CSV NOTES

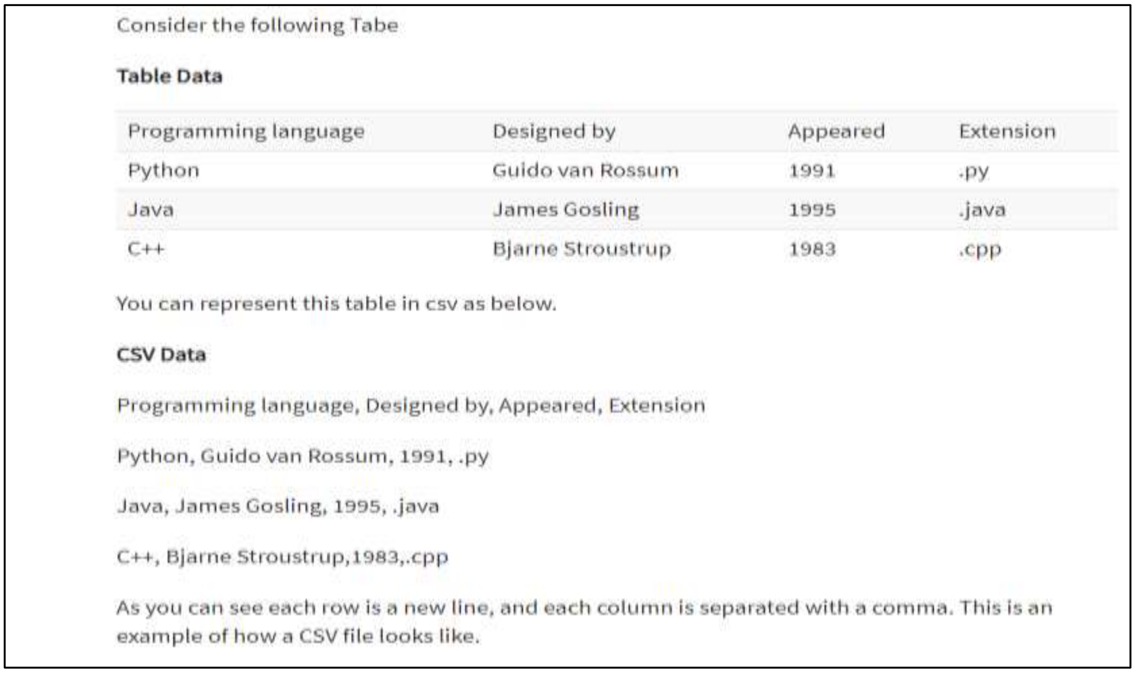
What is a CSV file?

A CSV file is a type of plain text file that uses specific structuring to arrange tabular data. CSV is a common format for data interchange as it's compact, simple and general. Many online services allow its users to export tabular data from the website into a CSV file.

Files of CSV will open into Excel, and nearly all databases have a tool to allow import from CSV file. The standard format is defined by rows and columns data. Moreover, each row is terminated by a newline to begin the next row. Also within the row, each column is separated by a comma.

CSV Sample File.

Data in the form of tables is also called CSV (comma separated values) - literally "comma-separated values." This is a text format intended for the presentation of tabular data. Each line of the file is one line of the table. The values of individual columns are separated by a separator symbol - a comma (,), a semicolon (;) or another symbol. CSV can be easily read and processed by Python.



Consider the following Tabe

Table Data

Programming language	Designed by	Appeared	Extension
Python	Guido van Rossum	1991	.py
Java	James Gosling	1995	.java
C++	Bjarne Stroustrup	1983	.cpp

You can represent this table in csv as below.

CSV Data

```
Programming language, Designed by, Appeared, Extension

Python, Guido van Rossum, 1991, .py

Java, James Gosling, 1995, .java

C++, Bjarne Stroustrup,1983,.cpp
```

As you can see each row is a new line, and each column is separated with a comma. This is an example of how a CSV file looks like.

