# IRIS VISUALIZATION USING SEABORN & MATPLOTLIB

In [1]:
```
import warnings
warnings.filterwarnings('ignore')
```

In [2]:
```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
```

In [3]:
```
iris=pd.read_csv("Iris.csv")
```

In [4]:
```
iris
```

Out[4]:

| | 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 [5]:
```
iris.head()
```

Out[5]:

| | 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 [6]:
```
iris.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 150 entries, 0 to 149
Data columns (total 6 columns):
 #   Column         Non-Null Count  Dtype  
---  ------         --------------  -----  
 0   Id             150 non-null    int64  
 1   SepalLengthCm  150 non-null    float64
 2   SepalWidthCm   150 non-null    float64
 3   PetalLengthCm  150 non-null    float64
 4   PetalWidthCm   150 non-null    float64
 5   Species        150 non-null    object 
dtypes: float64(4), int64(1), object(1)
memory usage: 7.2+ KB
```

In [7]:
```
iris.describe()
```

Out[7]:

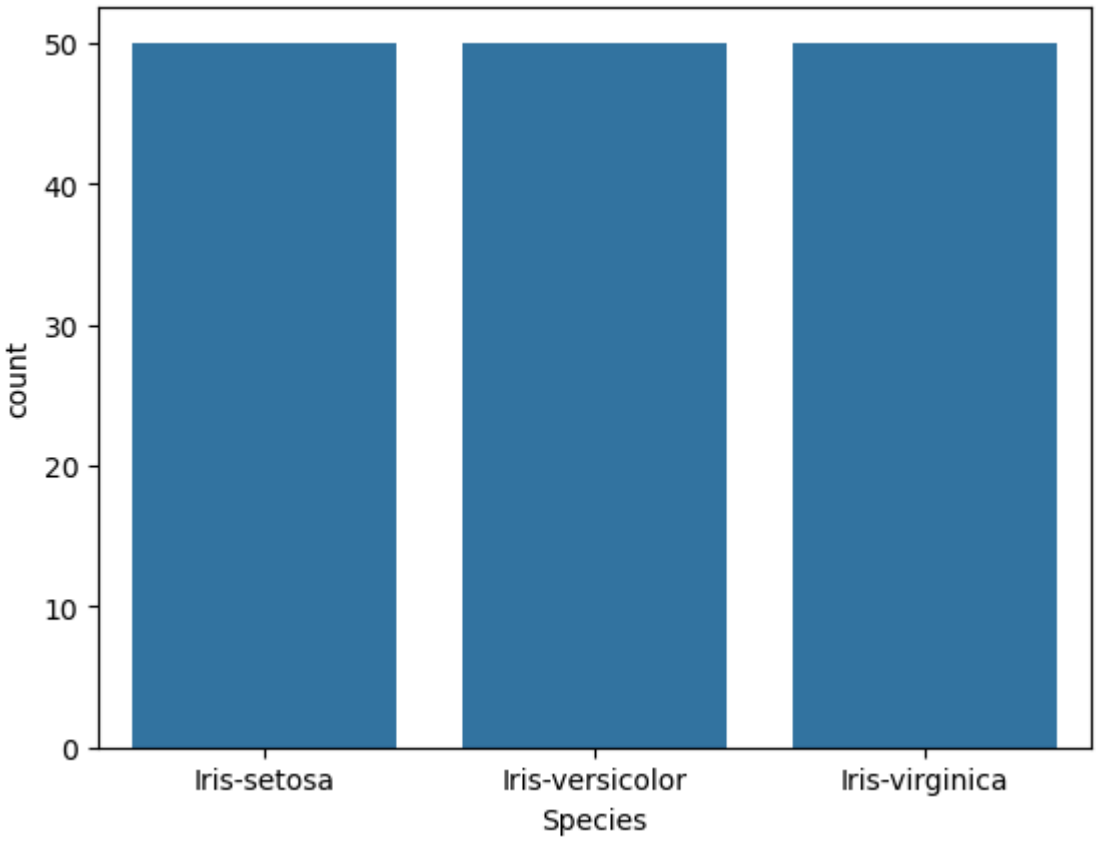
| | Id | SepalLengthCm | SepalWidthCm | PetalLengthCm | PetalWidthCm |
|---|---|---|---|---|---|
| **count** | 150.000000 | 150.000000 | 150.000000 | 150.000000 | 150.000000 |
| **mean** | 75.500000 | 5.843333 | 3.054000 | 3.758667 | 1.198667 |
| **std** | 43.445368 | 0.828066 | 0.433594 | 1.764420 | 0.763161 |
| **min** | 1.000000 | 4.300000 | 2.000000 | 1.000000 | 0.100000 |
| **25%** | 38.250000 | 5.100000 | 2.800000 | 1.600000 | 0.300000 |
| **50%** | 75.500000 | 5.800000 | 3.000000 | 4.350000 | 1.300000 |
| **75%** | 112.750000 | 6.400000 | 3.300000 | 5.100000 | 1.800000 |
| **max** | 150.000000 | 7.900000 | 4.400000 | 6.900000 | 2.500000 |

In [8]:
```
iris.drop('Id',axis=1,inplace=True)
```

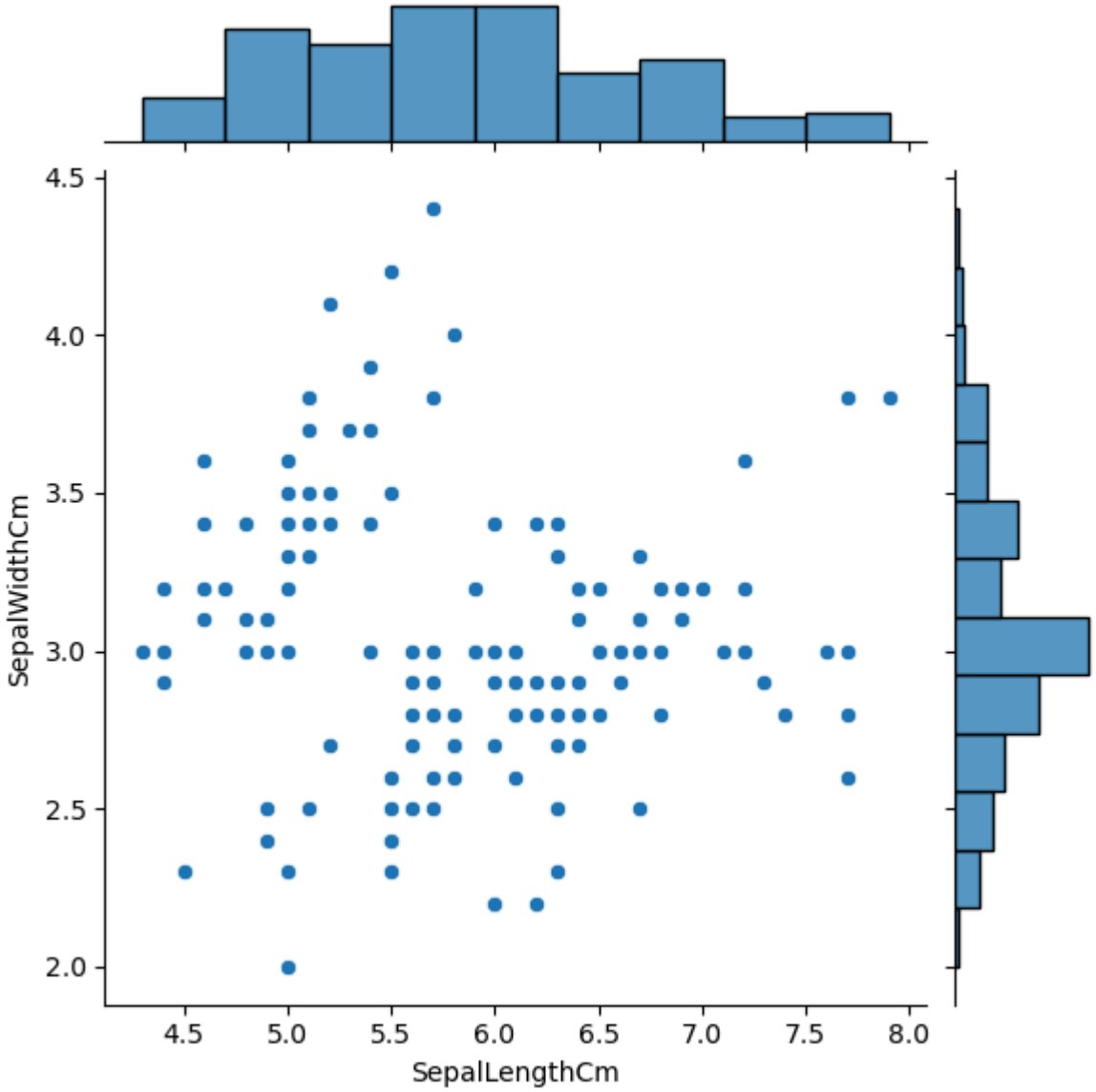
In [9]:
```
iris['Species'].value_counts()
```

Out[9]:
```
Iris-setosa        50
Iris-versicolor    50
Iris-virginica     50
Name: Species, dtype: int64
```

