Reading the Csv file

In [1]:
```
import pandas as pd  # Dataframe operations
import numpy as np   # Math operations
import matplotlib.pyplot as plt # Diagrams / plots
import seaborn as sns           # Diagrams / plots
```

In [ ]:
```
# data set name: visadataset
# read csv file : Comma separeated value
# extension :   .csv
# you can read this using pandas package

# read excel file
# extension: .xlsx
```

In [2]:
```
# path
#file location+filename+extension

path=r"C:\Users\omkar\OneDrive\Documents\Data science\Naresh IT\Datafiles\Vi
```

In [3]:
```
pd.read_csv(path)
```

Out[3]:

	case_id	continent	education_of_employee	has_job_experience	requires_job_traini
0	EZYV01	Asia	High School	N	
1	EZYV02	Asia	Master's	Y	
2	EZYV03	Asia	Bachelor's	N	
3	EZYV04	Asia	Bachelor's	N	
4	EZYV05	Africa	Master's	Y	
...	...	...	...	...	
25475	EZYV25476	Asia	Bachelor's	Y	
25476	EZYV25477	Asia	High School	Y	
25477	EZYV25478	Asia	Master's	Y	
25478	EZYV25479	Asia	Master's	Y	
25479	EZYV25480	Asia	Bachelor's	Y	

25480 rows × 12 columns

In [8]:
```
# Can you do bank data
# data set name= bank
path=r"C:\Users\omkar\OneDrive\Documents\Data science\Naresh IT\Datafiles\ba
pd.read_csv(path,
            sep=';')
```

Out[8]:

	age	job	marital	education	default	balance	housing	loan	contact	day	r
0	30	unemployed	married	primary	no	1787	no	no	cellular	19	
1	33	services	married	secondary	no	4789	yes	yes	cellular	11	
2	35	management	single	tertiary	no	1350	yes	no	cellular	16	
3	30	management	married	tertiary	no	1476	yes	yes	unknown	3	
4	59	blue-collar	married	secondary	no	0	yes	no	unknown	5	
...	...	...	...	...	...	...	...	...	...	...	
4516	33	services	married	secondary	no	-333	yes	no	cellular	30	
4517	57	self-employed	married	tertiary	yes	-3313	yes	yes	unknown	9	
4518	57	technician	married	secondary	no	295	no	no	cellular	19	
4519	28	blue-collar	married	secondary	no	1137	no	no	cellular	6	
4520	44	entrepreneur	single	tertiary	no	1136	yes	yes	cellular	3	

4521 rows × 17 columns

Creat dataframes using List

In [10]:
```
name=['Ramesh','Suresh','Sathish']
age=[30,35,40]
name,age
```

Out[10]:
```
(['Ramesh', 'Suresh', 'Sathish'], [30, 35, 40])
```

Step-1

create dataframe

In [11]:
```
pd.DataFrame()  # make the dataframe
```

Out[11]:

—

Step-2

provide data

In [12]:
```
pd.DataFrame(zip(name,age))
```

Out[12]:

	0	1
0	Ramesh	30
1	Suresh	35
2	Sathish	40

Step-3

provide columns

In [15]:
```
#Provide columns
data=zip(name,age)
cols=['Name','Age']
pd.DataFrame(data,columns=cols)
#pd.DataFrame(zip(name,age),columns=['Name','Age'])
```

Out[15]:

	Name	Age
0	Ramesh	30
1	Suresh	35
2	Sathish	40

Step-4

provide index

In [16]:
```
data=zip(name,age)
cols=['Name','Age']
ind=['A','B','C']
pd.DataFrame(data,
             columns=cols,
            index=ind)
```

Out[16]:

	Name	Age
A	Ramesh	30
B	Suresh	35
C	Sathish	40

Step-5

Add new column

In [17]:
```
name=['Ramesh','Suresh','Sathish']
age=[30,35,40]

data=zip(name,age)
cols=['Name','Age']
ind=['A','B','C']
df=pd.DataFrame(data,columns=cols,index=ind)
df
```

Out[17]:

	Name	Age
A	Ramesh	30
B	Suresh	35
C	Sathish	40

- if you want to add a new column
- df['new column']
- you need to have a list which is having some elements
- that elements need to equal to number of rows
- city_names=['Hyd','Blr','Chennai']
- df['city']=city_names

In [19]:
```
city_names=['Hyd','Blr','Chennai']
df['city']=city_names
df
```

Out[19]:

	Name	Age	city
A	Ramesh	30	Hyd
B	Suresh	35	Blr
C	Sathish	40	Chennai

Step-6

update the exsisting column

- if you want to create new column or update the old column
- both are same way

In [22]:
```
df['Name']=['Swamy','Asif','Sathwik']
df
```

Out[22]:

	Name	Age	city
A	Swamy	30	Hyd
B	Asif	35	Blr
C	Sathwik	40	Chennai

Step-7

drop the column

- In order to drop the column
- We need to use drop method
- It takes 3 parameters
  - drop column or row
  - mention the column name
  - axis
    - axis=1 reference as column
    - axis=0 reference as row
  - you want to create a new dataframe or
  - you want overwrite the existing dataframe
    - inplace= True

In [23]:
```
df.drop('city', # column name
        axis=1, # Column
       inplace=True) # overwrite the same
```

In [24]:
```
df
```

Out[24]:

	Name	Age
A	Swamy	30
B	Asif	35
C	Sathwik	40

In [25]:
```
name=['Ramesh','Suresh','Sathish']
age=[30,35,40]

df=pd.DataFrame(zip(name,age),
                columns=['Name','Age'],
                index=['A','B','C'])

city_names=['Hyd','Blr','Chennai']
df['city']=city_names

df.drop('city', # column name
        axis=1, # Column
       inplace=True) # overwrite the same
df
```

Out[25]:

	Name	Age
A	Ramesh	30
B	Suresh	35
C	Sathish	40

Step – 8

Drop rows

In [26]:
```
df.drop('C', # column name
        axis=0, # Column
       inplace=True) # overwrite the same
df
```

Out[26]:

	Name	Age
A	Ramesh	30
B	Suresh	35

Step-9

save the dataframe

In [27]:
```
df.to_csv("output.csv")
# while saving index consider as extra column
df.to_excel("output.xlsx")
```

In [28]:
```
# read output csv
pd.read_csv("output.csv")
```

Out[28]:

	Unnamed: 0	Name	Age
0	A	Ramesh	30
1	B	Suresh	35

Step-10

Remove The Index

In [29]:
```
# To avoid the above problem
# give index=False
df.to_csv("output.csv",index=False)
```

In [30]:
```
pd.read_csv("output.csv")
```

Out[30]:

	Name	Age
0	Ramesh	30
1	Suresh	35

Creat dataframes using dictionary

In [32]:
```
d1={"NAME":['Ramesh','Suresh','Sathish'],
   "AGE":[30,35,40]}

pd.DataFrame(d1)

# No need of zip
# No need of column names
```

Out[32]:

	NAME	AGE
0	Ramesh	30
1	Suresh	35
2	Sathish	40

In [ ]: