Ex.No.3

WORKING WITH PANDAS DATA FRAMES

Aim:

Write a python program to work with Panda data frames.

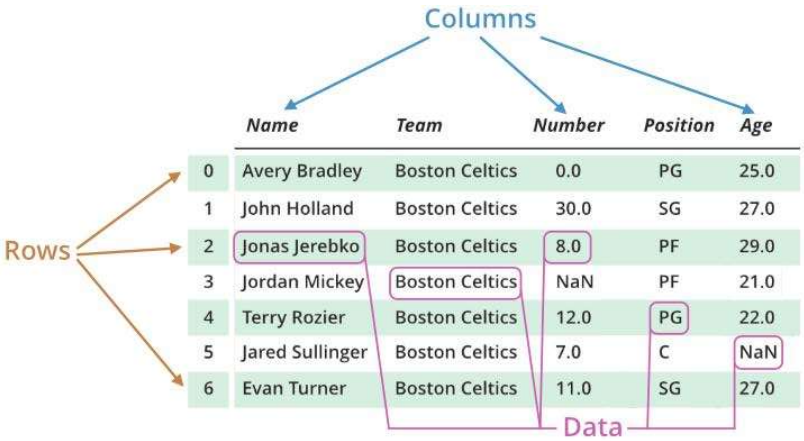
Pandas

Pandas is an open-source library that is built on top of NumPy library. It is a Python package that offers various data structures and operations for manipulating numerical data and time series. It is mainly popular for importing and analyzing data much easier. Pandas is fast and it has high-performance & productivity for users.

Pandas DataFrame

In the real world, a Pandas DataFrame will be created by loading the datasets from existing storage, storage can be SQL Database, CSV file, and Excel file. Pandas DataFrame can be created from the lists, dictionary, and from a list of dictionary etc.

Pandas DataFrame is two-dimensional size-mutable, potentially heterogeneous tabular data structure with labeled axes (rows and columns). A Data frame is a two-dimensional data structure, i.e., data is aligned in a tabular fashion in rows and columns. Pandas DataFrame consists of three principal components, the **data**, **rows**, and **columns**.



	Name	Team	Number	Position	Age
0	Avery Bradley	Boston Celtics	0.0	PG	25.0
1	John Holland	Boston Celtics	30.0	SG	27.0
2	Jonas Jerebko	Boston Celtics	8.0	PF	29.0
3	Jordan Mickey	Boston Celtics	NaN	PF	21.0
4	Terry Rozier	Boston Celtics	12.0	PG	22.0
5	Jared Sullinger	Boston Celtics	7.0	C	NaN
6	Evan Turner	Boston Celtics	11.0	SG	27.0

Creating a Panda Data Frames

A pandas DataFrame can be created using the following constructor −

pandas.DataFrame(data, index, columns, dtype, copy)

.

Creating an empty dataframe :
A basic DataFrame, which can be created is an Empty Dataframe. An Empty Dataframe is created just by calling a dataframe constructor.

Creating a dataframe using List:

DataFrame can be created using a single list or a list of lists.

Creating dataframe from dict of ndarray/lists:

To create dataframe from dict of narray/list, all the narray must be of same length. If index is passed then the length index should be equal to the length of arrays. If no index is passed, then by default, index will be range(n) where n is the array length.

Iterating over rows :

In order to iterate over rows, we can use three function iteritems(), iterrows(), itertuples() . These three function will help in iteration over rows.

Program

```
import pandas as pd


# Calling DataFrame constructor
print("Empty dataframe")
df = pd.DataFrame()
print(df)
print("Dataframe creation using List")
# list of strings
lst = ['Geeks', 'For', 'Geeks', 'is', 'portal', 'for', 'Geeks']
# Calling DataFrame constructor on list
df = pd.DataFrame(lst)
print(df)
# initialise data of lists.
Data = {'Name':['Tom', 'nick', 'krish', 'jack'], 'Age':[20, 21, 19, 18]}
# Create dataframe
```

```
df = pd.DataFrame(Data)
# Print the output.
print(df)
print("Create dataframe from dictionoary of lists")
# dictionary of lists
dict = {'name':["aparna", "pankaj", "sudhir", "Geeku"],
        'Degree': ["MBA", "BCA", "M.Tech", "MBA"],
        'Score':[90, 40, 80, 98]}

# creating a dataframe from a dictionary
df = pd.DataFrame(dict)
print(df)
# iterating over rows using iterrows() function
for i, j in df.iterrows():
    print(i, j)
    print()
```

OUTPUT

```
Empty dataframe
Empty DataFrame
Columns: []
Index: []

Dataframe creation using List
        0
    Geeks
1     For
2   Geeks
3      is
4  portal
5     for
6   Geeks

      Tom  20
1   nick  21
2  krish  19
```

3 jack 18

Create dataframe from dictionoary of lists
```
     name Degree Score
 0 aparna     MBA      90
 1 pankaj     BCA      40
 2 sudhir M.Tech       80
 3  Geeku     MBA      98
```

```
0 name      aparna
Degree       MBA
Score         90
Name: 0, dtype: object
```

```
1 name      pankaj
Degree       BCA
Score         40
Name: 1, dtype: object
```

```
2 name      sudhir
Degree    M.Tech
Score         80
Name: 2, dtype: object
```

```
3 name      Geeku
Degree      MBA
Score        98
Name: 3, dtype: object
```