In [10]:
```
sns.countplot(x ='Species',data=iris)
plt.show()
```

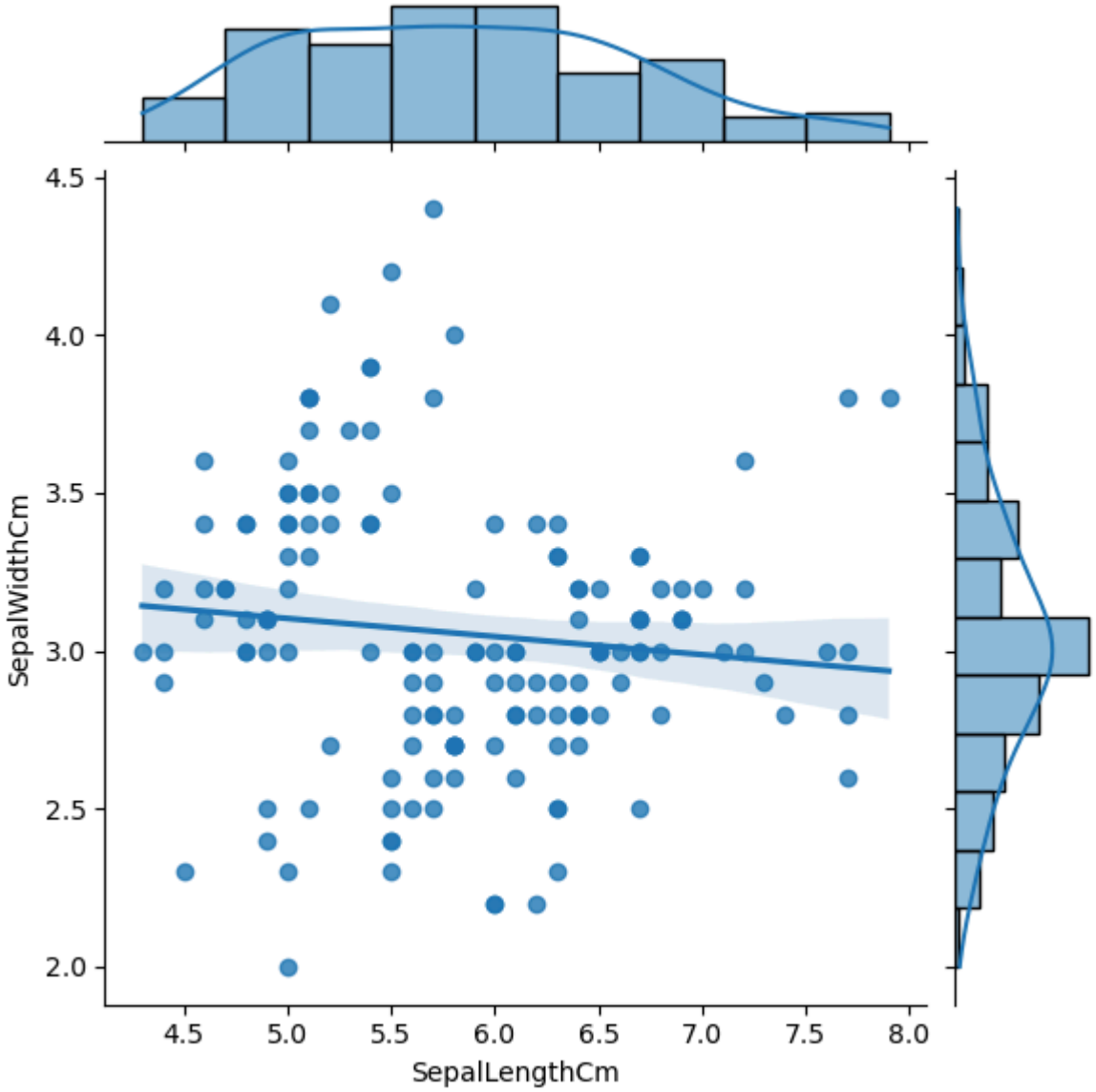



In [11]:
```
x = sns.jointplot(x='SepalLengthCm',y='SepalWidthCm',data=iris)
```

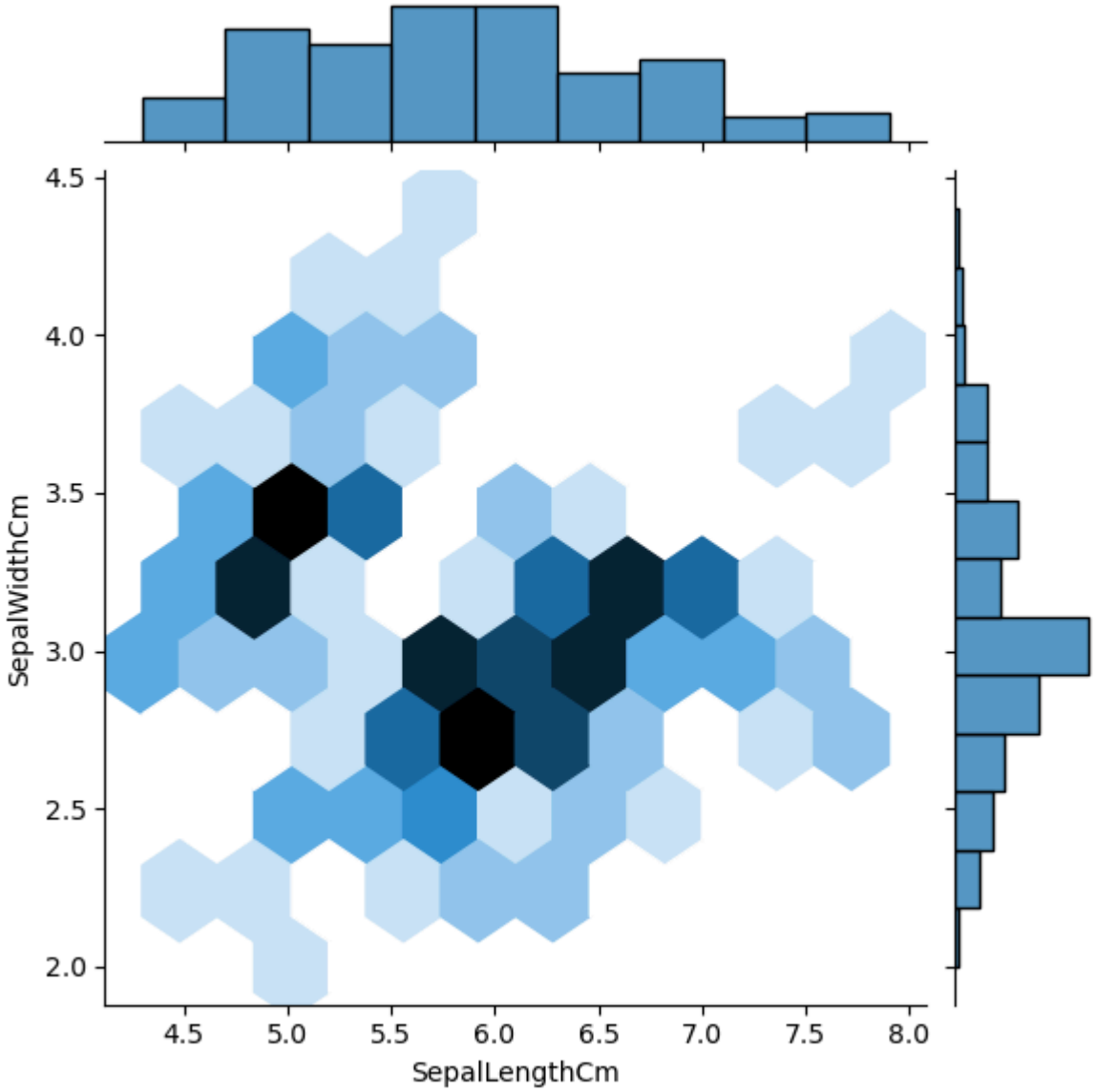



In [12]:
```
sns.jointplot(x = "SepalLengthCm", y = "SepalWidthCm", data=iris, kind="reg")
```

Out[12]:
```
<seaborn.axisgrid.JointGrid at 0x1d13534a150>
```

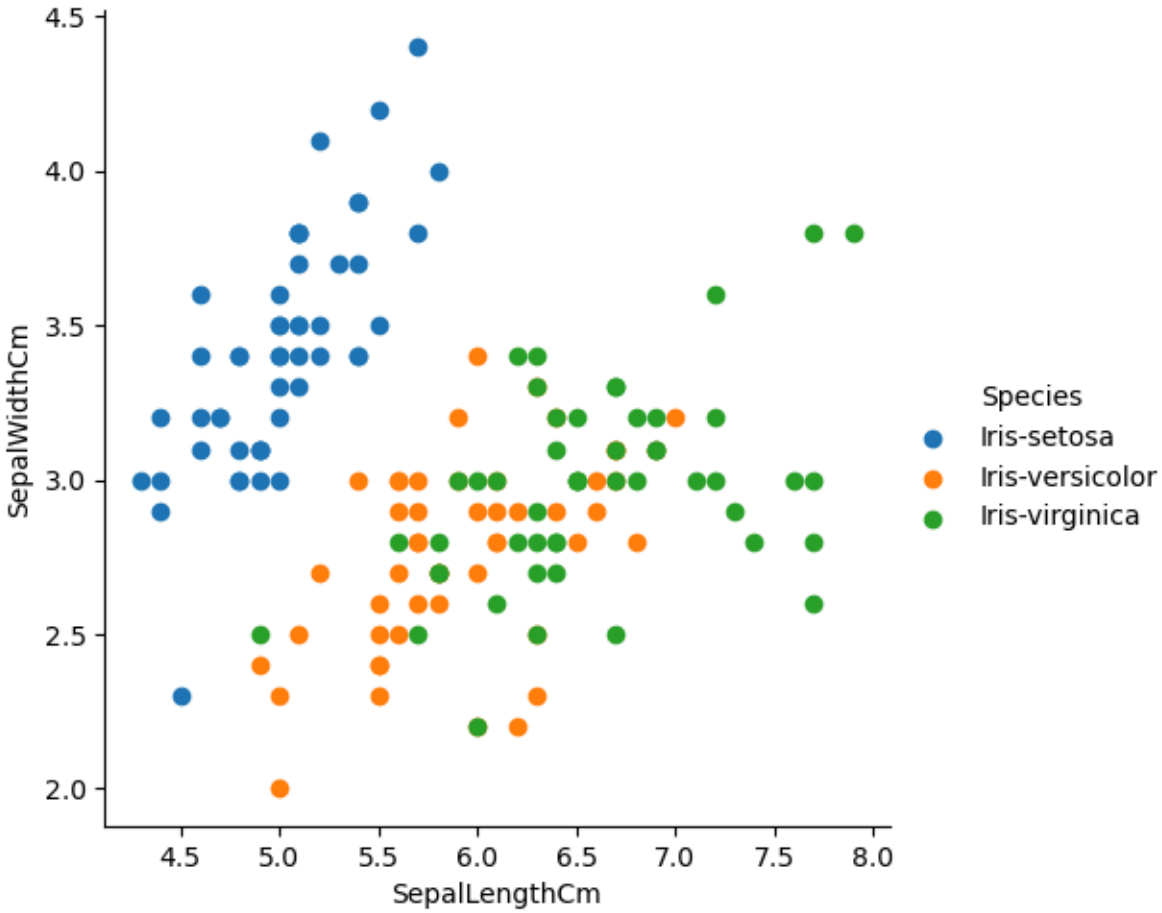



In [13]:
```
x1 = sns.jointplot(x='SepalLengthCm',y='SepalWidthCm',kind='hex',data=iris)
```

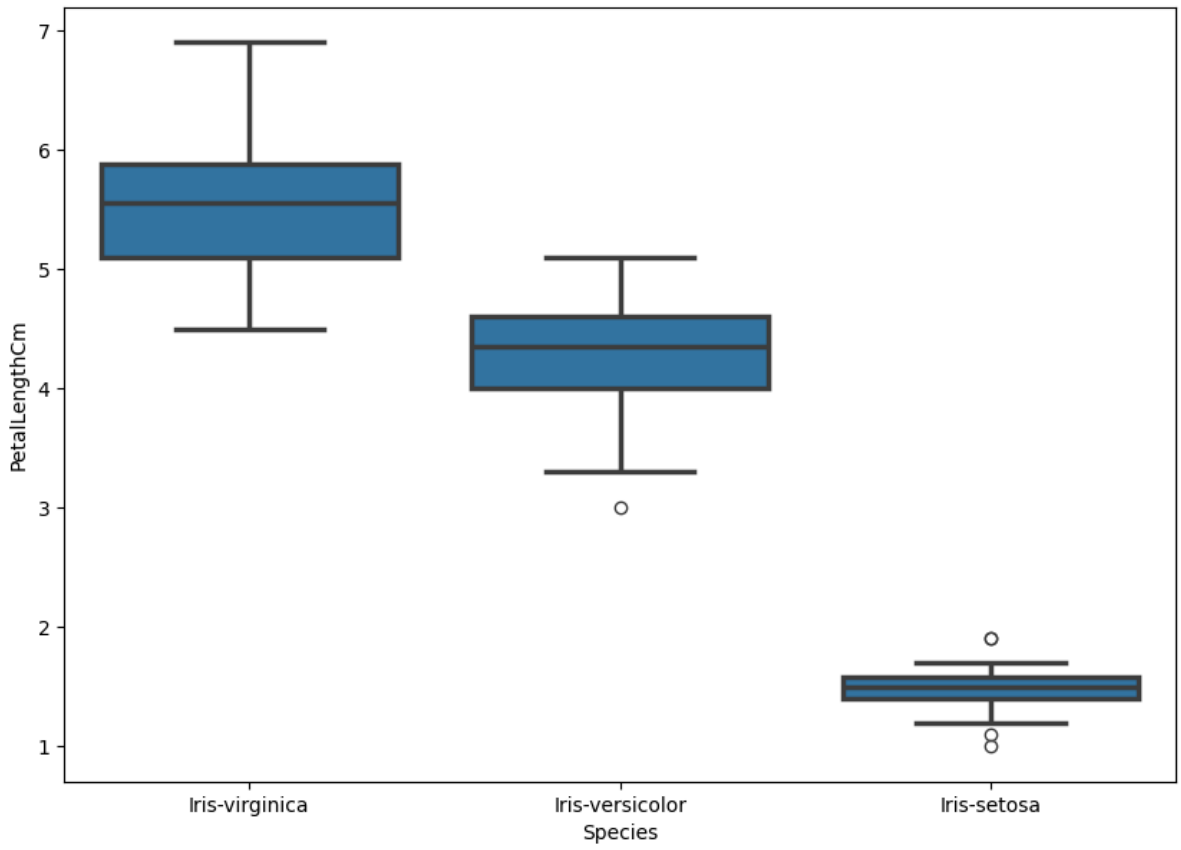



# FacetGrid Plot

In [14]:
```
g = sns.FacetGrid(iris, hue='Species', height=5)
g.map(plt.scatter, 'SepalLengthCm', 'SepalWidthCm')
g.add_legend()
plt.show()
```

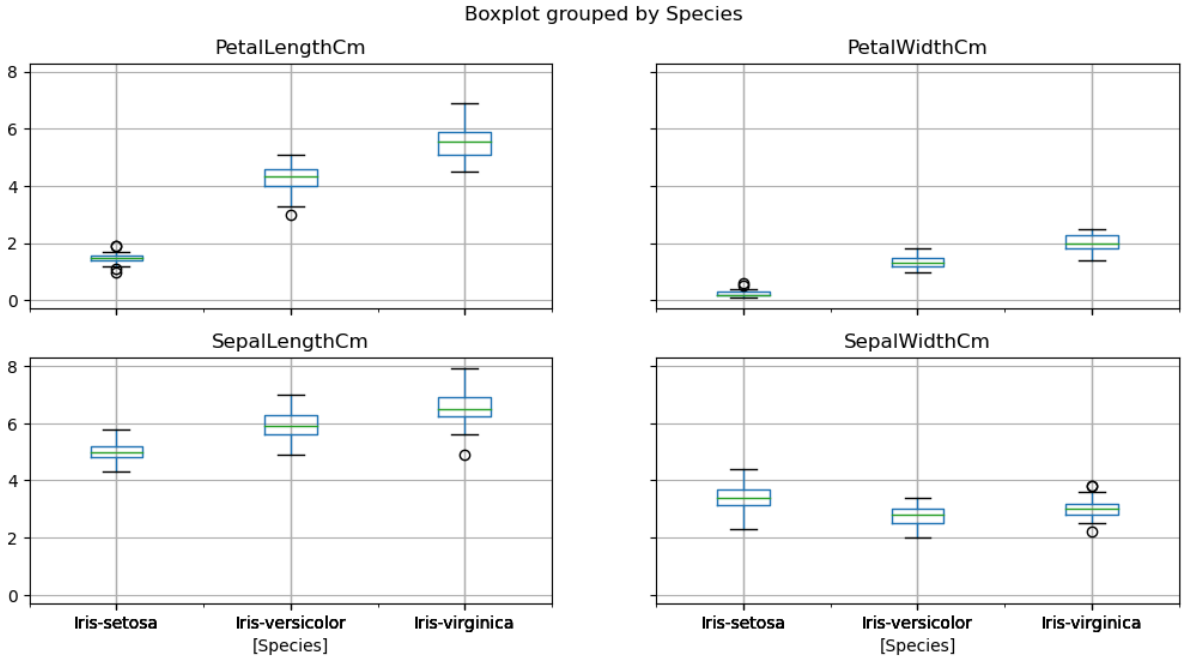



In [15]:
```
fig=plt.gcf()
fig.set_size_inches(10,7)
fig=sns.boxplot(x='Species',y='PetalLengthCm',data=iris,order=['Iris-virginica','Ir
```



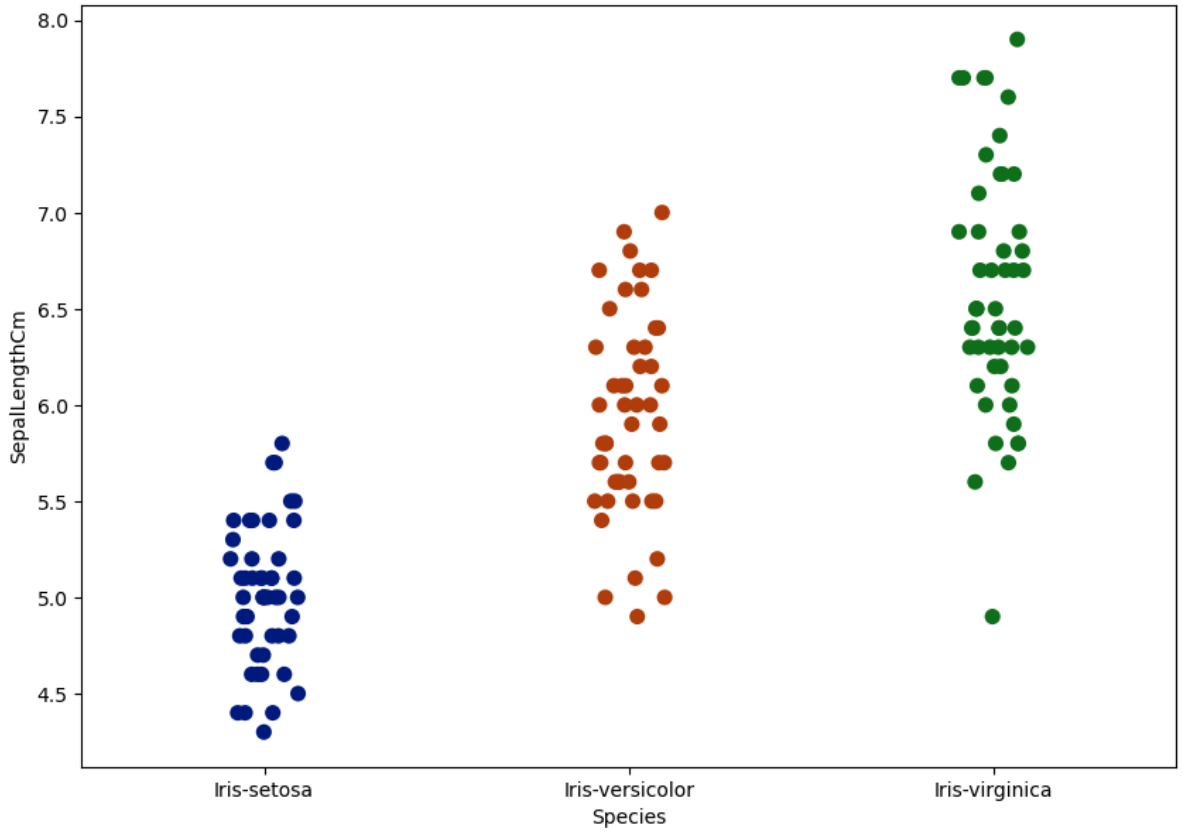

In [16]:
```
iris.boxplot(by="Species", figsize=(12, 6))
```

Out[16]:
```
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)
```



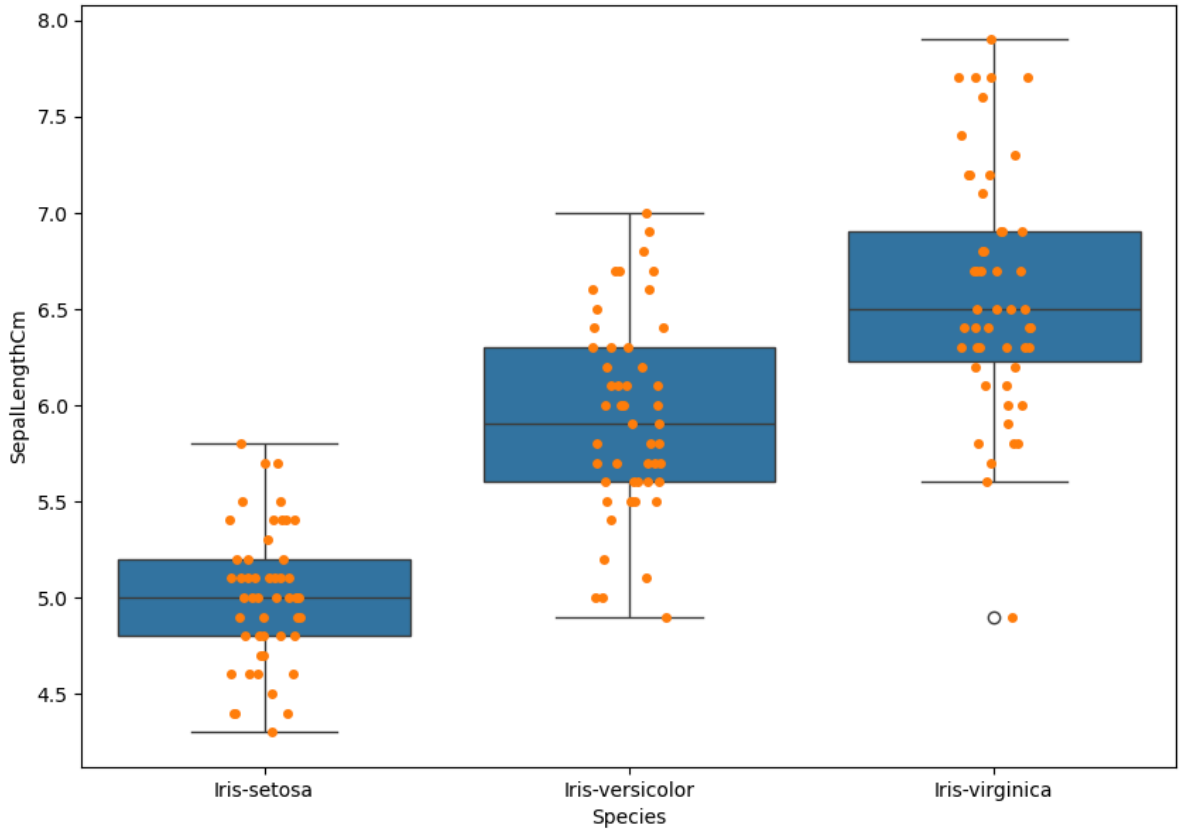

# Strip Plot

In [17]:
```
fig=plt.gcf()
fig.set_size_inches(10,7)
fig=sns.stripplot(x='Species',y='SepalLengthCm',data=iris,jitter=True,edgecolor='bl
```

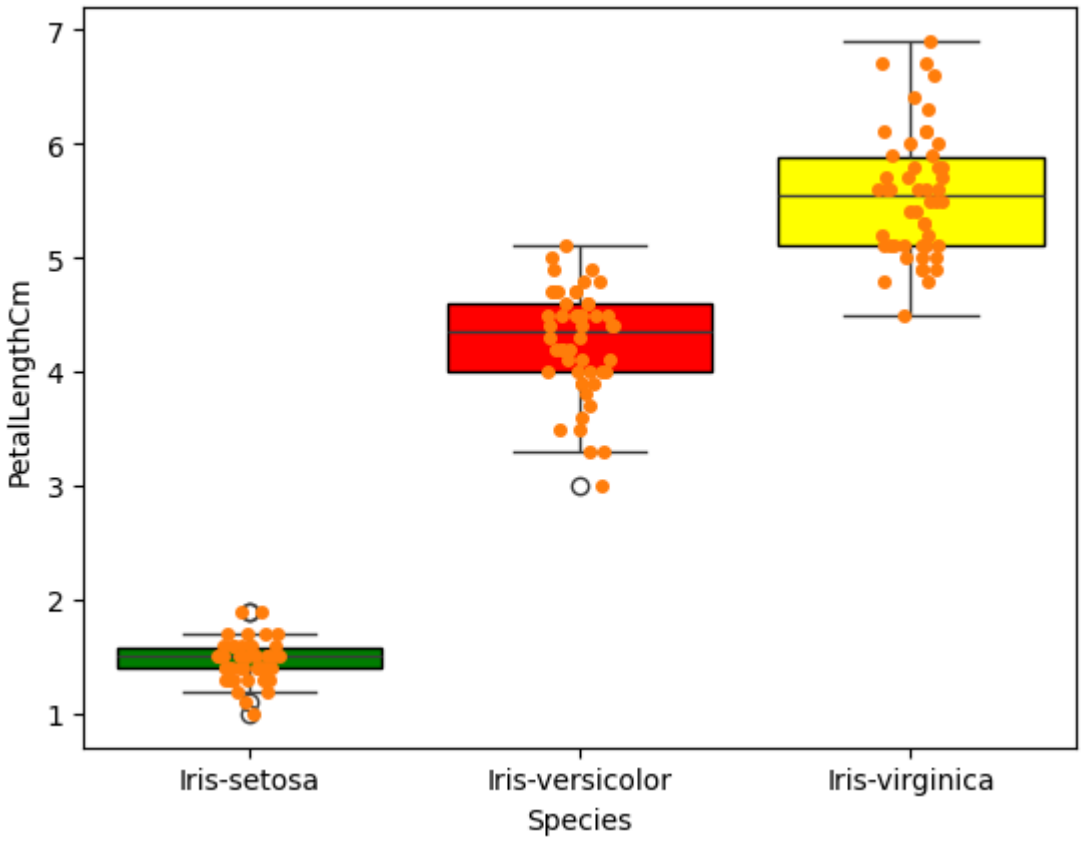


# Combining Box And Strip Plots

In [18]:
```
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='bl
```

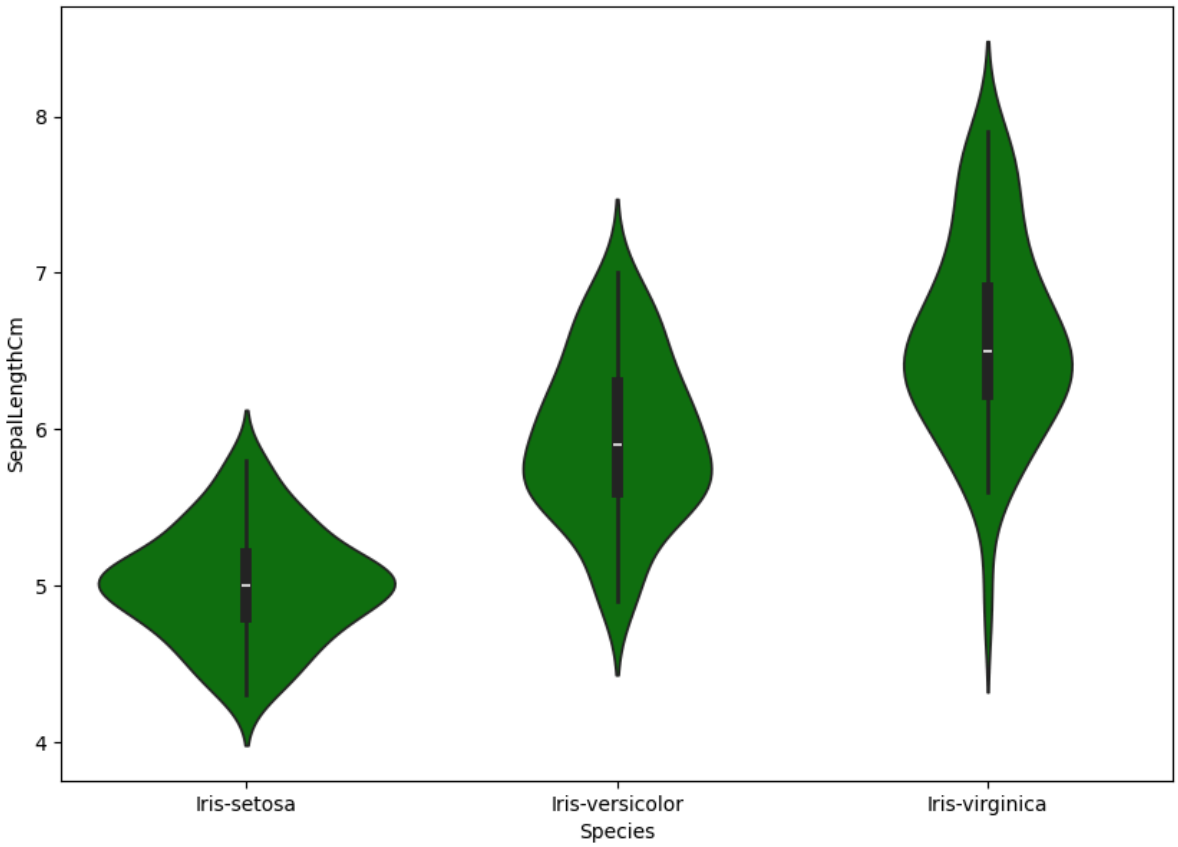



In [19]:
```
ax = sns.boxplot(x="Species", y="PetalLengthCm", data=iris)

# Add stripplot on same axes
sns.stripplot(x="Species", y="PetalLengthCm", data=iris,jitter=True, edgecolor="bla

# Get all box patches
boxes = ax.patches   # ✅ correct for new Seaborn/Matplotlib

# Change colors
boxes[0].set_facecolor('green')
boxes[0].set_edgecolor('black')

boxes[1].set_facecolor('red')
boxes[1].set_edgecolor('black')

boxes[2].set_facecolor('yellow')
boxes[2].set_edgecolor('black')

plt.show()
```



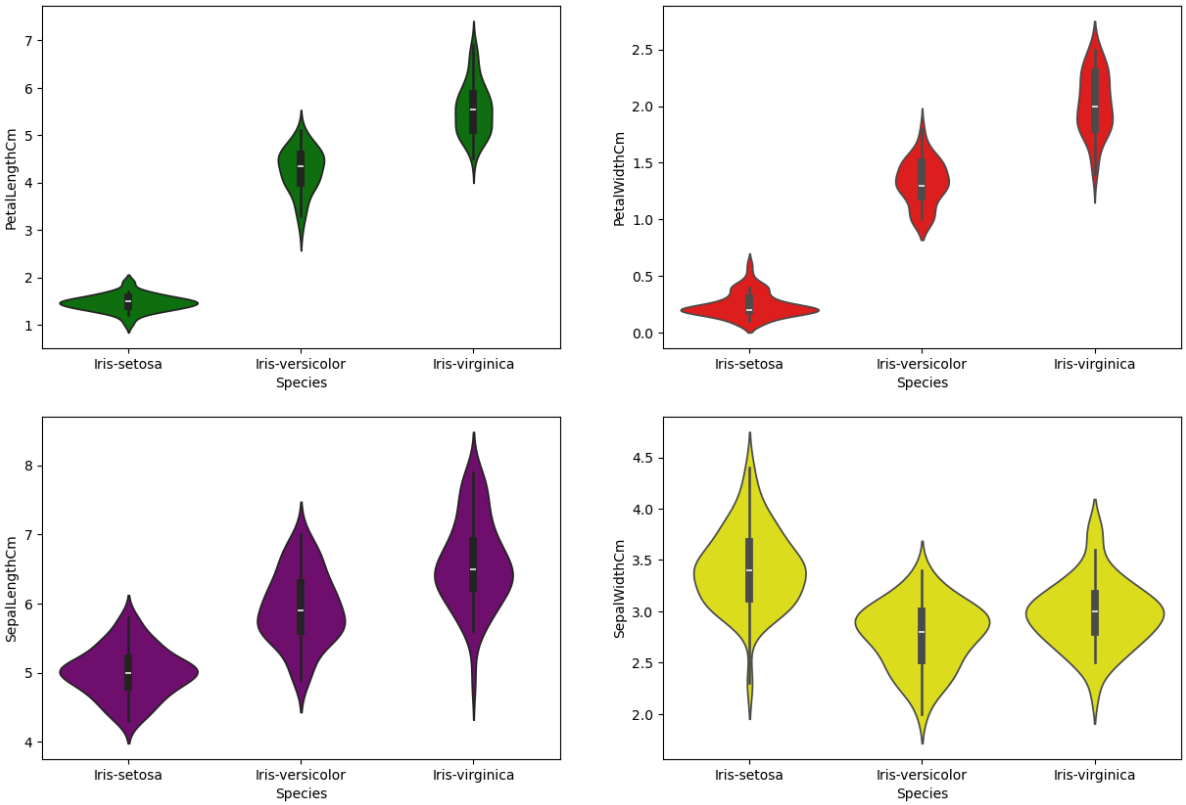

# Violin Plot

In [20]:
```
plt.figure(figsize=(10, 7))
sns.violinplot(x='Species', y='SepalLengthCm', data=iris, color='green')
plt.show()
```



In [21]:
```
plt.figure(figsize=(15,10))
plt.subplot(2,2,1)
sns.violinplot(x='Species',y='PetalLengthCm',data=iris,color='green')
plt.subplot(2,2,2)
sns.violinplot(x='Species',y='PetalWidthCm',data=iris, color='red')
plt.subplot(2,2,3)
sns.violinplot(x='Species',y='SepalLengthCm',data=iris, color='purple')
plt.subplot(2,2,4)
sns.violinplot(x='Species',y='SepalWidthCm',data=iris,color='yellow')
```

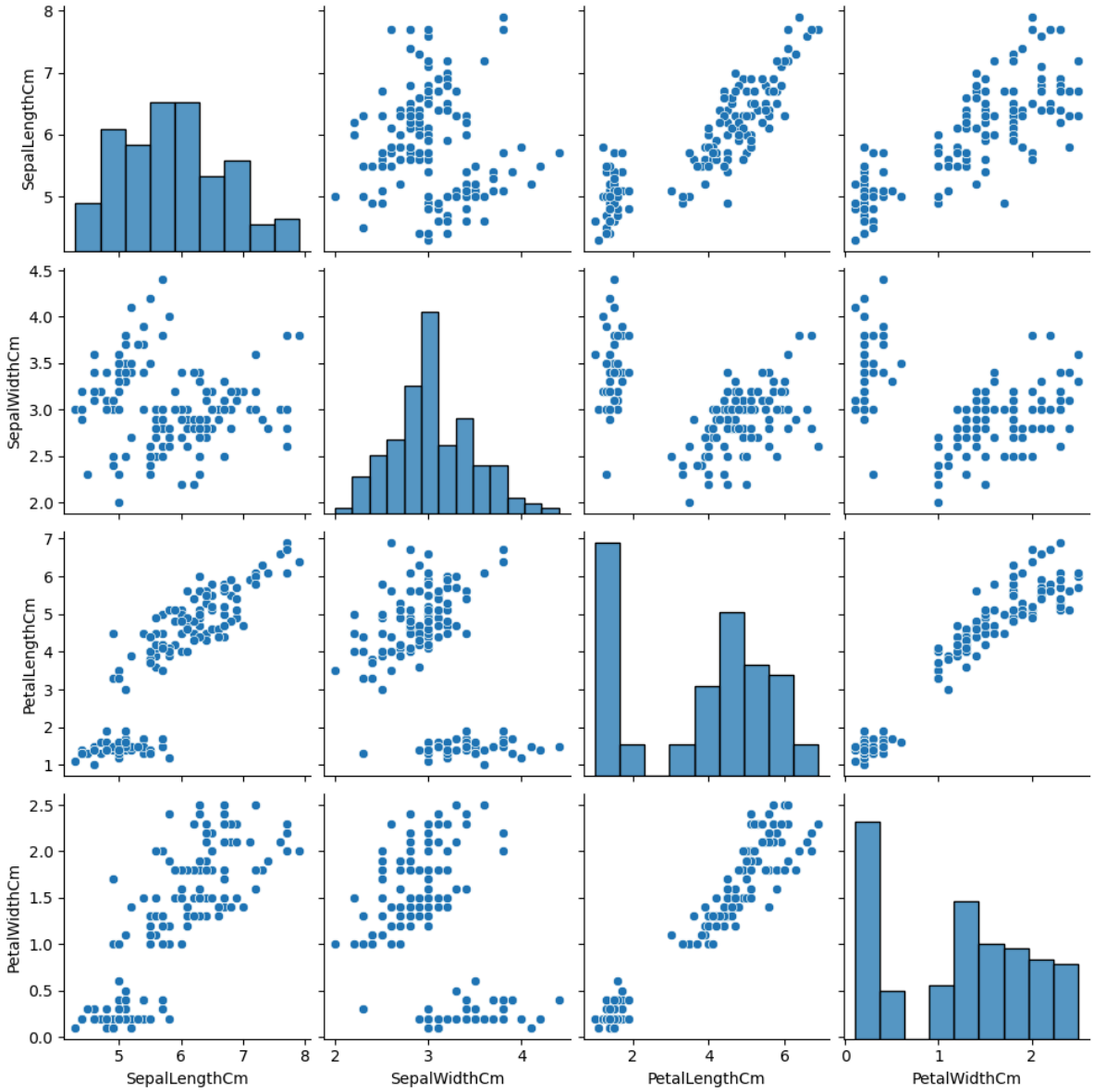
Out[21]:
```
<Axes: xlabel='Species', ylabel='SepalWidthCm'>
```



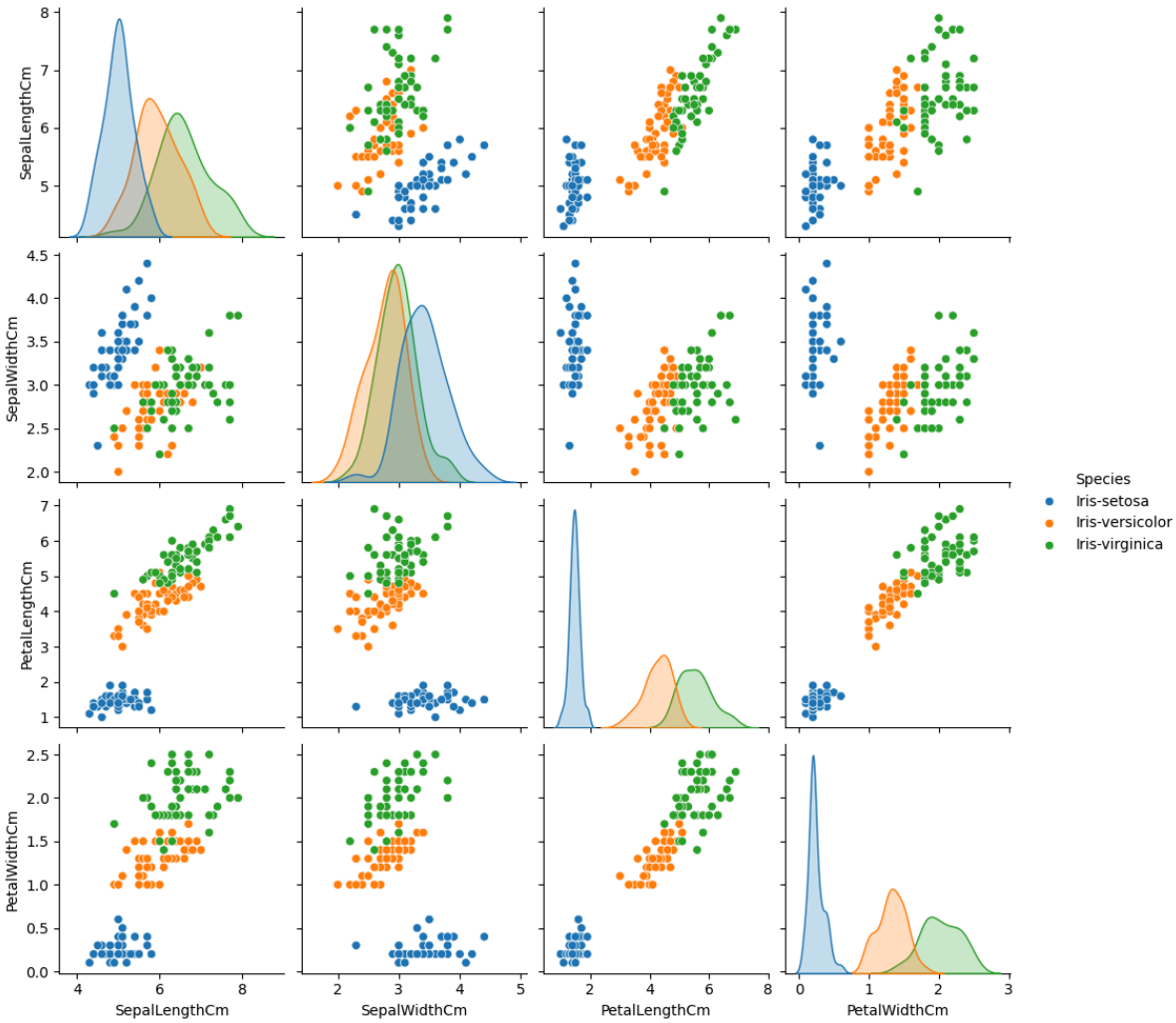

# Pair Plot

In [22]:
```
sns.pairplot(data=iris,kind='scatter')
```

Out[22]:
```
<seaborn.axisgrid.PairGrid at 0x1d1375e8610>
```

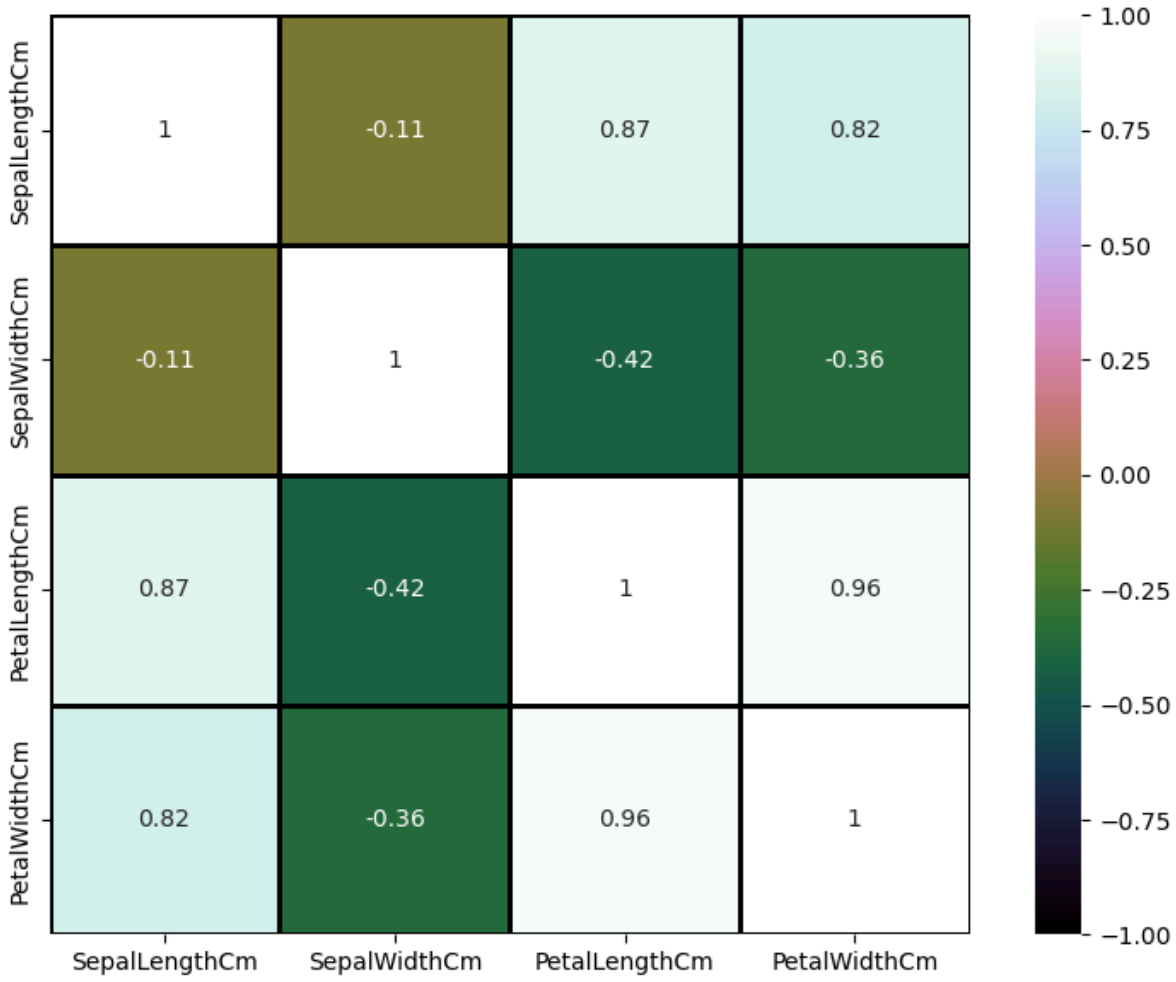



In [23]:
```
sns.pairplot(iris,hue='Species');
```



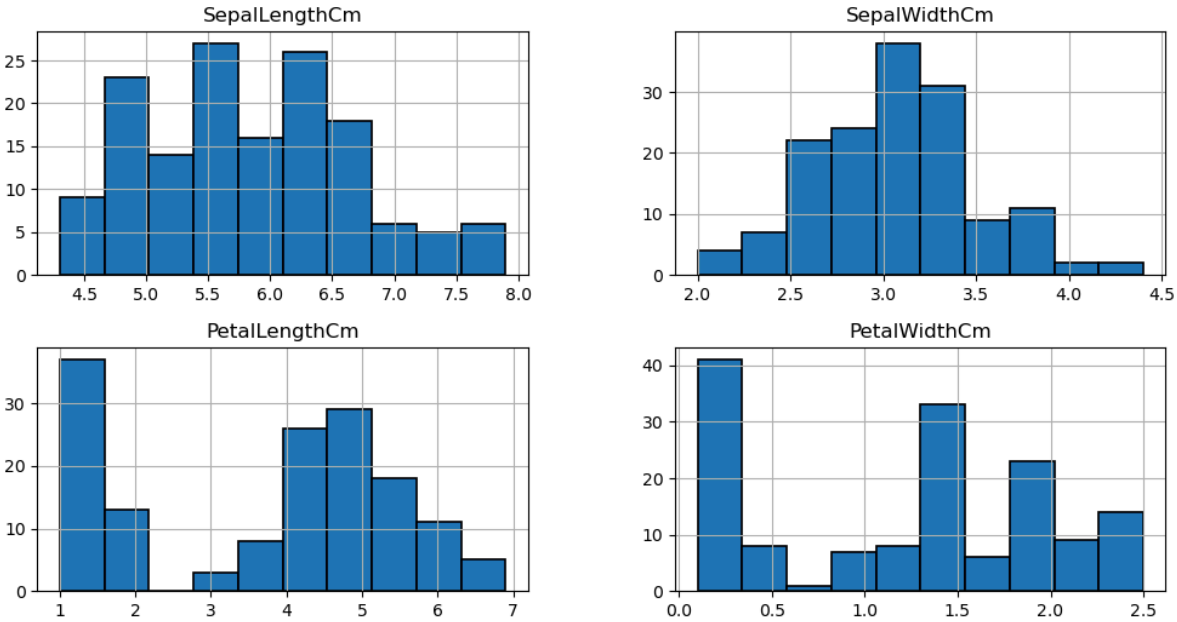

# Heat Map

In [24]:
```
fig=plt.gcf()
fig.set_size_inches(10,7)
fig=sns.heatmap(iris.corr(),annot=True,cmap='cubehelix',linewidths=1,linecolor='k',
```



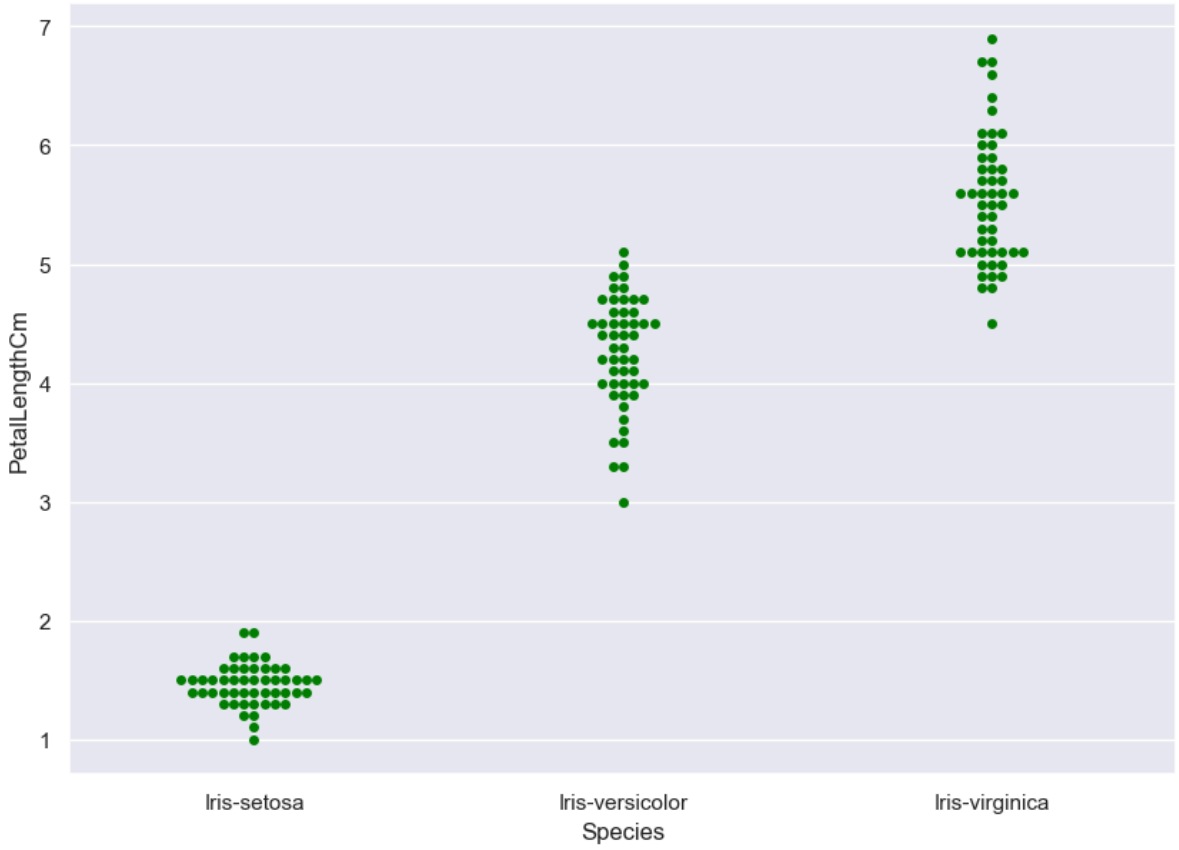

# Distribution Plot

In [25]:
```
iris.hist(edgecolor='black', linewidth=1.2)
fig=plt.gcf()
fig.set_size_inches(12,6)
```

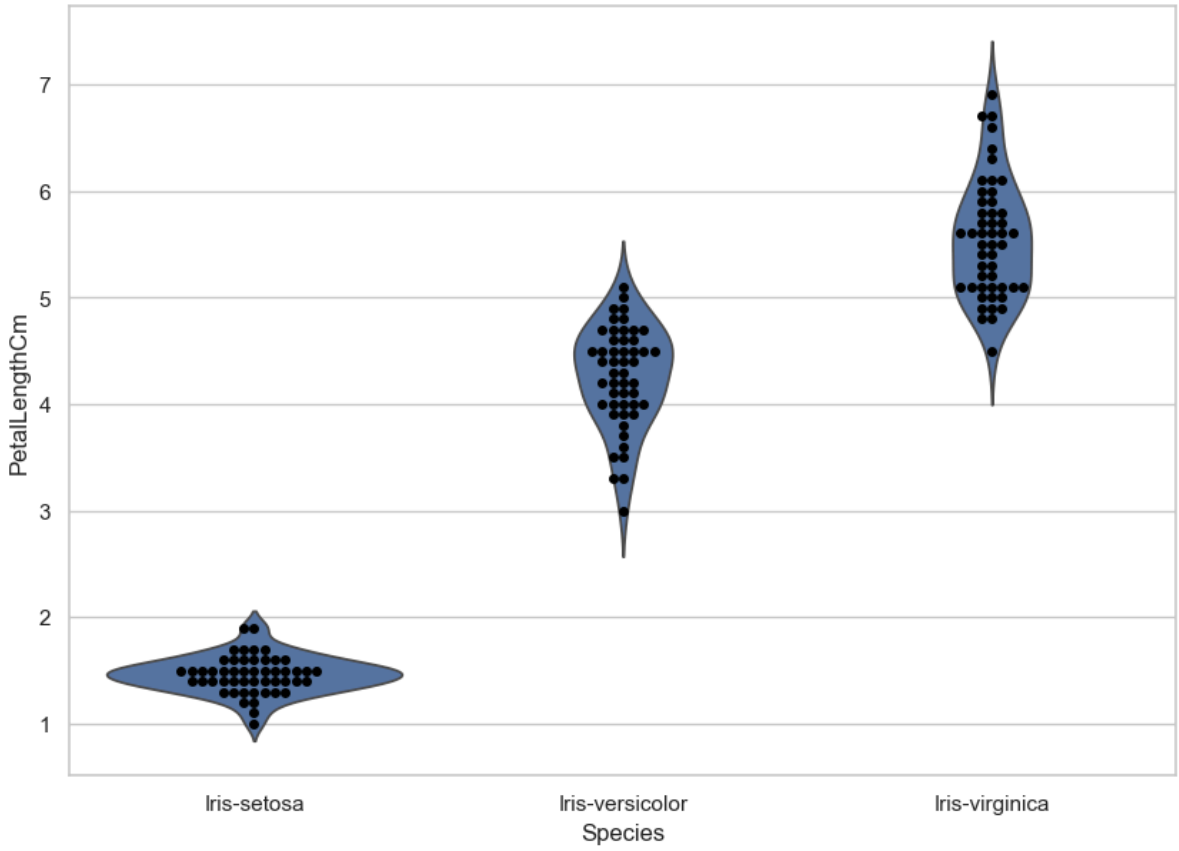



# Swarm Plot

In [26]:
```
sns.set(style="darkgrid")
fig=plt.gcf()
fig.set_size_inches(10,7)
fig = sns.swarmplot(x="Species", y="PetalLengthCm", data=iris, color='green')
```

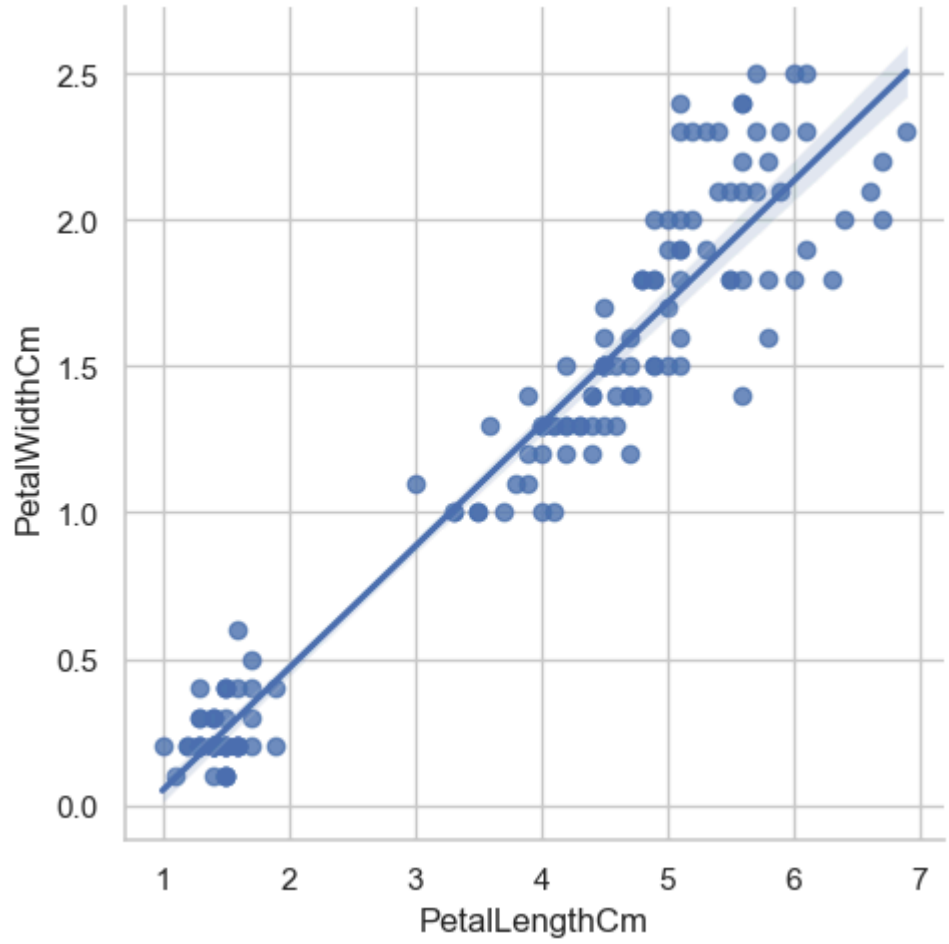



In [27]:
```
sns.set(style="whitegrid")
fig=plt.gcf()
fig.set_size_inches(10,7)
ax = sns.violinplot(x="Species", y="PetalLengthCm", data=iris, inner=None)
ax = sns.swarmplot(x="Species", y="PetalLengthCm", data=iris,color="black", edgecol
```




# LM Plot

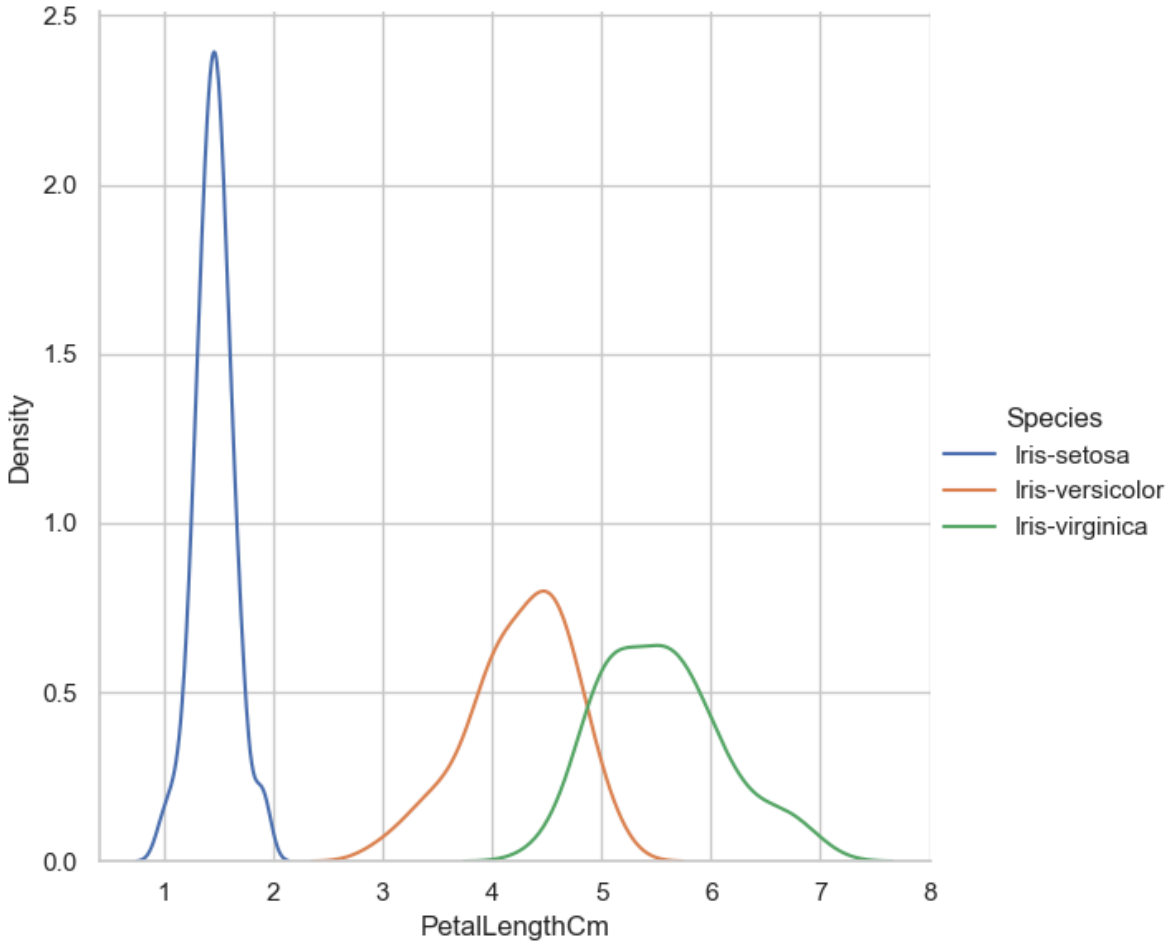
In [28]:
```
fig=sns.lmplot(x="PetalLengthCm", y="PetalWidthCm",data=iris)
```



# FacetGrid

In [29]:
```
g = sns.FacetGrid(iris, hue="Species", height=6)
g.map(sns.kdeplot, "PetalLengthCm")
g.add_legend()
```

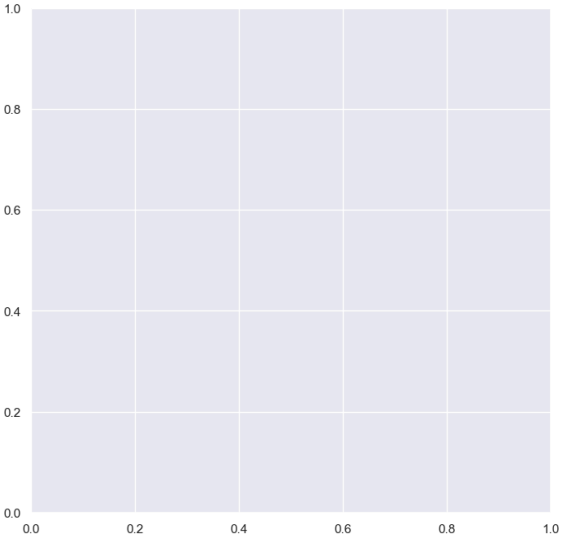
Out[29]:
```
<seaborn.axisgrid.FacetGrid at 0x1d13684ed90>
```

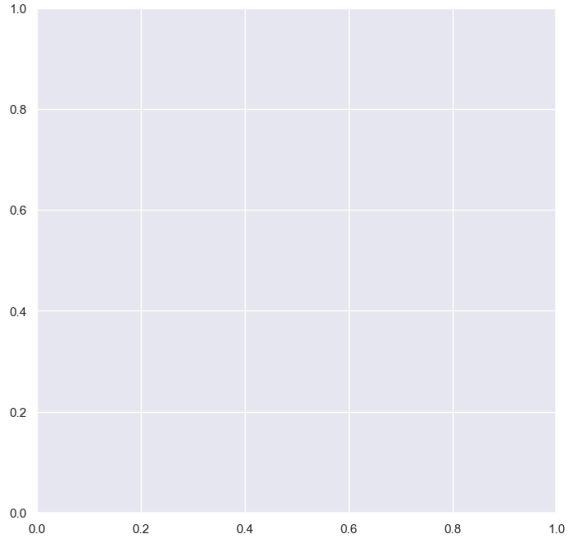



# Factor Plot

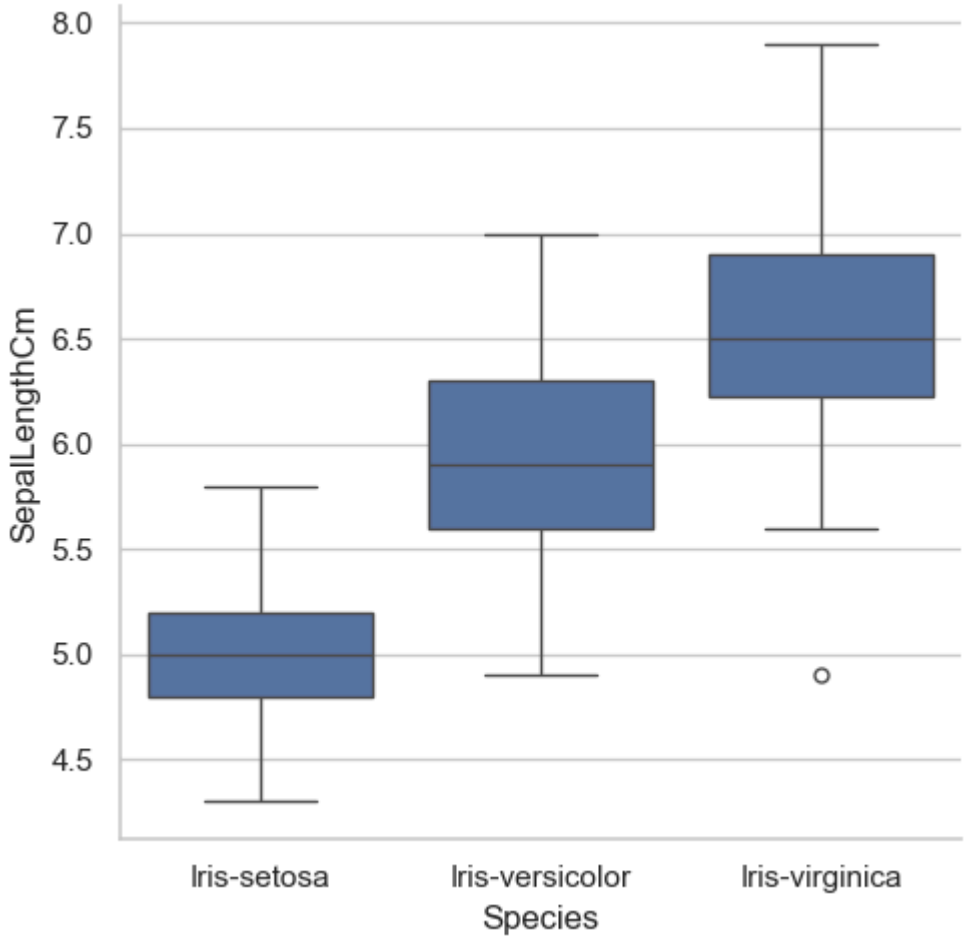
In [52]:
```
sns.factorplot('Species','SepalLengthCm',data=iris)
plt.ioff()
plt.show()
```

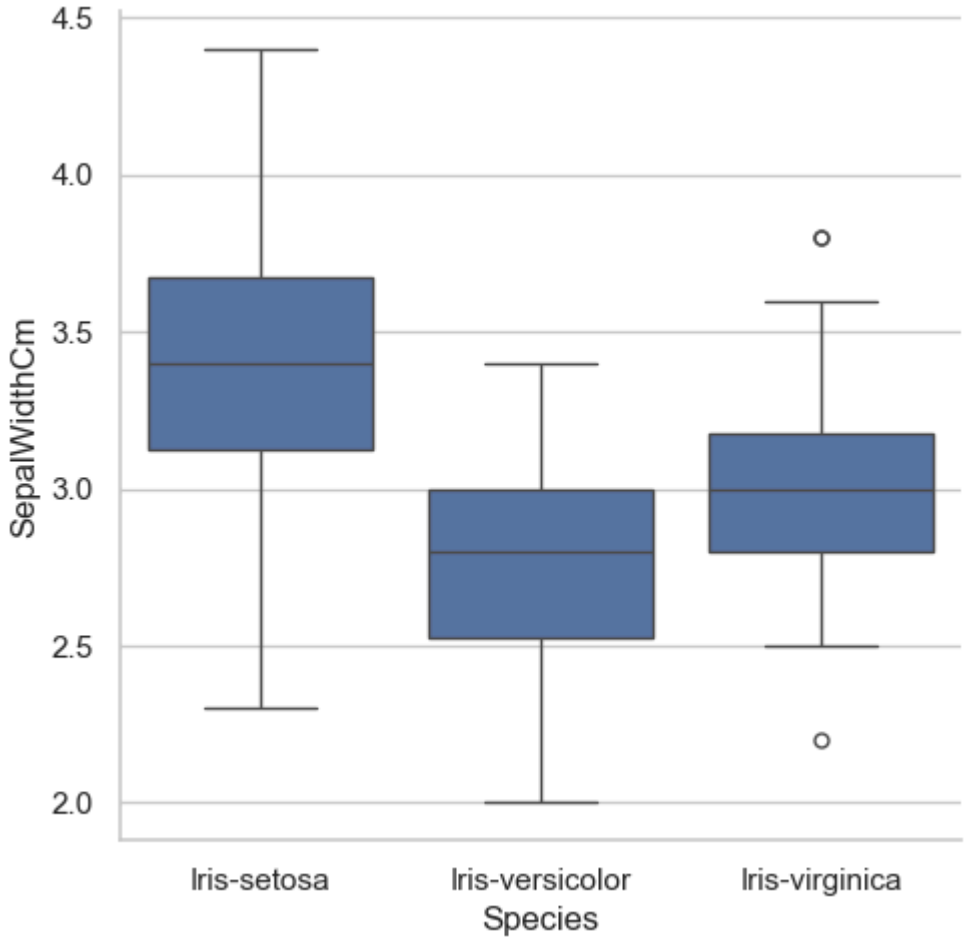
```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
Cell In[52], line 2
      1 f,ax=plt.subplots(1,2,figsize=(18,8))
----> 2 sns.factorplot('Species','SepalLengthCm',data=iris)
      3 plt.ioff()
      4 plt.show()

AttributeError: module 'seaborn' has no attribute 'factorplot'
```

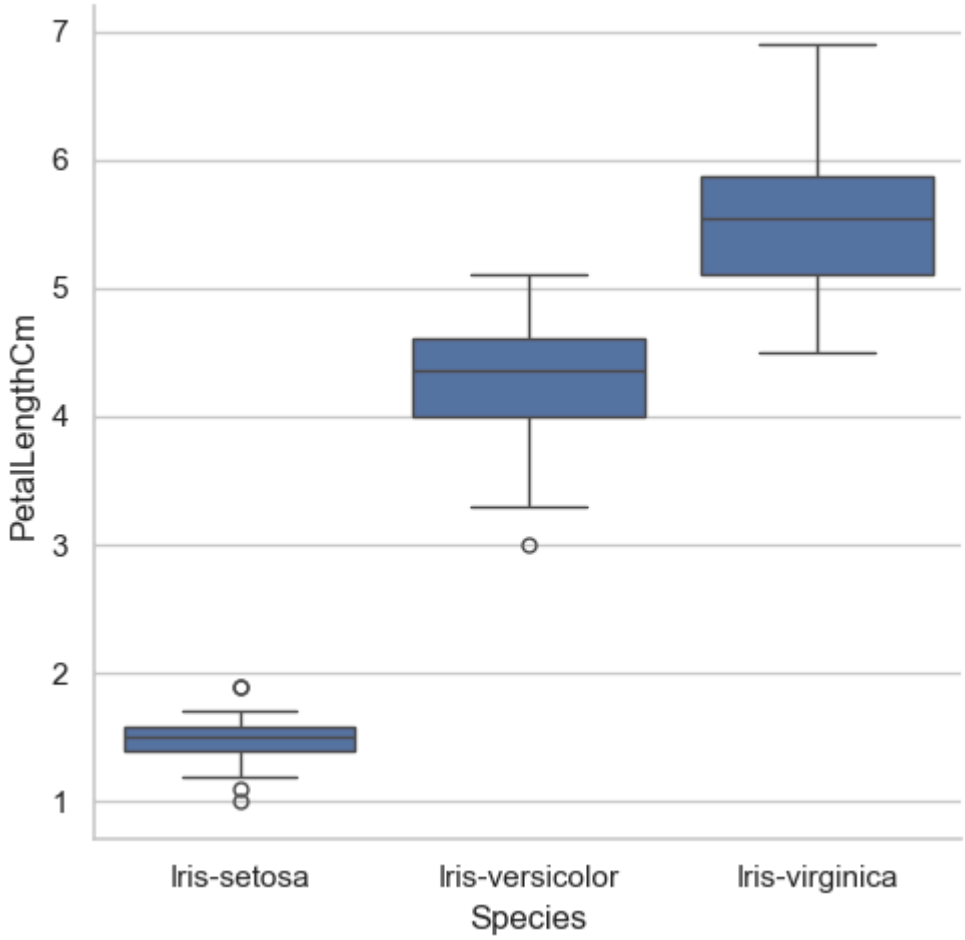


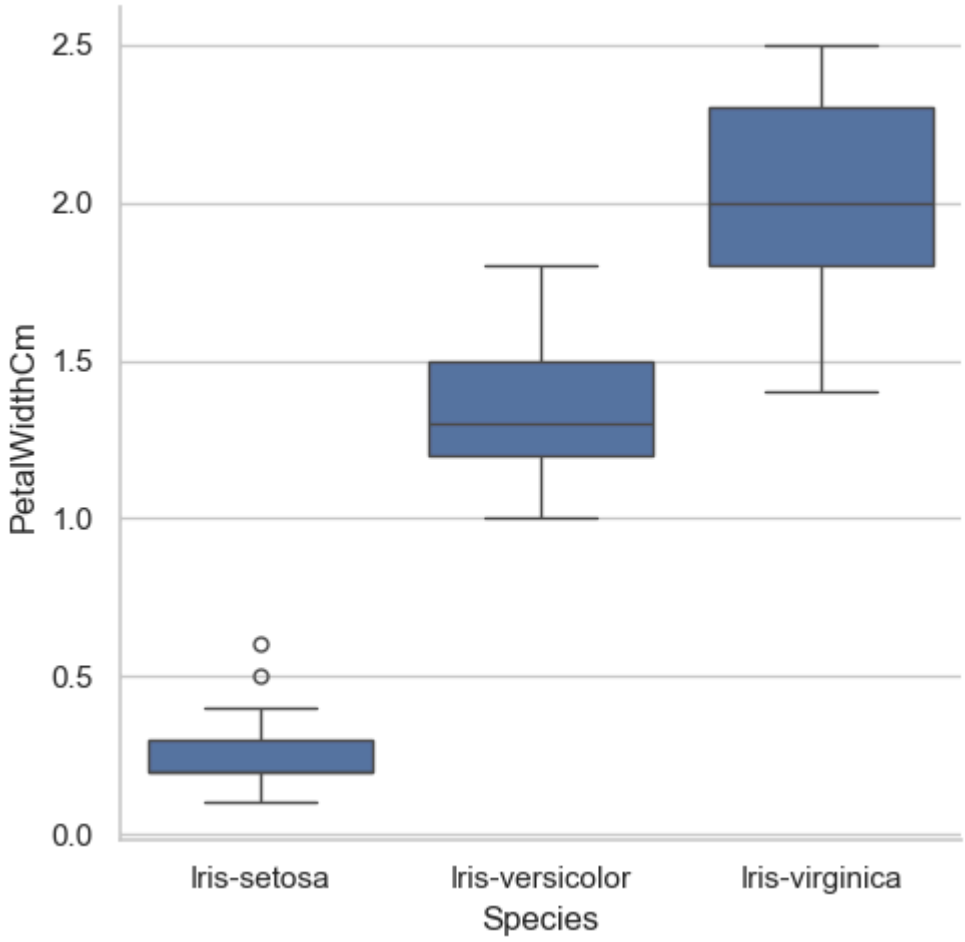




In [30]:
```
sns.catplot(x='Species', y='SepalLengthCm', data=iris, kind='box')
sns.catplot(x='Species', y='SepalWidthCm',  data=iris, kind='box')
sns.catplot(x='Species', y='PetalLengthCm', data=iris, kind='box')
sns.catplot(x='Species', y='PetalWidthCm',  data=iris, kind='box')
plt.show()
```

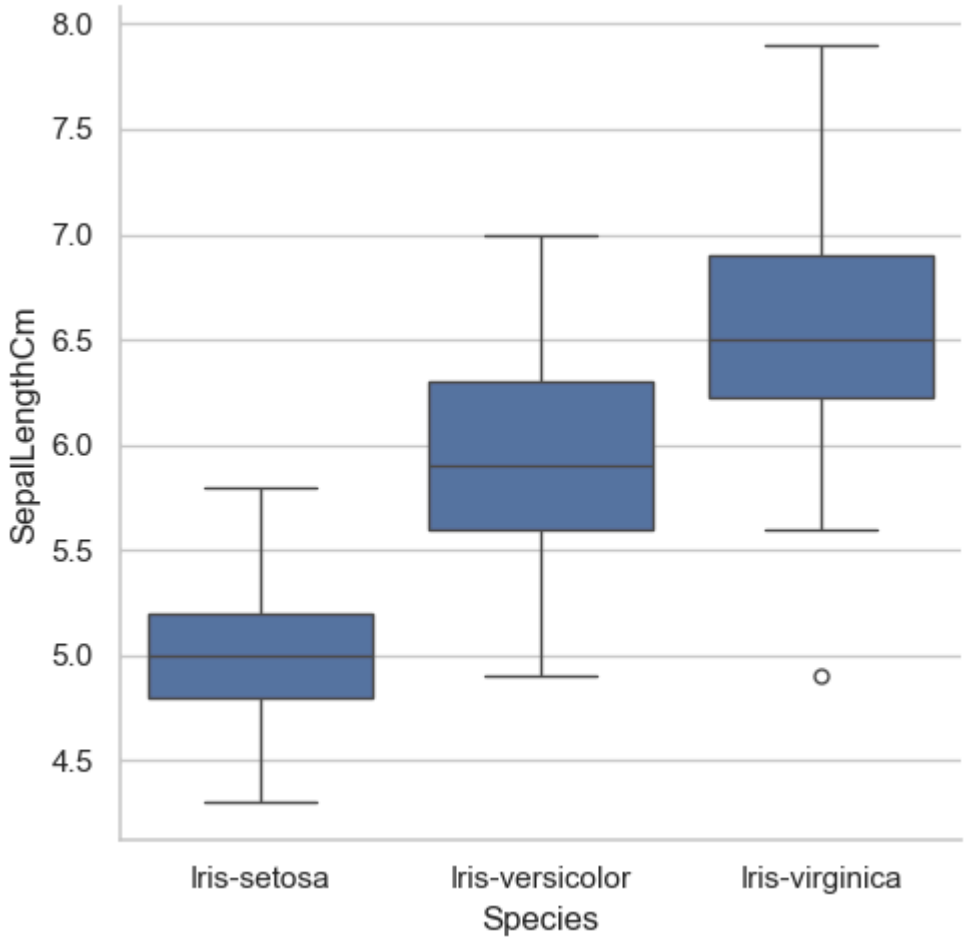



4.5
4.0
3.5
3.0
2.5
2.0
SepalWidthCm
Iris-setosa
Iris-versicolor
Iris-virginica
Species


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

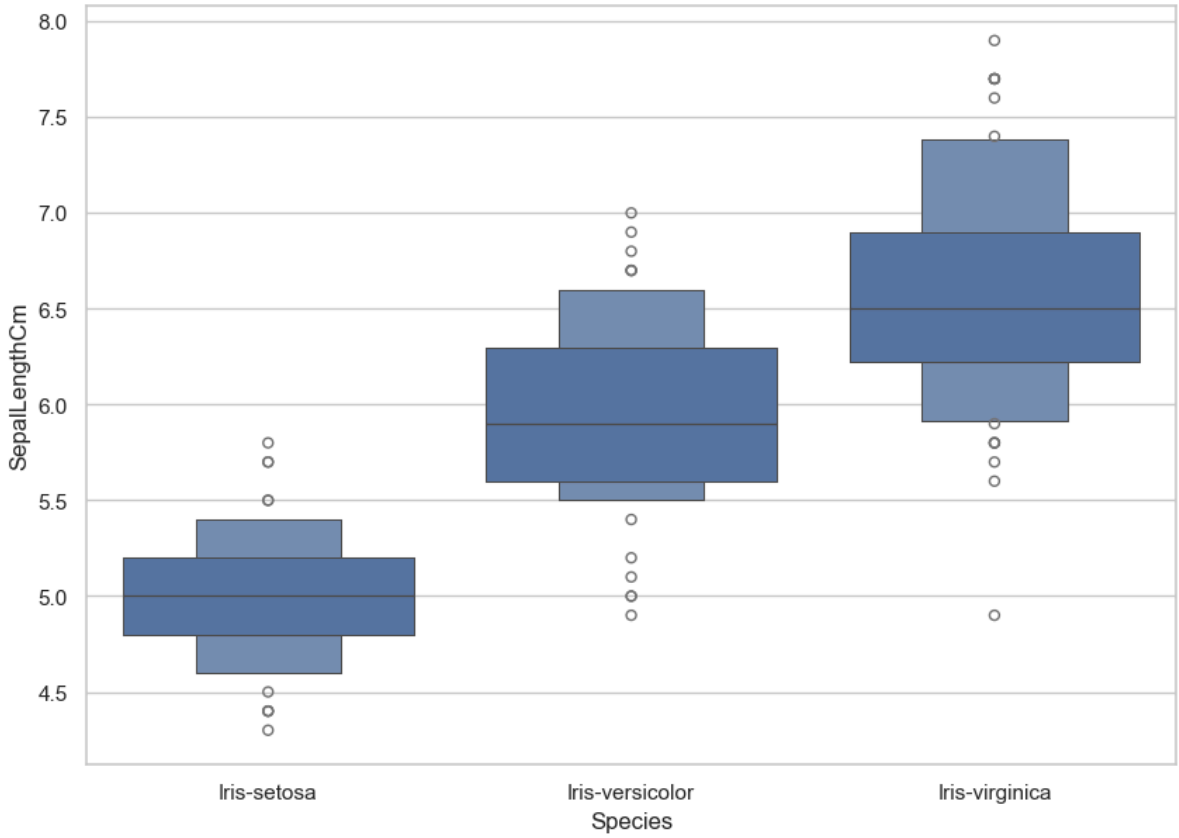



In [31]:
```
sns.catplot(x='Species', y='SepalLengthCm', data=iris, kind='box')
plt.show()
```



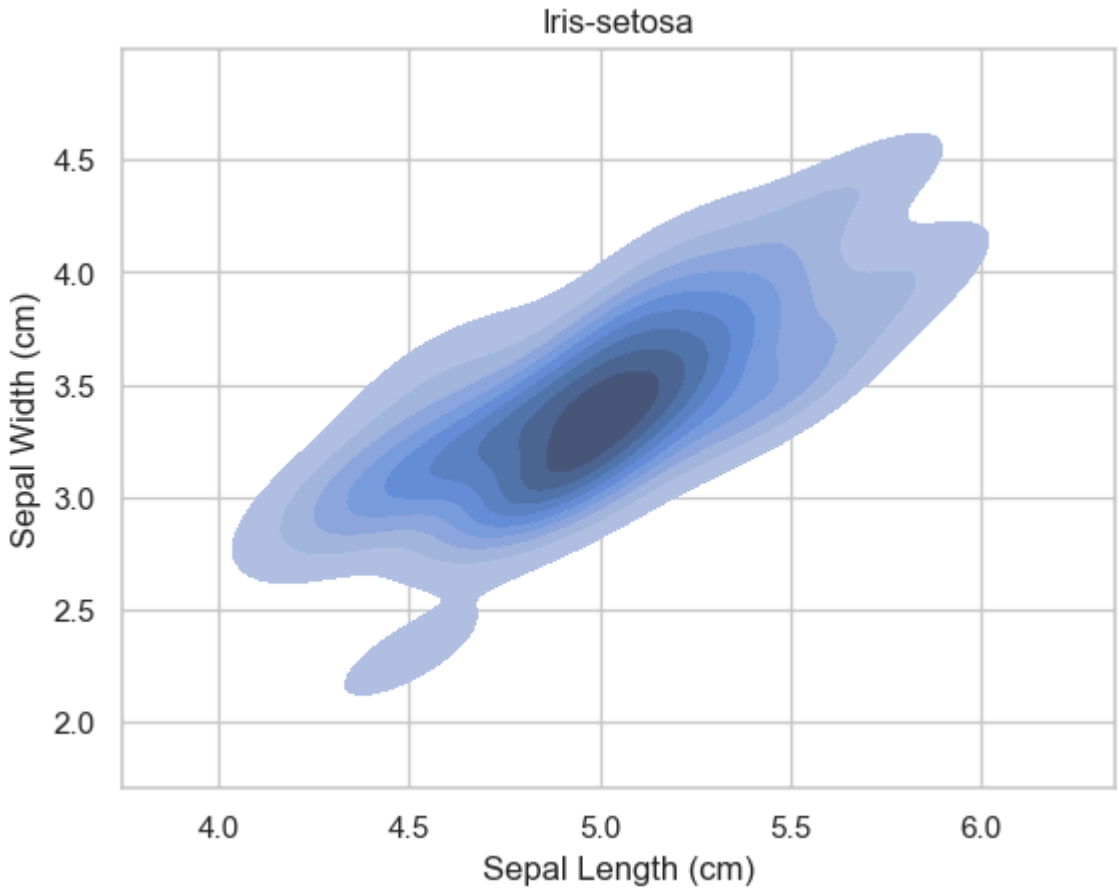
# Boxen Plot

In [32]:
```
fig=plt.gcf()
fig.set_size_inches(10,7)
fig=sns.boxenplot(x='Species',y='SepalLengthCm',data=iris)
```

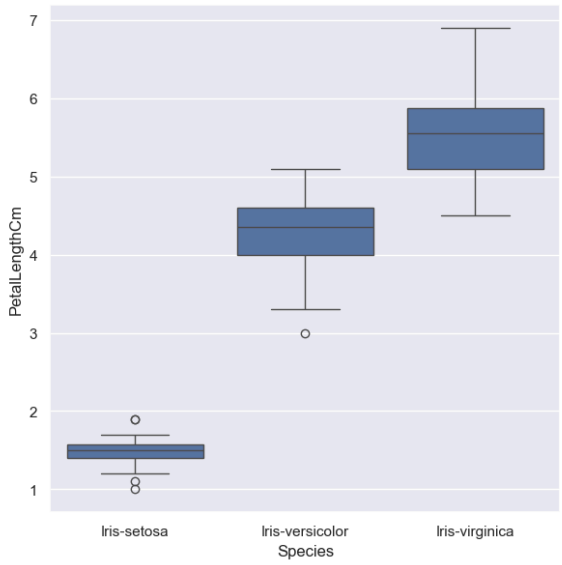


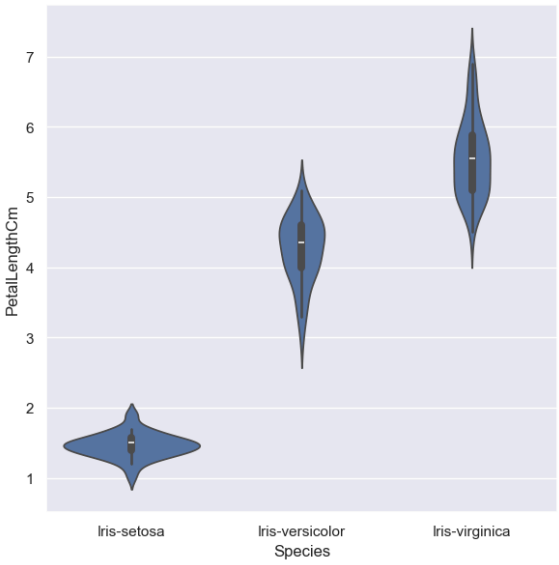

# KDE Plot

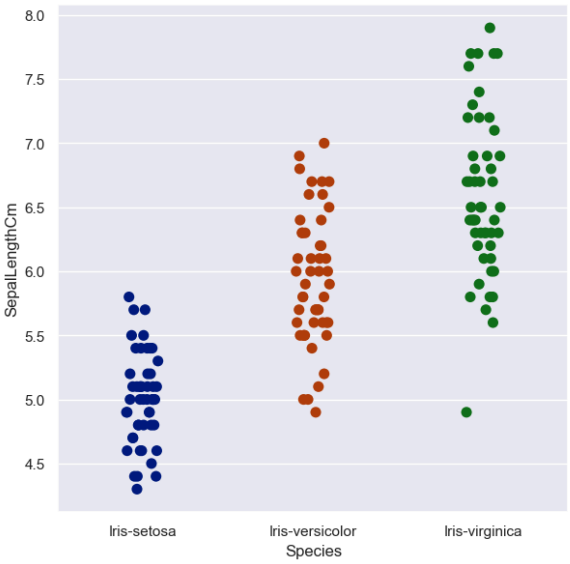
In [33]:
```
sub = iris[iris['Species'] == 'Iris-setosa']
sns.kdeplot(x='SepalLengthCm',y='SepalWidthCm',data=sub,fill=True)
plt.title('Iris-setosa')
plt.xlabel('Sepal Length (cm)')
plt.ylabel('Sepal Width (cm)')
plt.show()
```

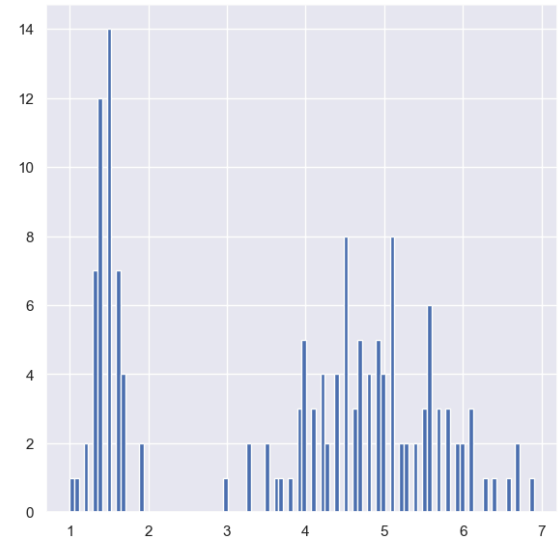



In [35]:
```
sns.set_style('darkgrid')
f,axes=plt.subplots(2,2,figsize=(15,15))

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='gra
#axes[1,1].hist(iris.hist,bin=10)
axes[1,1].hist(iris.PetalLengthCm,bins=100)
#k2.set(xlim=(-1,0.8))
plt.show()
```



# Stacked Histogram

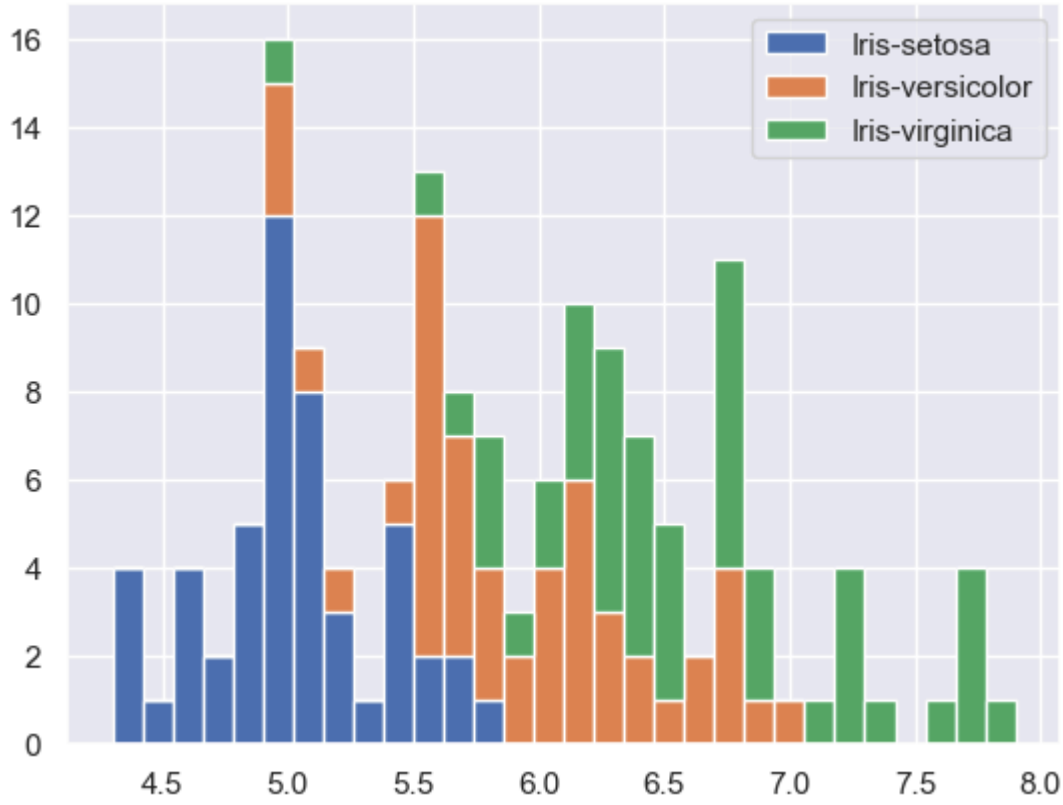
In [47]:
```
iris['Species'] = iris['Species'].astype('category')
iris.head()
```

Out[47]:

| | 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 |

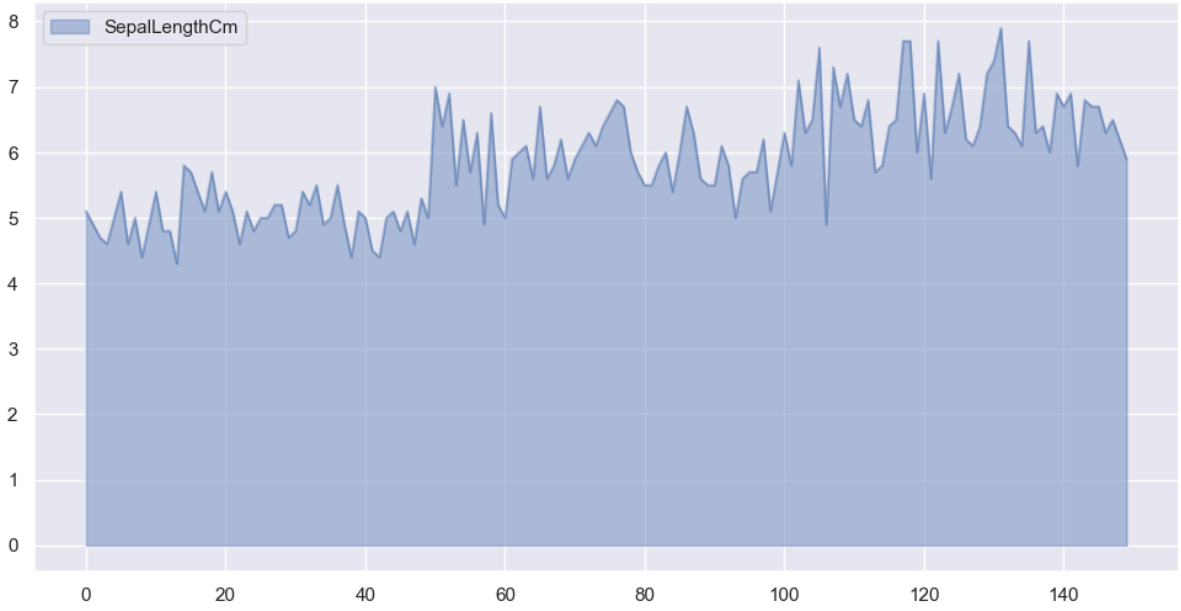
In [48]:
```
list1=list()
mylabels=list()
for gen in iris.Species.cat.categories:
    list1.append(iris[iris.Species==gen].SepalLengthCm)
    mylabels.append(gen)

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

```
plt.legend()
plt.show()
```



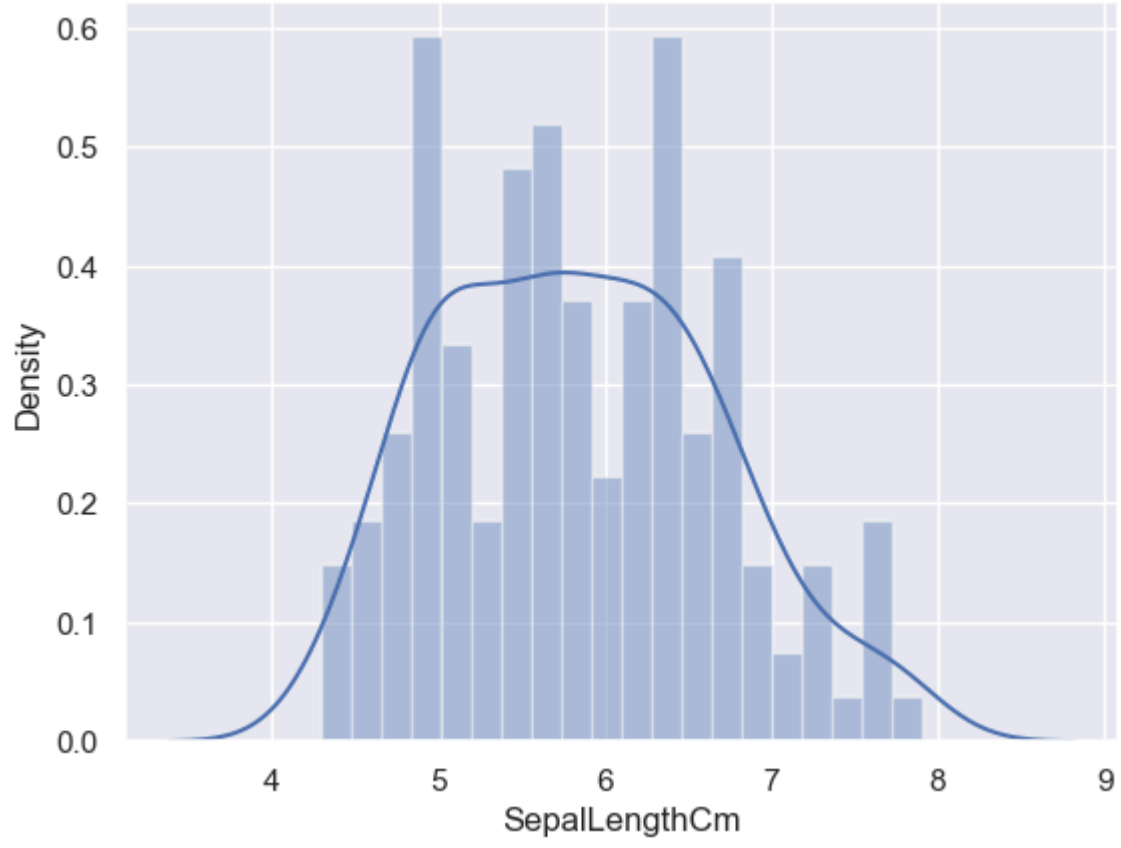

# Area Plot

In [50]:
```
iris['SepalLengthCm'] = iris['SepalLengthCm'].astype('float')
iris.plot.area(y='SepalLengthCm', alpha=0.4, figsize=(12, 6))
iris.plot.area(y=['SepalLengthCm','SepalWidthCm','PetalLengthCm','PetalWidthCm'], a
plt.show()
```




# Distplot

In [39]:
```
sns.distplot(iris['SepalLengthCm'],kde=True,bins=20);
```




In [ ]:

In [ ]: