sns.boxplot(wc_at_details['Waist'])
plt.show()
# Here in this type of data, mean > median, right whisker is larger than left whi
```



In [117]:
```
sns.boxplot(wc_at_details['Waist'])
plt.show()
#mean>median,both whisker are of same lenght, median is slightly shifted towards
```



Q. 22 Answer

In [119]:
```
from scipy import stats
from scipy.stats import norm
```

In [120]:
```
stats.norm.ppf(0.95) # Z-score of 90% confidence interval
```

Out[120]: 1.6448536269514722

In [121]:
```
stats.norm.ppf(0.97) # Z-score of 94% confidence interval
```
Out[121]: 1.8807936081512509

In [122]:
```
stats.norm.ppf(0.8) # Z-score of 60% confidence interval
```
Out[122]: 0.8416212335729143

Q. 23 Answer

In [ ]:
```
 # df = n-1 = 25-1 = 24
```

In [123]:
```
stats.t.ppf(0.975,24) # t-scores of 95% confidence interval for sample size of 25
```
Out[123]: 2.0638985616280205

In [124]:
```
stats.t.ppf(0.98,24) # t-scores of 96% confidence interval for sample size of 25
```
Out[124]: 2.1715446760080677

In [125]:
```
stats.t.ppf(0.995,24) # t-scores of 99% confidence interval for sample size of 25
```
Out[125]: 2.796939504772804

Q. 24 Answer

In [126]:
```
# Null Hypothesis is: H0 = Avg life of Bulb >= 260 days
# Alternate Hypothesis is: Ha = Avg life of Bulb < 260 days
```

In [ ]:
```
# To find t-scores at x=260;
# t = (sample mean - Popilation mean) / (Std. Dev / sqrt(n))
```

In [128]:
```
t=(260-270)/(90/18**0.5) # t_scores
t
```
Out[128]: -0.4714045207910317

In [129]:
```
# To Find P(X >=260) for Null hypothesis H0, df = n-1 = 18-1 = 17
```

In [131]:
```
stats.t.cdf(abs(-0.4714),df=17)
```
Out[131]: 0.6783258831553944

In [132]:
```
p_value = 1-stats.t.cdf(abs(-0.4714),df=17) # p_value=1-stats.t.cdf(abs(t_scores)
p_value
```
Out[132]: 0.32167411684460556

In [135]:
```
# OR here is also an alternative way to find out p_value.
p_value=stats.t.sf(abs(-0.4714),df=17)
p_value
```

Out[135]:
```
0.32167411684460556
```

====================The End==========================