In [5]:
```
import numpy as np # linear algebra
import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)
```

In [6]:
```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
#plt.style.use('fivethirtyeight')
import warnings
warnings.filterwarnings('ignore')  #this will ignore the warnings.it wont displa
```

Importing Iris Data set

In [8]:
```
iris=pd.read_csv(r"D:\NIT Daily Task\Sep\5th, 6th - Sql workshop\5th, 6th - Sql
```

In [9]:
```
iris
```

Out[9]:

	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species
0	1	5.1	3.5	1.4	0.2	Iris-setosa
1	2	4.9	3.0	1.4	0.2	Iris-setosa
2	3	4.7	3.2	1.3	0.2	Iris-setosa
3	4	4.6	3.1	1.5	0.2	Iris-setosa
4	5	5.0	3.6	1.4	0.2	Iris-setosa
...	...	...	...	...	...	...
145	146	6.7	3.0	5.2	2.3	Iris-virginica
146	147	6.3	2.5	5.0	1.9	Iris-virginica
147	148	6.5	3.0	5.2	2.0	Iris-virginica
148	149	6.2	3.4	5.4	2.3	Iris-virginica
149	150	5.9	3.0	5.1	1.8	Iris-virginica

150 rows × 6 columns

Displaying Data

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

Out[11]:

	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species
0	1	5.1	3.5	1.4	0.2	Iris-setosa
1	2	4.9	3.0	1.4	0.2	Iris-setosa
2	3	4.7	3.2	1.3	0.2	Iris-setosa
3	4	4.6	3.1	1.5	0.2	Iris-setosa
4	5	5.0	3.6	1.4	0.2	Iris-setosa

In [12]:
```
iris.drop('Id',axis=1,inplace=True)
```

In [13]:
```
iris.head()
```

Out[13]:

	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species
0	5.1	3.5	1.4	0.2	Iris-setosa
1	4.9	3.0	1.4	0.2	Iris-setosa
2	4.7	3.2	1.3	0.2	Iris-setosa
3	4.6	3.1	1.5	0.2	Iris-setosa
4	5.0	3.6	1.4	0.2	Iris-setosa

Checking if there are missing values

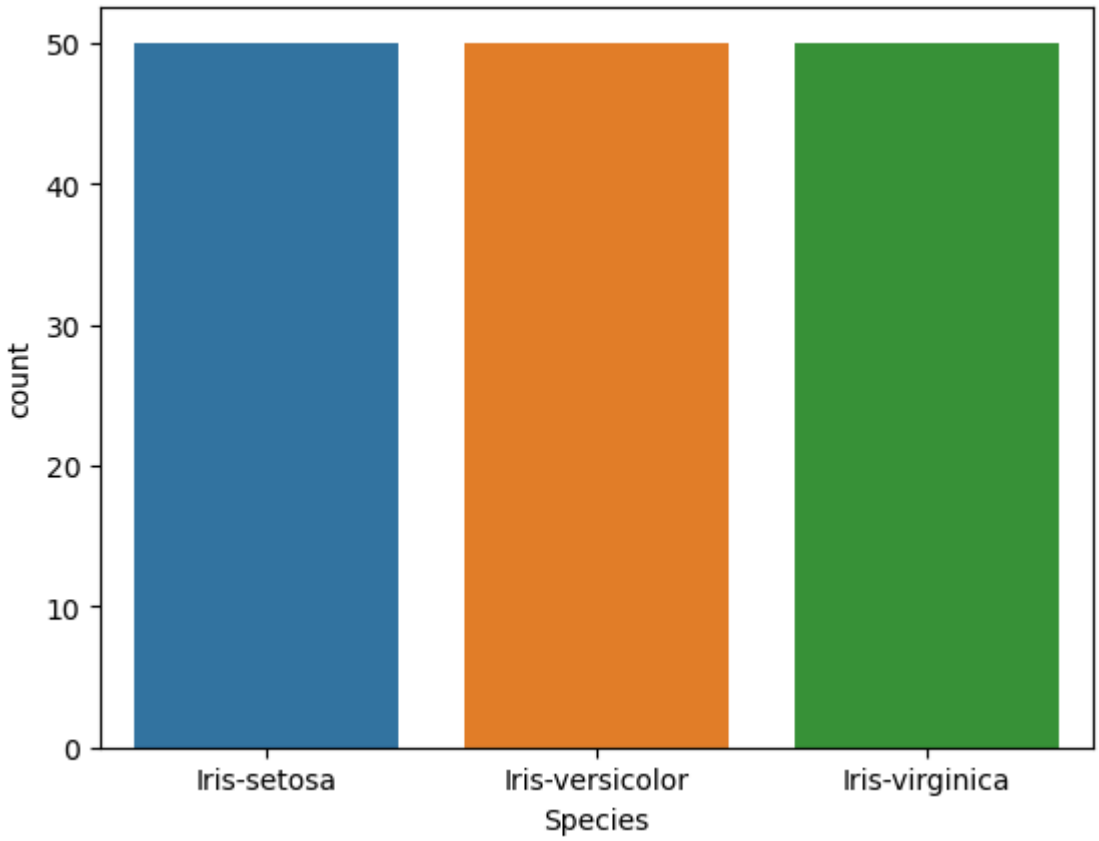
In [15]:
```
iris.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 150 entries, 0 to 149
Data columns (total 5 columns):
 #   Column         Non-Null Count  Dtype  
---  ------         --------------  -----  
 0   SepalLengthCm  150 non-null    float64
 1   SepalWidthCm   150 non-null    float64
 2   PetalLengthCm  150 non-null    float64
 3   PetalWidthCm   150 non-null    float64
 4   Species        150 non-null    object 
dtypes: float64(4), object(1)
memory usage: 6.0+ KB
```

In [16]:
```
iris['Species'].value_counts()
```

Out[16]:
```
Species
Iris-setosa        50
Iris-versicolor    50
Iris-virginica     50
Name: count, dtype: int64
```

In [17]:
```
sns.countplot(x='Species',hue='Species',data=iris)
plt.show()
```

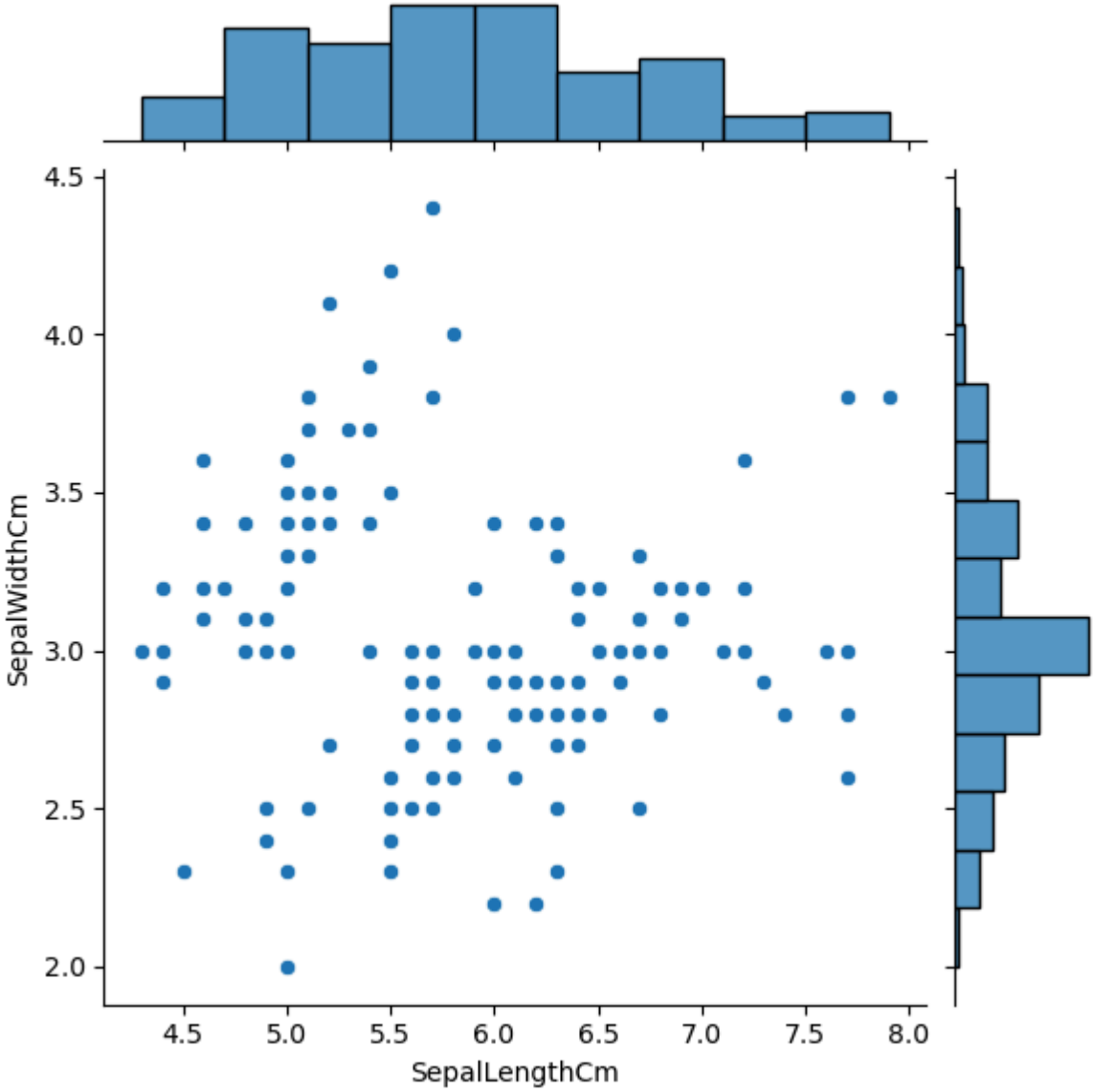


In [18]:
```
iris.head()
```

Out[18]:

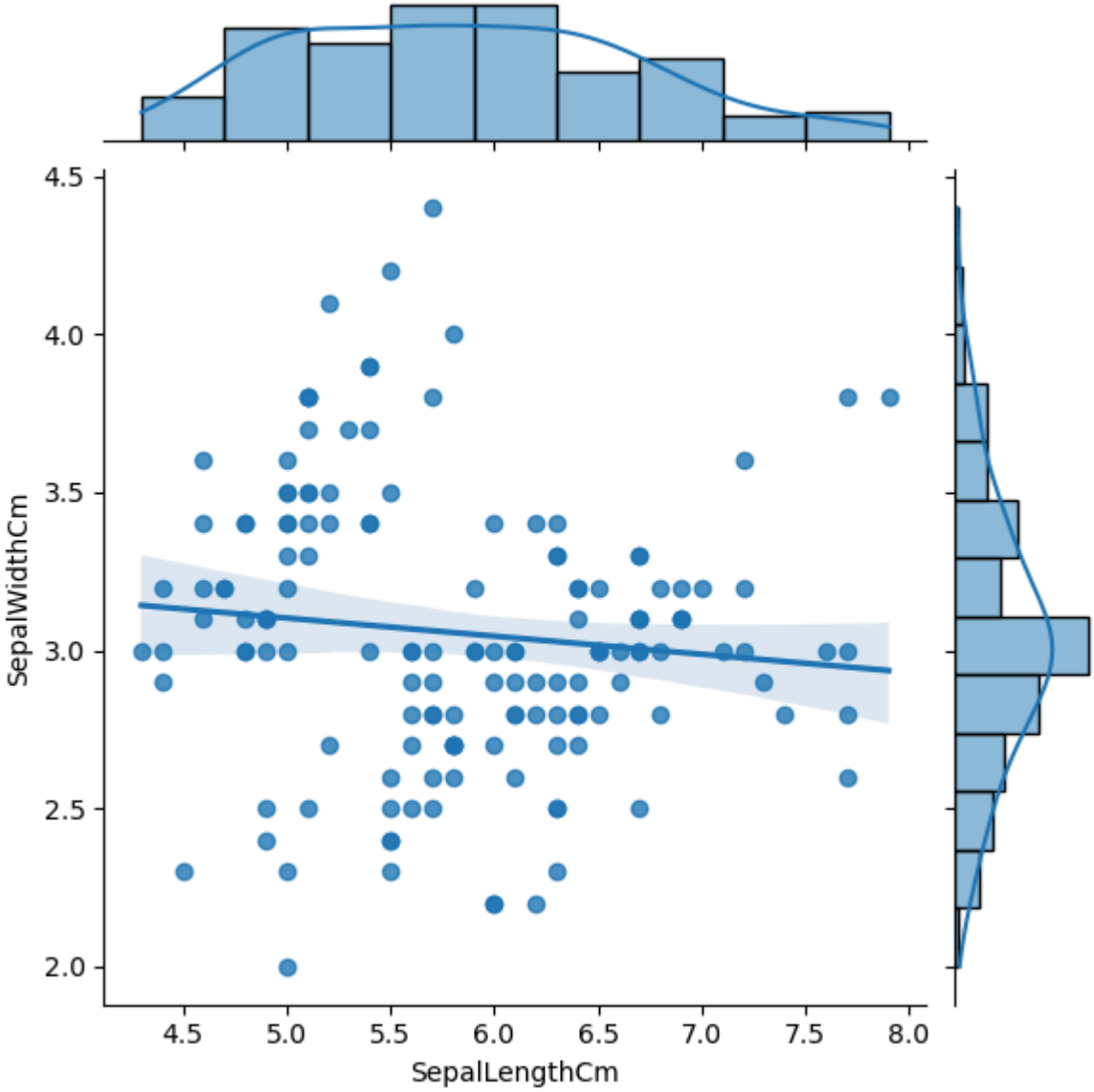
	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species
0	5.1	3.5	1.4	0.2	Iris-setosa
1	4.9	3.0	1.4	0.2	Iris-setosa
2	4.7	3.2	1.3	0.2	Iris-setosa
3	4.6	3.1	1.5	0.2	Iris-setosa
4	5.0	3.6	1.4	0.2	Iris-setosa

In [19]:
```
fig=sns.jointplot(x='SepalLengthCm',y='SepalWidthCm',data=iris)
```

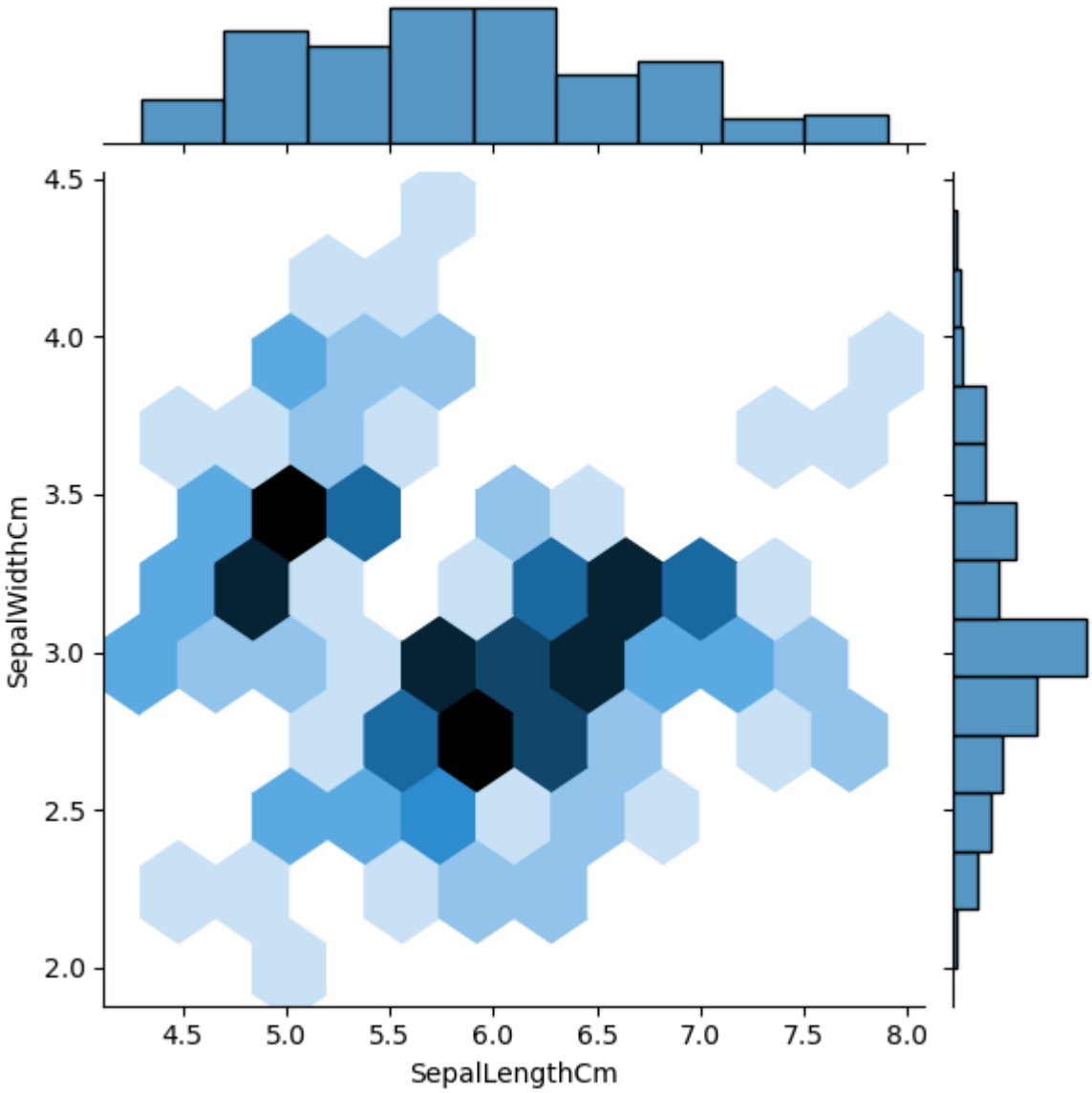


In [20]:
```
sns.jointplot(x="SepalLengthCm", y="SepalWidthCm", data=iris, kind="reg")
```

Out[20]:
```
<seaborn.axisgrid.JointGrid at 0x1733c3f58b0>
```



In [21]:
```
fig=sns.jointplot(x='SepalLengthCm',y='SepalWidthCm',kind='hex',data=iris)
```

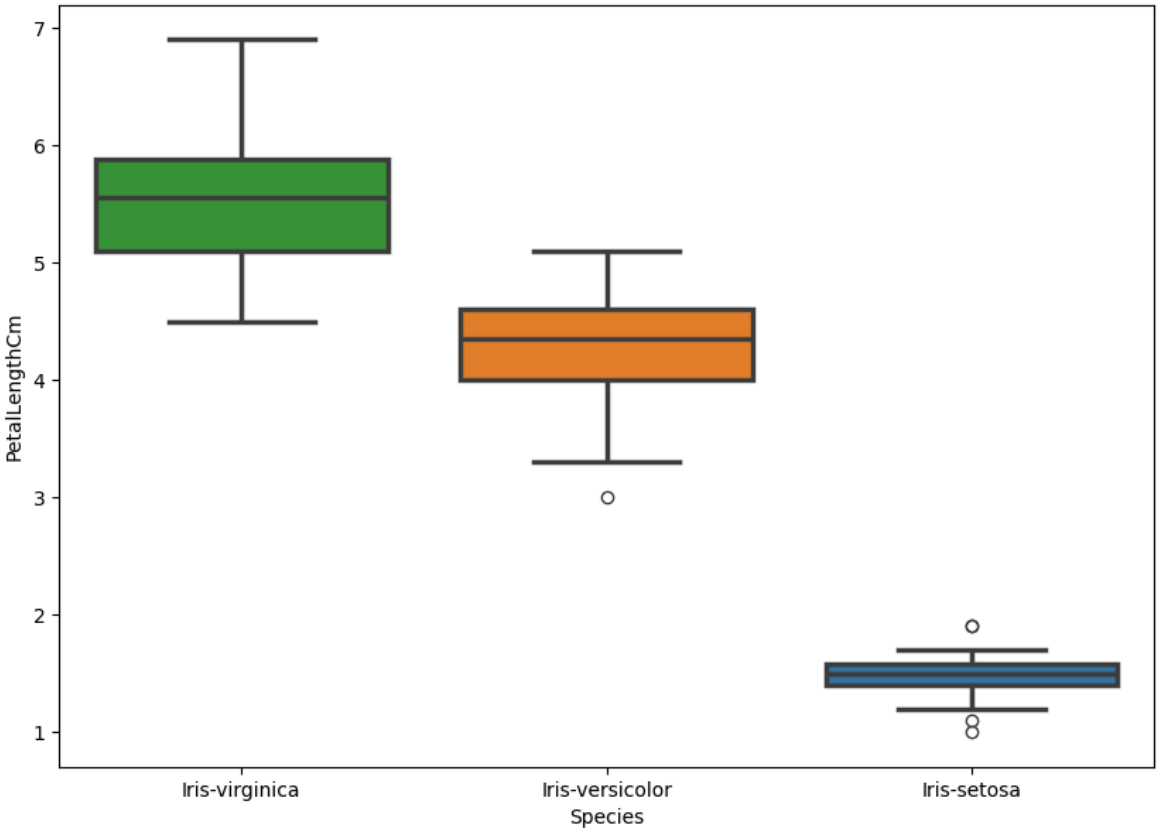


In [23]:
```
iris.head()
```

Out[23]:

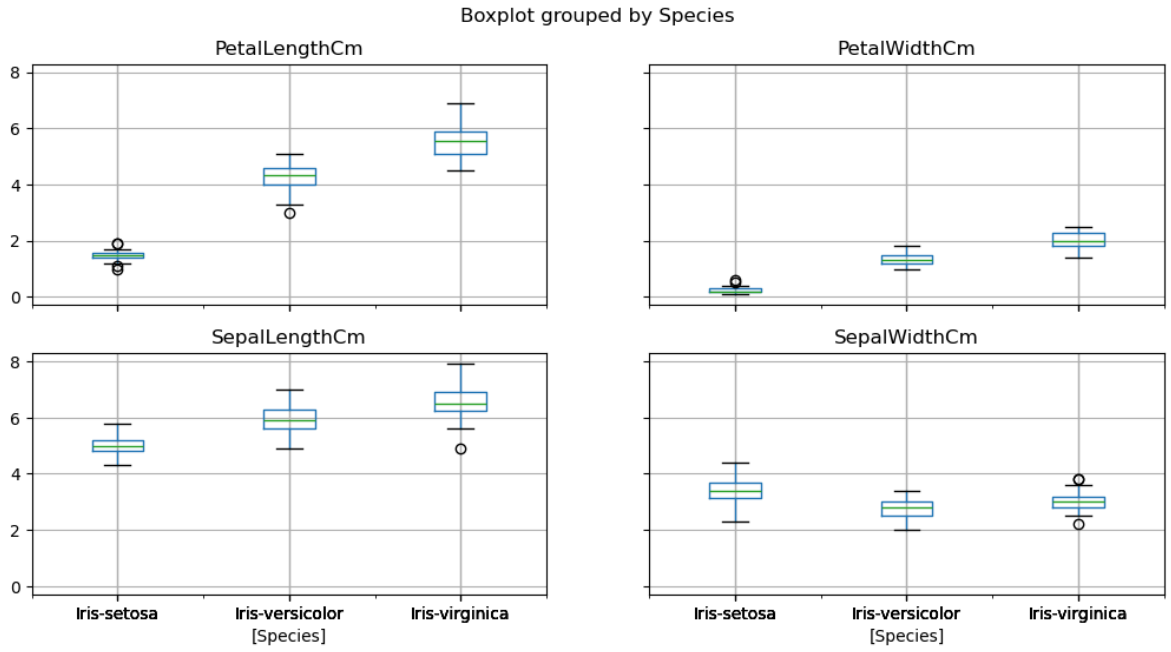
	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species
0	5.1	3.5	1.4	0.2	Iris-setosa
1	4.9	3.0	1.4	0.2	Iris-setosa
2	4.7	3.2	1.3	0.2	Iris-setosa
3	4.6	3.1	1.5	0.2	Iris-setosa
4	5.0	3.6	1.4	0.2	Iris-setosa

In [27]:
```
fig=plt.gcf()
fig.set_size_inches(10,7)
fig=sns.boxplot(x='Species',y='PetalLengthCm',hue='Species',data=iris,order=['Ir
```



In [29]:
```
#iris.drop("Id", axis=1).boxplot(by="Species", figsize=(12, 6))
iris.boxplot(by="Species", figsize=(12, 6))
```

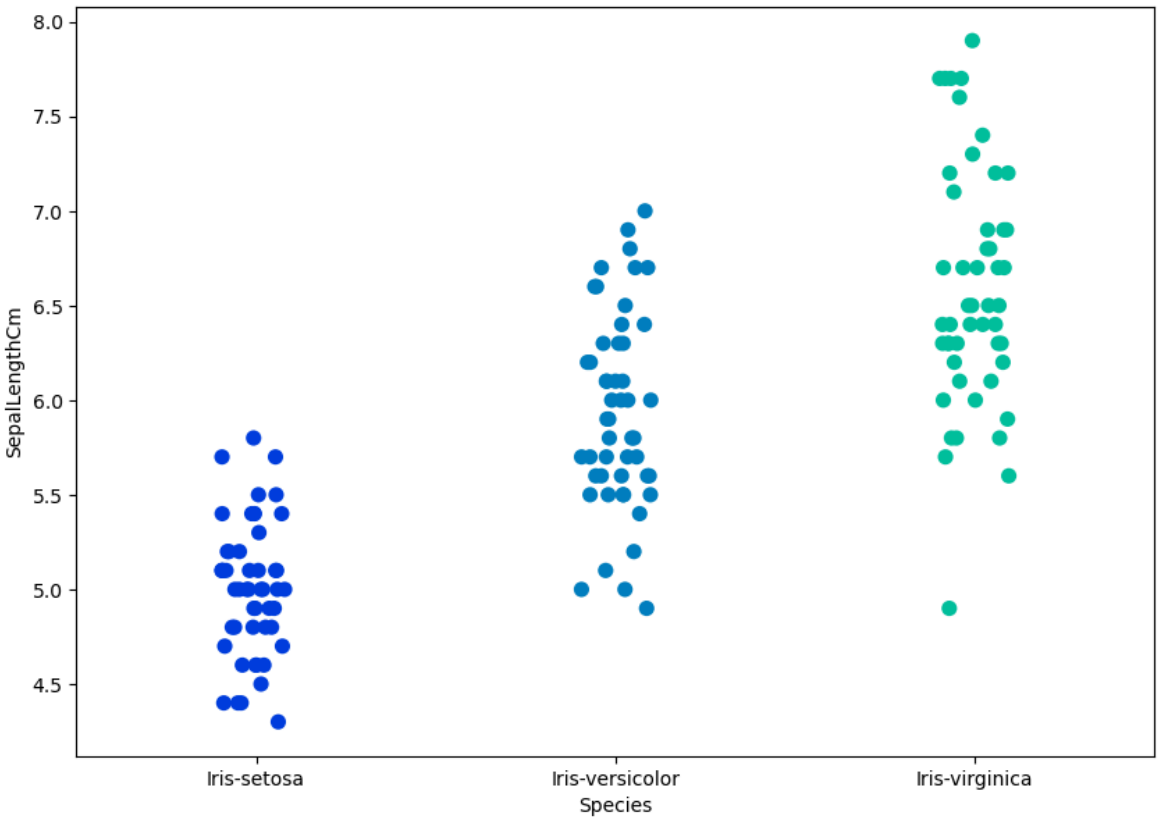
Out[29]:
```
array([[<Axes: title={'center': 'PetalLengthCm'}, xlabel='[Species]'>,
        <Axes: title={'center': 'PetalWidthCm'}, xlabel='[Species]'>],
       [<Axes: title={'center': 'SepalLengthCm'}, xlabel='[Species]'>,
        <Axes: title={'center': 'SepalWidthCm'}, xlabel='[Species]'>]],
      dtype=object)
```



Strip Plot

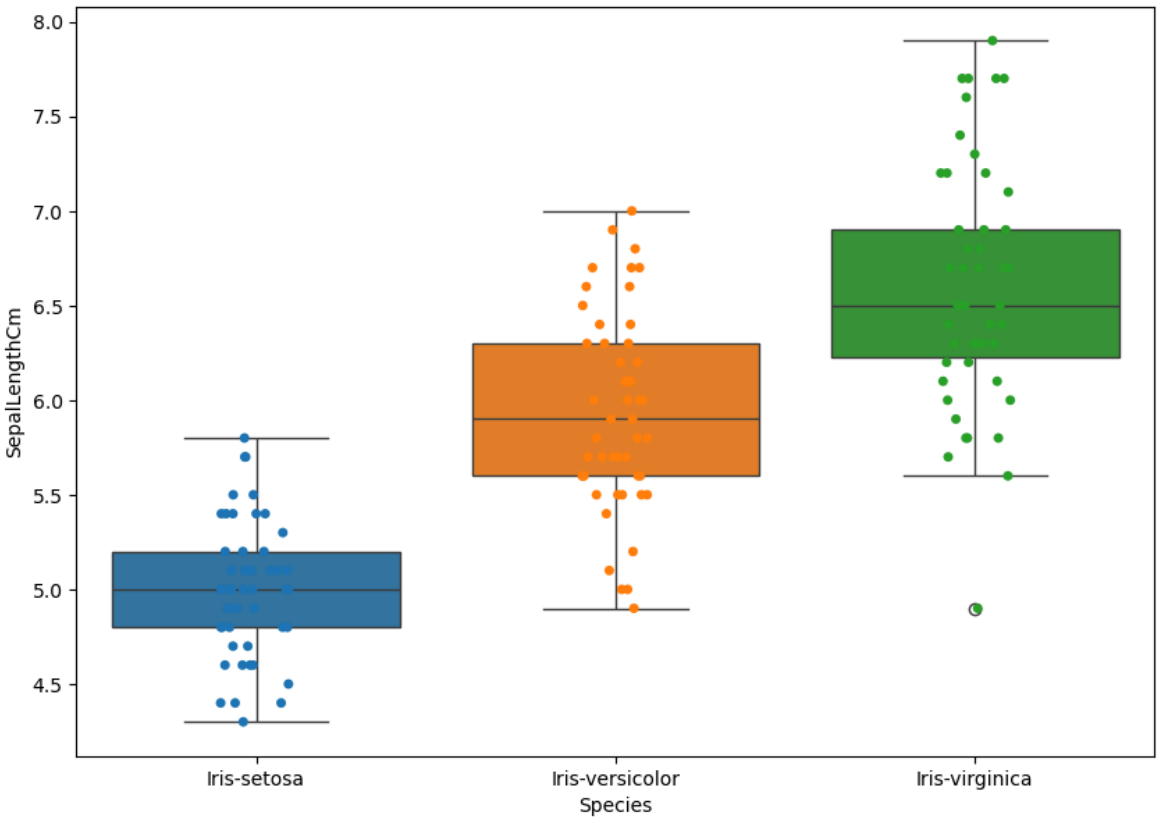
In [32]:
```
fig=plt.gcf()
fig.set_size_inches(10,7)
```

```
fig=sns.stripplot(x='Species',y='SepalLengthCm',data=iris,jitter=True,edgecolor=
```



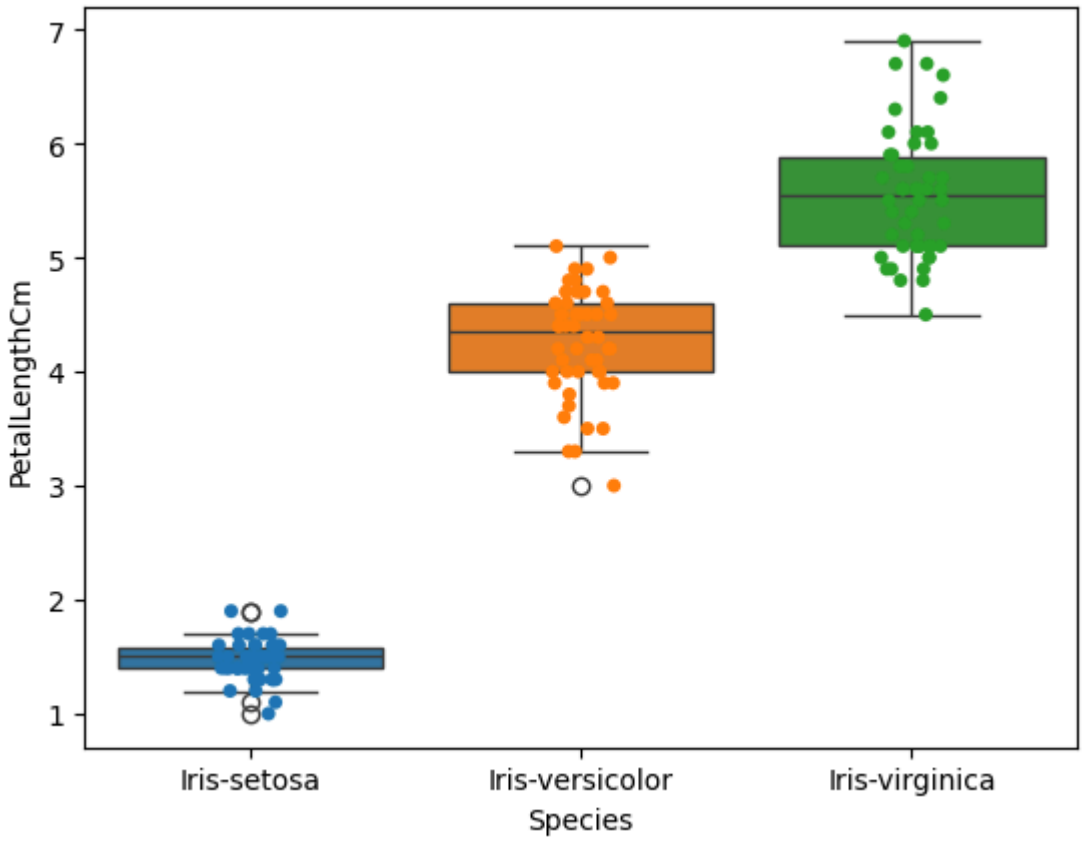
Combining Box and Strip Plots

In [37]:
```
fig=plt.gcf()
fig.set_size_inches(10,7)
fig=sns.boxplot(x='Species',y='SepalLengthCm',hue='Species',data=iris)
fig=sns.stripplot(x='Species',y='SepalLengthCm',hue='Species',data=iris,jitter=T
```



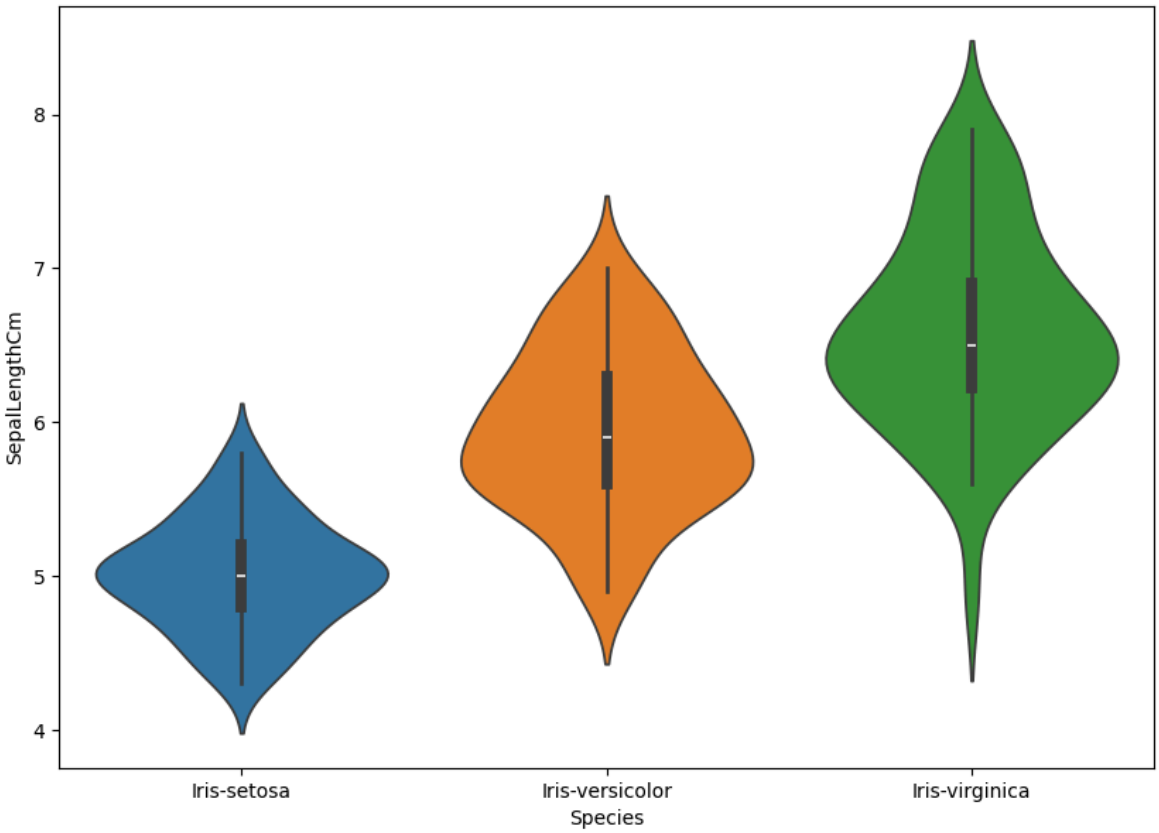
In [41]:
```
ax= sns.boxplot(x="Species", y="PetalLengthCm",hue='Species', data=iris)
ax= sns.stripplot(x="Species", y="PetalLengthCm",hue='Species', data=iris, jitte

boxtwo = ax.artists[2]
boxtwo.set_facecolor('yellow')
boxtwo.set_edgecolor('black')
boxthree=ax.artists[1]
boxthree.set_facecolor('red')
boxthree.set_edgecolor('black')
boxthree=ax.artists[0]
boxthree.set_facecolor('green')
boxthree.set_edgecolor('black')
```

```
---------------------------------------------------------------------------
IndexError                                Traceback (most recent call last)
Cell In[41], line 4
      1 ax= sns.boxplot(x="Species", y="PetalLengthCm",hue='Species', data=iris)
      2 ax= sns.stripplot(x="Species", y="PetalLengthCm",hue='Species', data=iri
s, jitter=True, edgecolor="gray")
----> 4 boxtwo = ax.artists[2]
      5 boxtwo.set_facecolor('yellow')
      6 boxtwo.set_edgecolor('black')

File ~\anaconda3\Lib\site-packages\matplotlib\axes\_base.py:1450, in _AxesBase.Ar
tistList.__getitem__(self, key)
   1449 def __getitem__(self, key):
-> 1450     return [artist
   1451             for artist in self._axes._children
   1452             if self._type_check(artist)][key]

IndexError: list index out of range
```



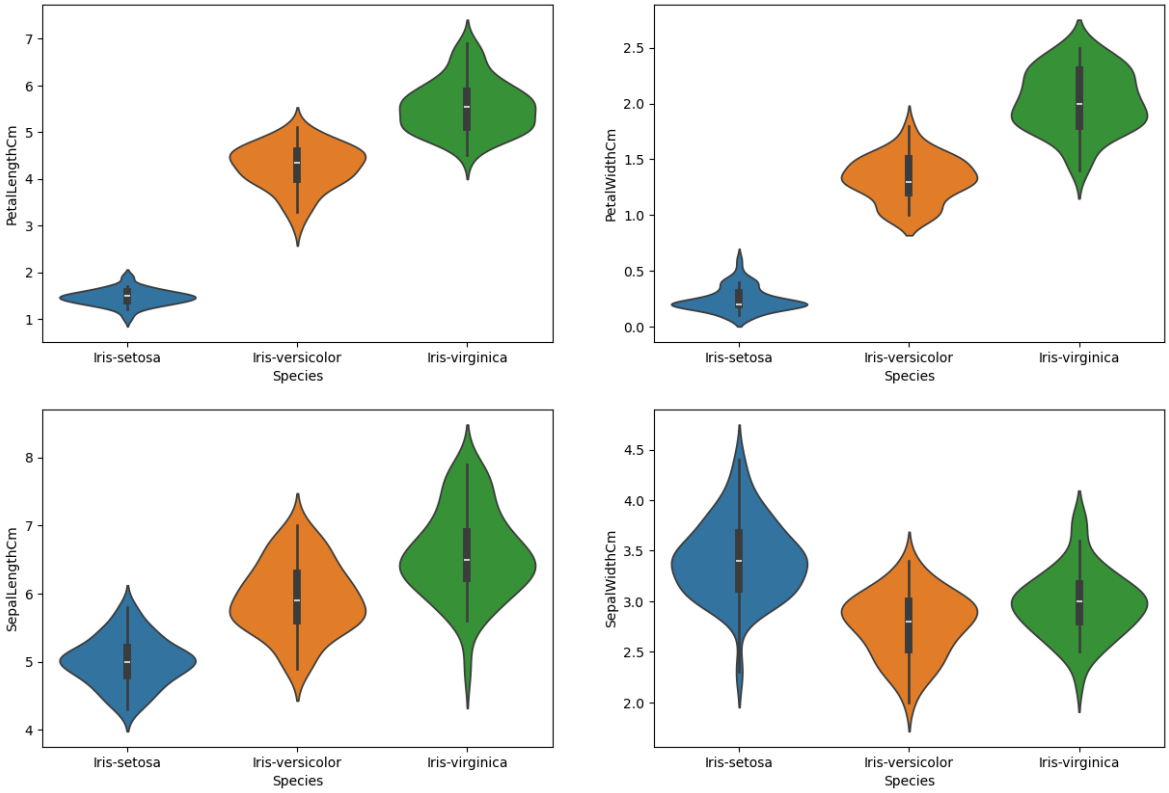
9. Violin Plot It is used to visualize the distribution of data and its probability distribution.This chart is a combination of a Box Plot and a Density Plot that is rotated and placed on each side, to show the distribution shape of the data. The thick black bar in the centre represents the interquartile range, the thin black line extended from it represents the 95% confidence intervals, and the white dot is the median.Box Plots are limited in their display of the data, as their visual simplicity tends to hide significant details about how values in the data are distributed

In [44]:
```
fig=plt.gcf()
fig.set_size_inches(10,7)
fig=sns.violinplot(x='Species',y='SepalLengthCm',hue='Species',data=iris)
```



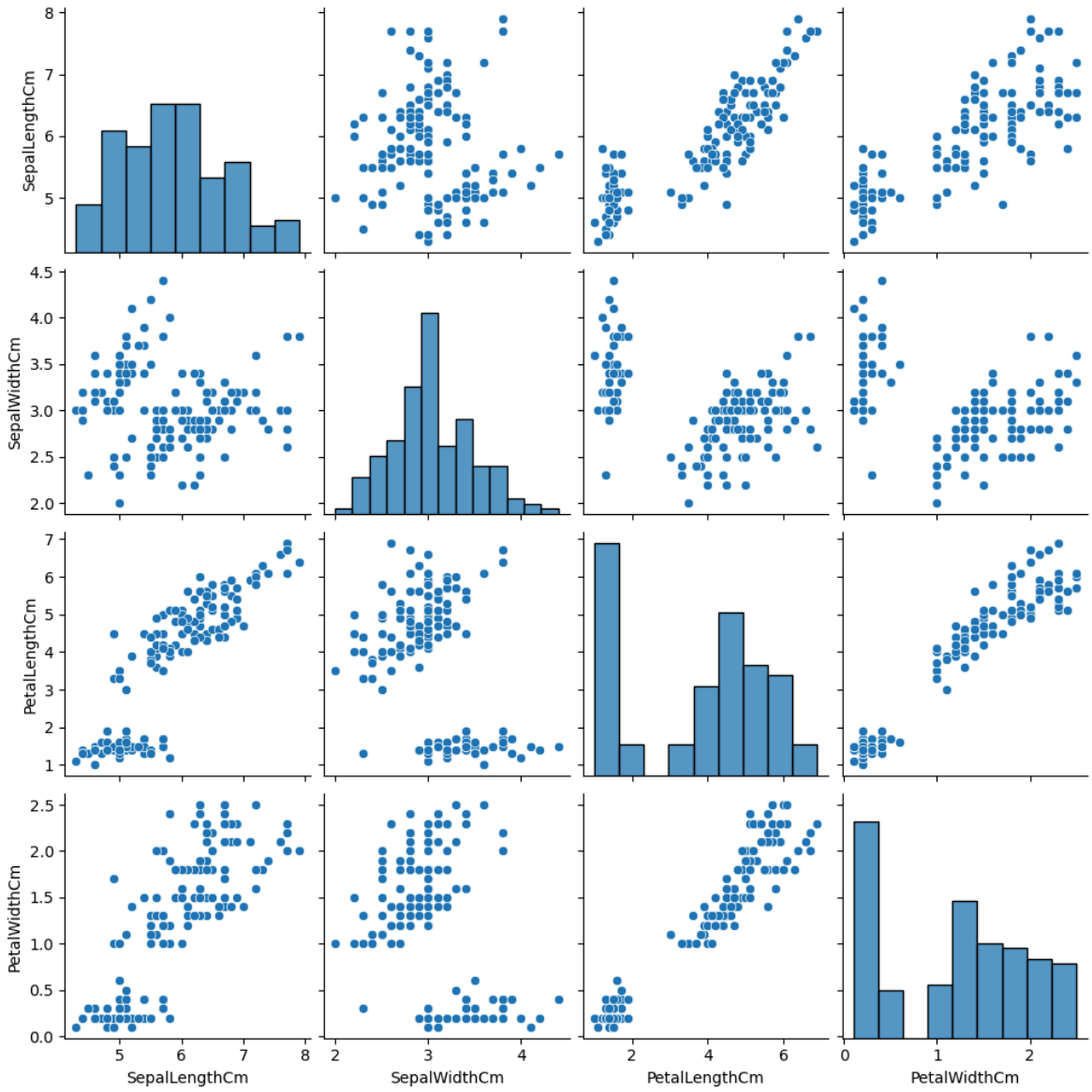
In [48]:
```
plt.figure(figsize=(15,10))
plt.subplot(2,2,1)
sns.violinplot(x='Species',y='PetalLengthCm',hue='Species',data=iris)
plt.subplot(2,2,2)
sns.violinplot(x='Species',y='PetalWidthCm',hue='Species',data=iris)
plt.subplot(2,2,3)
sns.violinplot(x='Species',y='SepalLengthCm',hue='Species',data=iris)
plt.subplot(2,2,4)
sns.violinplot(x='Species',y='SepalWidthCm',hue='Species',data=iris)
```

Out[48]: <Axes: xlabel='Species', ylabel='SepalWidthCm'>

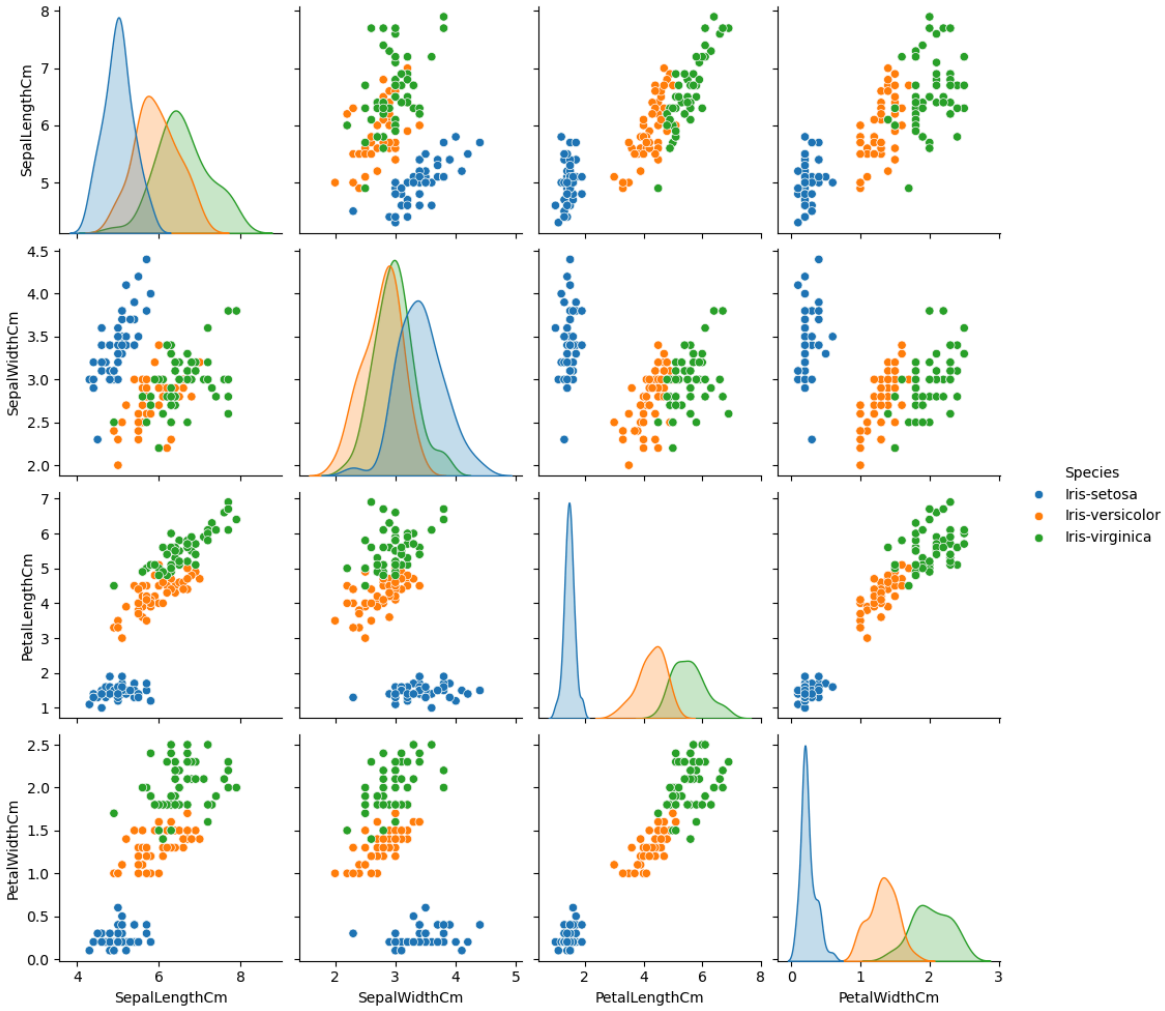


In [50]:
```
sns.pairplot(data=iris,kind='scatter')
```

Out[50]:
```
<seaborn.axisgrid.PairGrid at 0x17347d77cb0>
```



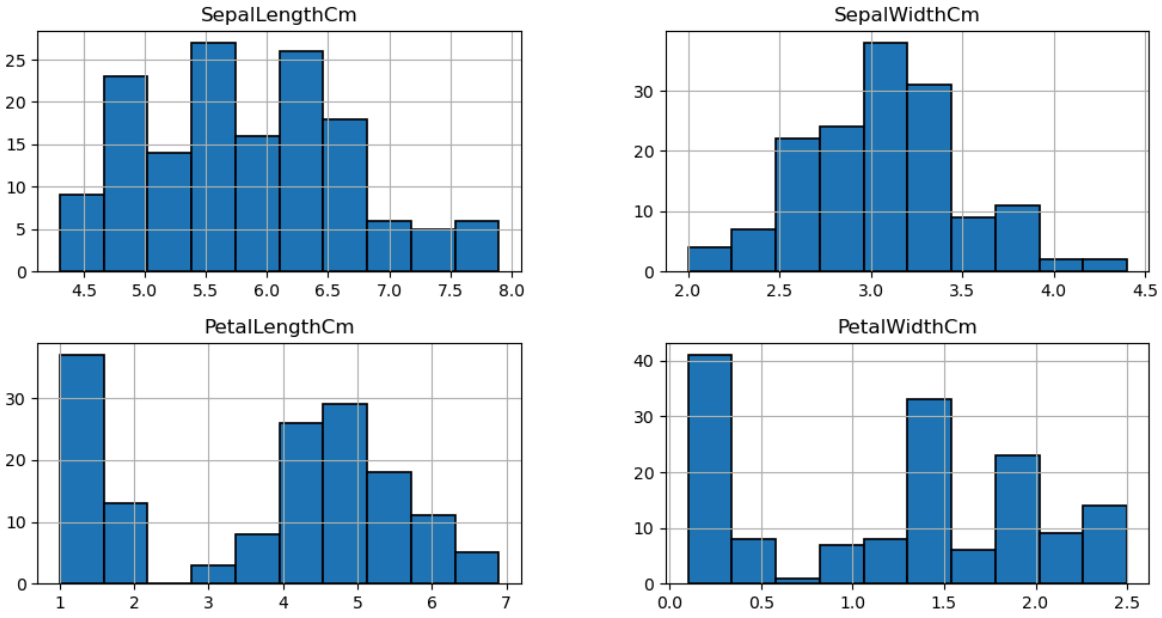
In [52]:
```
sns.pairplot(iris,hue='Species');
```



11. Heat map Heat map is used to find out the correlation between different features in the dataset.High positive or negative value shows that the features have high correlation.This helps us to select the parmeters for machine learning.

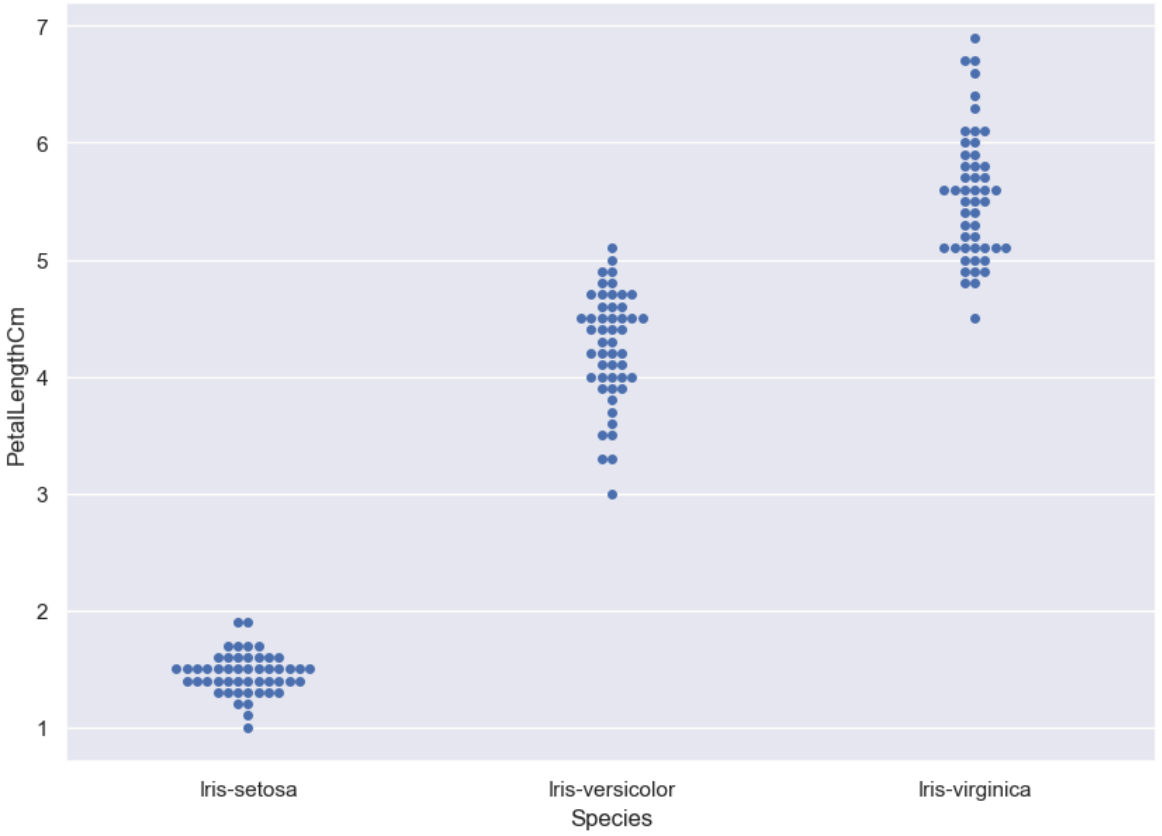
12. Distribution plot: The distribution plot is suitable for comparing range and distribution for groups of numerical data. Data is plotted as value points along an axis. You can choose to display only the value points to see the distribution of values, a bounding box to see the range of values, or a combination of both as shown here.The distribution plot is not relevant for detailed analysis of the data as it deals with a summary of the data distribution.

In [60]:
```
iris.hist(edgecolor='black', linewidth=1.2)
fig=plt.gcf()
fig.set_size_inches(12,6)
```



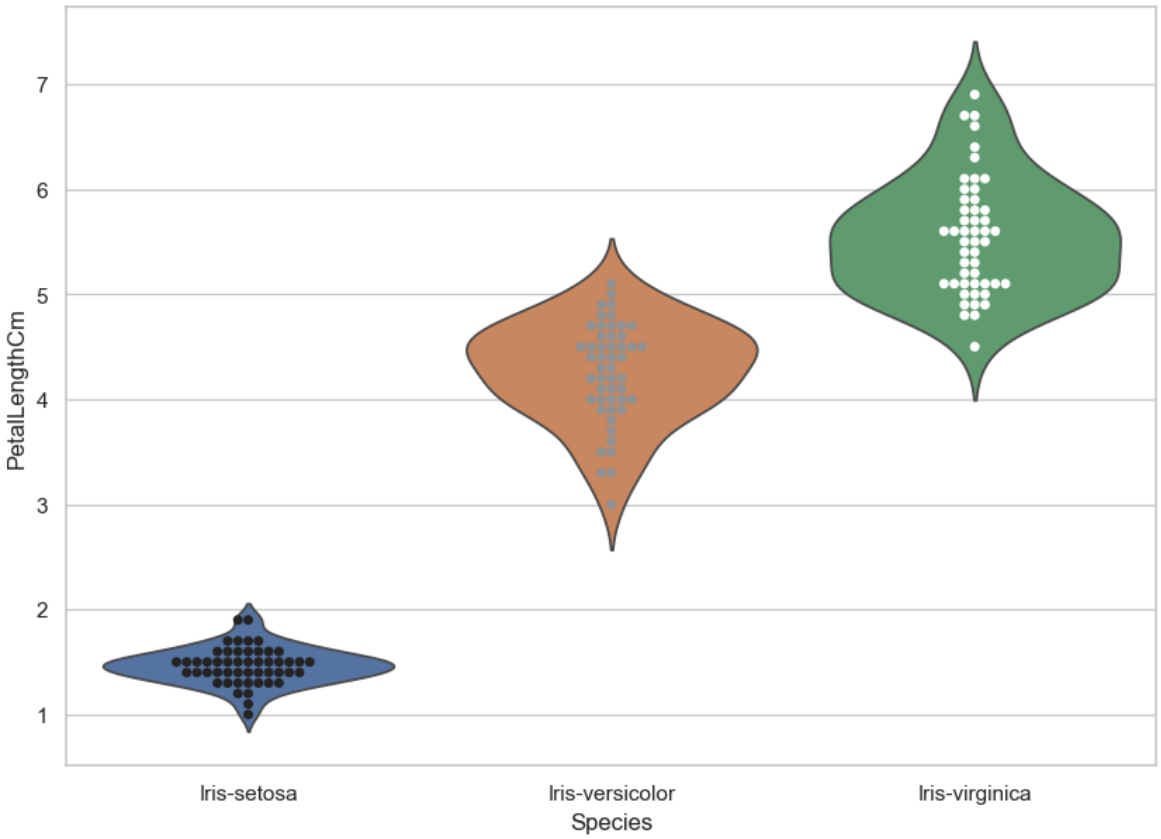
13. Swarm plot It looks a bit like a friendly swarm of bees buzzing about their hive. More importantly, each data point is clearly visible and no data are obscured by overplotting.A beeswarm plot improves upon the random jittering approach to move data points the minimum distance away from one another to avoid overlays. The result is a plot where you can see each distinct data point, like shown in below plot

In [63]:
```
sns.set(style="darkgrid")
fig=plt.gcf()
fig.set_size_inches(10,7)
fig = sns.swarmplot(x="Species", y="PetalLengthCm", data=iris)
```



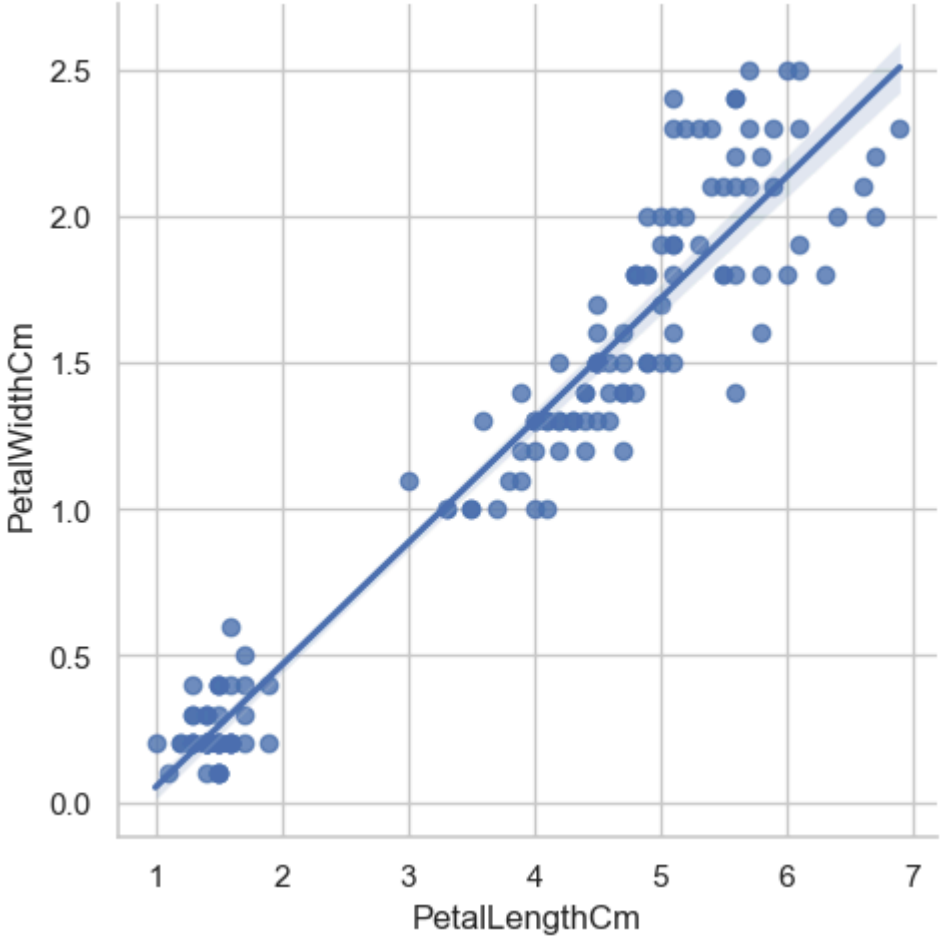
In [67]:
```
sns.set(style="whitegrid")
fig=plt.gcf()
```

```
fig.set_size_inches(10,7)
ax = sns.violinplot(x="Species", y="PetalLengthCm",hue='Species', data=iris, inn
ax = sns.swarmplot(x="Species", y="PetalLengthCm",hue='Species', data=iris,color
```



17.LM Plot

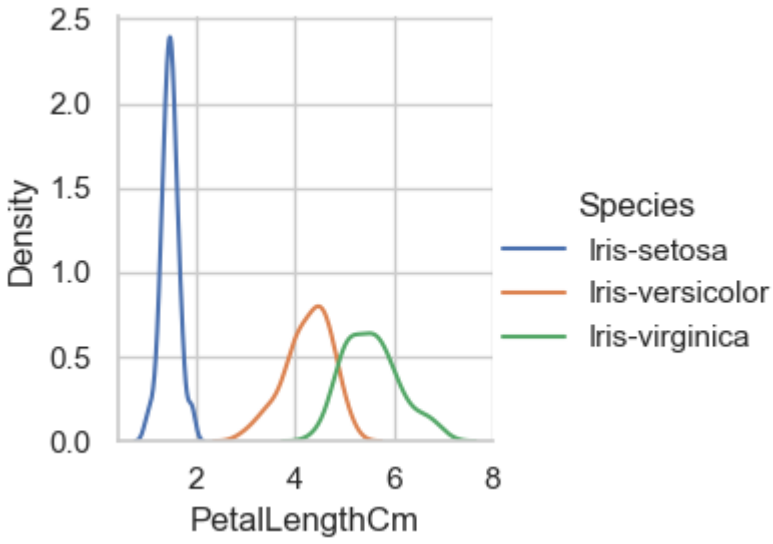
In [69]:
```
fig=sns.lmplot(x="PetalLengthCm", y="PetalWidthCm",data=iris)
```



18. FacetGrid

In [75]:
```
sns.FacetGrid(iris, hue="Species") \
   .map(sns.kdeplot, "PetalLengthCm") \
   .add_legend()
plt.ioff()
```

Out[75]: <contextlib.ExitStack at 0x1734ff23bc0>



** 22. Factor Plot **

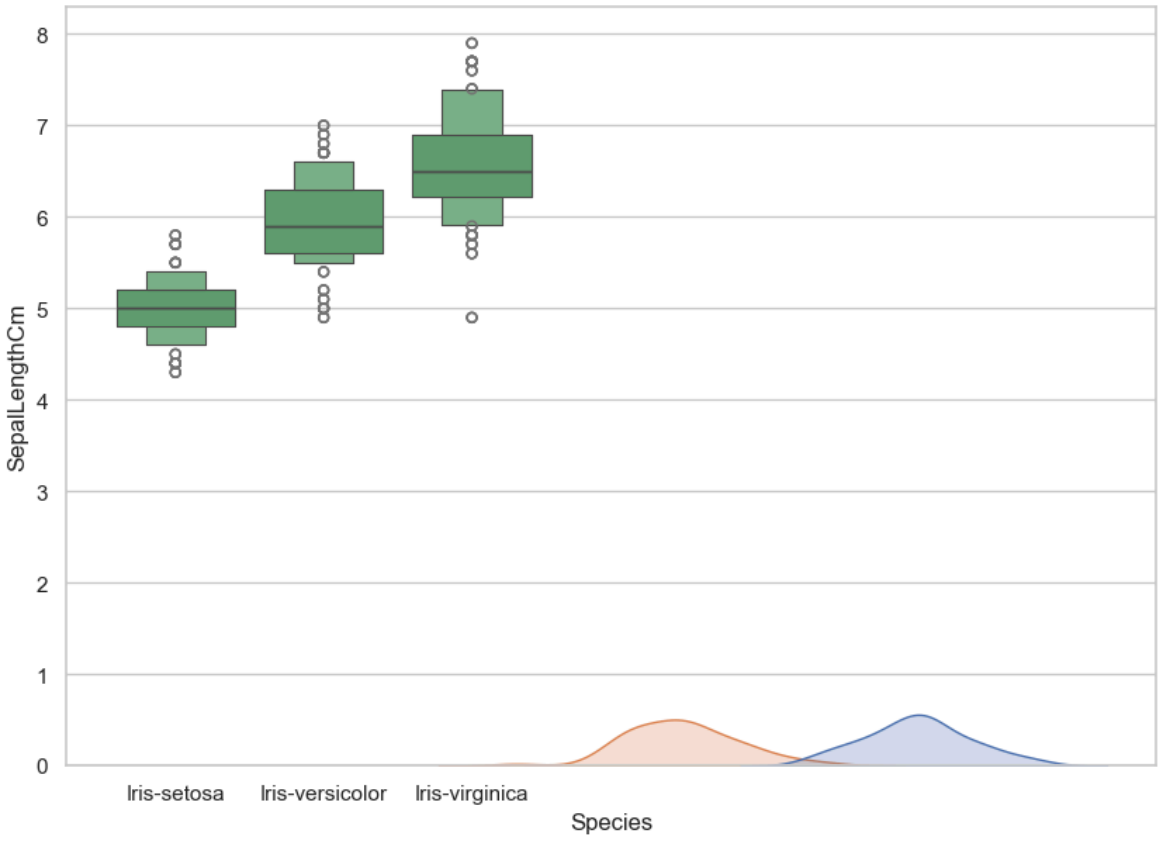
23. Boxen Plot

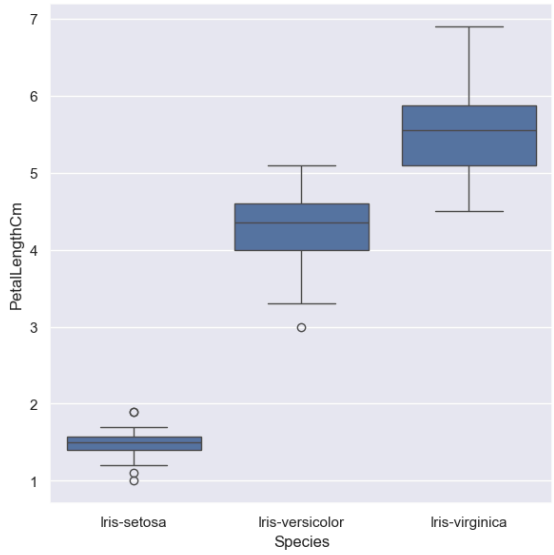
In [86]:
```
fig=plt.gcf()
fig.set_size_inches(10,7)
fig=sns.boxenplot(x='Species',y='SepalLengthCm',data=iris)
```

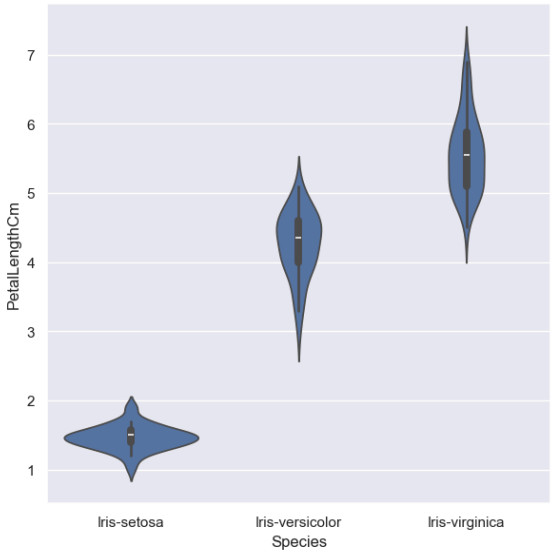
24.KDE Plot

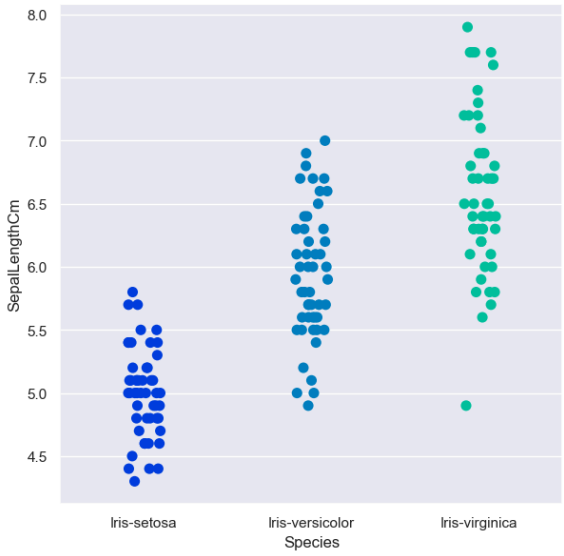
25.Dashboard

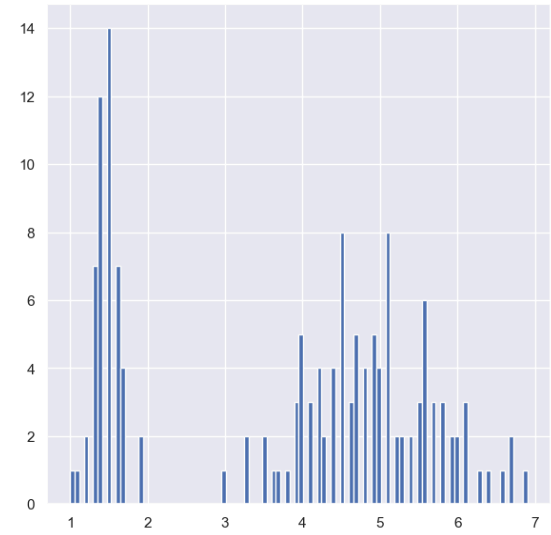
In [94]:
```
sns.set_style('darkgrid')
f,axes=plt.subplots(2,2,figsize=(15,15))

k1=sns.boxplot(x="Species", y="PetalLengthCm", data=iris,ax=axes[0,0])
k2=sns.violinplot(x='Species',y='PetalLengthCm',data=iris,ax=axes[0,1])
k3=sns.stripplot(x='Species',y='SepalLengthCm',data=iris,jitter=True,edgecolor='
#axes[1,1].hist(iris.hist,bin=10)
axes[1,1].hist(iris.PetalLengthCm,bins=100)
#k2.set(xlim=(-1,0.8))
plt.show()
```







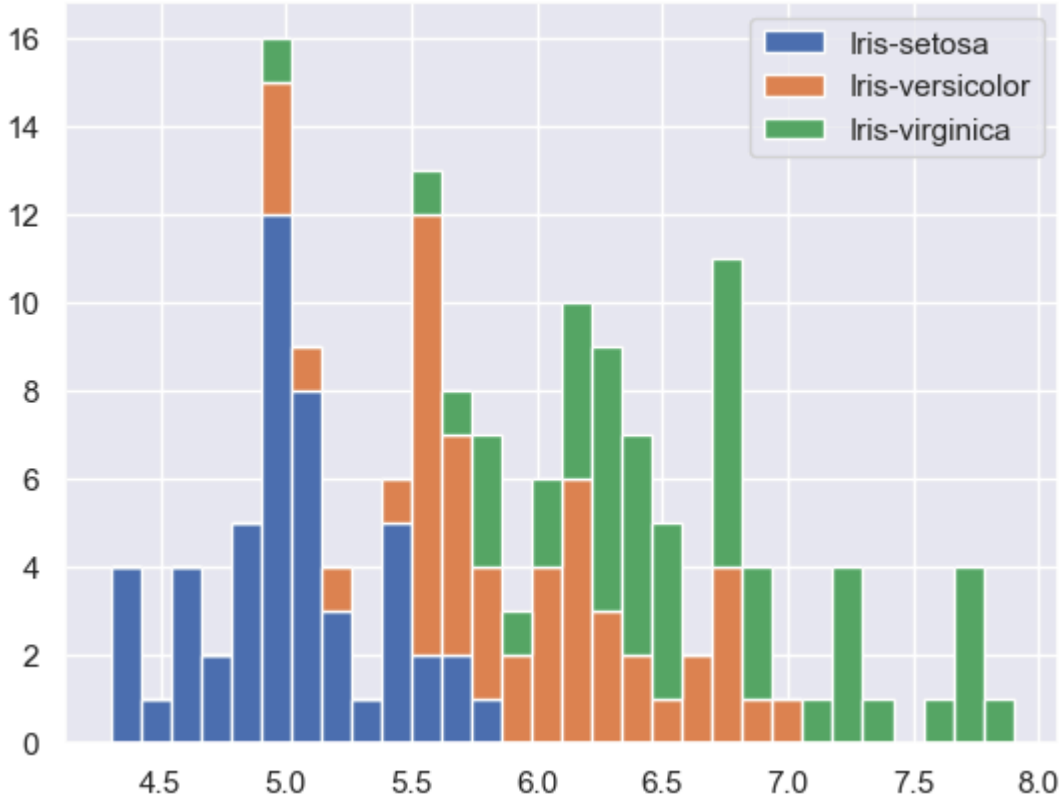




26.Stacked Histogram

In [99]:
```
iris['Species'] = iris['Species'].astype('category')
#iris.head()
```

In [101…
```
list1=list()
mylabels=list()
for gen in iris.Species.cat.categories:
    list1.append(iris[iris.Species==gen].SepalLengthCm)
    mylabels.append(gen)

h=plt.hist(list1,bins=30,stacked=True,rwidth=1,label=mylabels)
plt.legend()
plt.show()
```



27.Area Plot: Area Plot gives us a visual representation of Various dimensions of Iris flower and their range in dataset

In [104…
```
#iris['SepalLengthCm'] = iris['SepalLengthCm'].astype('category')
#iris.head()
#iris.plot.area(y='SepalLengthCm',alpha=0.4,figsize=(12, 6));
iris.plot.area(y=['SepalLengthCm','SepalWidthCm','PetalLengthCm','PetalWidthCm']
```

28.Distplot: It helps us to look at the distribution of a single variable.Kde shows the density of the distribution

In [111…
```
sns.distplot(iris['SepalLengthCm'],kde=True,bins=20);
```

THIS IS ALL ABOUT EDA COMPLETE

In [ ]:
```
```