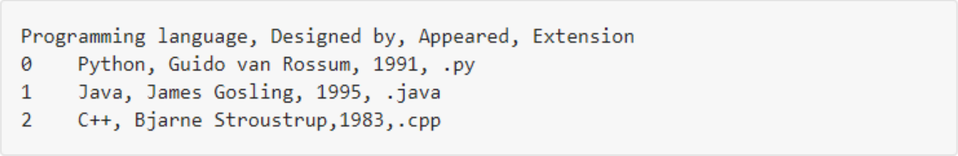
Reading CSV Files with Pandas

Pandas is an opensource library that allows to you perform data manipulation in Python. Pandas provide an easy way to create, manipulate and delete the data.

Reading the CSV into a pandas DataFrame is very quick and easy:

```
#import necessary modules
import pandas
result = pandas.read_csv('X:\data.csv')
print(result)
```

Result:



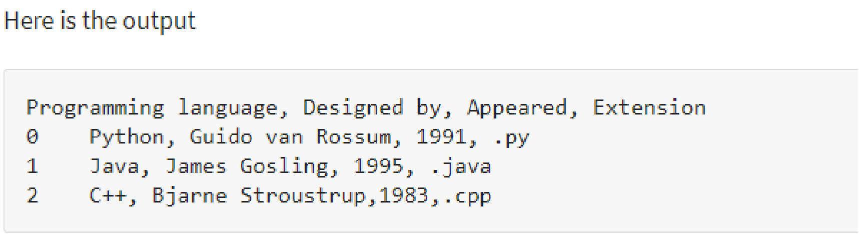
```
Programming language, Designed by, Appeared, Extension
0    Python, Guido van Rossum, 1991, .py
1    Java, James Gosling, 1995, .java
2    C++, Bjarne Stroustrup,1983,.cpp
```

Very useful library. In just three lines of code you the same result as earlier. Pandas know that the first line of the CSV contained column names, and it will use them automatically.

Writing to CSV Files with Pandas

Writing to CSV file with Pandas is as easy as reading. Here you can convince in it. First you must create DataFrame based on the following code.

```
from pandas import DataFrame
C = {'Programming language': ['Python','Java', 'C++'],
        'Designed by': ['Guido van Rossum', 'James Gosling', 'Bjarne Stroustrup'],
        'Appeared': ['1991', '1995', '1985'],
        'Extension': ['.py', '.java', '.cpp'],
    }
df = DataFrame(C, columns= ['Programming language', 'Designed by', 'Appeared', 'Extension'])
export_csv = df.to_csv (r'X:\pandaresult.csv', index = None, header=True) # here you have to write path, where result file will be stored
print (df)
```



Here is the output

```
Programming language, Designed by, Appeared, Extension
0    Python, Guido van Rossum, 1991, .py
1    Java, James Gosling, 1995, .java
2    C++, Bjarne Stroustrup,1983,.cpp
```

And CSV file is created at the specified location.

CSV files are widely used in software applications because they are easy to read and manage, and their small size makes them relatively fast for processing and transmission.

The csv module provides various functions and classes which allow you to read and write easily. CSV is the best way for saving, viewing, and sending data. Actually, it isn't so hard to learn as it seems at the beginning. But with a little practice, you'll master it.

Pandas is a great alternative to read CSV files.

PRACTICAL ASSIGNMENT

#Code to read from CSV file

import pandas as pd

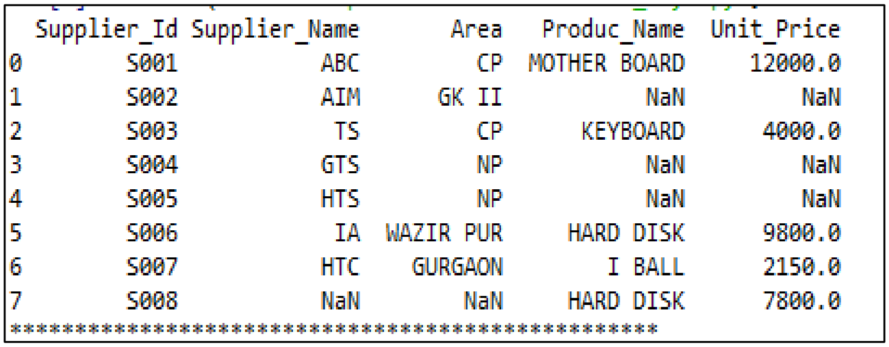
import numpy as np

df1 = pd.read_csv('F:\desktop\GRADE XII PYTHON/SUPPLIER.csv') #File in Excel

print(df1)

print("*"*50)

Supplier.csv

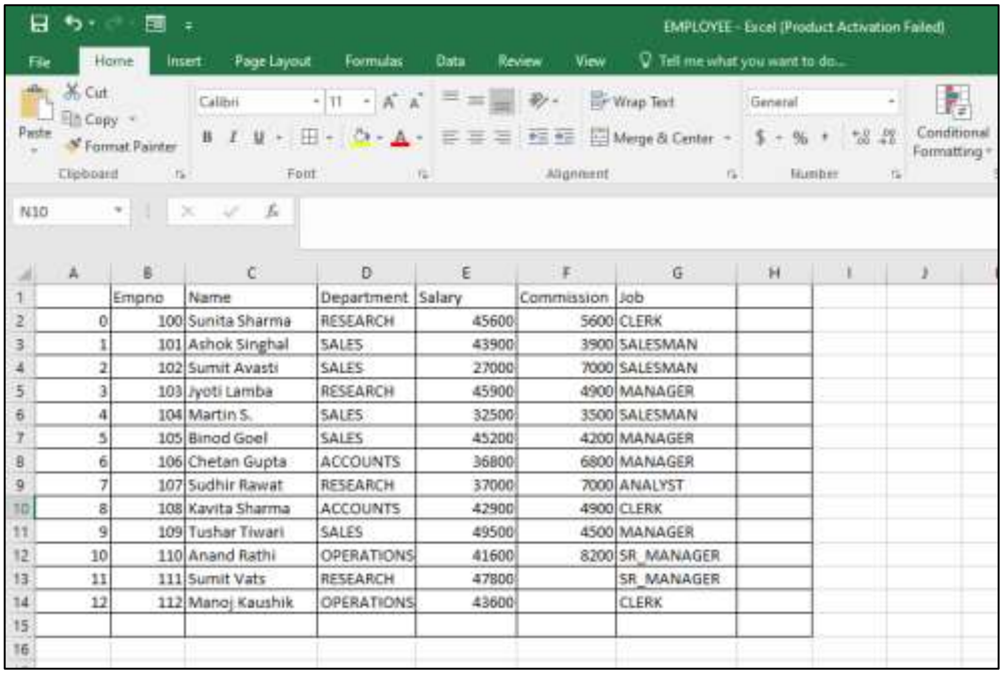


```
  Supplier_Id Supplier_Name       Area   Produc_Name  Unit_Price
0        S001           ABC         CP  MOTHER BOARD     12000.0
1        S002           AIM      GK II           NaN         NaN
2        S003            TS         CP      KEYBOARD      4000.0
3        S004           GTS         NP           NaN         NaN
4        S005           HTS         NP           NaN         NaN
5        S006            IA  WAZIR PUR     HARD DISK      9800.0
6        S007           HTC    GURGAON        I BALL      2150.0
7        S008           NaN        NaN     HARD DISK      7800.0
**************************************************
```

