Import the packages

In [1]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

Read the data

In [3]:
```
path=r"C:\Users\omkar\OneDrive\Documents\Data science\Naresh IT\Datafiles\V
df=pd.read_csv(path)
df
```

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

head

Top 5 rows

In [6]:
```
# dataframe name : df
# bydefault 5 rows
df.head(2)
```

Out[6]:

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

Tail

Last 5 rows

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

Out[9]:

	case_id	continent	education_of_employee	has_job_experience	requires_job_traini
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	

shape

Number of rows and number of columns

In [11]:
```
df.shape
```

Out[11]:
```
(25480, 12)
```

In [12]:
```
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
```

size

how many indices are there provided by size

In [13]:
```
df.size
```

Out[13]:
```
305760
```

In [14]:
```
25480*12
```

Out[14]:
```
305760
```

columns

In [15]:
```
df.columns # all the column values
```

Out[15]:
```
Index(['case_id', 'continent', 'education_of_employee', 'has_job_experienc
e',
       '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 [16]:
```
type(df)
```

Out[16]:
```
pandas.core.frame.DataFrame
```

In [17]:
```
type(df.columns)
```

Out[17]:
```
pandas.core.indexes.base.Index
```

$dtypes$

data types

In [18]:
```
df.dtypes

# Object means categorical
# other then object numerical (int or float)
```

Out[18]:
```
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 [19]:
```
type(df.dtypes)
```

Out[19]:
```
pandas.core.series.Series
```

$task-1$

Extract Numerical columns and categorical column sepearetly by using dtypes output

In [25]:
```
# Convert above one into dictionary
# key and values
# if for list
d1=dict(df.dtypes)
# for i in d1:
#     if d1[i]=='object':
#         print(i)

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

In [26]:
```
cat
```

Out[26]:
```
['case_id',
 'continent',
 'education_of_employee',
 'has_job_experience',
 'requires_job_training',
 'region_of_employment',
 'unit_of_wage',
 'full_time_position',
 'case_status']
```

In [28]:
```
# Categorical data avalaibale
df.select_dtypes(include='object').columns
```

Out[28]:
```
Index(['case_id', 'continent', 'education_of_employee', 'has_job_experienc
e',
       'requires_job_training', 'region_of_employment', 'unit_of_wage',
       'full_time_position', 'case_status'],
      dtype='object')
```

In [29]:
```
df.select_dtypes(exclude='object').columns
```

Out[29]:
```
Index(['no_of_employees', 'yr_of_estab', 'prevailing_wage'], dtype='objec
t')
```

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

isnull

identify if data has any missing values or Null values

In [31]:
```
df.isnull()
# True means (yes) there is a null value
# False maens (No) there is no null value
```

Out[31]:

	case_id	continent	education_of_employee	has_job_experience	requires_job_training	no_o
0	False	False	False	False	False	
1	False	False	False	False	False	
2	False	False	False	False	False	
3	False	False	False	False	False	
4	False	False	False	False	False	
...	...	...	...	...	...	
75	False	False	False	False	False	
76	False	False	False	False	False	
77	False	False	False	False	False	
78	False	False	False	False	False	
79	False	False	False	False	False	

80 rows × 12 columns

In [ ]:
```
# when you open excel sheet the data has empty
# which means that data is missed
# when you read that using panads
# at that particular postion it display as Null
```

In [32]:
```
df.isnull().sum()
```

Out[32]:
```
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
```

drop duplicates

Drop duplicate values

In [33]:
```
df.drop_duplicates()
```

Out[33]:

	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

info

In [35]:
```
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
```

In [36]:
```
len(df)
```

Out[36]: 25480

- head
- tail
- shape
- size

- columns
- dtypes
- isnull
- isnull().sum()
- drop duplicates
- info
- len

Bound method

- You need to keep brackets

Not callable

- you need to remove the brackets

Attribute error

- the method is not available
- check the spell mistake

In [ ]:
```
# we want read some sample of data
# we know head will give top5
# we know tail wil give last 5
# if you want specific rows or columns
```

take-loc-iloc

In [43]:
```
df.take((2,5,7))
# 2,3,4  are the columns or rows
# axis=1 reference as columns
# axis=0 reference as rows
# by default axis =0 , rows
```

Out[43]:

case_id	continent	education_of_employee	has_job_experience	requires_job_training	no_of_e
EZYV03	Asia	Bachelor's	N	Y	
EZYV06	Asia	Master's	Y	N	
EZYV08	North America	Bachelor's	Y	N	

In [41]:
```
df.take([2,5,7],axis=1)
# python index start with 0
```

Out[41]:

	education_of_employee	no_of_employees	region_of_employment
0	High School	14513	West
1	Master's	2412	Northeast
2	Bachelor's	44444	West
3	Bachelor's	98	West
4	Master's	1082	South
...	...	...	...
25475	Bachelor's	2601	South
25476	High School	3274	Northeast
25477	Master's	1121	South
25478	Master's	1918	West
25479	Bachelor's	3195	Midwest

25480 rows × 3 columns

In [44]:
```
df.take([100,200,300])
```

Out[44]:

	case_id	continent	education_of_employee	has_job_experience	requires_job_training	r
100	EZYV101	Asia	Master's	Y	N	
200	EZYV201	Asia	Doctorate	Y	N	
300	EZYV301	Asia	Master's	Y	N	

In [ ]:
```
# i want 100,200,300 rows from 4, 8, 11 columns
```

In [45]:
```
df.take([100,200,300]).take([4,8,11],axis=1)
```

Out[45]:

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

take does not take rows and columns at a time

iloc

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

In [46]:
```
df.iloc[5:10] # all the columns
```

Out[46]:

ent	education_of_employee	has_job_experience	requires_job_training	no_of_employees	yr_of_
sia	Master's	Y	N	2339	
sia	Bachelor's	N	N	4985	
rth ica	Bachelor's	Y	N	3035	
sia	Bachelor's	N	N	4810	
pe	Doctorate	Y	N	2251	

In [47]:
```
df.iloc[5:10,2:5]
```

Out[47]:

	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 [48]:
```
df.iloc[:,2:5]
```

Out[48]:

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

In [49]:
```
df.iloc[[100,200,300]]
```

Out[49]:

	case_id	continent	education_of_employee	has_job_experience	requires_job_training	
100	EZYV101	Asia	Master's	Y	N	
200	EZYV201	Asia	Doctorate	Y	N	
300	EZYV301	Asia	Master's	Y	N	

In [50]:
```
df.iloc[[100,200,300],[4,8,11]]
```

Out[50]:

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

In [ ]:
```
df.iloc[5:10] # all the columns
df.iloc[5:10,2:5] # specific rows and specific  columns
df.iloc[:,2:5] # all the rows
df.iloc[[100,200,300]]
df.iloc[[100,200,300],[4,8,11]]
```

In [51]:
```
df.columns
```

Out[51]:
```
Index(['case_id', 'continent', 'education_of_employee', 'has_job_experienc
e',
       '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 [55]:
```
# Only prevailing_Wage
df.iloc[[100,200,300],[8]]

# No bracket: Series
# Barcket is there : Data frame
```

Out[55]:

	prevailing_wage
100	28243.79
200	74441.11
300	101371.21

In [56]:
```
# Only full time
df.iloc[[100,200,300],[10]]

# iloc will consider column index
```

Out[56]:

	full_time_position
100	Y
200	Y
300	Y

loc

In [57]:
```
df.loc[[100,200,300],['full_time_position']]

# loc will consider directly column name
```

Out[57]:

	full_time_position
100	Y
200	Y
300	Y

In [59]:
```
#df.loc[[100,200,300],[10]]
```

In [ ]:
```

```