1) Import Necessary Libraries

In [49]:
```
import pandas as pd
from matplotlib import pyplot as plt
import seaborn as sns

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

2) Import Data

In [50]:
```
cars_data=pd.read_csv('Cars.csv')
cars_data
```

Out[50]:

	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

3) Data Understanding

3.1 Initial Analysis

In [4]:
```
cars_data.shape
```

Out[4]:
```
(81, 5)
```

In [5]:
```
cars_data.isna().sum()
```

Out[5]:
```
HP     0
MPG    0
VOL    0
SP     0
WT     0
dtype: int64
```

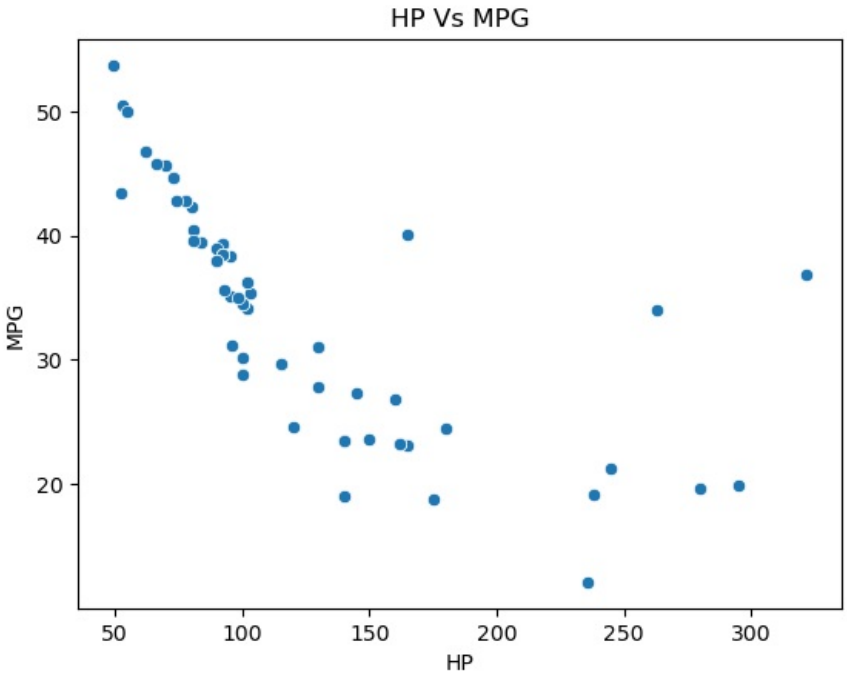
In [6]:
```
cars_data.dtypes
```

Out[6]:
```
HP       int64
MPG    float64
VOL      int64
SP     float64
WT     float64
dtype: object
```

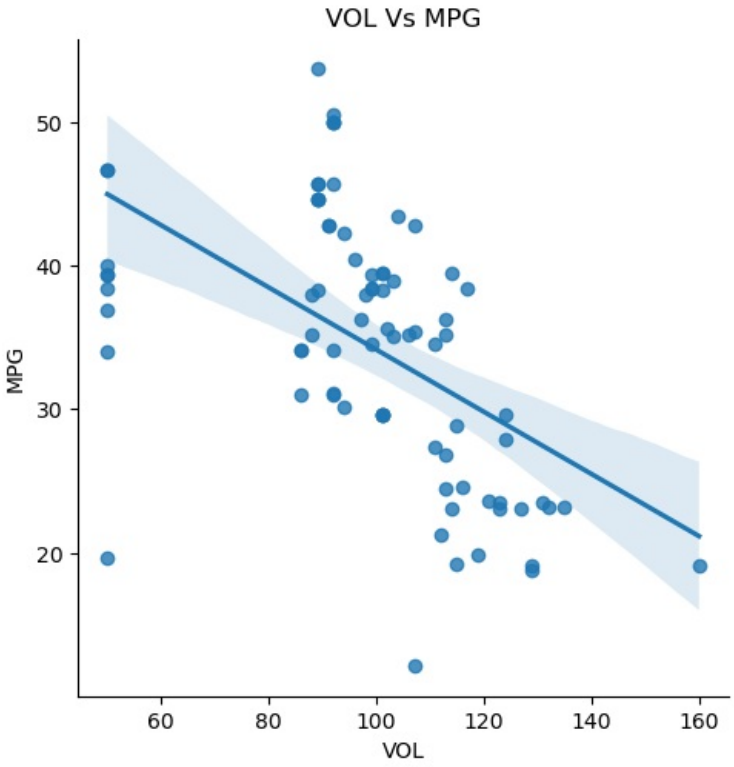
3.2 Assumptions Check

Assumption 1: Test For Linearity

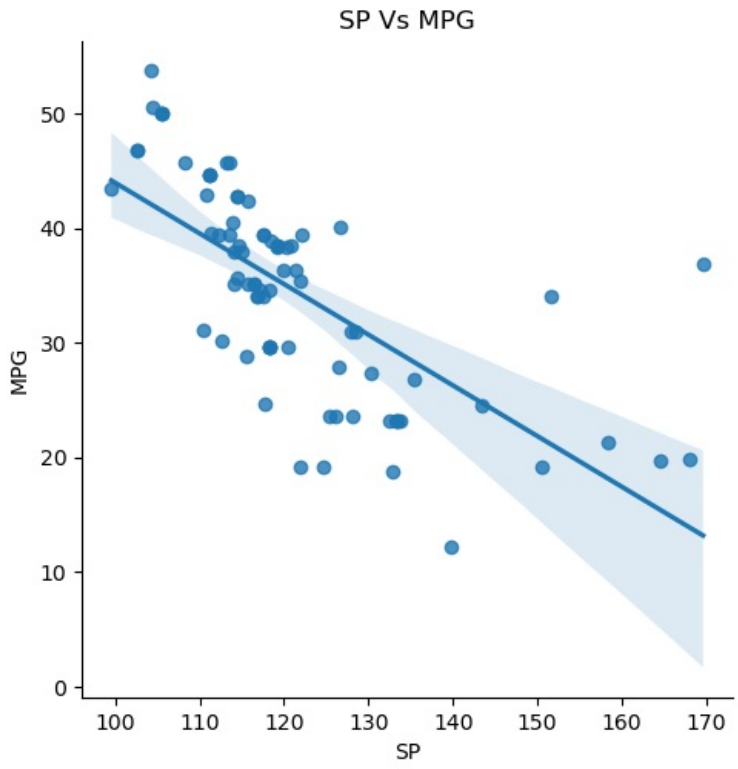
In [7]:
```
sns.scatterplot(x='HP',y='MPG',data=cars_data)
plt.title('HP Vs MPG')
plt.show()
```



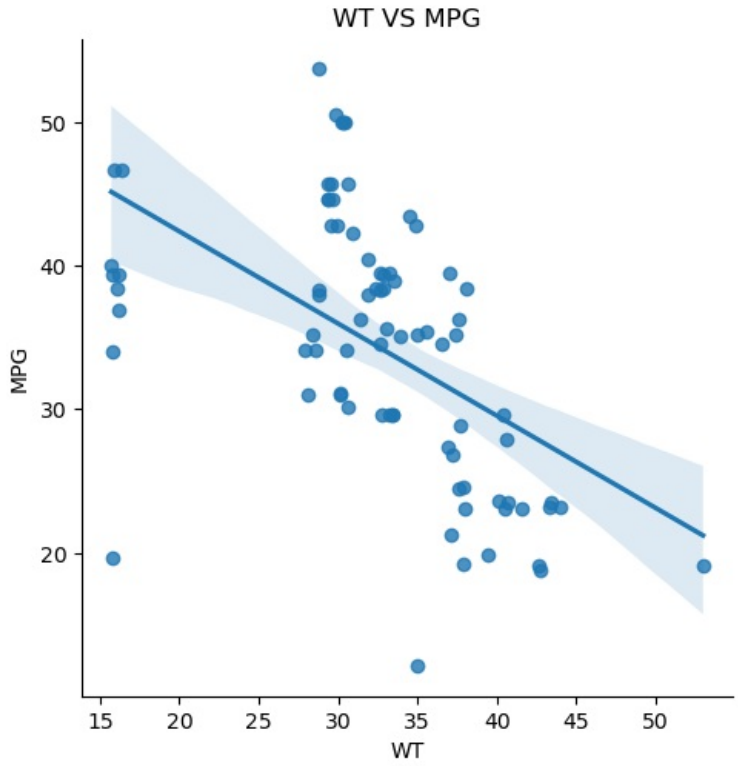
In [8]:
```
sns.lmplot(x='VOL',y='MPG',data=cars_data)
plt.title('VOL Vs MPG')
plt.show()
```



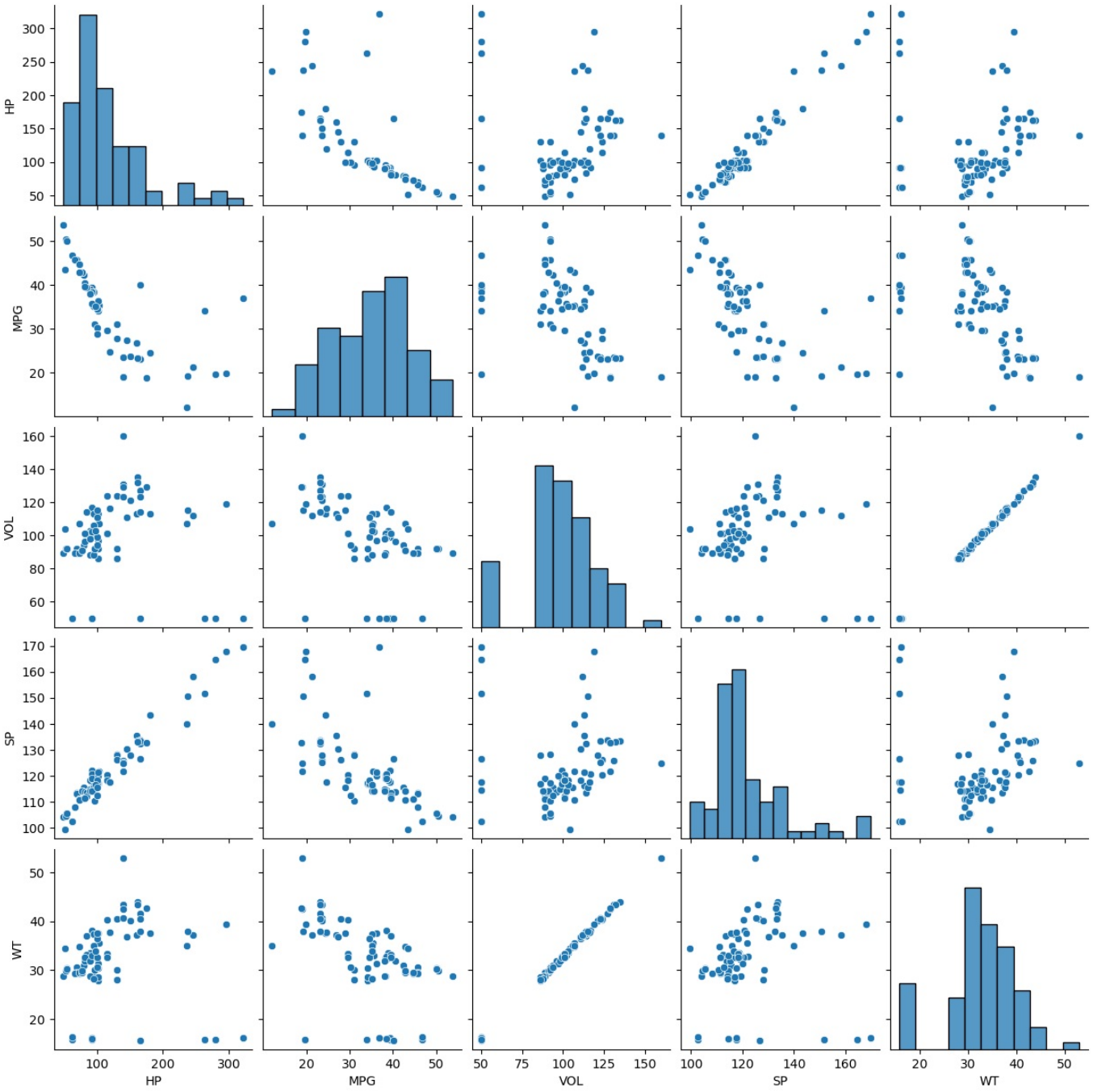
In [10]:
```
sns.lmplot(x='SP',y='MPG',data=cars_data)
plt.title('SP Vs MPG')
plt.show()
```



In [12]:
```
sns.lmplot(x='WT',y='MPG',data=cars_data)
plt.title('WT VS MPG')
plt.show()
```



In [14]:
```
sns.pairplot(data=cars_data)
plt.show()
```



In [15]:
```
cars_data.corr()
```

Out[15]:

	HP	MPG	VOL	SP	WT
HP	1.000000	-0.725038	0.077459	0.973848	0.076513
MPG	-0.725038	1.000000	-0.529057	-0.687125	-0.526759
VOL	0.077459	-0.529057	1.000000	0.102170	0.999203
SP	0.973848	-0.687125	0.102170	1.000000	0.102439
WT	0.076513	-0.526759	0.999203	0.102439	1.000000

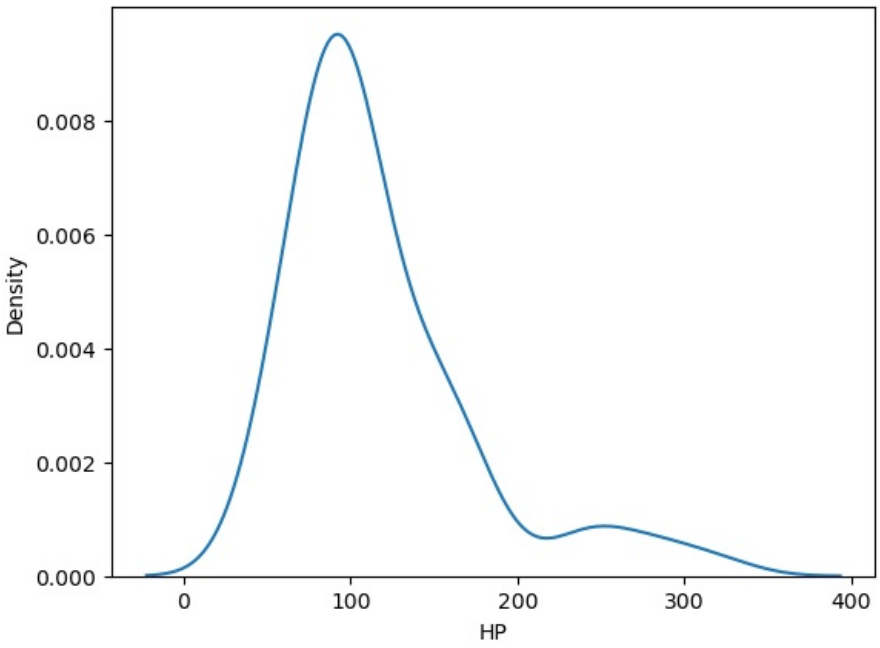
Observation

The linearity test failed

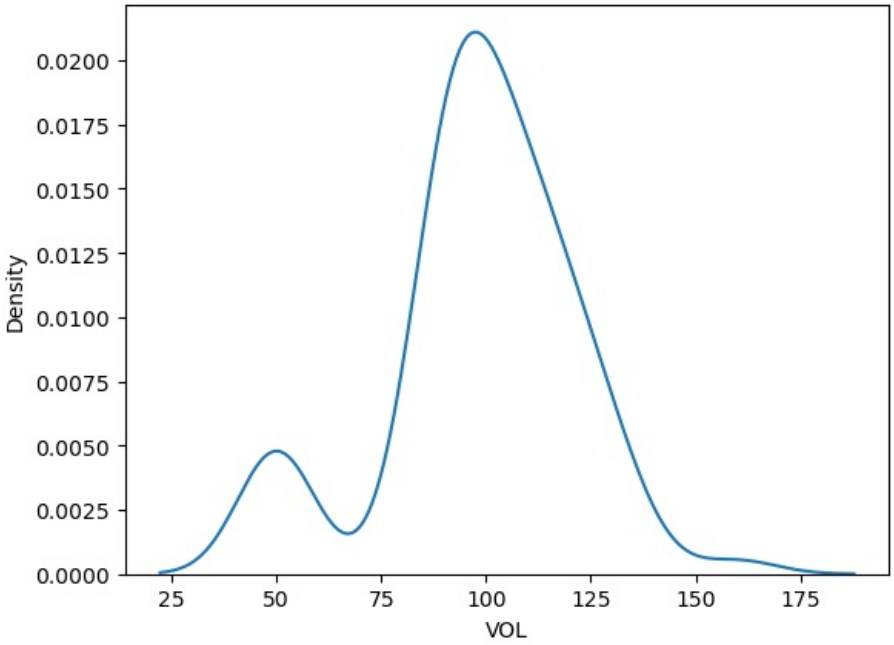
Asssumptions 2: Test for Linearity

2.1 : Using Distplot

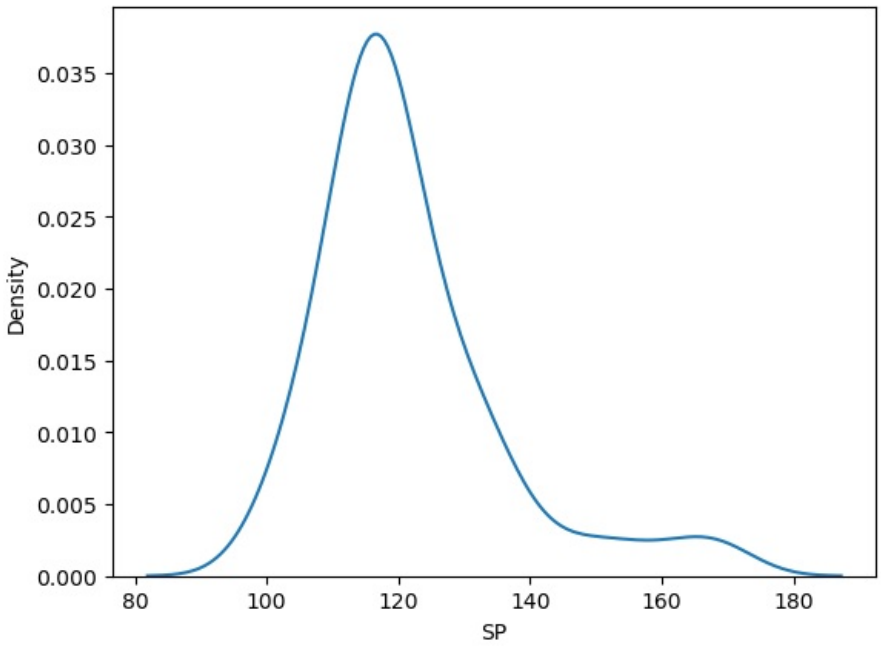
In [9]:
```
sns.distplot(a=cars_data['HP'],hist=False)
plt.show()
```



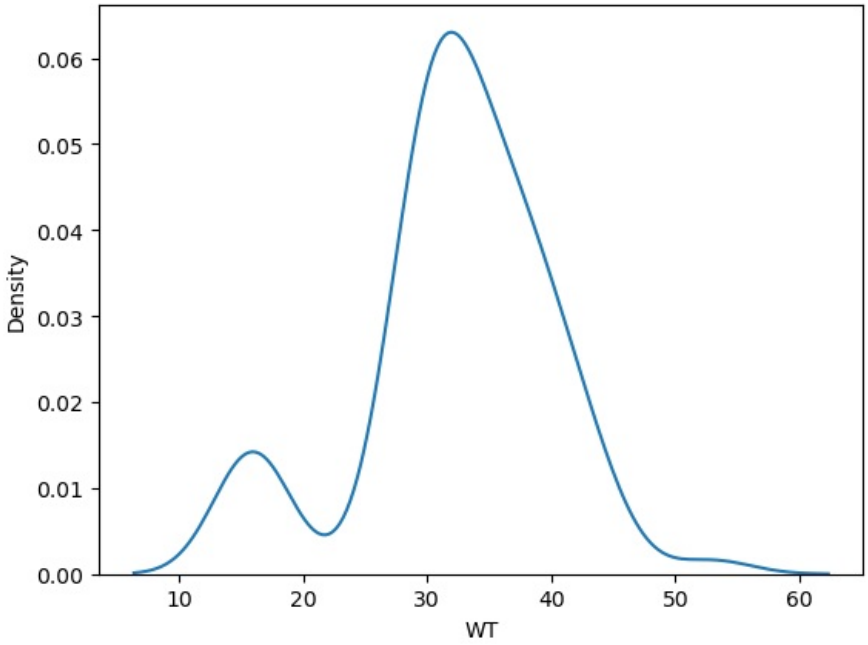
In [10]:
```
sns.distplot(a=cars_data['VOL'],hist=False)
plt.show()
```



In [11]:
```
sns.distplot(a=cars_data['SP'],hist=False)
plt.show()
```



In [12]:
```
sns.distplot(a=cars_data['WT'],hist=False)
plt.show()
```

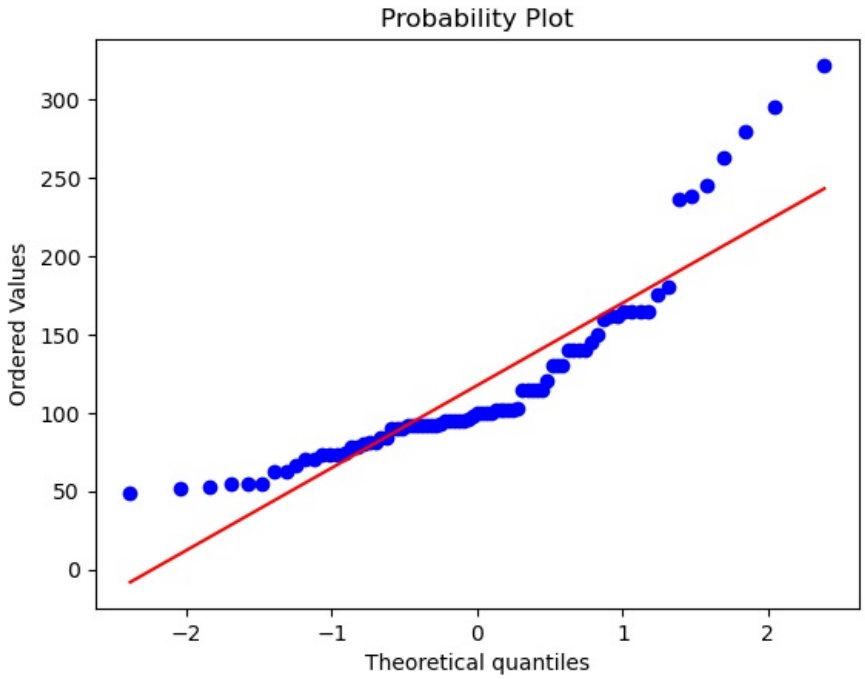


Observation

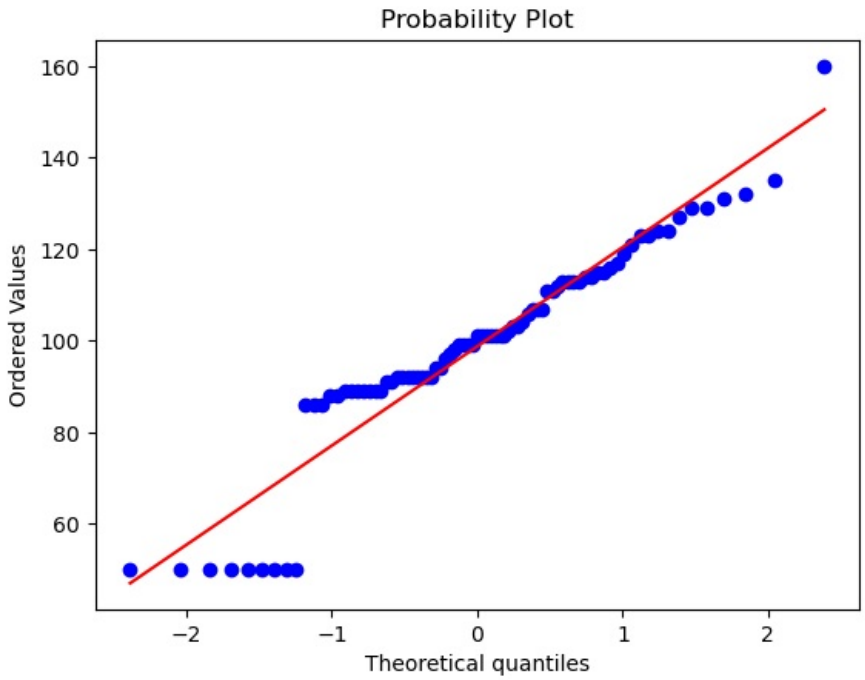
Normality test failed

2.2:Using Probplot

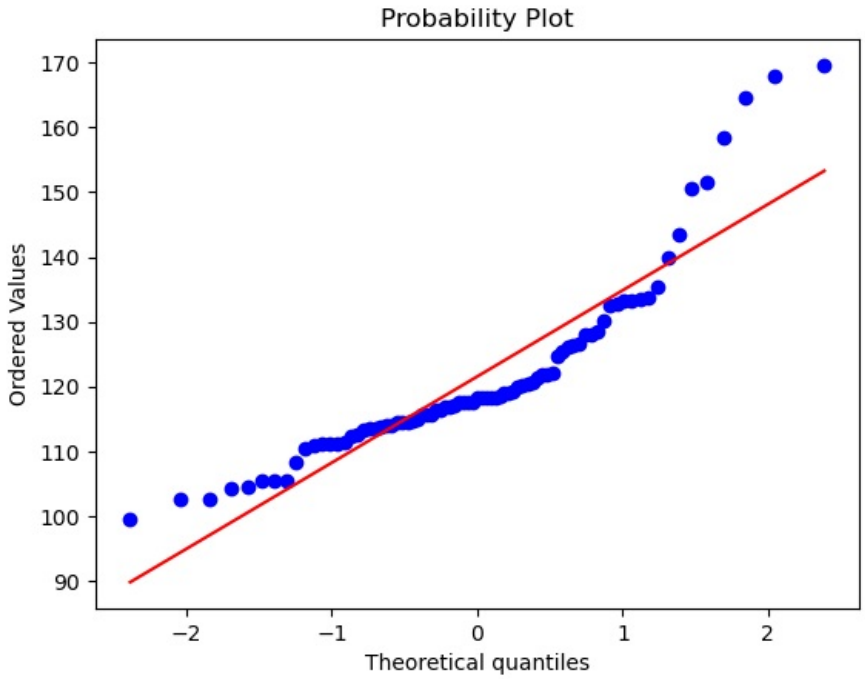
In [15]:
```
from scipy import stats
stats.probplot(x=cars_data['HP'],dist='norm',plot=plt)
plt.show()
```



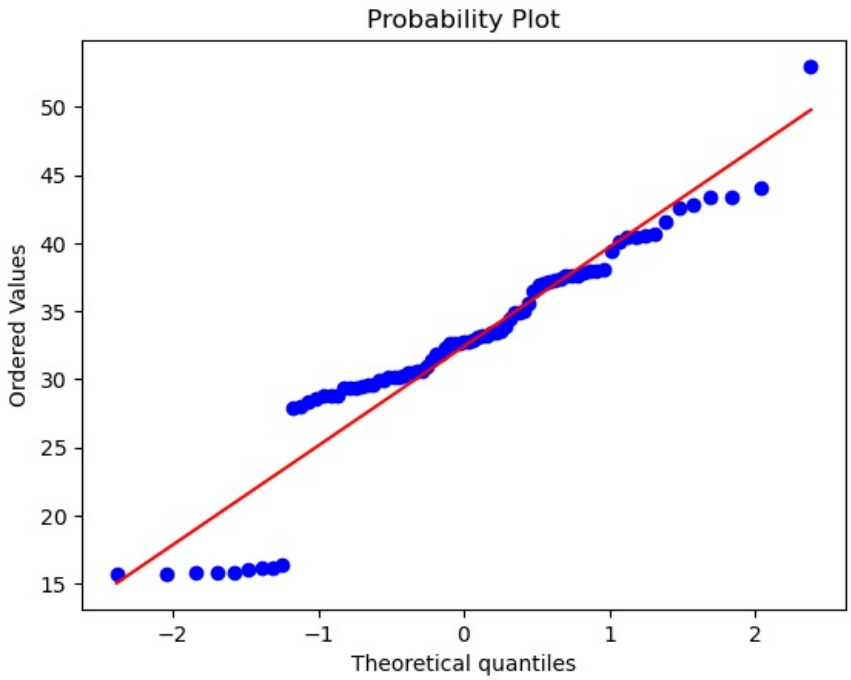
In [16]:
```
stats.probplot(x=cars_data['VOL'],dist='norm',plot=plt)
plt.show()
```



In [17]:
```
stats.probplot(x=cars_data['SP'],dist='norm',plot=plt)
plt.show()
```



In [18]:
```
stats.probplot(x=cars_data['WT'],dist='norm',plot=plt)
plt.show()
```



Observation

Test for linearity failed

Assumption 3: Test for Multi collinearity

By using 2 Techniques

1.Correlation Matrix

2.Variance Inflation Factor (VIF)

3.1 Correlation Test

In [ ]:
```
corr_matrix=cars_data.corr().round()
corr_matrix
```

In [ ]:
```
sns.heatmap(data=corr_matrix,annot=True)
plt.show()
```

Observation

There is multicollinearity in my data, so this test also fails.

4 No Auto Regression Passed

Model Building

In [25]:
```
#x=cars_data.drop(labels='MPG',axis=1,inplace=True)
x=cars_data[['HP','VOL','SP','WT']]
y=cars_data['MPG']
```

cars_data.drop(['MPG'],axis=1)

In [26]:
```
x
```

Out[26]:

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

81 rows × 4 columns

In [27]:
```
y
```

Out[27]:
```
0     53.700681
1     50.013401
2     50.013401
3     45.696322
4     50.504232
        ...
76    36.900000
77    19.197888
78    34.000000
79    19.833733
80    12.101263
Name: MPG, Length: 81, dtype: float64
```

Model Validation / Data Validation

In [33]:
```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2,random_state=35)
```

In [34]:
```
print(x_train.shape,x_test.shape,y_train.shape,y_test.shape)
```

```
(64, 4) (17, 4) (64,) (17,)
```

In [35]:
```
from sklearn.linear_model import LinearRegression
```

In [37]:
```
mlr = LinearRegression() #model Building
mlr.fit(x_train,y_train)
y_pred_train=mlr.predict(x_train)
y_pred_test=mlr.predict(x_test)
```

In [ ]:
```
x1_train= hp - mpg
x2_train=hp,vol - mpg
x3_train=hp,vol,sp - mpg
```

In [39]:
```
y_pred_test
```

Out[39]:
```
array([41.93599129, 36.64877574, 37.21897411, 33.80175655, 39.2043074 ,
       34.13391752, 35.61010219, 38.83215133, 35.77879072, 33.53455187,
       23.02614465, 43.54665895, 11.49873125, 34.36553633, 21.54290653,
       24.03684324, 48.98206504])
```

In [40]:
```
error_train=y_train-y_pred_train
error_train
```

Out[40]:
```
14    3.440987
8    -2.879342
47   -0.711218
52    0.889579
49   -4.197923
        ...
63   -3.055257
33    1.492869
55   -0.136305
15    2.445640
73    0.577655
Name: MPG, Length: 64, dtype: float64
```

In [45]:
```
error_test=y_test-y_pred_test
error_test
```

Out[45]:
```
13     2.716843
48    -5.634645
22     1.091632
51    -4.171821
38    -5.133639
46     0.427581
41    -0.457375
27    -0.421148
58    -5.646868
50    -3.904616
67     0.077027
0     10.154022
79     8.335002
40     0.787191
71     1.660662
68    -0.933672
7     -2.265511
Name: MPG, dtype: float64
```

In [51]:
```
from sklearn.metrics import r2_score
r2_train=r2_score(y_train,y_pred_train)
print(r2_train)
```

```
0.7689342881829975
```

In [52]:
```
r2_test=r2_score(y_test,y_pred_test)
print(r2_test)
```

```
0.7618997645852453
```