#Reading CSV file with specific Columns

```
df2 = pd.read_csv('F:\desktop\GRADE XII PYTHON/SUPPLIER.csv',
usecols=['Supplier_Name', 'Area', 'Unit_Price'])

print(df2)

print("*"*50)


#Reading CSV file without Header

df3 = pd.read_csv('F:\desktop\GRADE XII PYTHON/SUPPLIER.csv', header=None)

print(df3)

print("*"*50)


#PRINT PRICE IN ASCENDING OREDR

df2=df1.sort_values(by=['Unit_Price'])

print(df2)


#STORING SORTED VALUES TO NEW DATAFRAME NAMED ASCENDINGSALES

unitprice=pd.DataFrame(df2)

unitprice.to_csv('F:\desktop\GRADE XII PYTHON/Ascendingsales.csv')

print(unitprice)


#Reading CSV file without Index

df4 = pd.read_csv('F:\desktop\GRADE XII PYTHON/SUPPLIER.csv', index_col=0)

print(df4)

print("*"*50)


#Reading CSV file with new Column Names

new_names = ['S_IDENTITY', 'S_NAME' , 'PLACE', 'P_NAME', 'COST']

df2 = pd.read_csv('F:\desktop\GRADE XII PYTHON/SUPPLIER.csv', skiprows=1,
names=new_names)

print(df2)
```

```
#*****************Saving DataFrame as CSV File*******************************
dfE=pd.DataFrame({'Empno':[100,101,102,103,104,105,106,107,108,109,110,111,112],

              'Name':['Sunita Sharma','Ashok Singhal',
'Sumit Avasti','Jyoti Lamba','Martin S.','Binod Goel',
'Chetan Gupta','Sudhir Rawat','Kavita Sharma',
'Tushar Tiwari','Anand Rathi','Sumit Vats','Manoj Kaushik'],

'Department':['RESEARCH','SALES','SALES',
'RESEARCH','SALES','SALES','ACCOUNTS','RESEARCH',
'ACCOUNTS','SALES','OPERATIONS','RESEARCH','OPERATIONS'],

'Salary':[45600,43900,27000,45900,32500,45200,36800,
37000,42900,49500,41600,47800,43600],

'Commission':[5600,3900,7000,4900,3500,4200,6800,7000,
4900,4500,8200,np.nan,np.nan],

'Job':['CLERK','SALESMAN','SALESMAN','MANAGER',
'SALESMAN','MANAGER','MANAGER','ANALYST','CLERK',
'MANAGER','SR_MANAGER','SR_MANAGER','CLERK']})
print(dfE)
print("*"*50)
dfE.to_csv('F:\desktop\GRADE XII PYTHON/EMPLOYEE.csv')
#Open EMPLYEE.csv in EXCEL and check..
```



	A	B	C	D	E	F	G	H
1		Empno	Name	Department	Salary	Commission	Job	
2	0	100	Sunita Sharma	RESEARCH	45600	5600	CLERK	
3	1	101	Ashok Singhal	SALES	43900	3900	SALESMAN	
4	2	102	Sumit Avasti	SALES	27000	7000	SALESMAN	
5	3	103	Jyoti Lamba	RESEARCH	45900	4900	MANAGER	
6	4	104	Martin S.	SALES	32500	3500	SALESMAN	
7	5	105	Binod Goel	SALES	45200	4200	MANAGER	
8	6	106	Chetan Gupta	ACCOUNTS	36800	6800	MANAGER	
9	7	107	Sudhir Rawat	RESEARCH	37000	7000	ANALYST	
10	8	108	Kavita Sharma	ACCOUNTS	42900	4900	CLERK	
11	9	109	Tushar Tiwari	SALES	49500	4500	MANAGER	
12	10	110	Anand Rathi	OPERATIONS	41600	8200	SR_MANAGER	
13	11	111	Sumit Vats	RESEARCH	47800		SR_MANAGER	
14	12	112	Manoj Kaushik	OPERATIONS	43600		CLERK	
15								
16								

If you do not need the index column, mention index = False in to_csv() function.

```
#To read the Emp.csv file into a DataFrame dfF
print("*"*50)
df1 = pd.read_csv('F:\desktop\GRADE XII PYTHON/EMPLOYEE.csv',index_col=0)
print(df1)
```

x---x--x