Pandas Dataframe visualization
Retrieving data from the web

In[1]:

```
import pandas as pd
url =
'https://github.com/chris1610/pbpython/blob/master/data/2018_Sales_Total_v2.xlsx?raw=True'
df = pd.read_excel(url)
df
```

OUTPUT

	account number	name	sku	quantity	unit price	ext price	date
0	740150	Barton LLC	B1-20000	39	86.69	3380.91	2018-01-01 07:21:51
1	714466	Trantow-Barrows	S2-77896	-1	63.16	-63.16	2018-01-01 10:00:47
2	218895	Kulas Inc	B1-69924	23	90.70	2086.10	2018-01-01 13:24:58
3	307599	Kassulke, Ondricka and Metz	S1-65481	41	21.05	863.05	2018-01-01 15:05:22
4	412290	Jerde-Hilpert	S2-34077	6	83.21	499.26	2018-01-01 23:26:55
...	...	...	...	...	...	...	...
1502	424914	White-Trantow	B1-69924	37	42.77	1582.49	2018-11-27 14:29:02
1503	424914	White-Trantow	S1-47412	16	65.58	1049.28	2018-12-19 15:15:41
1504	424914	White-Trantow	B1-86481	75	28.89	2166.75	2018-12-29 13:03:54
1505	424914	White-Trantow	S1-82801	20	95.75	1915.00	2018-12-22 03:31:36
1506	424914	White-Trantow	S2-83881	100	88.19	8819.00	2018-12-16 00:46:26

1507 rows × 7 columns

Pandas for retrieving data from the csv file

In[2]:

```
import pandas as pd

data = pd.read_csv(r'C:\Users\HI\Downloads\PythonDataScienceHandbook-
master\notebooks\data\iris.csv')
df = pd.DataFrame(data)

print (df)
```

Out[2]:

```
     sepallength  sepalwidth  petallength  petalwidth           class
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
..           ...         ...          ...         ...             ...
145          6.7         3.0          5.2         2.3  Iris-virginica
146          6.3         2.5          5.0         1.9  Iris-virginica
147          6.5         3.0          5.2         2.0  Iris-virginica
148          6.2         3.4          5.4         2.3  Iris-virginica
149          5.9         3.0          5.1         1.8  Iris-virginica

[150 rows x 5 columns]
```

Questions:

1) Write a Pandas program to select the specified columns and rows from a given data frame. Sample Python dictionary data and list labels:
Select 'name' and 'score' columns in rows 1, 3, 5, 6 from the following data frame.

```
exam_data = {'name': ['Anastasia', 'Dima', 'Katherine', 'James', 'Emily', 'Michael', 'Matthew', 'Laura', 'Kevin', 'Jonas'],
'score': [12.5, 9, 16.5, np.nan, 9, 20, 14.5, np.nan, 8, 19],
'attempts': [1, 3, 2, 3, 2, 3, 1, 1, 2, 1],
'qualify': ['yes', 'no', 'yes', 'no', 'no', 'yes', 'yes', 'no', 'no', 'yes']}
labels = ['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j']
```

Expected Output:
Select specific columns and rows:
score qualify
b 9.0 no
d NaN no
f 20.0 yes
g 14.5 yes

2) Write a Pandas program to count the number of rows and columns of a DataFrame. Sample Python dictionary data and list labels:
exam_data = {'name': ['Anastasia', 'Dima', 'Katherine', 'James', 'Emily', 'Michael', 'Matthew', 'Laura', 'Kevin', 'Jonas'],
'score': [12.5, 9, 16.5, np.nan, 9, 20, 14.5, np.nan, 8, 19],
'attempts': [1, 3, 2, 3, 2, 3, 1, 1, 2, 1],
'qualify': ['yes', 'no', 'yes', 'no', 'no', 'yes', 'yes', 'no', 'no', 'yes']}
labels = ['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j']
Expected Output:
Number of Rows: 10
Number of Columns: 4

3) Write a Pandas program to select the rows where the number of attempts in the examination is greater than 2.
Sample Python dictionary data and list labels:
exam_data = {'name': ['Anastasia', 'Dima', 'Katherine', 'James', 'Emily', 'Michael', 'Matthew', 'Laura', 'Kevin', 'Jonas'],
'score': [12.5, 9, 16.5, np.nan, 9, 20, 14.5, np.nan, 8, 19],
'attempts': [1, 3, 2, 3, 2, 3, 1, 1, 2, 1],
'qualify': ['yes', 'no', 'yes', 'no', 'no', 'yes', 'yes', 'no', 'no', 'yes']}
labels = ['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j']
Expected Output:
Number of attempts in the examination is greater than 2:
name score attempts qualify
b Dima 9.0 3 no
d James NaN 3 no
f Michael 20.0 3 yes

4) Write a Pandas program to get the first 3 rows of a given DataFrame.
Sample Python dictionary data and list labels:
exam_data = {'name': ['Anastasia', 'Dima', 'Katherine', 'James', 'Emily', 'Michael', 'Matthew', 'Laura', 'Kevin', 'Jonas'],
'score': [12.5, 9, 16.5, np.nan, 9, 20, 14.5, np.nan, 8, 19],
'attempts': [1, 3, 2, 3, 2, 3, 1, 1, 2, 1],
'qualify': ['yes', 'no', 'yes', 'no', 'no', 'yes', 'yes', 'no', 'no', 'yes']}
labels = ['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j']
Expected Output:
First three rows of the data frame:
attempts name qualify score
a 1 Anastasia yes 12.5
b 3 Dima no 9.0
c 2 Katherine yes 16.5

Result:

Thus the python program is written to show the working of pandas dataframes

Ex.No.4	READING DATA FROM TEXT FILES, EXCEL AND THE WEB AND EXPLORING VARIOUS COMMANDS FOR DOING DESCRIPTIVE ANALYTICS ON THE IRIS DATA SET

Aim:

Reading data from text files, excel and the web and exploring various commands for doing descriptive analytics on the Iris data set.

What is Exploratory Data Analysis?

Exploratory Data Analysis (EDA) is a technique to analyze data using some visual Techniques. With this technique, we can get detailed information about the statistical summary of the data. We will also be able to deal with the duplicates values, outliers, and also see some trends or patterns present in the dataset.
Now let's see a brief about the Iris dataset.

Iris Dataset
If you are from a data science background you all must be familiar with the Iris Dataset. If you are not then don't worry we will discuss this here.

Iris Dataset is considered as the Hello World for data science. It contains five columns namely – Petal Length, Petal Width, Sepal Length, Sepal Width, and Species Type. Iris is a flowering plant, the researchers have measured various features of the different iris flowers and recorded them digitally.

4A). Aim:

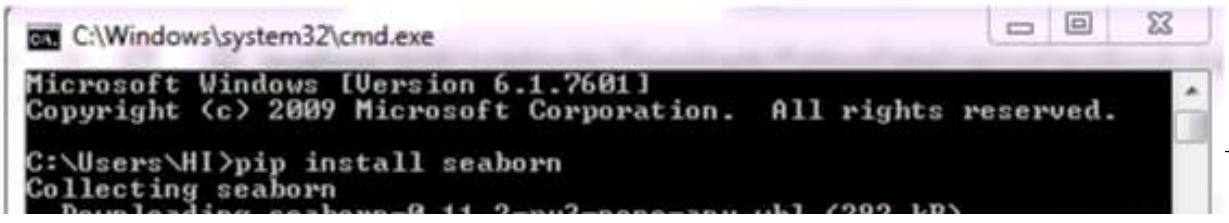
Reading data from Text file and exploring various commands for doing descriptive analytics on the Iris data set.

Seaborn Package:

Seaborn has many of its own high-level plotting routines, but it can also overwrite Matplotlib's default parameters and in turn get even simple Matplotlib scripts to produce vastly superior output. We can set the style by calling Seaborn's set() method. By convention, Seaborn is imported as sns:

Seaborn package is installed by typing the following command in the command prompt

pip install seaborn



Step 2:

After the successful insertion of seaborn package launch into jupyter using jupyter notebook command. Type the following program. Give the correct path name for iris dataset. The Iris dataset, which lists measurements of petals and sepals of three iris species.

Step 1:

Import Packages

In[1]:

```
import numpy as np
import pandas as pd # package for working with data frames in python
import seaborn as sns # package for visualization (more on seaborn later)
import matplotlib.pyplot as plt
%matplotlib inline
```

Step 2:

Import iris dataset

In[2]:

```
iris = sns.load_dataset('iris')
my_data_frame = pd.DataFrame(iris)
my_data_frame.head()
```

OUTPUT

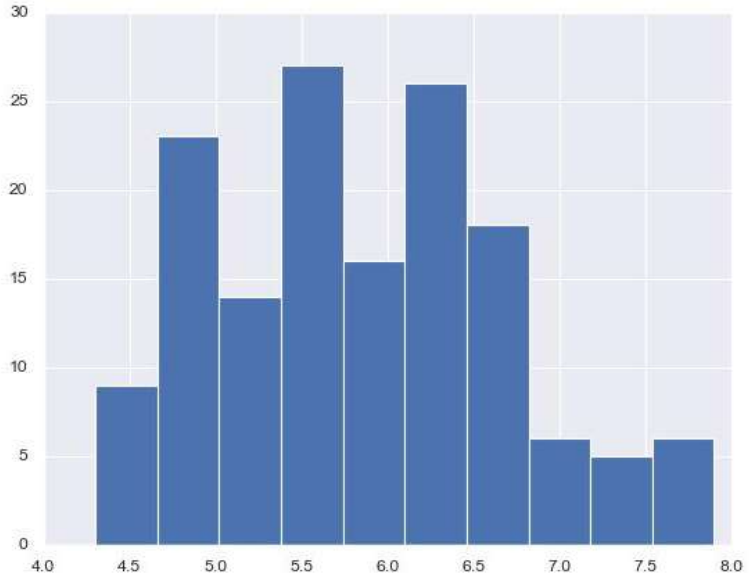
	sepal_length	sepal_width	petal_length	petal_width	species
0	5.1	3.5	1.4	0.2	setosa
1	4.9	3.0	1.4	0.2	setosa
2	4.7	3.2	1.3	0.2	setosa
3	4.6	3.1	1.5	0.2	setosa
4	5.0	3.6	1.4	0.2	setosa

Step 3: Simple plot

In[3]:

```
p=plt.hist(my_data_frame.sepal_length)
```

OUTPUT

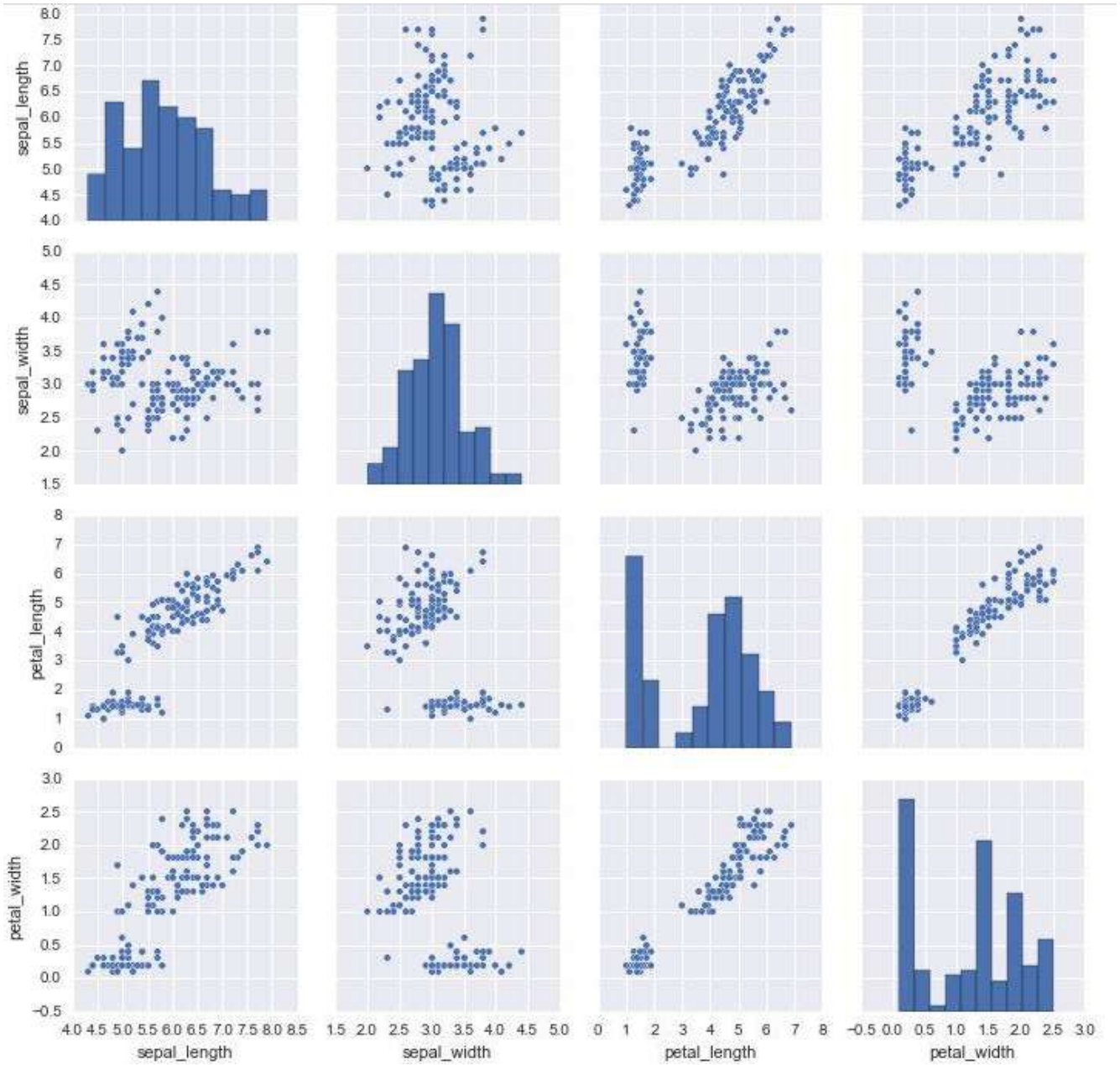


Step: 4 Plot using Seaborn

In[4]:

```
g = sns.pairplot(my_data_frame)
```

OUTPUT

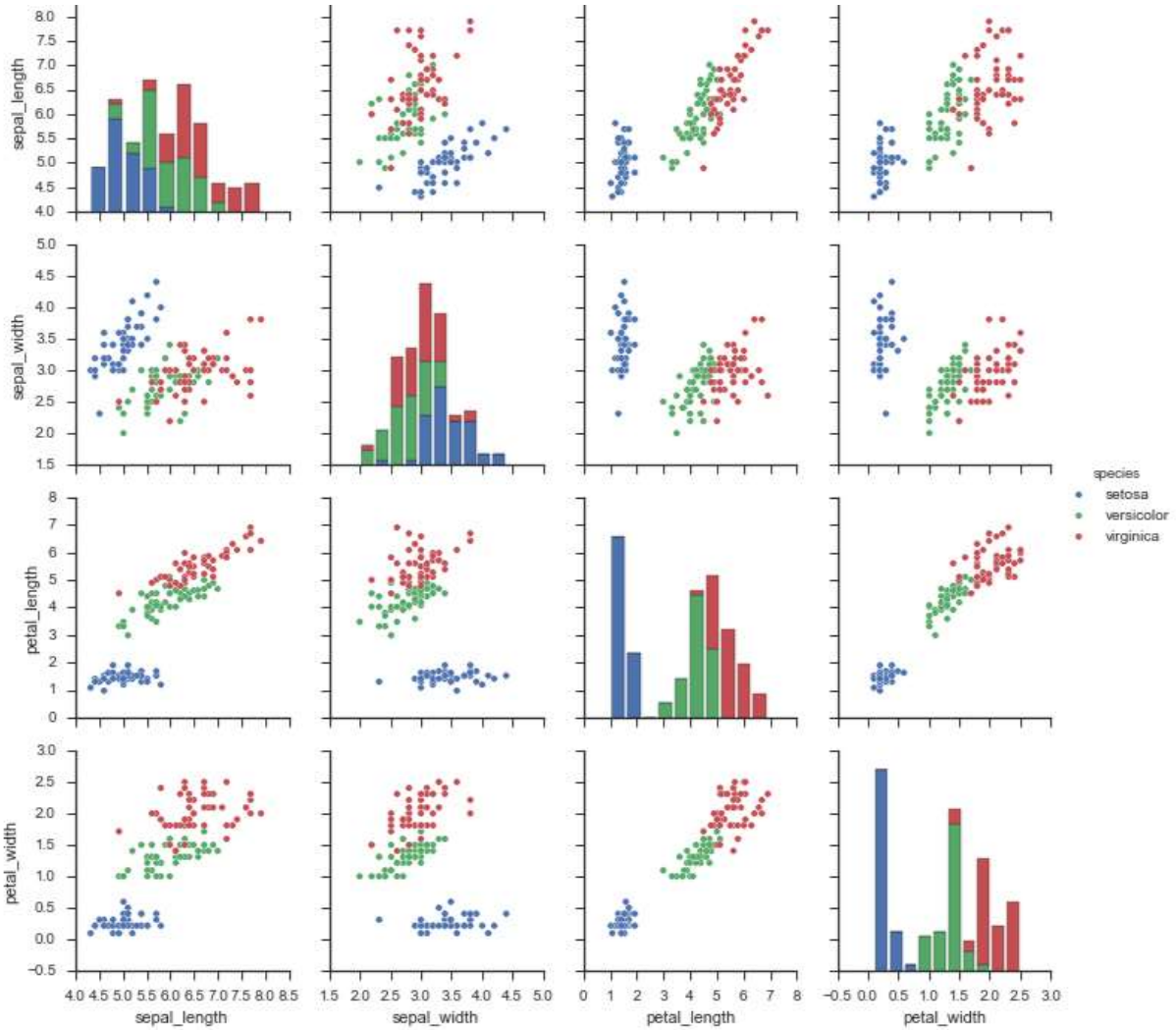


Step 5: Colour plot

In[5]:

```
