Perform the following preprocessing tasks on the dirty_iris (https://raw.githubusercontent.com/edwindj/datacleaning/ dataset.

In [ ]:
```
import pandas as pd
df=pd.read_csv("https://raw.githubusercontent.com/edwindj/datacleaning/master/data/dirty_iris.csv")
df
```

Out[ ]:

	Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species
0	6.4	3.2	4.5	1.5	versicolor
1	6.3	3.3	6.0	2.5	virginica
2	6.2	NaN	5.4	2.3	virginica
3	5.0	3.4	1.6	0.4	setosa
4	5.7	2.6	3.5	1.0	versicolor
...	...	...	...	...	...
145	6.7	3.1	5.6	2.4	virginica
146	5.6	3.0	4.5	1.5	versicolor
147	5.2	3.5	1.5	0.2	setosa
148	6.4	3.1	NaN	1.8	virginica
149	5.8	2.6	4.0	NaN	versicolor

150 rows × 5 columns

Calculate the number and percentage of observations that are complete.

In [ ]:
```
c=len(df.dropna())
print("Number of observations that are complete:",c)
print("Percentage of observations that are complete:",c/len(df)*100,"%")
```

```
Number of observations that are complete: 96
Percentage of observations that are complete: 64.0 %
```

Replace all the special values in data with NA.

In [ ]:
```
df.fillna("NA")
```

Out[ ]:

	Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species
0	6.4	3.2	4.5	1.5	versicolor
1	6.3	3.3	6	2.5	virginica
2	6.2	NA	5.4	2.3	virginica
3	5	3.4	1.6	0.4	setosa
4	5.7	2.6	3.5	1	versicolor
...	...	...	...	...	...
145	6.7	3.1	5.6	2.4	virginica
146	5.6	3	4.5	1.5	versicolor
147	5.2	3.5	1.5	0.2	setosa
148	6.4	3.1	NA	1.8	virginica
149	5.8	2.6	4	NA	versicolor

150 rows × 5 columns

Define these rules in a separate text file and read them.

In [ ]:
```
from google.colab import drive
drive.mount('/content/drive')
```

```
Mounted at /content/drive
```

In [ ]:
```
%cd drive/MyDrive/Colab Notebooks
```

```
/content/drive/MyDrive/Colab Notebooks
```

In [ ]:
```
import Rules as RL
R=RL.Rules
```

In [ ]:
```
R.keys()
```

Out[ ]:
```
dict_keys(['speciescheck', 'allcheck', 'lengthcheck', 'sepalcheck', 'flowerch
eck'])
```

– Species should be one of the following values: setosa, versicolor or virginica.

In [ ]:
```
R1=df.apply(R['speciescheck'],axis=1).rename('speciescheck')
R1
```

Out[ ]:
```
0      True
1      True
2      True
3      True
4      True
       ...
145    True
146    True
147    True
148    True
149    True
Name: speciescheck, Length: 150, dtype: bool
```

–All measured numerical properties of an iris should be positive.

In [ ]:
```
R2=df.apply(R['allcheck'],axis=1).rename('allcheck')
R2
```

Out[ ]:
```
0       True
1       True
2      False
3       True
4       True
       ...
145     True
146     True
147     True
148    False
149    False
Name: allcheck, Length: 150, dtype: bool
```

–The petal length of an iris is atleast 2 times its petal width.

In [ ]:
```
R3=df.apply(R['lengthcheck'],axis=1).rename('lengthcheck')
R3
```

Out[ ]:
```
0       True
1       True
2       True
3       True
4       True
       ...
145     True
146     True
147     True
148    False
149    False
Name: lengthcheck, Length: 150, dtype: bool
```

–The sepal length of an iris cannot exceed 30cm.

In [ ]:
```
R4=df.apply(R['sepalcheck'],axis=1).rename('sepalcheck')
R4
```

Out[ ]:
```
0      True
1      True
2      True
3      True
4      True
       ...
145    True
146    True
147    True
148    True
149    True
Name: sepalcheck, Length: 150, dtype: bool
```

–The sepals of an iris are longer than its petals.

In [ ]:
```
R5=df.apply(R['flowercheck'],axis=1).rename('flowercheck')
R5
```

Out[ ]:
```
0       True
1       True
2       True
3       True
4       True
       ...
145     True
146     True
147     True
148    False
149     True
Name: flowercheck, Length: 150, dtype: bool
```

In [ ]:
```
Result=pd.DataFrame([R1,R2,R3,R4,R5])
Result=Result.transpose()
Result
```

Out[ ]:

	speciescheck	allcheck	lengthcheck	sepalcheck	flowercheck
0	True	True	True	True	True
1	True	True	True	True	True
2	True	False	True	True	True
3	True	True	True	True	True
4	True	True	True	True	True
...	...	...	...	...	...
145	True	True	True	True	True
146	True	True	True	True	True
147	True	True	True	True	True
148	True	False	False	True	False
149	True	False	False	True	True

150 rows × 5 columns

Determine how often each rule is broken (violatedEdits). Also summarize and plot the result.

In [ ]:
```
RD={'speciescheck':'Rule: Species should be one of the following values: setos
a,versicolor or virginica.','allcheck':"Rule: All properties should be greater
than 0",'lengthcheck':'Rule: petal length should be 2 times the petal width',
'sepalcheck':"Rule: Sepal length should be less than 30",'flowercheck':"Rule:
 Sepal length should be greater than petal length"}
V=[]
for i in R.keys():
    print(i)
    print(RD[i])
    print('No of violations: ',end='')
    violations=Result[Result[i]==False].shape[0]
    print(violations)
    V.append(violations)
```

```
speciescheck
Rule: Species should be one of the following values: setosa,versicolor or vir
ginica.
No of violations: 0
allcheck
Rule: All properties should be greater than 0
No of violations: 57
lengthcheck
Rule: petal length should be 2 times the petal width
No of violations: 34
sepalcheck
Rule: Sepal length should be less than 30
No of violations: 12
flowercheck
Rule: Sepal length should be greater than petal length
No of violations: 30
```

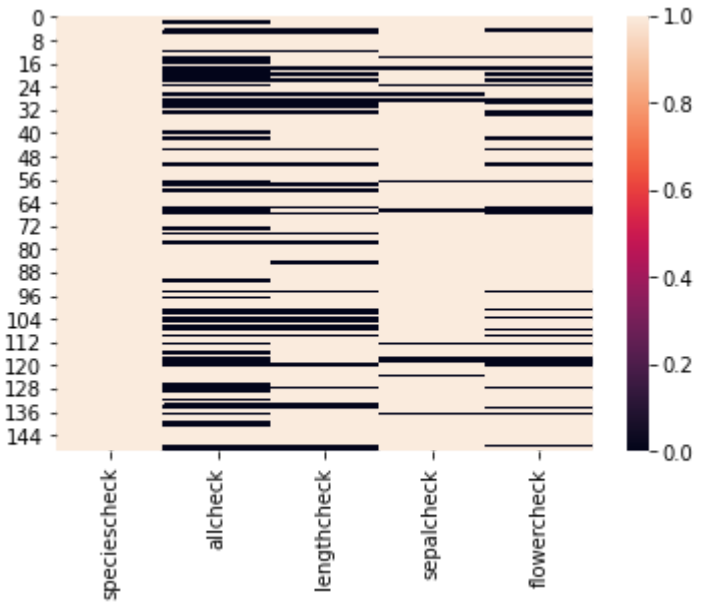
In [ ]:
```
Result.describe()
```

Out[ ]:

	speciescheck	allcheck	lengthcheck	sepalcheck	flowercheck
count	150	150	150	150	150
unique	1	2	2	2	2
top	True	True	True	True	True
freq	150	93	116	138	120

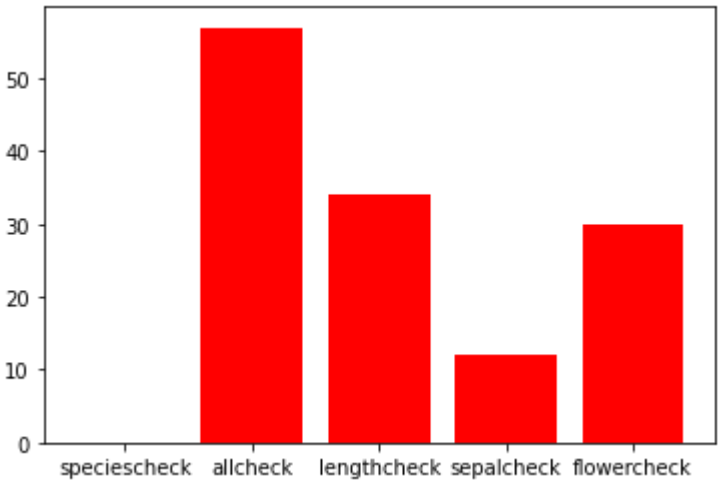
In [ ]:
```
import matplotlib.pyplot as plt
import seaborn as sns
sns.heatmap(Result,square=False,)
```

Out[ ]: <matplotlib.axes._subplots.AxesSubplot at 0x7f1aa90baed0>



In [ ]:
```
plt.bar(Result.columns,V,color='Red')
```

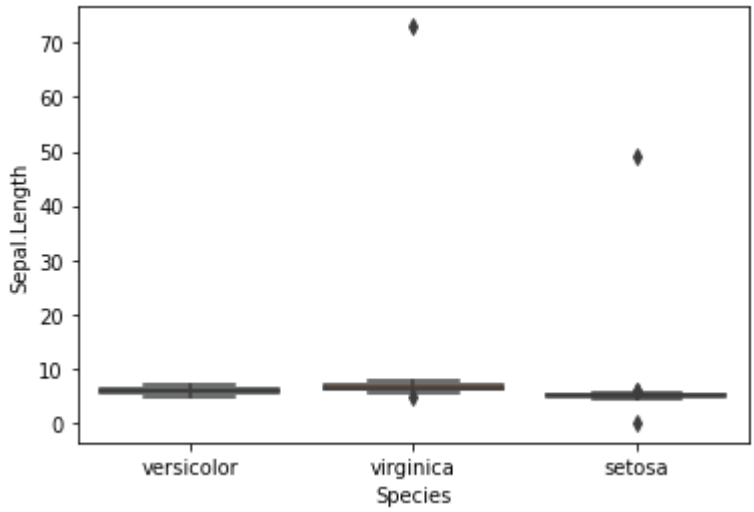
Out[ ]: <BarContainer object of 5 artists>



Find outliers in sepal length using boxplot and boxplot.stats

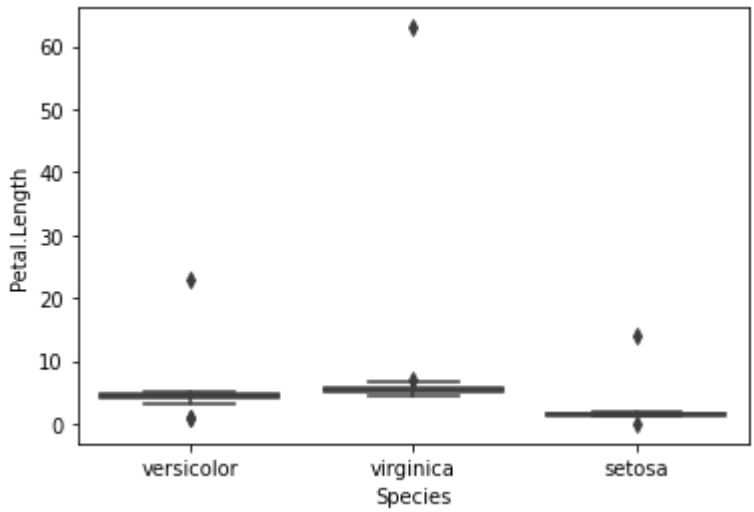
In [ ]:
```
import seaborn as sns
sns.boxplot(x=df['Species'],y=df['Sepal.Length'])
```

Out[ ]: <matplotlib.axes._subplots.AxesSubplot at 0x7f1ab3d04c90>



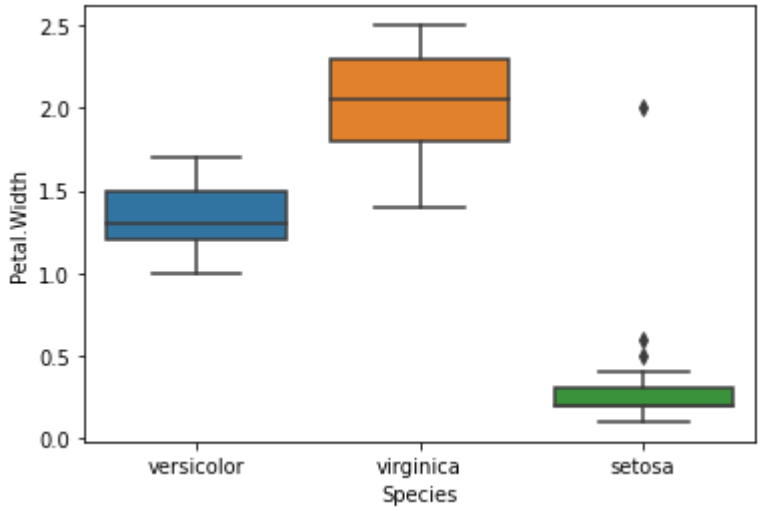
In [ ]:
```
sns.boxplot(x=df['Species'],y=df['Petal.Length'])
```

Out[ ]: <matplotlib.axes._subplots.AxesSubplot at 0x7f1ab3bed590>



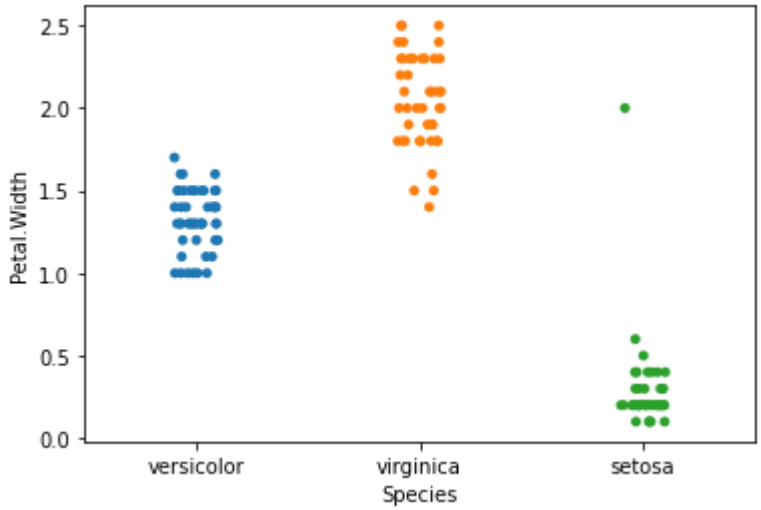
In [ ]:
```
sns.boxplot(x=df['Species'],y=df['Petal.Width'])
```

Out[ ]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f1ab3d61690>
```



In [ ]:
```
sns.stripplot(x=df['Species'],y=df['Petal.Width'],data=df)
```

Out[ ]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f1ab3985910>
```



In [141]:
```
from google.colab import drive
drive.mount('/content/drive')
!cp  "/content/drive/MyDrive/Colab Notebooks/Practical 2.ipynb" ./
!jupyter nbconvert --to html "Practical 2.ipynb"
```

```
Drive already mounted at /content/drive; to attempt to forcibly remount, call
drive.mount("/content/drive", force_remount=True).
cp: '/content/drive/MyDrive/Colab Notebooks/Practical 2.ipynb' and './Practic
al 2.ipynb' are the same file
[NbConvertApp] Converting notebook Practical 2.ipynb to html
[NbConvertApp] Writing 374112 bytes to Practical 2.html
```