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

- Seaborn

In [7]:
```
path =r'C:\Users\aramaiah.ASUAD\Naresh_IT\MyDataScience\Data_Files\Visadataset.csv'
df=pd.read_csv(path)
df
```

Out[7]:

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

25480 rows × 12 columns

In [9]:
```
df.head()
```

Out[9]:

	case_id	continent	education_of_employee	has_job_experience	requires_job_training	no_of_employees
0	EZYV01	Asia	High School	N	N	14513
1	EZYV02	Asia	Master's	Y	N	2412
2	EZYV03	Asia	Bachelor's	N	Y	44444
3	EZYV04	Asia	Bachelor's	N	N	98
4	EZYV05	Africa	Master's	Y	N	1082

In [8]:
```
# data frame name: df
#  by default 5 rows
df.head(2)
```

Out[8]:

	case_id	continent	education_of_employee	has_job_experience	requires_job_training	no_of_employees
0	EZYV01	Asia	High School	N	N	14513
1	EZYV02	Asia	Master's	Y	N	2412

In [10]:
```
# last 5 rows we use tail
df.tail()
```

Out[10]:

	case_id	continent	education_of_employee	has_job_experience	requires_job_training	no_of_emp
25475	EZYV25476	Asia	Bachelor's	Y	Y	
25476	EZYV25477	Asia	High School	Y	N	
25477	EZYV25478	Asia	Master's	Y	N	
25478	EZYV25479	Asia	Master's	Y	Y	
25479	EZYV25480	Asia	Bachelor's	Y	N	

In [11]:
```
df.tail(2)
```

Out[11]:

	case_id	continent	education_of_employee	has_job_experience	requires_job_training	no_of_emp
25478	EZYV25479	Asia	Master's	Y	Y	
25479	EZYV25480	Asia	Bachelor's	Y	N	

Shape

NUmber of rows and number of coulmns

In [12]:
```
df.shape
```

Out[12]:
```
(25480, 12)
```

In [14]:
```
print("The number of rows:", df.shape[0])
print("The number of columns:", df.shape[1])
```

```
The number of rows: 25480
The number of columns: 12
```

- How many indices are provided by size

In [15]:
```
df.size
```

Out[15]:
```
305760
```

In [16]:
```
25480*12 is the size of the
```

Out[16]:
```
305760
```

Columns

In [17]:
```
df.columns
```

Out[17]:
```
Index(['case_id', 'continent', 'education_of_employee', 'has_job_experience',
       'requires_job_training', 'no_of_employees', 'yr_of_estab',
       'region_of_employment', 'prevailing_wage', 'unit_of_wage',
       'full_time_position', 'case_status'],
      dtype='object')
```

In [18]:
```
type(df)
```

Out[18]:
```
pandas.core.frame.DataFrame
```

In [19]:
```
type(df.columns)
```

Out[19]:
```
pandas.core.indexes.base.Index
```

dtypes

In [21]:
```
df.dtypes
# object means categorical
# other than obejct is numerical (int or float)
```

Out[21]:
```
case_id                 object
continent               object
education_of_employee   object
has_job_experience      object
requires_job_training   object
no_of_employees          int64
yr_of_estab              int64
region_of_employment    object
prevailing_wage        float64
unit_of_wage            object
full_time_position      object
case_status             object
dtype: object
```

In [23]:
```
type(df.dtypes)
```

Out[23]:
```
pandas.core.series.Series
```

task1 **Extarct numerical columns and Categorical columns seperately by using dtypes output**

In [24]:
```
# convert above one into dictionary
# key and values
# if for list
dict(df.dtypes)
```

Out[24]:
```
{'case_id': dtype('O'),
 'continent': dtype('O'),
 'education_of_employee': dtype('O'),
 'has_job_experience': dtype('O'),
 'requires_job_training': dtype('O'),
 'no_of_employees': dtype('int64'),
 'yr_of_estab': dtype('int64'),
 'region_of_employment': dtype('O'),
 'prevailing_wage': dtype('float64'),
 'unit_of_wage': dtype('O'),
 'full_time_position': dtype('O'),
 'case_status': dtype('O')}
```

In [28]:
```
d1=dict(df.dtypes)
for i in d1:
#     print(i,d1[i])
    if d1[i]=='object':
        print(i)
```

```
case_id
continent
education_of_employee
has_job_experience
requires_job_training
region_of_employment
unit_of_wage
full_time_position
case_status
```

In [30]:
```
cat=[i for i in d1 if d1[i] == 'object']
num=[i for i in d1 if d1[i] != 'object']
cat
```

Out[30]:
```
['case_id',
 'continent',
 'education_of_employee',
 'has_job_experience',
 'requires_job_training',
 'region_of_employment',
 'unit_of_wage',
 'full_time_position',
 'case_status']
```

In [31]:
```
num
```

Out[31]:
```
['no_of_employees', 'yr_of_estab', 'prevailing_wage']
```

In [34]:
```
df.select_dtypes(include='object').columns
# this displays categorical data available
```

Out[34]:
```
Index(['case_id', 'continent', 'education_of_employee', 'has_job_experience',
       'requires_job_training', 'region_of_employment', 'unit_of_wage',
       'full_time_position', 'case_status'],
      dtype='object')
```

In [35]:
```
df.select_dtypes(exclude='object').columns
```

Out[35]:
```
Index(['no_of_employees', 'yr_of_estab', 'prevailing_wage'], dtype='object')
```

In [ ]:
```
# df has 12 columns
# df.select_dtypes(include='object') has 9 columns
# df.select_dtypes(exclude='object') has 3 columns
```

***isnull* identifies if data has any missing values or null values**

In [36]:
```
df.isnull()
# True means there is a null values
# False means there is no null value
```

Out[36]:

	has_job_experience	requires_job_training	no_of_employees	yr_of_estab	region_of_employment	prevailing
	False	False	False	False	False	
	False	False	False	False	False	
	False	False	False	False	False	
	False	False	False	False	False	
	False	False	False	False	False	
	...	...	...	...	...	
	False	False	False	False	False	
	False	False	False	False	False	
	False	False	False	False	False	
	False	False	False	False	False	
	False	False	False	False	False	

In [ ]:
```
# when you open excel sheet the data is empty which means the data is missed
#  when you read that using pandas at that particular position it will displasy as null
```

In [37]:
```
df.isnull().sum()
```

Out[37]:
```
case_id                 0
continent               0
education_of_employee   0
has_job_experience      0
requires_job_training   0
no_of_employees         0
yr_of_estab             0
region_of_employment    0
prevailing_wage         0
unit_of_wage            0
full_time_position      0
case_status             0
dtype: int64
```

In [39]:
```
df.drop_duplicates()
```

Out[39]:

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

25480 rows × 12 columns

DropDuplicatevalues

In [40]:
```
df.drop_duplicates() #this actually drops duplicate values
```

Out[40]:

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

25480 rows × 12 columns

info

In [42]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 25480 entries, 0 to 25479
Data columns (total 12 columns):
 #   Column                 Non-Null Count  Dtype  
---  ------                 --------------  -----  
 0   case_id                25480 non-null  object 
 1   continent              25480 non-null  object 
 2   education_of_employee  25480 non-null  object 
 3   has_job_experience     25480 non-null  object 
 4   requires_job_training  25480 non-null  object 
 5   no_of_employees        25480 non-null  int64  
 6   yr_of_estab            25480 non-null  int64  
 7   region_of_employment   25480 non-null  object 
 8   prevailing_wage        25480 non-null  float64
 9   unit_of_wage           25480 non-null  object 
 10  full_time_position     25480 non-null  object 
 11  case_status            25480 non-null  object 
dtypes: float64(1), int64(2), object(9)
memory usage: 2.3+ MB
```

Bound method - you need to keep brackets **Not callable -** you neeed to remove the barckets **Attribute error -** the method is not available - check the spell mistake

we need to read sme sample of data

we know head will give top 5

we know tal will give last 5

if you want specific rows or columns

$take - loc - iloc$

In [43]:
```
df.take([2,3,4])
# 2,3,4 are the columns or rows
# axis=1 refernce as columns
#  axis=0 referance as rows
#  by deafsult axis =0,rows
```

Out[43]:

	case_id	continent	education_of_employee	has_job_experience	requires_job_training	no_of_employees
2	EZYV03	Asia	Bachelor's	N	Y	44444
3	EZYV04	Asia	Bachelor's	N	N	98
4	EZYV05	Africa	Master's	Y	N	1082

In [44]:
```
df.take([100,200,300]).take([4,8,11],axis=1)
```

Out[44]:

	requires_job_training	prevailing_wage	case_status
100	N	28243.79	Certified
200	N	74441.11	Certified
300	N	101371.21	Certified

In [ ]:
```
# take doesnt take rows and coulmns at a time hence we can make use of iloc
```

$iloc$

In [ ]:
```
# df.iloc[<rows>,<columns>]
# df.iloc[<start:end>,<start:end>]
# rows=[]
# cols=[]
# df.iloc[rows,cols]
```

In [45]:
```
df.iloc[5:10] #all the columns are displayed
```

Out[45]:

	case_id	continent	education_of_employee	has_job_experience	requires_job_training	no_of_employees
5	EZYV06	Asia	Master's	Y	N	2339
6	EZYV07	Asia	Bachelor's	N	N	4985
7	EZYV08	North America	Bachelor's	Y	N	3035
8	EZYV09	Asia	Bachelor's	N	N	4810
9	EZYV10	Europe	Doctorate	Y	N	2251

In [46]:
```
df.iloc[5:10,2:5]
```

Out[46]:

	education_of_employee	has_job_experience	requires_job_training
5	Master's	Y	N
6	Bachelor's	N	N
7	Bachelor's	Y	N
8	Bachelor's	N	N
9	Doctorate	Y	N

In [47]:
```
df.iloc[:,2:5]
```

Out[47]:

	education_of_employee	has_job_experience	requires_job_training
0	High School	N	N
1	Master's	Y	N
2	Bachelor's	N	Y
3	Bachelor's	N	N
4	Master's	Y	N
...	...	...	...
25475	Bachelor's	Y	Y
25476	High School	Y	N
25477	Master's	Y	N
25478	Master's	Y	Y
25479	Bachelor's	Y	N

25480 rows × 3 columns

In [ ]:
```
df.iloc[5:10] #all the columns
df.iloc[5:10,2:5]#specfic rows and specific columns
df.iloc[:,2:5] #all the rows
```

In [ ]:
```
df.iloc[[100,200,300],[4,8,11]]
```

In [ ]:
```
# only previlage_wage
df.iloc[[100,200,300],[8]]
# no bracekt :Series
# Bracket is there : Data Frame
```

In [ ]:
```
# only  full time
df.iloc[[100]]
# iloc will consider the index while loc will directly consider the loc
```

In [48]:
```
df.loc[[100,200,300],['full_time_position']]
```

Out[48]:

	full_time_position
100	Y
200	Y
300	Y

In [ ]:
```

```