sns.set(style="ticks", color_codes=True) # change style g
= sns.pairplot(iris, hue="species")
```

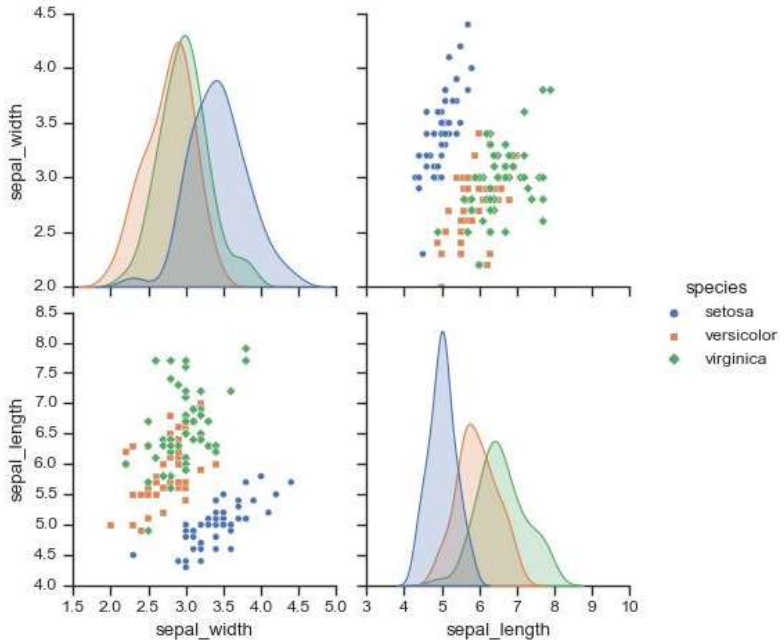
OUTPUT:



In [6]:

```
g = sns.pairplot(iris, height=3, vars=["sepal_width", "sepal_length"], \
                 markers=["o", "s", "D"], hue="species")
```

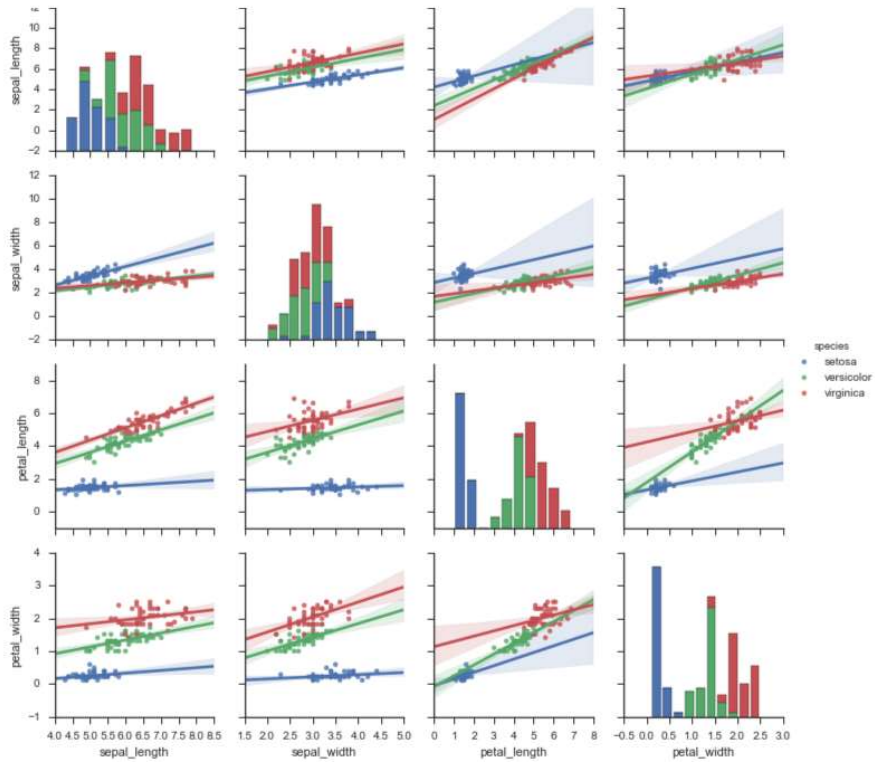
OUTPUT



In [7]:

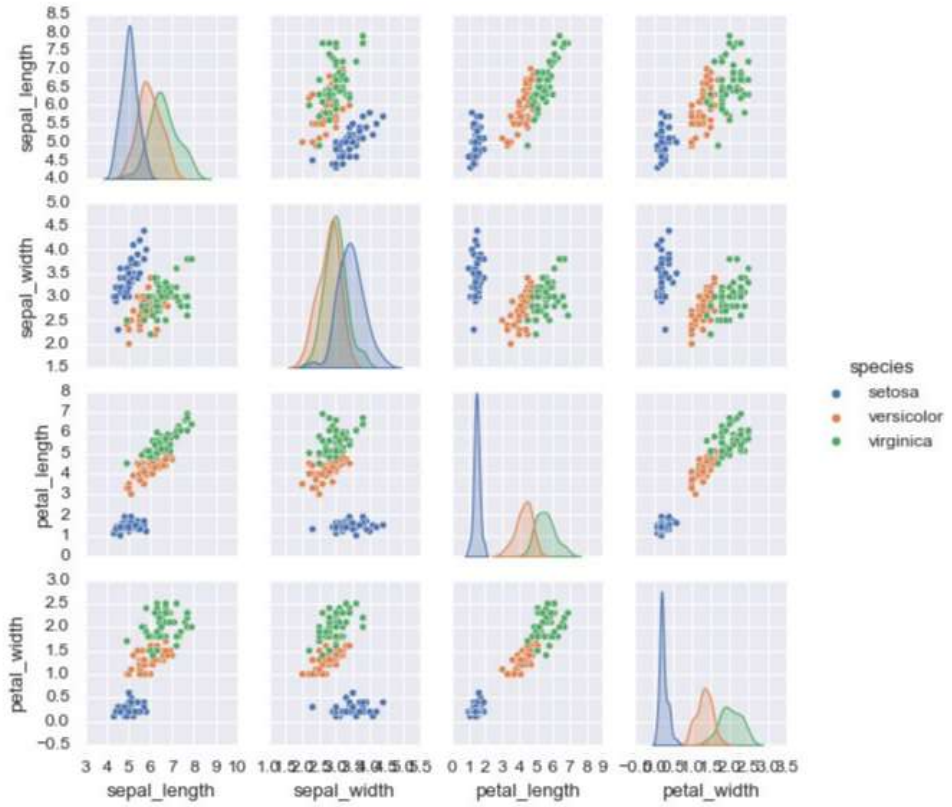
```
g = sns.pairplot(iris, kind="reg", hue="species")
```

OUTPUT



In[8]:

```
sns.set(style="ticks", color_codes=True) # change
style g = sns.pairplot(iris, hue="species")
```



Result:

Thus reading data from Text file and exploring various commands for doing descriptive analytics on the Iris data set is executed.

4 B). Aim:

Reading data from web and exploring various commands for doing descriptive analytics on the iris dataset.

Step 1:

Download the Iris dataset from the UCI machine learning repository.by providind the corresponding URL.

In [1]:

```
import pandas as pd

data = pd.read_csv('http://archive.ics.uci.edu/ml/machine-learning-databases/iris/iris.data',header=None)
data.columns = ['sepal length', 'sepal width', 'petal length', 'petal width', 'class']
data.head()
```

Out[1]:

	sepal length	sepal width	petal length	petal width	class
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

Step 2:

For each quantitative attribute, calculate its average, standard deviation, minimum, and maximum values.

In [2]:

```
from pandas.api.types import is_numeric_dtype

for col in data.columns:
    if is_numeric_dtype(data[col]):
        print('%s:' % (col))
        print('\t Mean = %.2f' % data[col].mean())
        print('\t Standard deviation = %.2f' % data[col].std())
        print('\t Minimum = %.2f' % data[col].min())
```

```
        print('\t Maximum = %.2f' % data[col].max())
```

Out[2]:

```
sepal length:
        Mean = 5.84
        Standard deviation = 0.83
        Minimum = 4.30
        Maximum = 7.90
sepal width:
        Mean = 3.05
        Standard deviation = 0.43
        Minimum = 2.00
        Maximum = 4.40
petal length:
        Mean = 3.76
        Standard deviation = 1.76
        Minimum = 1.00
        Maximum = 6.90
petal width:
        Mean = 1.20
        Standard deviation = 0.76
        Minimum = 0.10
        Maximum = 2.50
```

Step 3:

For the qualitative attribute (class), count the frequency for each of its distinct values.

In [3]:

```
        data['class'].value_counts()
```

Out [3]:

```
Iris-setosa        50
Iris-versicolor    50
Iris-virginica     50
Name: class, dtype: int64
```

Step 4:

It is also possible to display the summary for all the attributes simultaneously in a table using the describe() function. If an attribute is quantitative, it will display its mean, standard deviation and various quantiles (including minimum, median, and maximum) values. If an attribute is qualitative, it will display its number of unique values and the top (most frequent) values.

In [4]:

```
data.describe(include='all')
describe()
```

Out [4]:

	sepal length	sepal width	petal length	petal width	class
count	150.000000	150.000000	150.000000	150.000000	150
unique	NaN	NaN	NaN	NaN	3
top	NaN	NaN	NaN	NaN	Iris-setosa
freq	NaN	NaN	NaN	NaN	50
mean	5.843333	3.054000	3.758667	1.198667	NaN
std	0.828066	0.433594	1.764420	0.763161	NaN
min	4.300000	2.000000	1.000000	0.100000	NaN
25%	5.100000	2.800000	1.600000	0.300000	NaN
50%	5.800000	3.000000	4.350000	1.300000	NaN
75%	6.400000	3.300000	5.100000	1.800000	NaN
max	7.900000	4.400000	6.900000	2.500000	NaN

Step :5

For multivariate statistics, you can compute the covariance and correlation between pairs of attributes.

In [5]:

```
print('Covariance:')
data.cov()
```

Out[5]:

```
Covariance:
```

	sepal length	sepal width	petal length	petal width
sepal length	0.685694	-0.039268	1.273682	0.516904
sepal width	-0.039268	0.188004	-0.321713	-0.117981
petal length	1.273682	-0.321713	3.113179	1.296387
petal width	0.516904	-0.117981	1.296387	0.582414

In [6]:

```
print('Correlation:')
data.corr()
```

Out[6]:

Correlation:

	sepal length	sepal width	petal length	petal width
sepal length	1.000000	-0.109369	0.871754	0.817954
sepal width	-0.109369	1.000000	-0.420516	-0.356544
petal length	0.871754	-0.420516	1.000000	0.962757
petal width	0.817954	-0.356544	0.962757	1.000000

Result:

Thus the reading data from web and exploring various commands for doing descriptive analytics on the iris dataset is executed.

4C). Aim:

Reading data from Excel file and exploring various commands for doing descriptive analytics on the Iris data set.

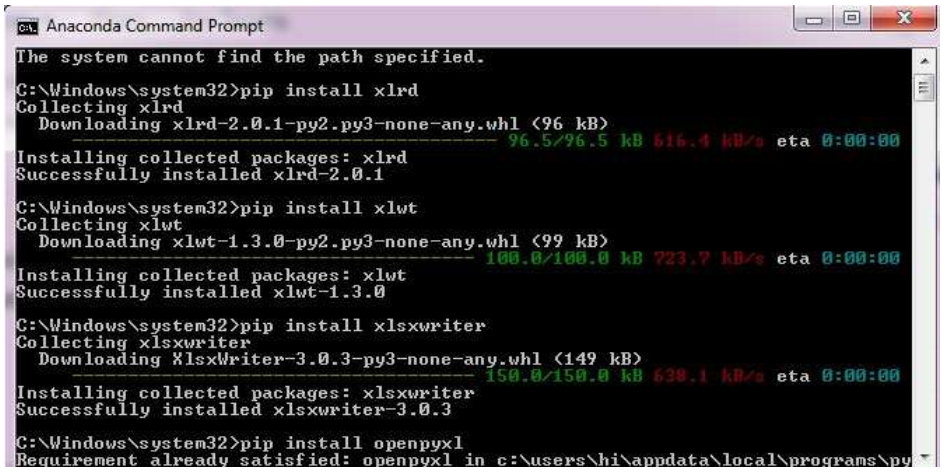
Required python packages

- matplotlib – data visualization
- NumPy – numerical data functionality
- OpenPyXL – read/write Excel 2010 xlsx/xlsm files
- pandas – data import, clean-up, exploration, and analysis
- xlrd – read Excel data
- xlwt – write to Excel
- XlsxWriter – write to Excel (xlsx) files

You can install the required modules using pip. Open your command line program and execute command pip install <module name> to install a module. You should replace <module name> with the actual name of the module you are trying to install.

For example, to install pandas, you would execute command –

eg: pip install pandas.



Step 1:

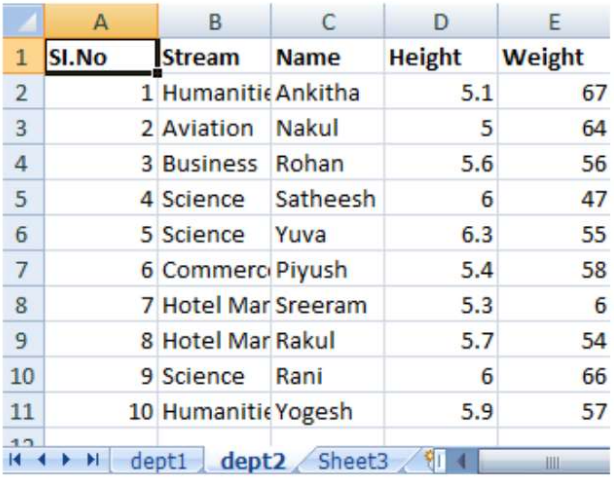
Create a excel file by name dept.. It can be saved as **dept.xlsx**

Sheet 1

	A	B	C	D	E
1	Sl.No	Stream	Name	Height	Weight
2	1	Science	Ayush	5.6	67
3	2	Science	Aniban	6	56
4	3	Commerce	Saurav	5.7	76
5	4	Humanities	laxman	5.8	58
6	5	Commerce	Rahul	6.1	49
7	6	Science	Sneha	6	55
8	7	Hotel Management	Harshit	5.11	68
9	8	Humanitics	Biiju	5.9	63
10	9	Aviation	karan	5.8	59
11	10	Physical Education	Rinu	5.5	65

dept1 dept2 Sheet3

Sheet 2



	A	B	C	D	E
1	Sl.No	Stream	Name	Height	Weight
2	1	Humanitie	Ankitha	5.1	67
3	2	Aviation	Nakul	5	64
4	3	Business	Rohan	5.6	56
5	4	Science	Satheesh	6	47
6	5	Science	Yuva	6.3	55
7	6	Commerc	Piyush	5.4	58
8	7	Hotel Mar	Sreeram	5.3	6
9	8	Hotel Mar	Rakul	5.7	54
10	9	Science	Rani	6	66
11	10	Humanitie	Yogesh	5.9	57

dept1 dept2 Sheet3

Step 2:

Now we can import the excel file using the read_excel function in pandas, as shown below: #place "r" before the path string to address special character, such as '\'. Don't forget to put the file name at the end of the path + '.xlsx'

In [1]:

```
import pandas as pd
df = pd.read_excel (r'C:\Users\HI\Downloads\dept.xlsx')

print (df)
```

Out [1]:

```
   SI.No              Stream     Name  Height  Weight
0      1             Science    Ayush    5.60      67
1      2             Science   Aniban    6.00      56
2      3            Commerce   Saurav    5.70      76
3      4          Humanities   laxman    5.80      58
4      5            Commerce    Rahul    6.10      49
5      6             Science    Sneha    6.00      55
6      7   Hotel Management  Harshit    5.11      68
7      8          Humanitics    Biiju    5.90      63
8      9            Aviation    karan    5.80      59
9     10  Physical Education     Rinu    5.50      65
```

Step 2:

The second statement reads the data from excel and stores it into a pandas Data Frame which is represented by the variable newData. If there are multiple sheets in the excel workbook, the command will import data of the first sheet. To make a data frame with all the sheets in the workbook, the easiest method is to create different data frames separately and then concatenate them.

The read_excel method takes argument sheet_name and index_col where we can specify the sheet of which the data frame should be made of and specifies the title column.

In [2]:

```
sheet1 = pds.read_excel(file, sheet_name = 0, index_col = 0)

sheet2 = pds.read_excel(file, sheet_name = 1, index_col = 0)

newData = pds.concat([sheet1, sheet2])

newData
```

Out [2]:

Sl.No				
1	Science	Ayush	5.60	67
2	Science	Aniban	6.00	56
3	Commerce	Saurav	5.70	76
4	Humanities	laxman	5.80	58
5	Commerce	Rahul	6.10	49
6	Science	Sneha	6.00	55
7	Hotel Management	Harshit	5.11	68
8	Humanitics	Biiju	5.90	63
9	Aviation	karan	5.80	59
10	Physical Education	Rinu	5.50	65
1	Humanities	Ankitha	5.10	67
2	Aviation	Nakul	5.00	64
3	Business	Rohan	5.60	56
4	Science	Satheesh	6.00	47
5	Science	Yuva	6.30	55
6	Commerce	Piyush	5.40	58
7	Hotel Management	Sreeram	5.30	6
8	Hotel Management	Rakul	5.70	54
9	Science	Rani	6.00	66
10	Humanities	Yogesh	5.90	57

Step 3:

To view 5 columns from the top and from the bottom of the data frame, we can run the command

In [3] :

```
newData.tail()
```

Out[3]:

Sl.No	Stream	Height	Weight	Name
6	Commerce	5.4	58	Piyush
7	Hotel Management	5.3	6	Sreeram
8	Hotel Management	5.7	54	Rakul
9	Science	6.0	66	Rani
10	Humanities	5.9	57	Yogesh

In [4] :

```
newData.head()
```

Out[4]:

Sl.No	Stream	Name	Height	Weight
1	Science	Ayush	5.6	67
2	Science	Aniban	6.0	56
3	Commerce	Saurav	5.7	76
4	Humanities	laxman	5.8	58
5	Commerce	Rahul	6.1	49

Step 5:

If any column contains numerical data, we can sort that column using the sort_values() method in pandas as follows:

In[5]:

```
sorted_column = newData.sort_values(['Weight'], ascending = True)

sorted_column.head(5)
```

Out[5]:

Sl.No	Stream	Name	Height	Weight
7	Hotel Management	Sreeram	5.3	6
4	Science	Satheesh	6.0	47
5	Commerce	Rahul	6.1	49
8	Hotel Management	Rakul	5.7	54
6	Science	Sneha	6.0	55

Step 6:

Our data is mostly numerical. We can get the statistical information like mean, max, min, etc. about the data frame using the describe() method as shown below:

In [6]:

```
newData.describe()
```

Out [6]:

	Height	Weight
count	20.000000	20.000000
mean	5.690500	57.300000
std	0.361742	13.955191
min	5.000000	6.000000
25%	5.475000	55.000000
50%	5.750000	58.000000
75%	6.000000	65.250000
max	6.300000	76.000000

Result:

Thus reading data from Excel file and exploring various commands for doing descriptive analytics has been executed.