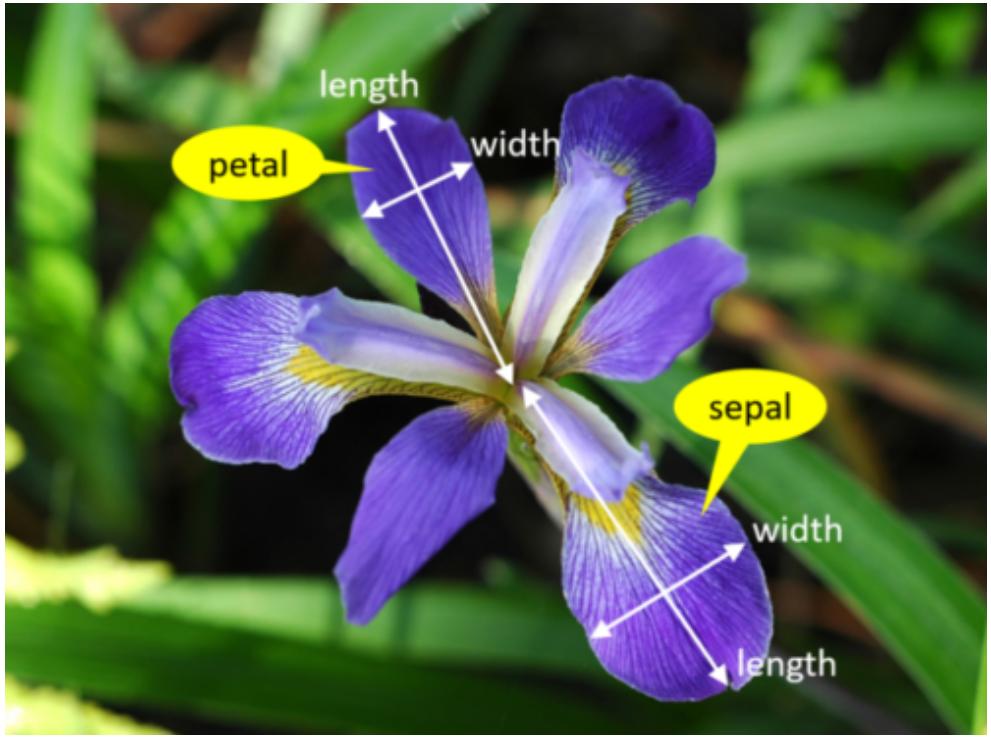
DATASET VISUALIZATION(SEABORN,MATPLOTLIB)



In [3]:
```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
```

In [4]:
```
import warnings
warnings.filterwarnings('ignore')
```

IMPORTING IRIS DATA SET

In [6]:
```
iris=pd.read_csv(r'D:\Data Science\AUG\IRIS DATASET _ ADVANCE VISUALIZATION _ EDA 2\Iris.csv')
```

In [7]:
```
iris
```

Out[7]:

	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

In [8]:
```
iris.shape # Return a tuple representing the dimensionality of the DataFrame.
```

Out[8]:
```
(150, 6)
```

In [9]:
```
iris.head()#Return the first `n` rows.
```

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

In [10]:
```
iris.tail()#Return the last `n` rows.
```

Out[10]:

	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species
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

In [11]:
```
iris.drop('Id',axis=1,inplace=True)#Drop specified labels from rows or columns.drop th Id attribute
```

In [12]:
```
iris.head(3)
```

Out[12]:

	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

In [13]:
```
iris.columns
```

Out[13]:
```
Index(['SepalLengthCm', 'SepalWidthCm', 'PetalLengthCm', 'PetalWidthCm',
       'Species'],
      dtype='object')
```

In [14]:
```
iris.info()#This method prints information about a DataFrame including
           #the index dtype and columns, non-null values and memory usage.
```

```
<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 [15]:
```
iris['SepalLengthCm'].value_counts()
```

Out[15]:
```
SepalLengthCm
5.0    10
5.1     9
6.3     9
5.7     8
6.7     8
5.8     7
5.5     7
6.4     7
4.9     6
5.4     6
6.1     6
6.0     6
5.6     6
4.8     5
6.5     5
6.2     4
7.7     4
6.9     4
4.6     4
5.2     4
5.9     3
4.4     3
7.2     3
6.8     3
6.6     2
4.7     2
7.6     1
7.4     1
7.3     1
7.0     1
7.1     1
5.3     1
4.3     1
4.5     1
7.9     1
Name: count, dtype: int64
```

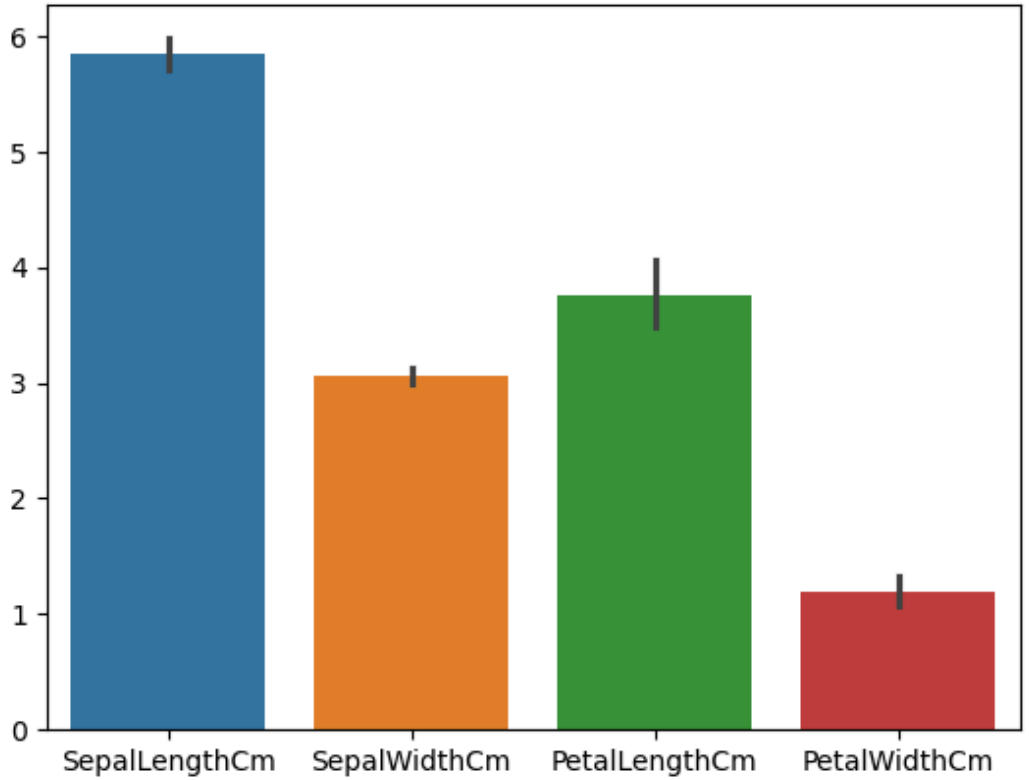
In [16]:
```
iris['Species'].value_counts()
#Return a Series containing the frequency of each distinct row in the Dataframe.
```

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

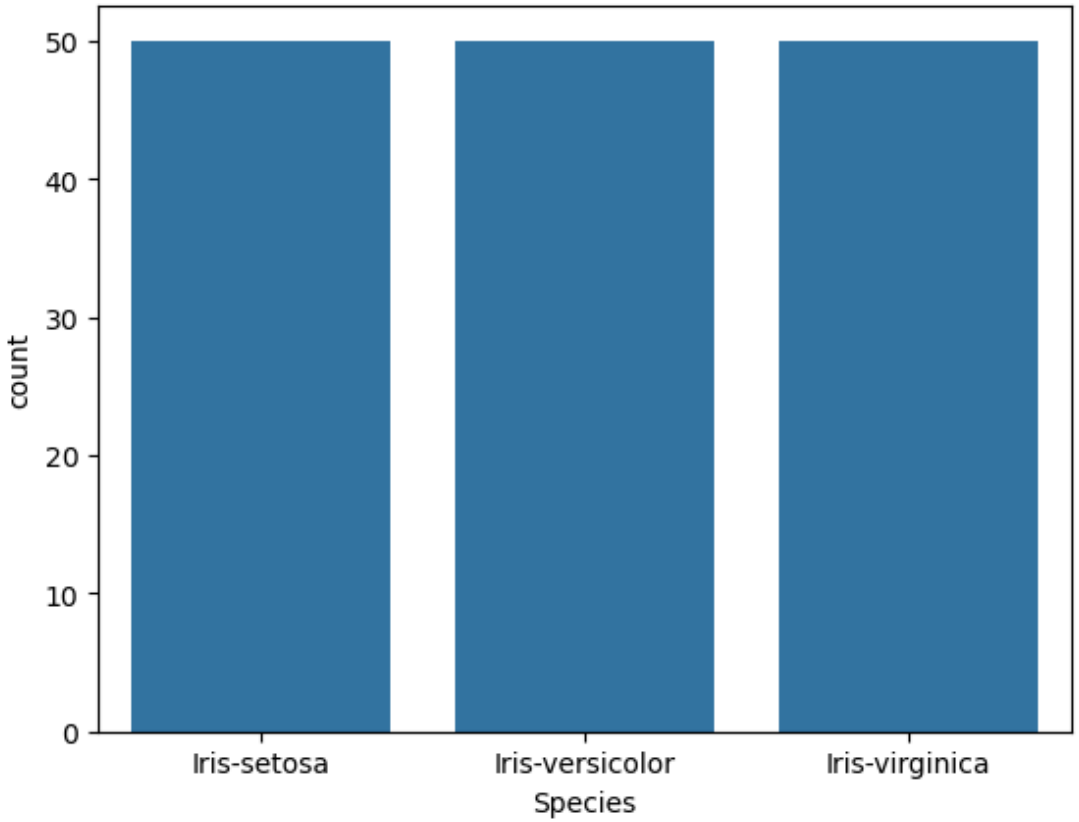
BAR PLOT

: Here the frequency of the observation is plotted.In this case we are plotting the frequency of the three species in the Iris Dataset

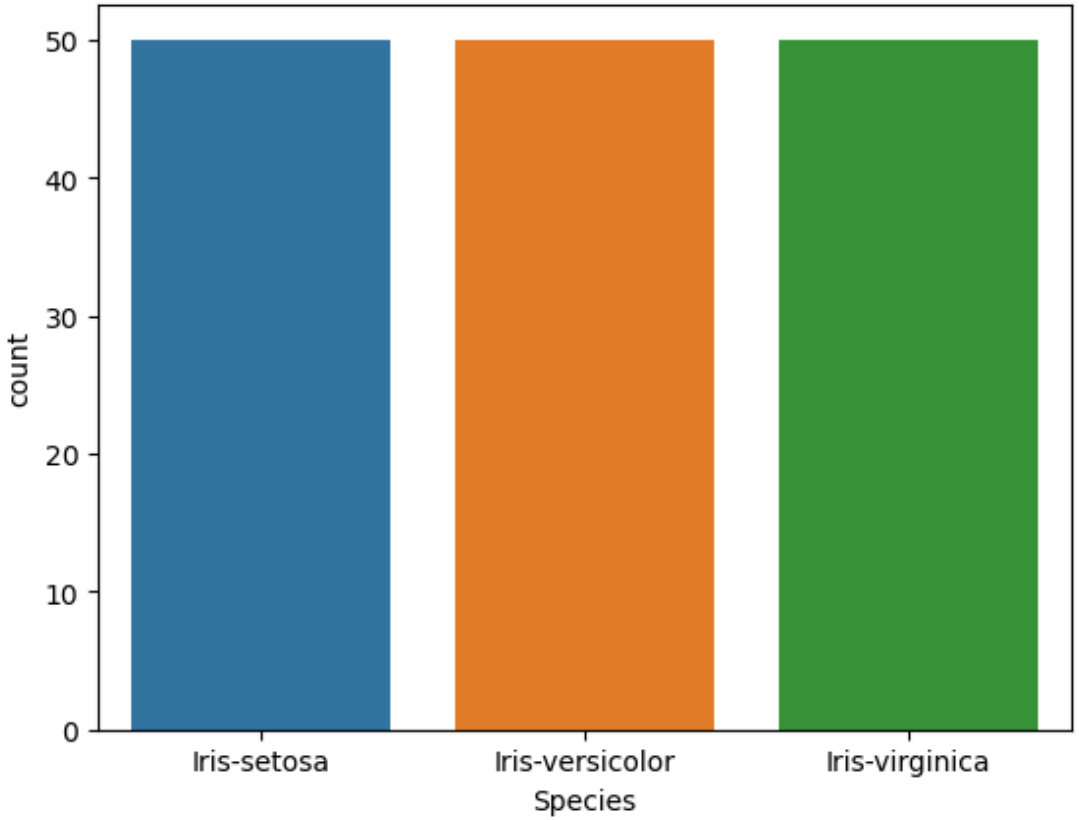
In [18]:
```
sns.barplot(data= iris,)
plt.show()#Display all open figures.
```



In [19]:
```
sns.countplot(data=iris,x='Species')
plt.show()
```



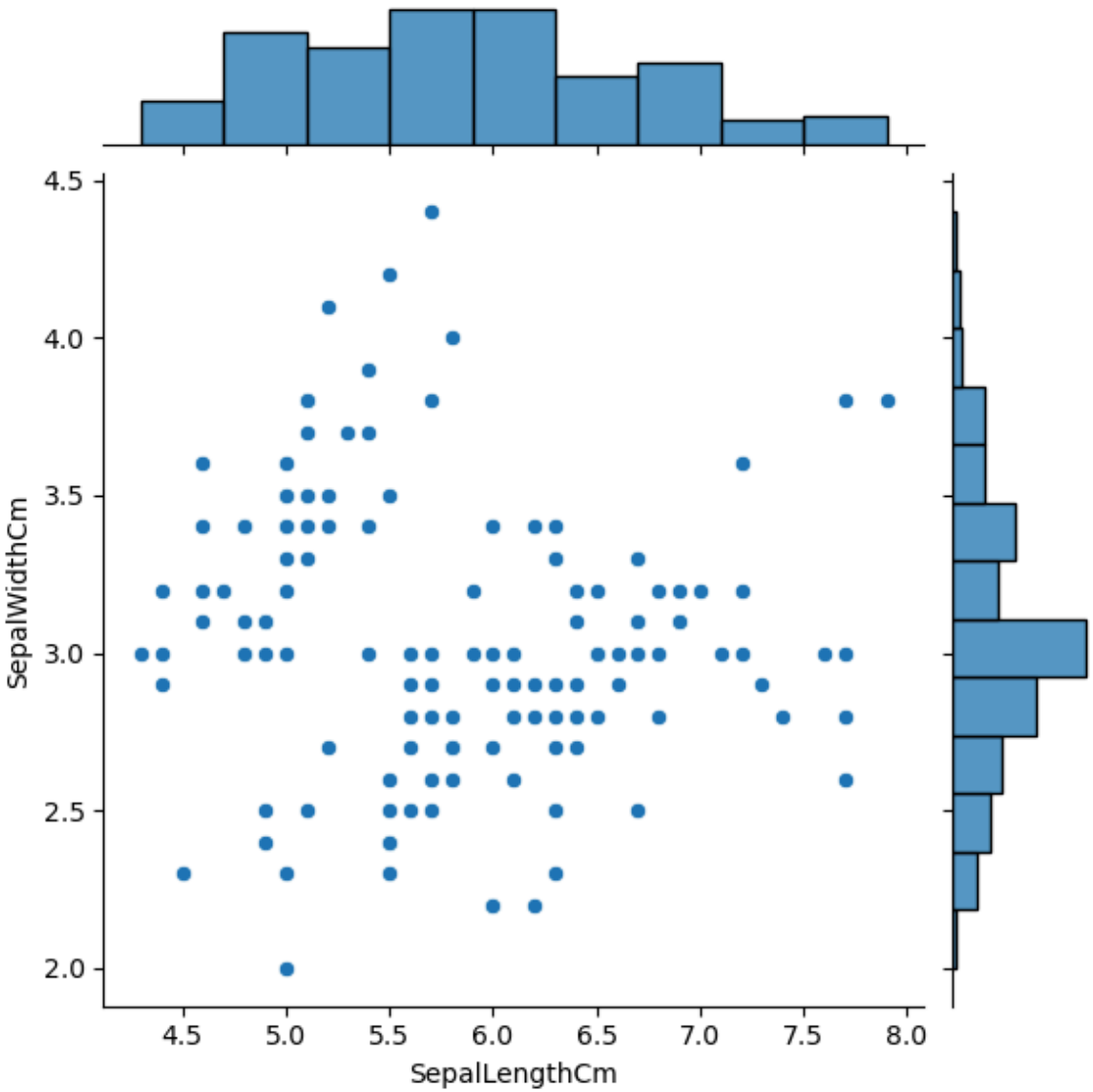
In [20]:
```
sns.countplot(data=iris,x='Species',hue='Species')
plt.show()
```



4. Joint plot:

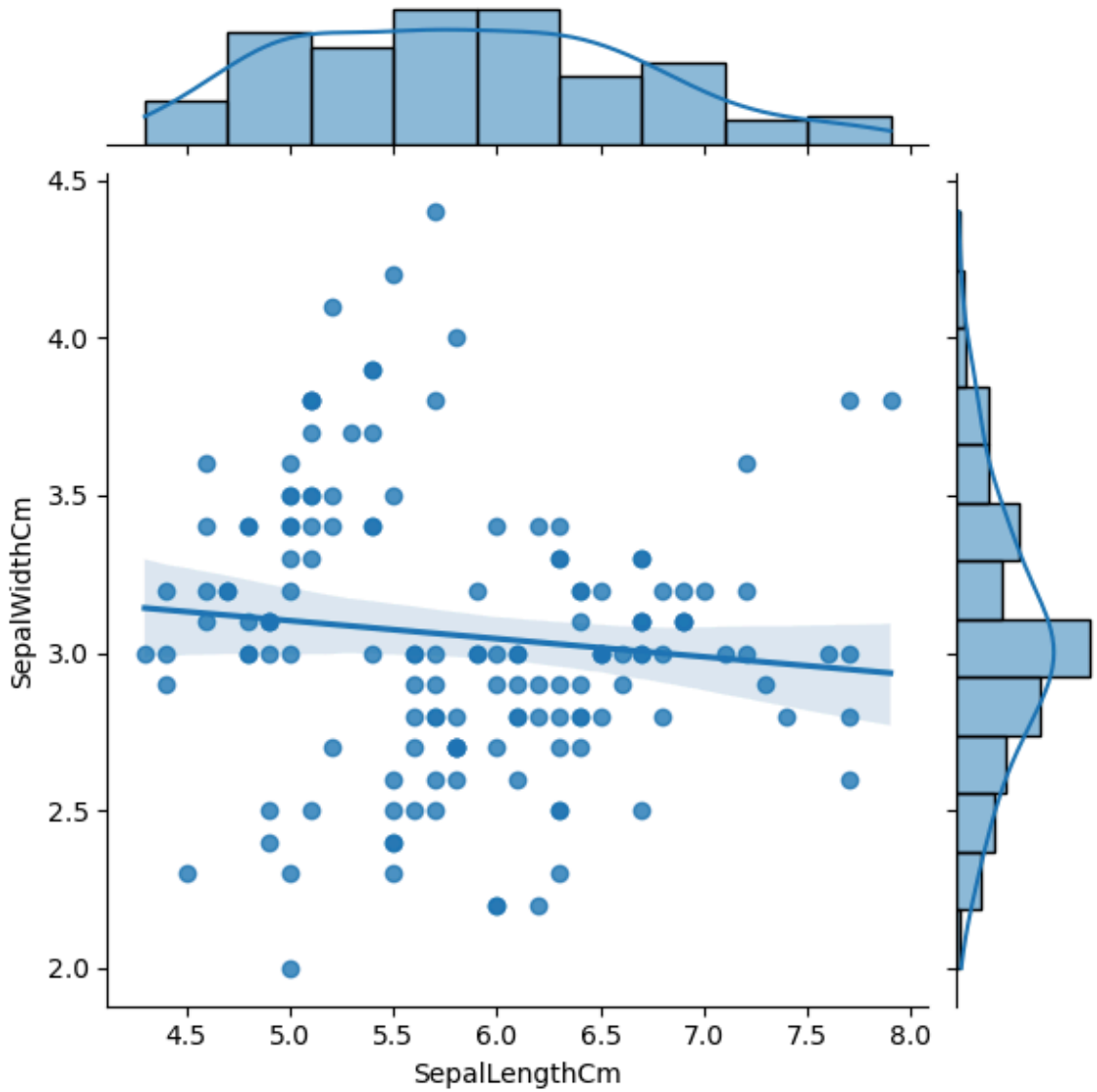
Jointplot is seaborn library specific and can be used to quickly visualize and analyze the relationship between two variables and describe their individual distributions on the same plot.

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

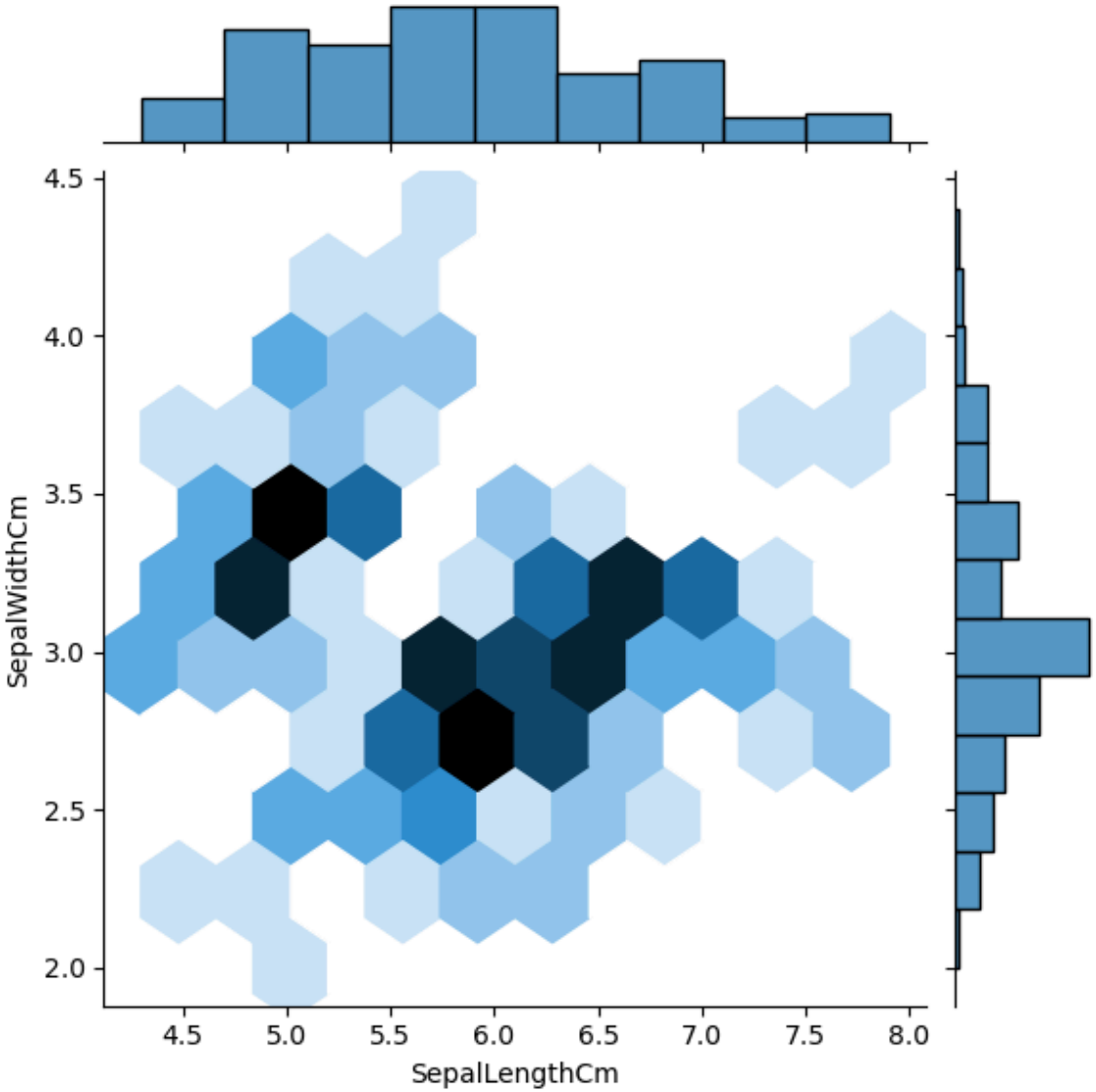


In [22]:
```
sns.jointplot(x="SepalLengthCm", y="SepalWidthCm", data=iris, kind="reg")
```

Out[22]:
```
<seaborn.axisgrid.JointGrid at 0x227a0d5f6b0>
```



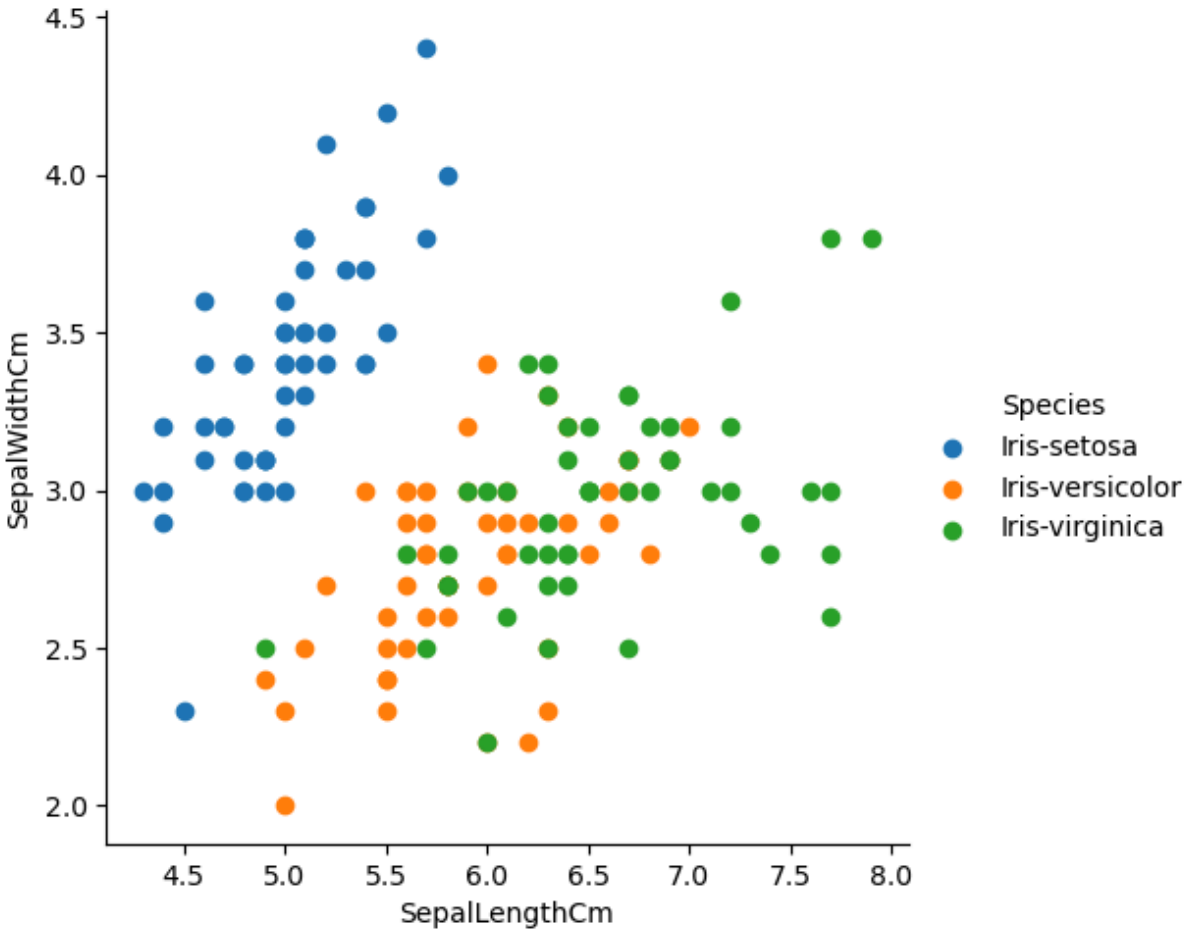
In [23]:
```
fig=sns.jointplot(x='SepalLengthCm',y='SepalWidthCm',kind='hex',data=iris)
```



FacetGrid plot

In [25]:
```
sns.FacetGrid(iris , hue='Species' ,height=5)\
.map(plt.scatter,'SepalLengthCm','SepalWidthCm')\
.add_legend()
```

Out[25]:
```
<seaborn.axisgrid.FacetGrid at 0x227a18c85c0>
```



In [27]:
```
iris.head(4)
```

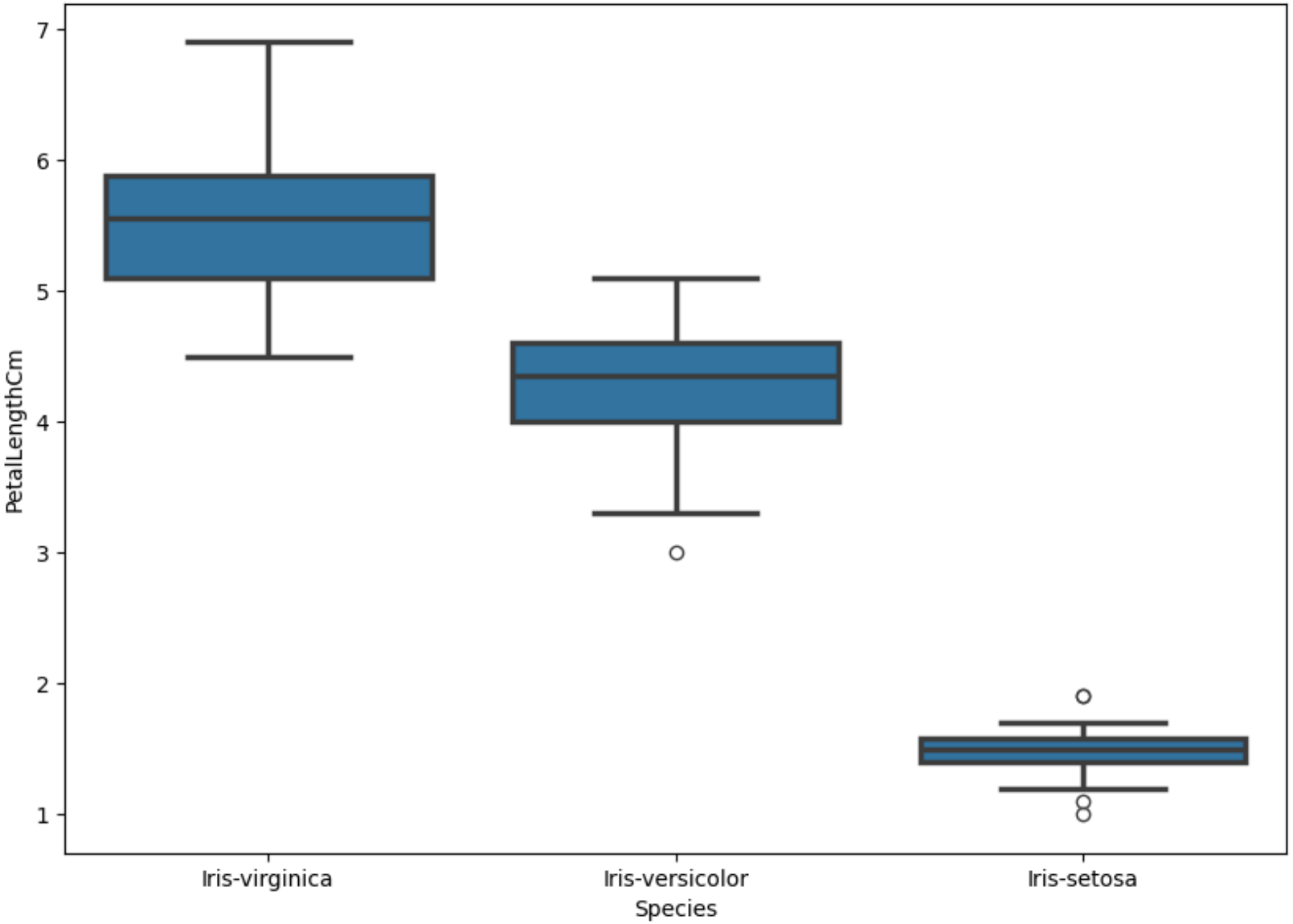
Out[27]:

	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

BOXPLOT OR WHISKER

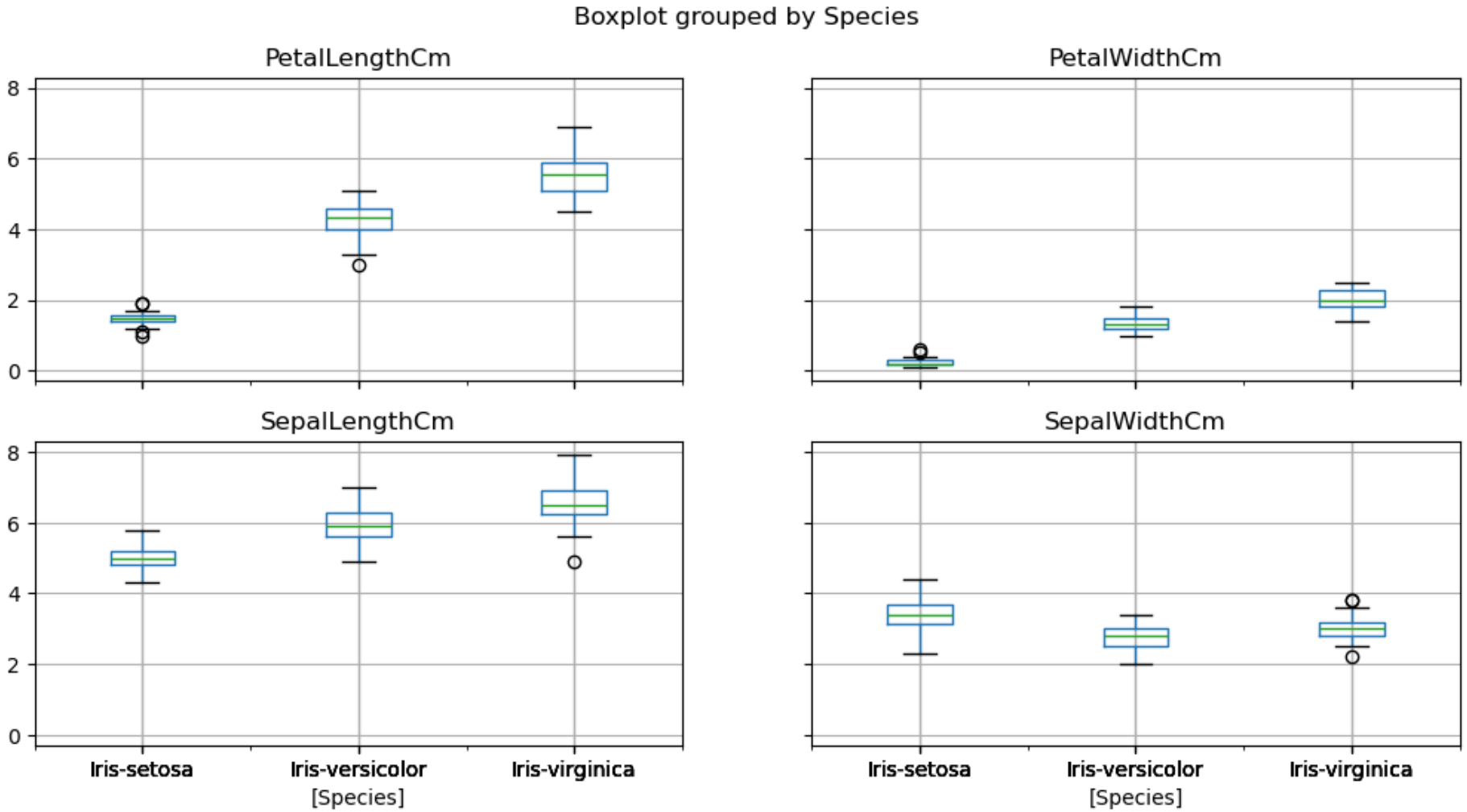
. Boxplot or Whisker plot Box plot was was first introduced in year 1969 by Mathematician John Tukey.Box plot give a statical summary of the features being plotted.Top line represent the max value,top edge of box is third Quartile, middle edge represents the median,bottom edge represents the first quartile value.The bottom most line respresent the minimum value of the feature.The height of the box is called as Interquartile range.The black dots on the plot represent the outlier values in the data.

In [29]:
```
fig=plt.gcf()
fig.set_size_inches(10,7)
fig=sns.boxplot(x='Species',y='PetalLengthCm',data=iris,order=['Iris-virginica','Iris-versicolor','Iris-setosa'],linewidth=2.5,or
```



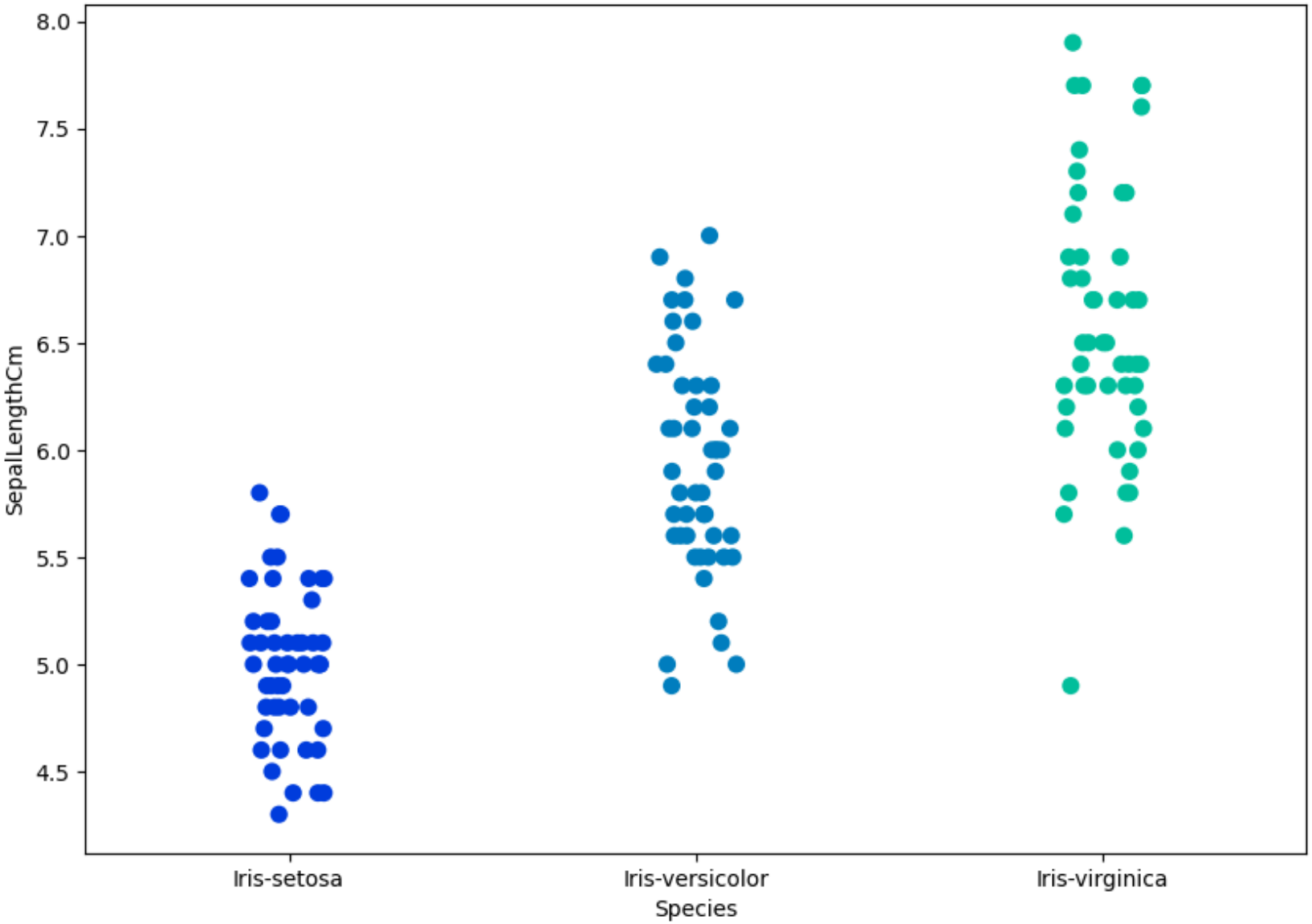
In [30]:
```
#iris.drop("Id", axis=1).boxplot(by="Species", figsize=(12, 6))
iris.boxplot(by="Species", figsize=(12, 6))
```

Out[30]:
```
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)
```



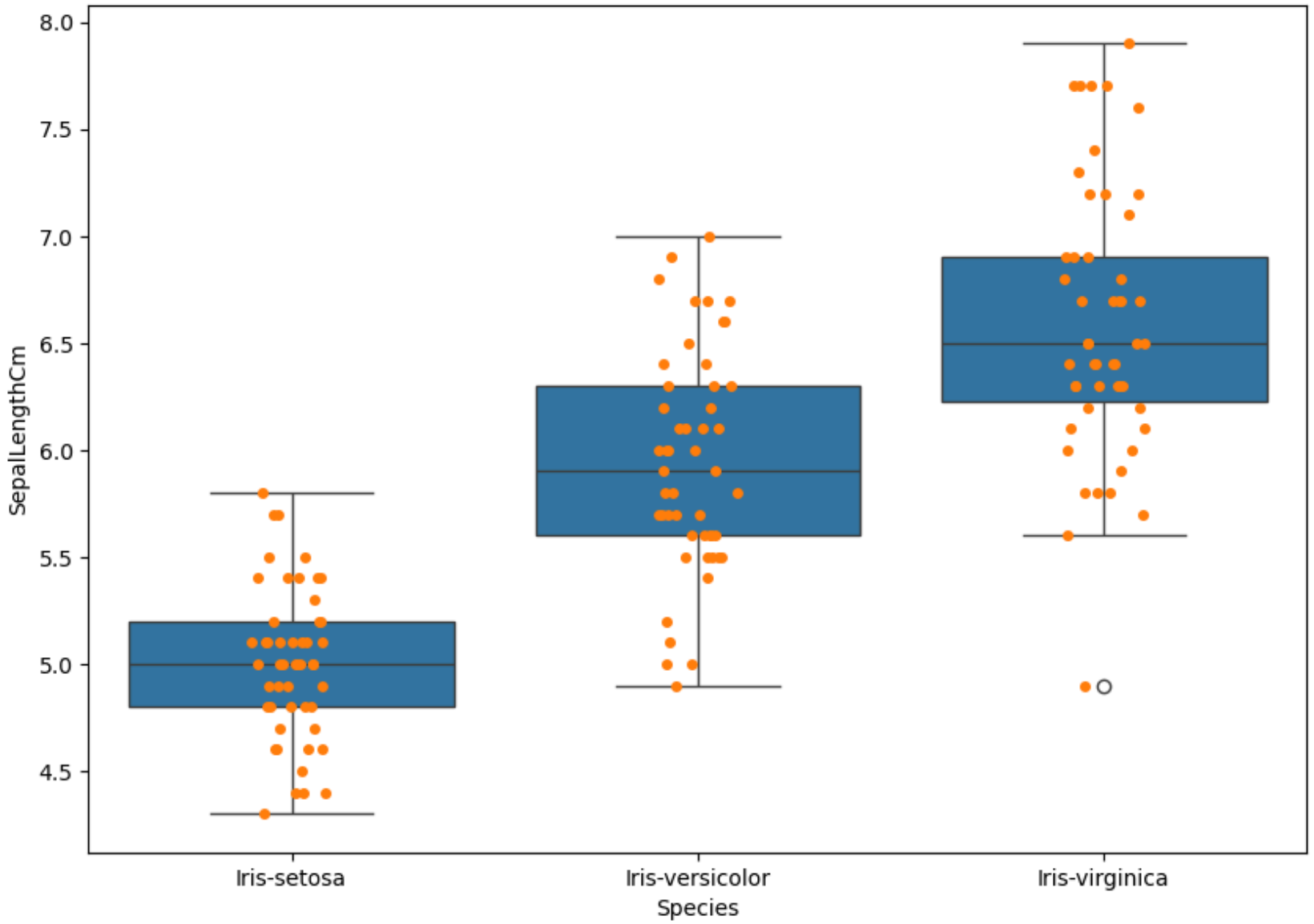
stripplot

In [31]:
```
fig=plt.gcf()
fig.set_size_inches(10,7)
fig=sns.stripplot(x='Species',y='SepalLengthCm',data=iris,jitter=True,edgecolor='gray',size=8,palette='winter',orient='v')
```

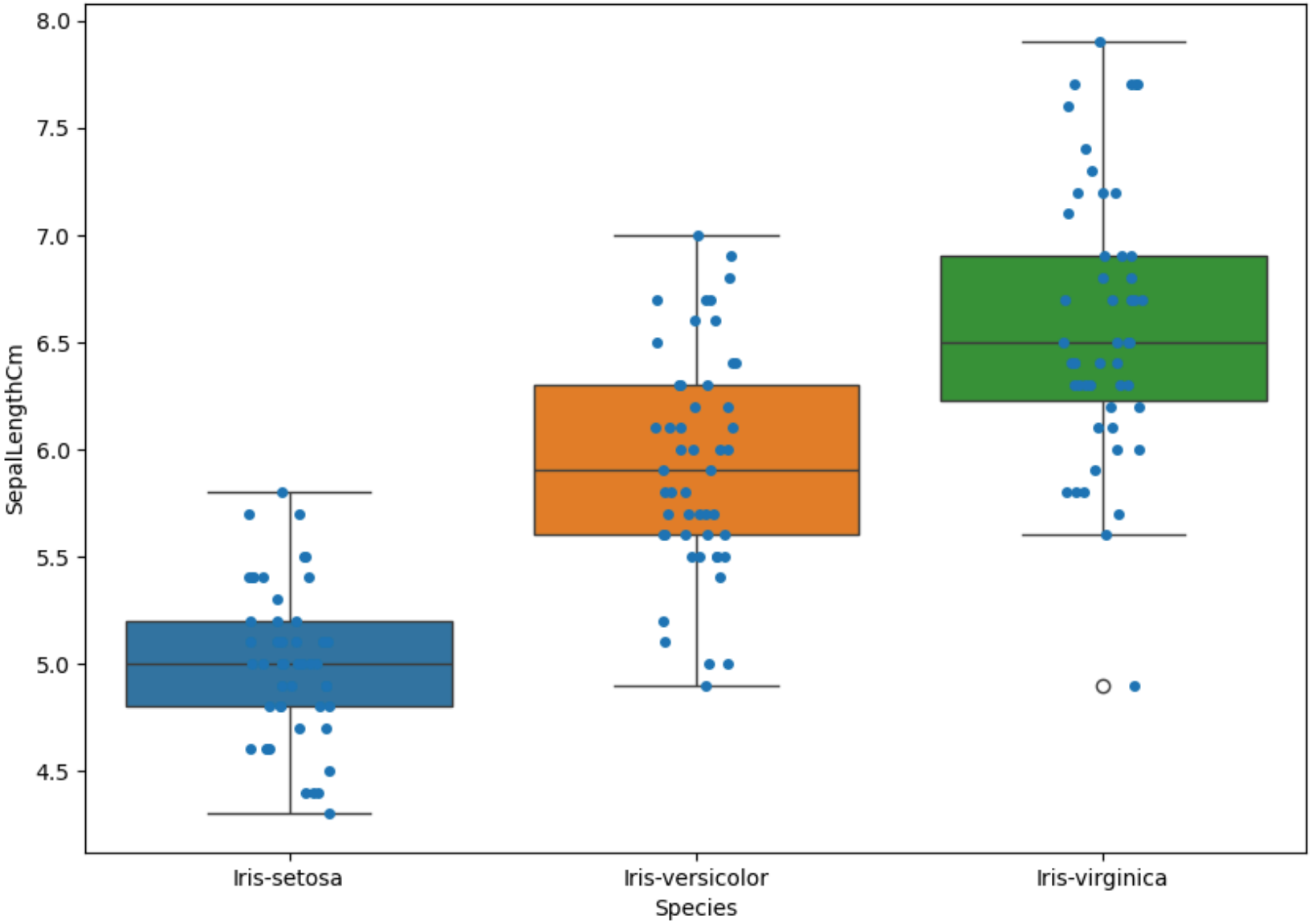


Combining Box and Strip Plots

In [33]:
```
fig=plt.gcf()
fig.set_size_inches(10,7)
fig=sns.boxplot(x='Species',y='SepalLengthCm',data=iris,)
fig=sns.stripplot(x='Species',y='SepalLengthCm',data=iris,jitter=True,edgecolor='gray',)
```



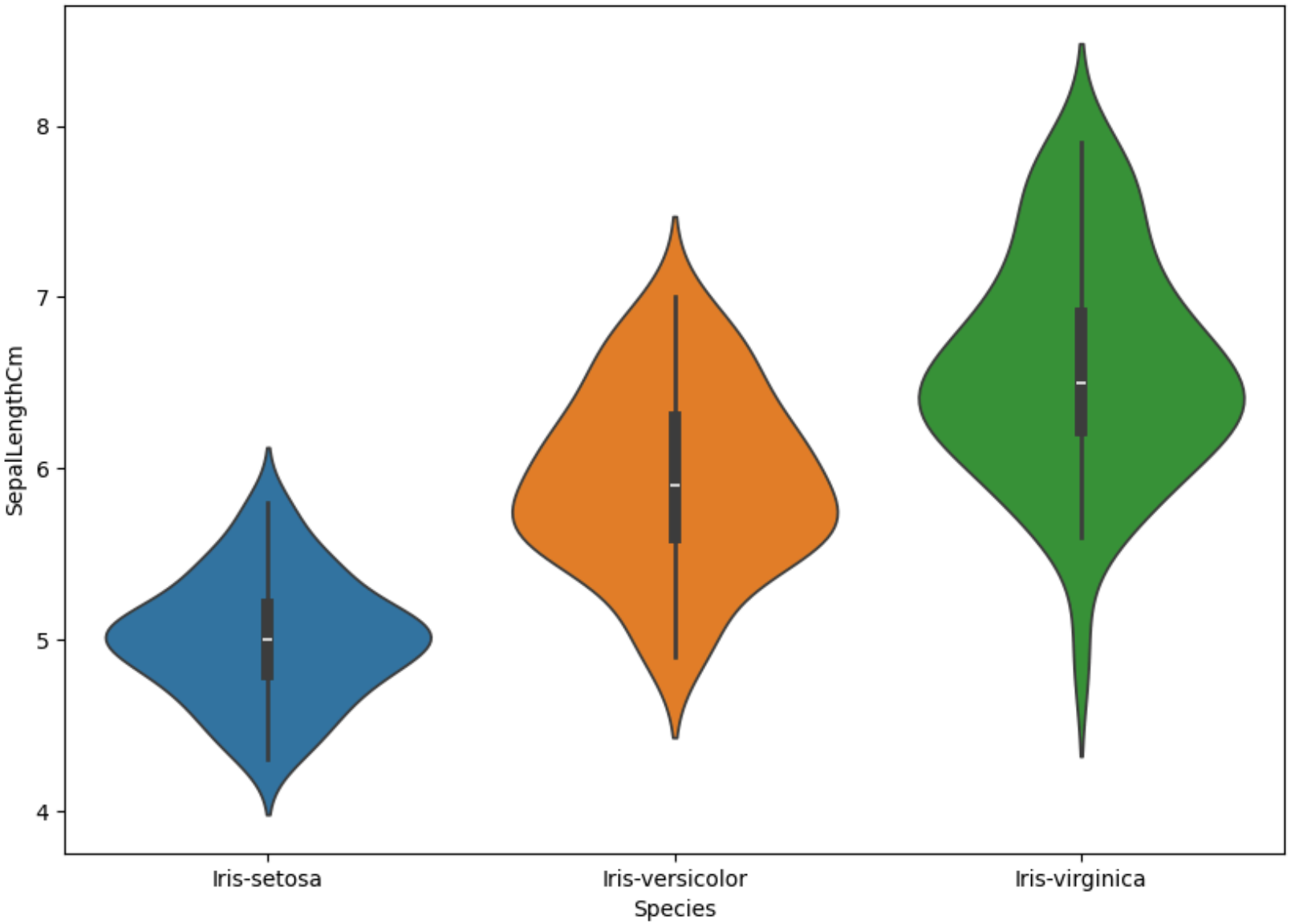
In [34]:
```
fig=plt.gcf()
fig.set_size_inches(10,7)
fig=sns.boxplot(x='Species',y='SepalLengthCm',data=iris, hue = 'Species')
fig=sns.stripplot(x='Species',y='SepalLengthCm',data=iris,jitter=True,edgecolor='gray',)
```


8.0
7.5
7.0
6.5
6.0
5.5
5.0
4.5
SepalLengthCm
Iris-setosa
Iris-versicolor
Iris-virginica
Species

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

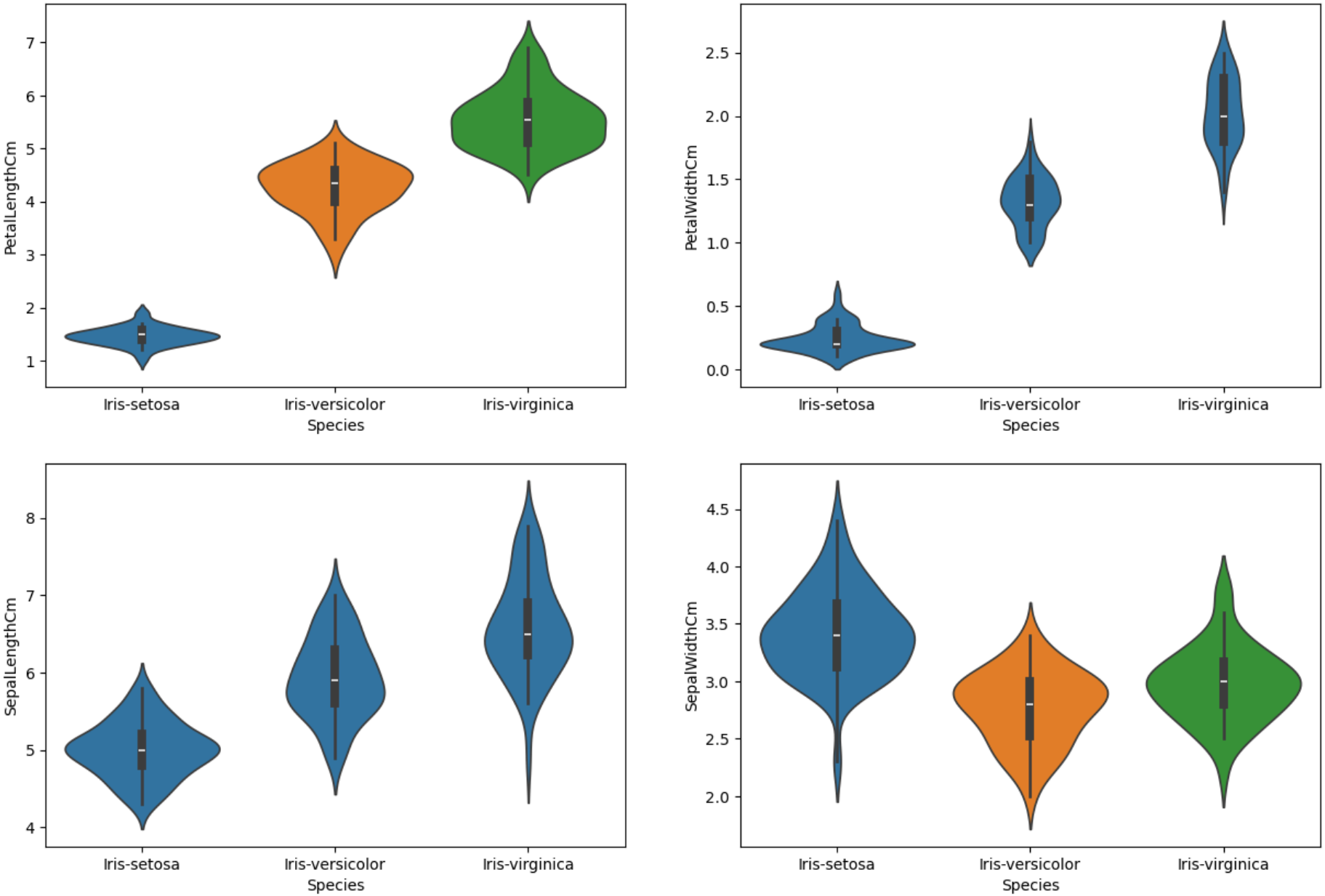
In [36]:
```
fig=plt.gcf()
fig.set_size_inches(10,7)
fig=sns.violinplot(x='Species',y='SepalLengthCm',data=iris,hue="Species")
# hue ='arrgument' for plot diffent colours for different violinplot
```



In [37]:
```
plt.figure(figsize=(15,10))
plt.subplot(2,2,1)
sns.violinplot(x='Species',y='PetalLengthCm',data=iris,hue='Species')
plt.subplot(2,2,2)
sns.violinplot(x='Species',y='PetalWidthCm',data=iris)
```

```
plt.subplot(2,2,3)
sns.violinplot(x='Species',y='SepalLengthCm',data=iris)
plt.subplot(2,2,4)
sns.violinplot(x='Species',y='SepalWidthCm',data=iris,hue='Species')
```

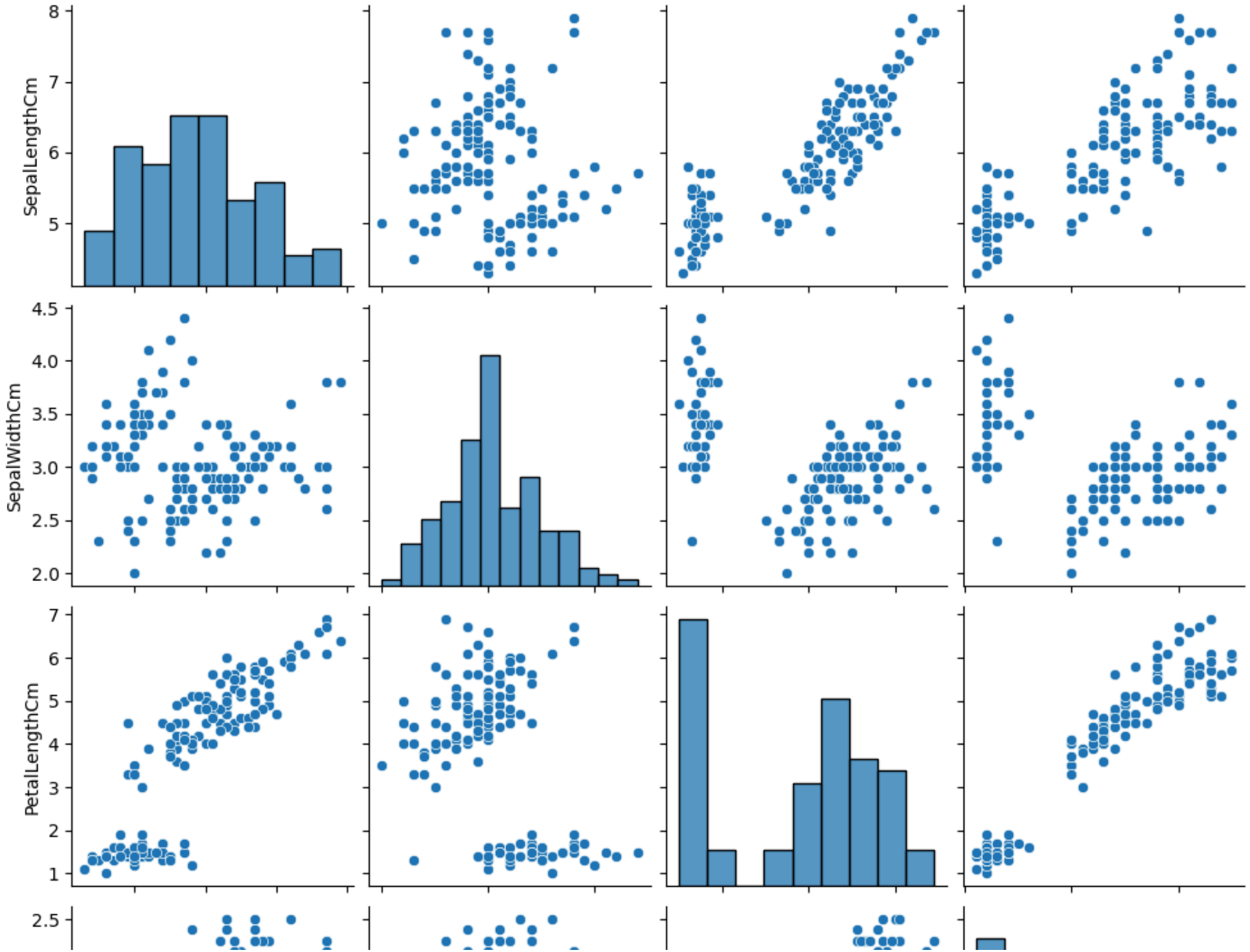
Out[37]: <Axes: xlabel='Species', ylabel='SepalWidthCm'>

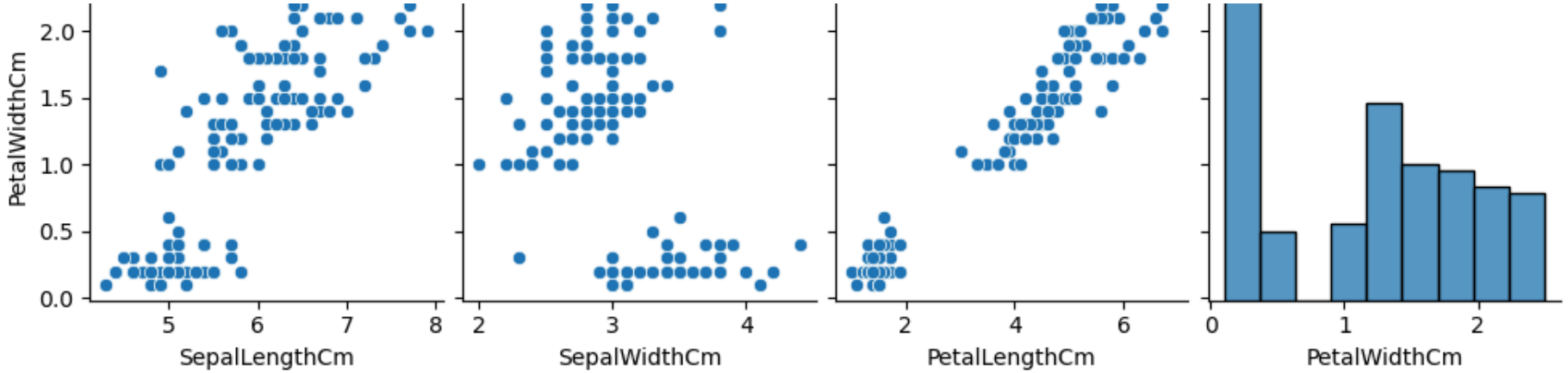


10. Pair Plot: A “pairs plot” is also known as a scatterplot, in which one variable in the same data row is matched with another variable's value, like this: Pairs plots are just elaborations on this, showing all variables paired with all the other variables.

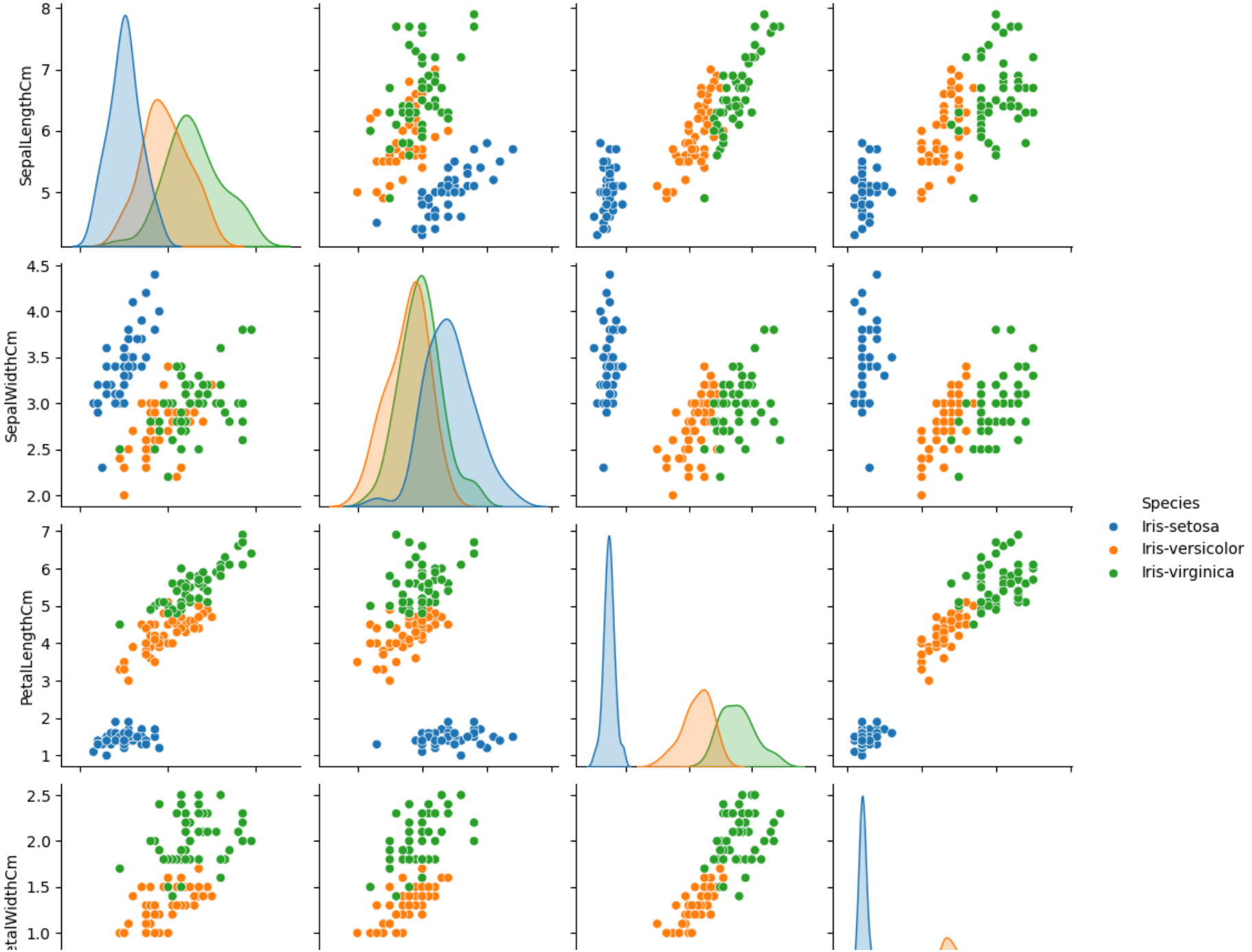
In [38]:
```
sns.pairplot(data=iris,kind='scatter',)
```

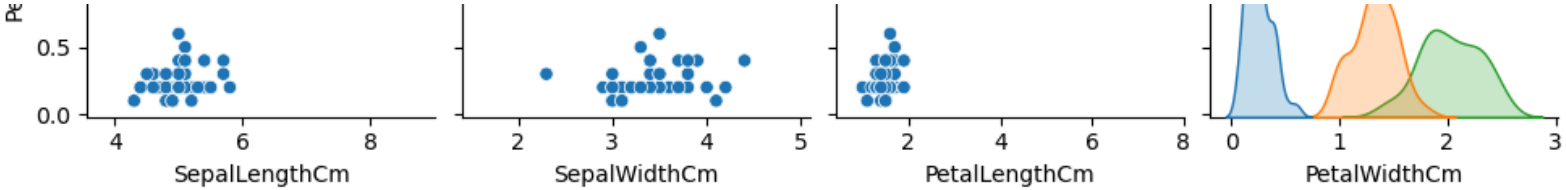
Out[38]:
```
<seaborn.axisgrid.PairGrid at 0x227a73ab620>
```


SepalLengthCm
SepalWidthCm
PetalLengthCm
8
7
6
5
4.5
4.0
3.5
3.0
2.5
2.0
7
6
5
4
3
2
1
2.5



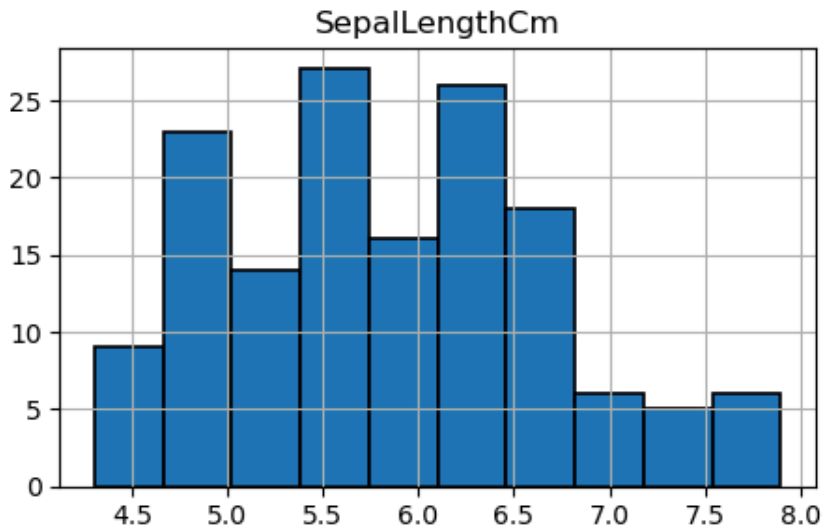
In [39]:
```
sns.pairplot(iris,hue='Species');
```

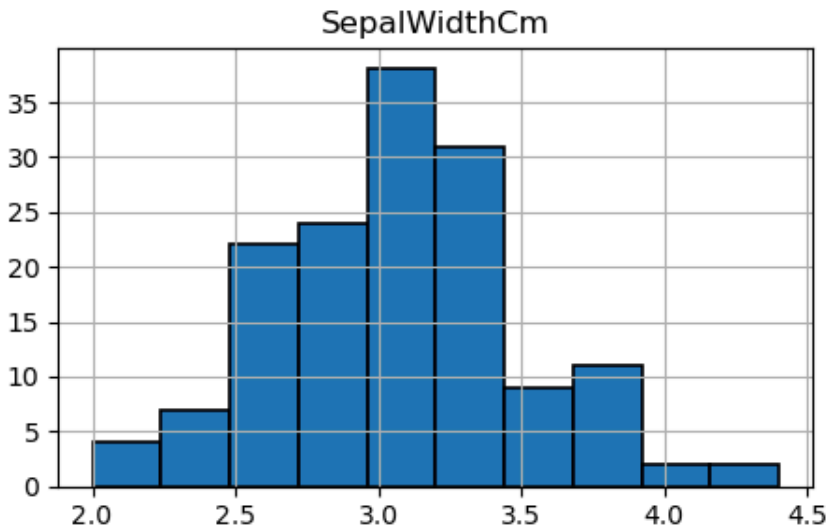

SepalLengthCm
SepalWidthCm
PetalLengthCm
PetalWidthCm
Species
Iris-setosa
Iris-versicolor
Iris-virginica

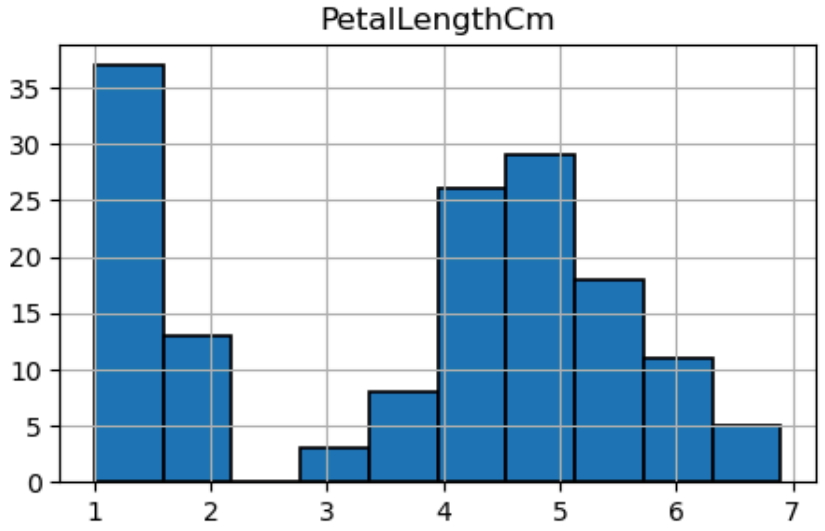


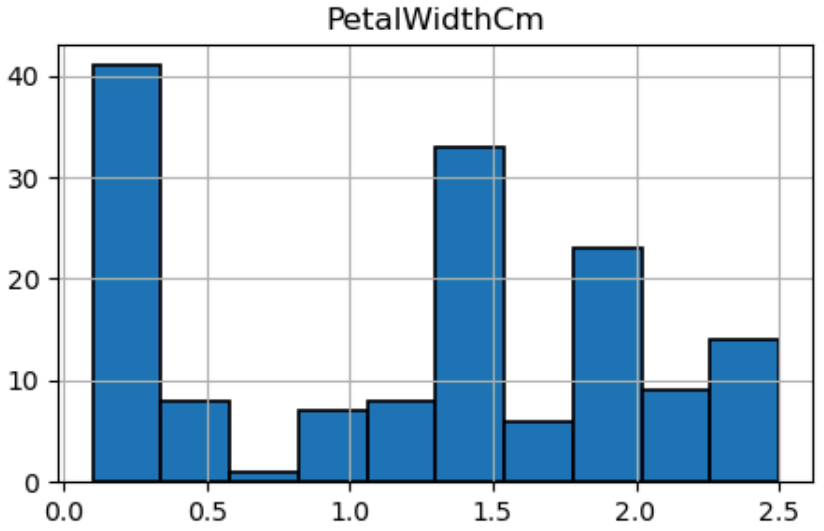
. **Distribution plot**: The distribution plot is suitable for comparing range and distribution for groups of numerical data. Data is plotted as value points along an axis. You can choose to display only the value points to see the distribution of values, a bounding box to see the range of values, or 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 [41]:
```
iris.hist(edgecolor='k',linewidth=1.2)
fig=plt.gcf()
fig.set_size_inches(12,7)
```



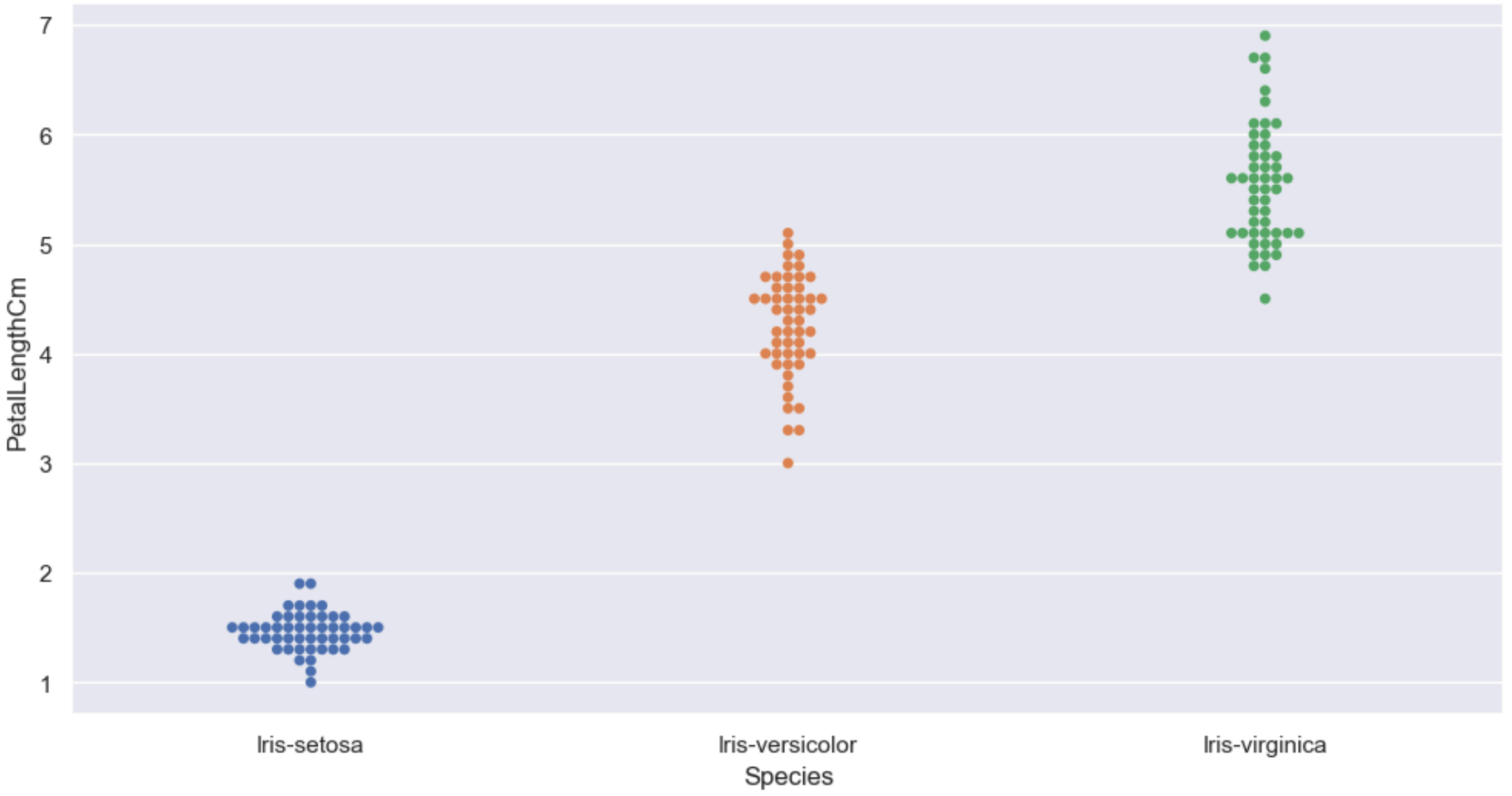






Swarm plot : It looks a bit like a friendly swarm of bees buzzing about their hive. More importantly, each data point is clearly visible and no data are obscured by overplotting.A beeswarm plot improves upon the random jittering approach to move data points the minimum distance away from one another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

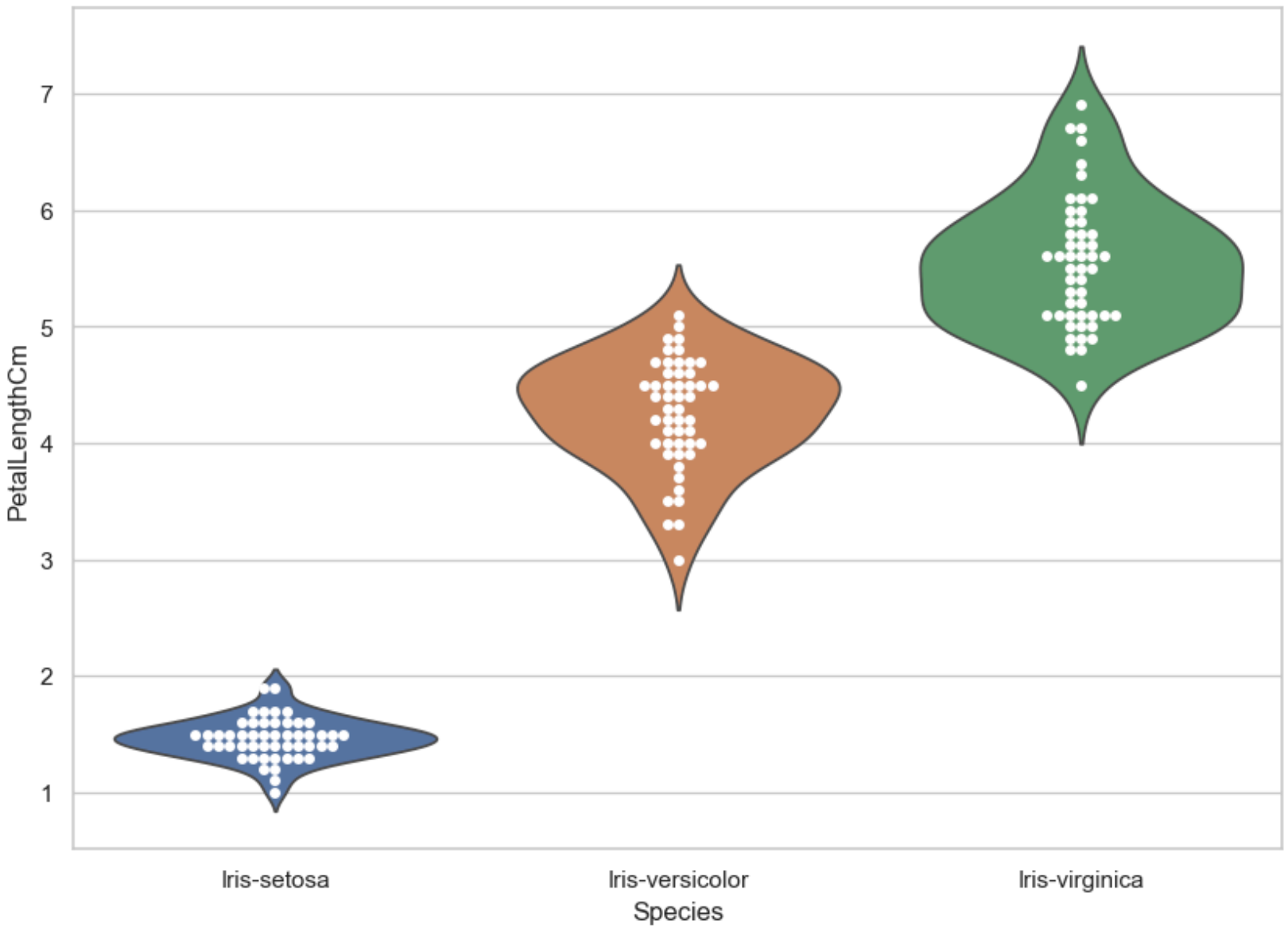
In [43]:
```
sns.set(style="darkgrid")
fig=plt.gcf()
fig.set_size_inches(12,6)
fig=sns.swarmplot(x="Species",y="PetalLengthCm",data=iris,hue="Species")
```



combination of violinplot & swarmplot

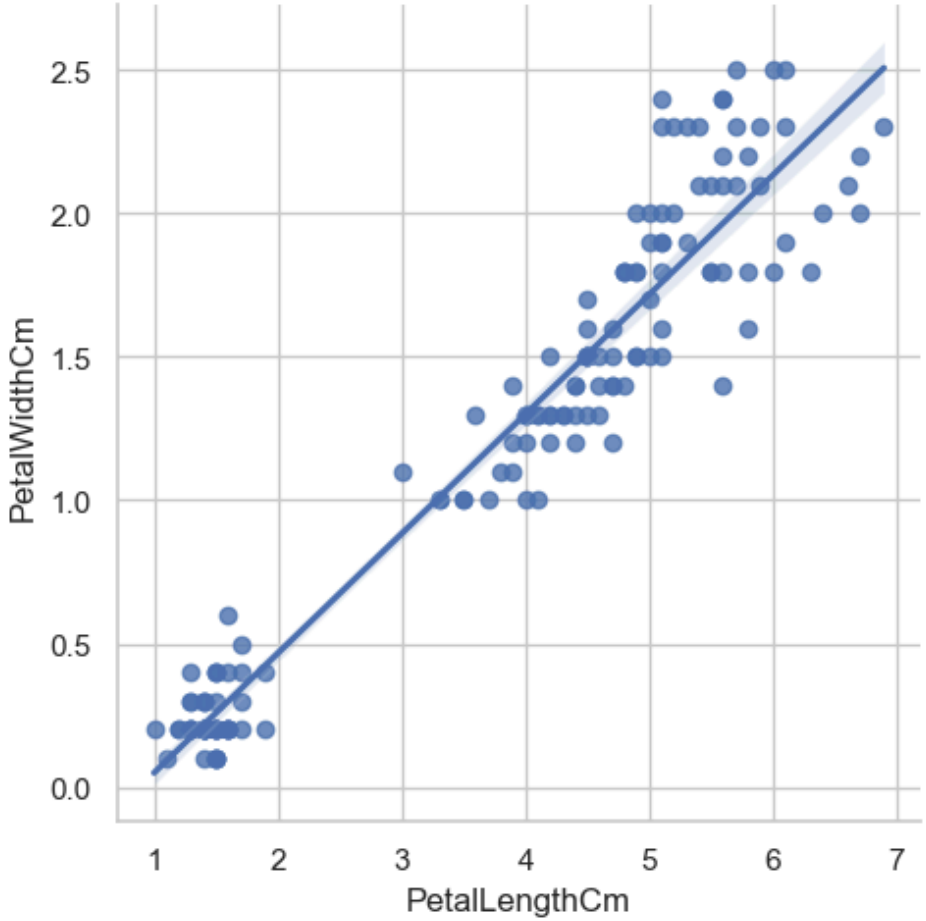
In [45]:

```
sns.set(style="whitegrid")
fig=plt.gcf()
fig.set_size_inches(10,7)
ax = sns.violinplot(x="Species", y="PetalLengthCm", data=iris, inner=None,hue="Species")
ax = sns.swarmplot(x="Species", y="PetalLengthCm", data=iris,color="white", edgecolor="black")
```


PetalLengthCm
7
6
5
4
3
2
1
Iris-setosa
Iris-versicolor
Iris-virginica
Species

LM plot

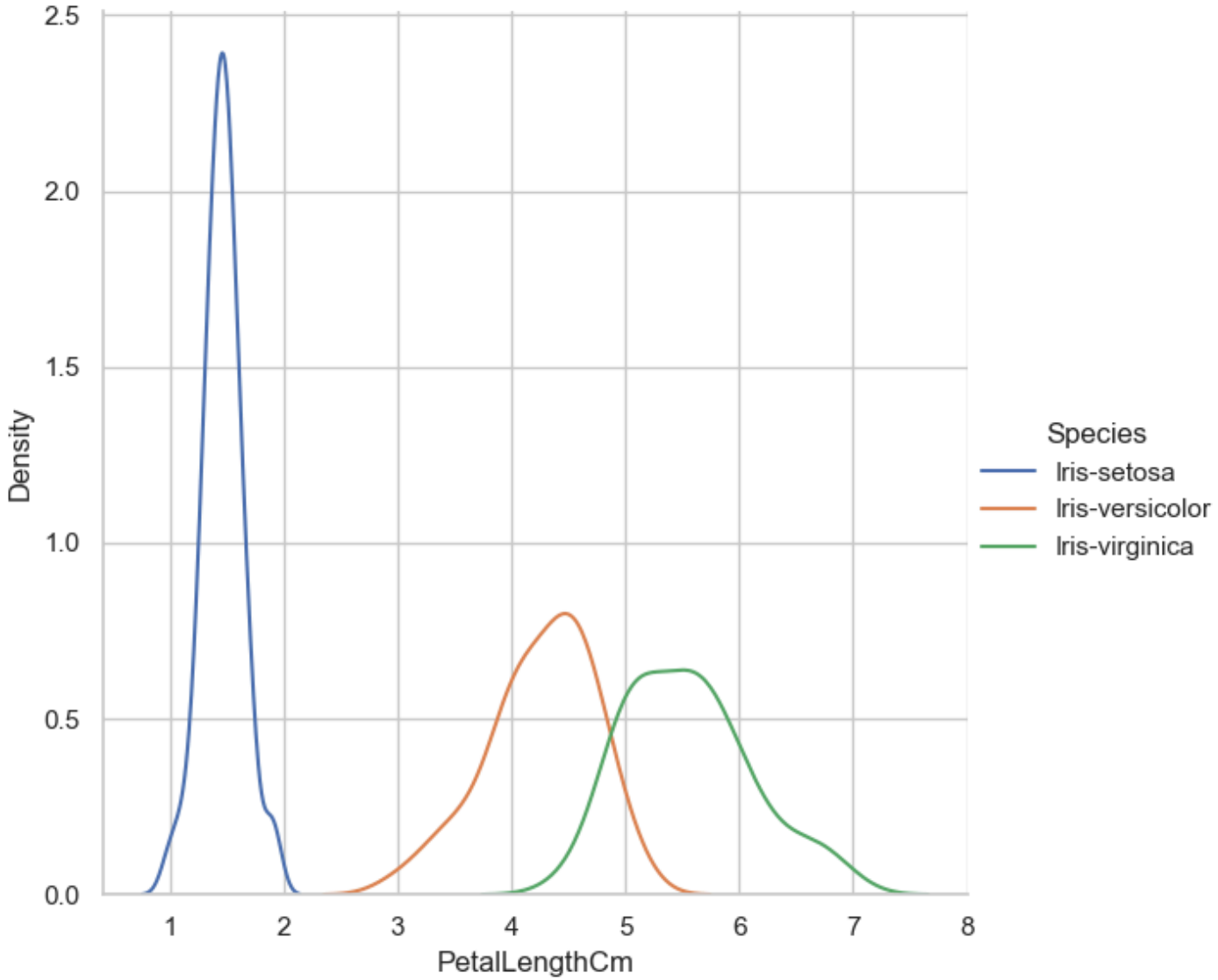
In [47]:
```
fig=sns.lmplot(x="PetalLengthCm", y="PetalWidthCm",data=iris,height=5)
```



FacetGrid

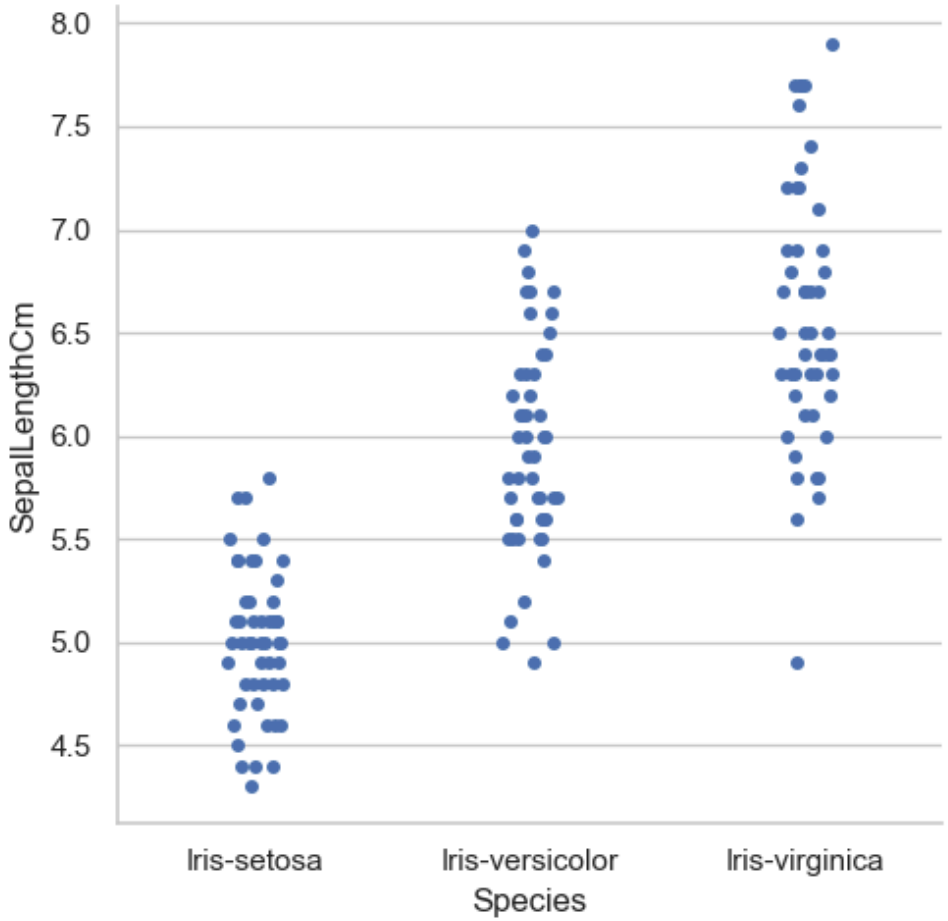
In [49]:
```
sns.FacetGrid(iris, hue="Species", height=6) \
   .map(sns.kdeplot, "PetalLengthCm") \
   .add_legend()
plt.ioff()
```

Out[49]: <contextlib.ExitStack at 0x227a19ec770>



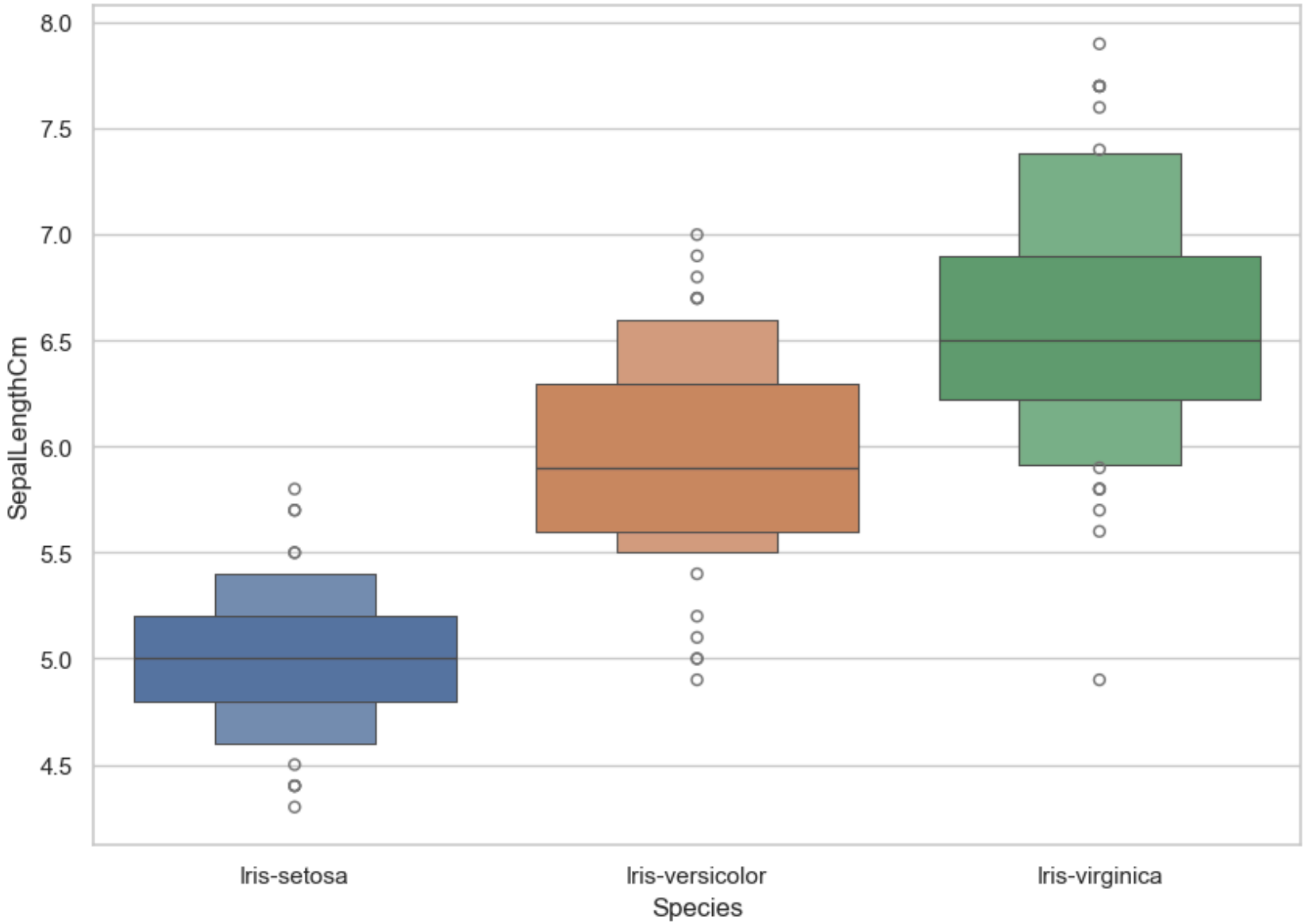
cat Plot

In [51]:
```
#f,ax=plt.subplots(1,2,figsize=(18,8))
sns.catplot(iris,x='Species',y='SepalLengthCm')
plt.ioff()
plt.show()
#sns.factorplot('Species','SepalLengthCm',data=iris,ax=ax[0][0])
#sns.factorplot('Species','SepalWidthCm',data=iris,ax=ax[0][1])
#sns.factorplot('Species','PetalLengthCm',data=iris,ax=ax[1][0])
#sns.factorplot('Species','PetalWidthCm',data=iris,ax=ax[1][1])
```



Boxen plot

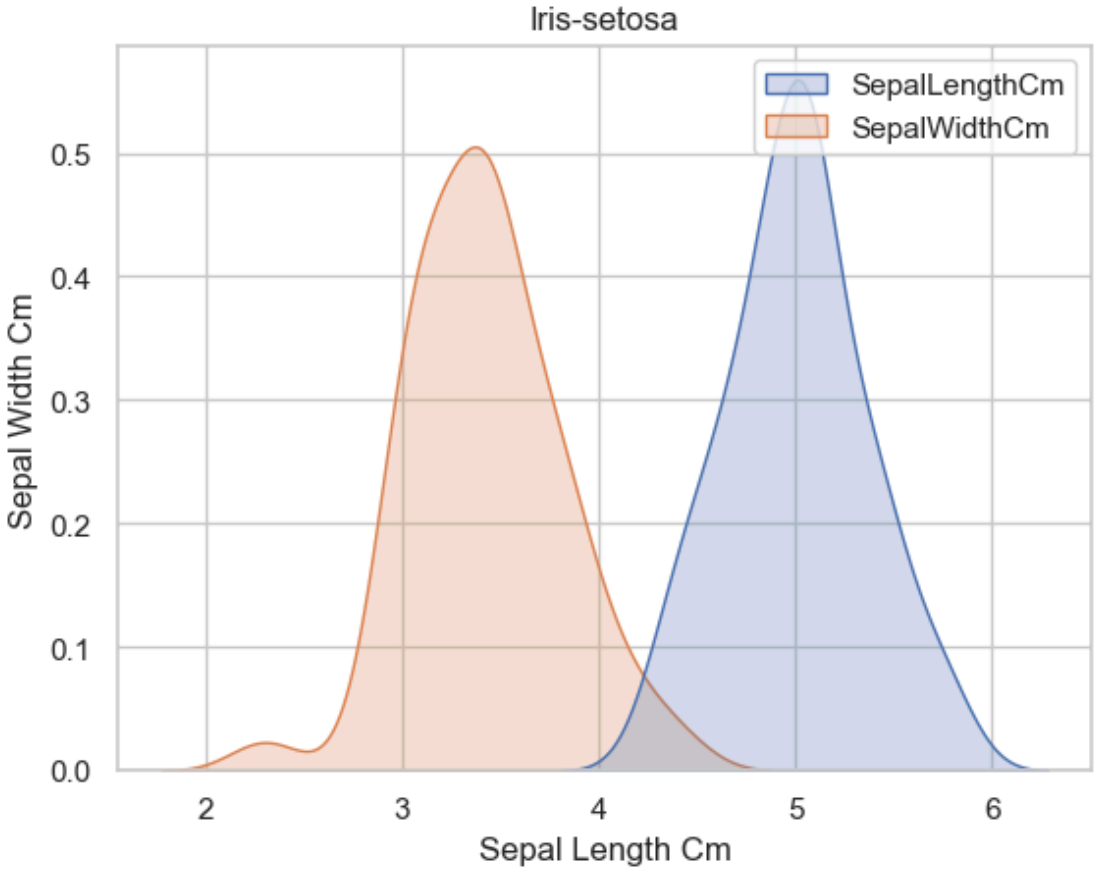
In [53]:
```
fig=plt.gcf()
fig.set_size_inches(10,7)
fig=sns.boxenplot(x='Species',y='SepalLengthCm',data=iris,hue="Species")
plt.show(fig)
```


8.0
7.5
7.0
6.5
6.0
5.5
5.0
4.5
SepalLengthCm
Iris-setosa
Iris-versicolor
Iris-virginica
Species

kde plot

- A kernel density estimate (KDE) plot is a method for visualizing the distribution of observations in a dataset, analogous to a histogram.

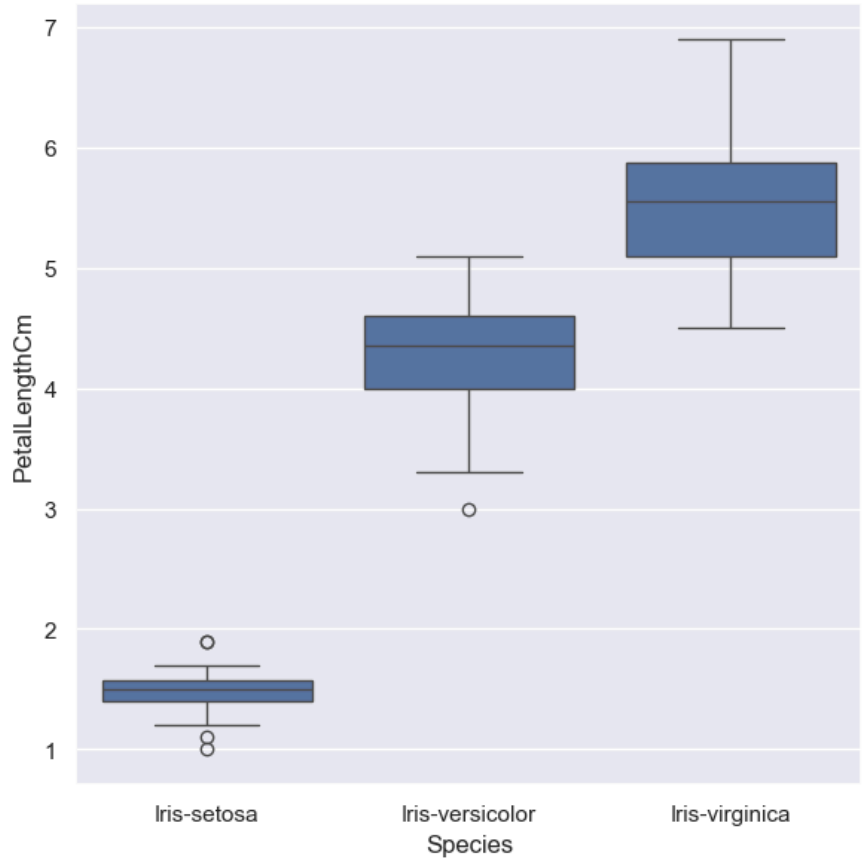
In [55]:
```
# Create a kde plot of sepal_length versus sepal width for setosa species of flower.
sub=iris[iris['Species']=='Iris-setosa']
sns.kdeplot(data=sub[['SepalLengthCm','SepalWidthCm']], shade=True, shade_lowest=False)
plt.title('Iris-setosa')
plt.xlabel('Sepal Length Cm')
plt.ylabel('Sepal Width Cm')
plt.show()
```

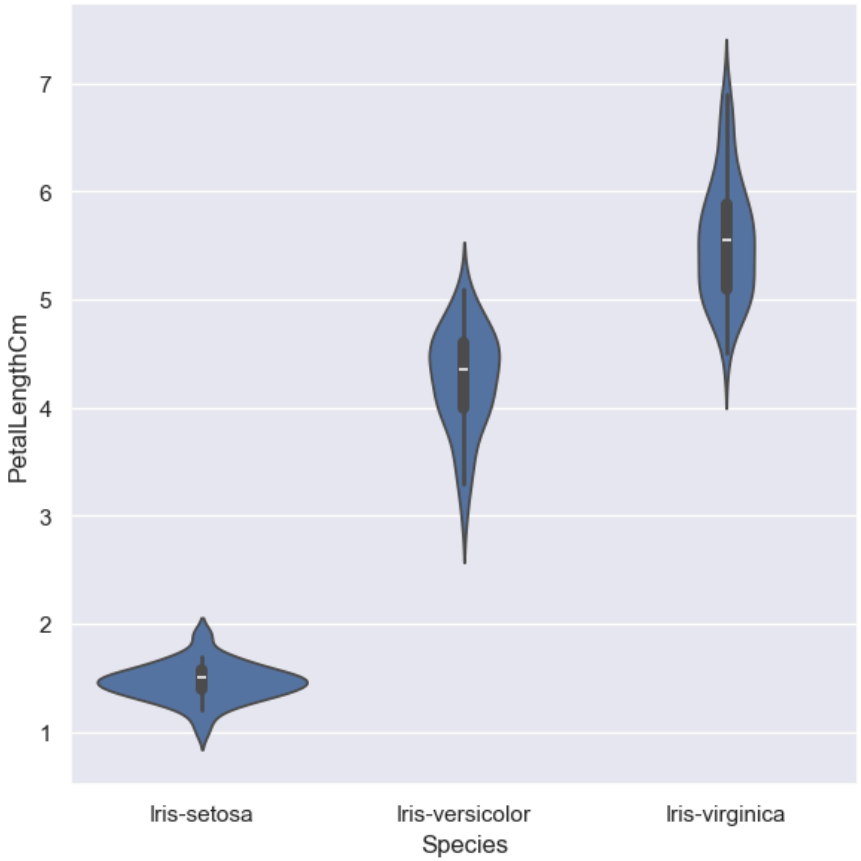


Dashboard

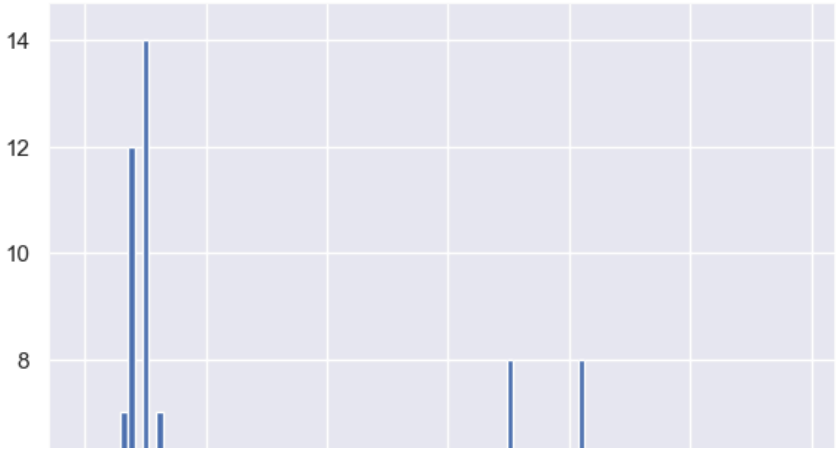
In [57]:

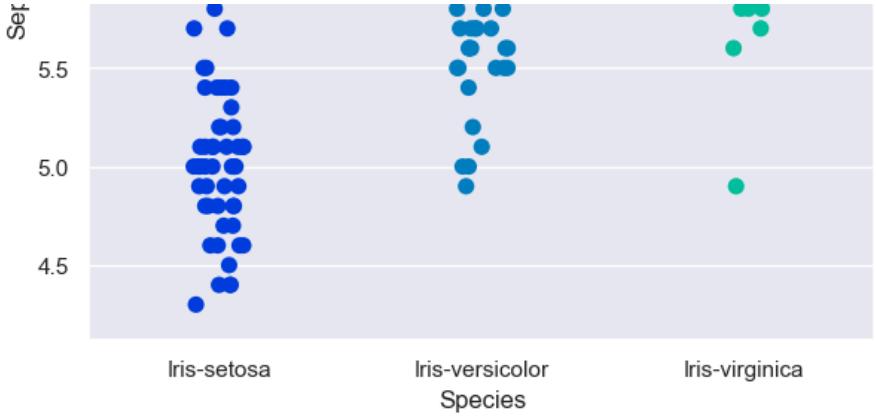
```
sns.set_style('darkgrid')
f,axes=plt.subplots(2,2,figsize=(15,15))

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='gray',size=8,palette='winter',orient='v',ax=axes[
#axes[1,1].hist(iris.hist,bin=10)
axes[1,1].hist(iris.PetalLengthCm,bins=100)
#k2.set(xlim=(-1,0.8))
plt.show()
```

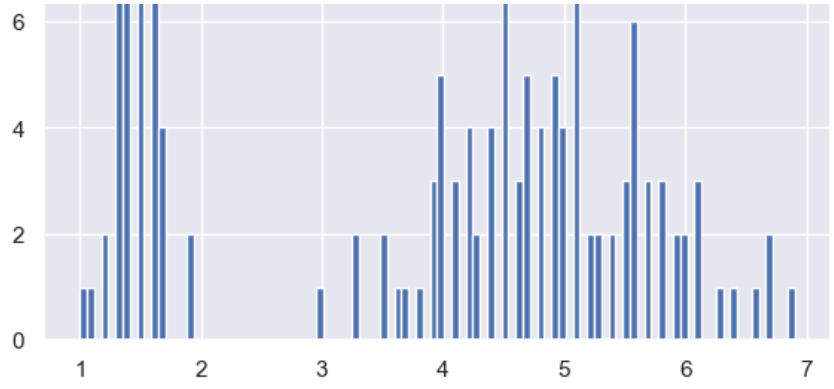

PetalLengthCm
7
6
5
4
3
2
1
Iris-setosa
Iris-versicolor
Iris-virginica
Species


PetalLengthCm
7
6
5
4
3
2
1
Iris-setosa
Iris-versicolor
Iris-virginica
Species


8.0
7.5
7.0
6.5
6.0


14
12
10
8

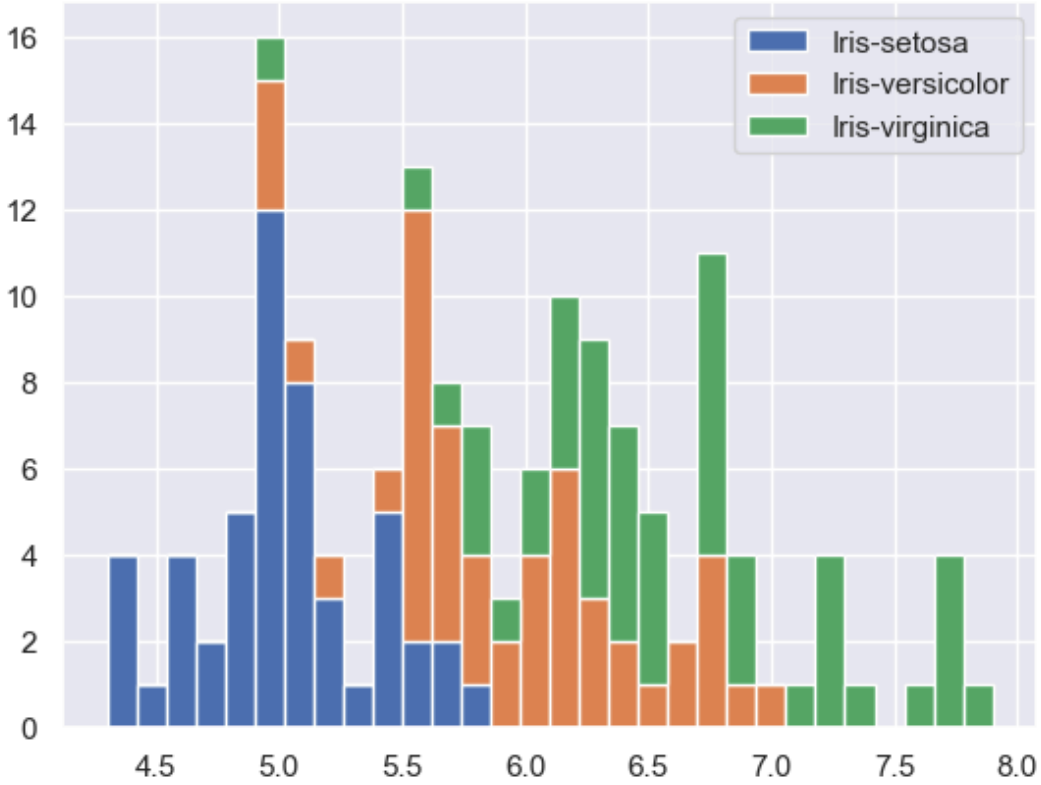




In the dashboard we have shown how to create multiple plots to foam a dashboard using Python.In this plot we have demonstrated how to plot Seaborn and Matplotlib plots on the same Dashboard.

Stacked Histogram

In [60]:
```
iris['Species'] = iris['Species'].astype('category')
#iris.head()
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()
```



With Stacked Histogram we can see the distribution of Sepal Length of Different Species together.This shows us the range of Sepan Length for the three different Species of Iris Flower.

32.Area Plot:

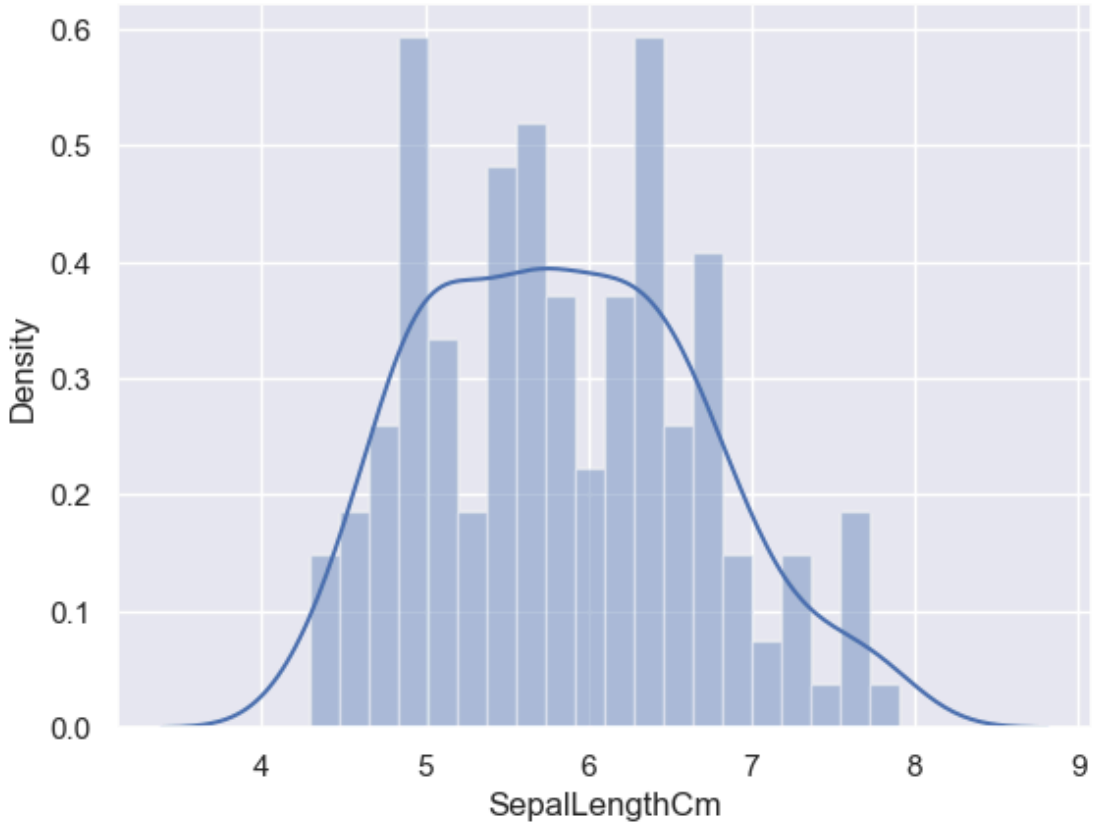
Area Plot gives us a visual representation of Various dimensions of Iris flower and their range in dataset.

In [63]:
```
#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'],alpha=0.4,figsize=(12, 6));
plt.show()
```

33.Distplot:

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

In [65]:
```
sns.distplot(iris['SepalLengthCm'],kde=True,bins=20);
plt.show()
```



In [ ]: