

Date-08-12-2023*Step – 1***Import Required Packages**

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

*Step – 2***Read the Data**

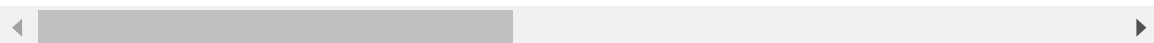
```
In [4]: file_path='C:\\Users\\kurre\\OneDrive\\Documents\\Naresh IT\\datafiles\\Visa_data.csv'
visa_df=pd.read_csv(file_path)
```

```
In [5]: visa_df
```

```
Out[5]:
```

	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

*Shape*

```
In [6]: # dataframe name is visa_df

visa_df.shape
```

```
Out[6]: (25480, 12)
```

```
In [10]: print('the number of rows =',visa_df.shape[0])
         print('the number of rows =',visa_df.shape[1])
```

```
the number of rows = 25480
the number of rows = 12
```

```
In [7]: type(visa_df)
```

```
Out[7]: pandas.core.frame.DataFrame
```

```
In [ ]: # dataframe = table (where data is present)
```

```
In [ ]: # <package_name>.<method_name>()
         # how can we find () is required or not
```

```
In [11]: visa_df.shape()
```

```
-----
-
TypeError                                Traceback (most recent call last)
Cell In[11], line 1
----> 1 visa_df.shape()
```

TypeError: 'tuple' object is not callable

- Function are used to call
- Function means()
- Not callable means,remove the brackets

Size

```
In [ ]: # no. of rows * no. of columns
```

```
In [12]: visa_df.shape[0]*visa_df.shape[1]
         # 25480*12
```

```
Out[12]: 305760
```

Columns

```
In [13]: visa_df.columns
```

```
Out[13]: 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 [14]: type(visa_df.columns)
```

```
Out[14]: pandas.core.indexes.base.Index
```

```
In [15]: # convert tuple to dictionary
```

```
visa_df.columns.to_list
```

```
Out[15]: <bound method IndexOpsMixin.tolist of 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 [16]: # bound method = add brackets  
# bound method means use the brackets
```

```
visa_df.columns.to_list()
```

```
Out[16]: ['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']
```

```
In [ ]: dict()====>to_dict  
list()====>to_list
```

dtypes

```
In [18]: visa_df.dtypes
```

```
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 [25]: type(visa_df.dtypes.values)

# numpy n dimensional array
```

Out[25]: numpy.ndarray

```
In [27]: list1=[1,2,3]
list1

[1,2,3]+[4,5,6]
```

Out[27]: [1, 2, 3, 4, 5, 6]

```
In [28]: np.array([1,2,3])+np.array([4,5,6])
```

Out[28]: array([5, 7, 9])

```
In [ ]: array[]==> numpy
index[]==> pandas index
pandas.dataframe
pandas.series
```

Task – 1

seperate categorical and numerical columns in list

```
In [ ]: # use visa_df.dtypes
# create two list
# cat
# num
```

```
In [29]: dict(visa_df.dtypes)
```

Out[29]: {'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 [30]: dict(visa_df.dtypes)['case_id']
```

Out[30]: dtype('O')

```
In [5]: dict1=dict(visa_df.dtypes)
        for i,j in dict1.items():
            if j=='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 [6]: [i for i,j in dict1.items() if j=='object']
```

```
Out[6]: ['case_id',
        'continent',
        'education_of_employee',
        'has_job_experience',
        'requires_job_training',
        'region_of_employment',
        'unit_of_wage',
        'full_time_position',
        'case_status']
```

```
In [35]: # in single line
```

```
cat=[key for key,value in dict1.items() if value=='object']
cat
```

```
Out[35]: ['case_id',
        'continent',
        'education_of_employee',
        'has_job_experience',
        'requires_job_training',
        'region_of_employment',
        'unit_of_wage',
        'full_time_position',
        'case_status']
```

head

```
In [ ]: # top 5 rows
        # starting index with zero
```

In [37]: `visa_df.head()`

Out[37]:

	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	
2	EZYV03	Asia	Bachelor's	N	Y	
3	EZYV04	Asia	Bachelor's	N	N	
4	EZYV05	Africa	Master's	Y	N	

In [38]: `visa_df.head(2) # if i want only top two`

Out[38]:

	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

In [39]: `visa_df.tail()`

Out[39]:

	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	

In [40]: `visa_df.tail(2) # if i want only last two`

Out[40]:

	case_id	continent	education_of_employee	has_job_experience	requires_job_traini
25478	EZYV25479	Asia	Master's	Y	
25479	EZYV25480	Asia	Bachelor's	Y	

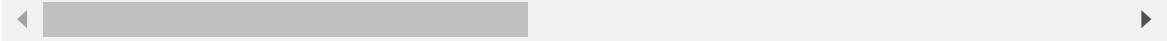
take

In []: `# take random list which you want`

```
In [41]: list1=[100,200,300]
visa_df.take(list1)
```

Out[41]:

	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 [42]: list1=[100,200,300]
visa_df.take(list1,axis=1)

# axis=1 column
# 100,200,300
# it gives error because there is only 12 columns
```

```

-----
-
IndexError                                Traceback (most recent call last)
Cell In[42], line 2
      1 list1=[100,200,300]
----> 2 visa_df.take(list1,axis=1)

File ~\anaconda3\Lib\site-packages\pandas\core\generic.py:3909, in NDFrame
.take(self, indices, axis, **kwargs)
    3833 """
    3834 Return the elements in the given *positional* indices along an axis.
    3835
    (...)
    3904 3    lion    mammal    80.5
    3905 """
    3907 nv.validate_take((), kwargs)
-> 3909 return self._take(indices, axis)

File ~\anaconda3\Lib\site-packages\pandas\core\generic.py:3932, in NDFrame
._take(self, indices, axis, convert_indices)
    3924 if (
    3925     axis == 0
    3926     and indices.ndim == 1
    3927     and using_copy_on_write()
    3928     and is_range_indexer(indices, len(self))
    3929 ):
    3930     return self.copy(deep=None)
-> 3932 new_data = self._mgr.take(
    3933     indices,
    3934     axis=self._get_block_manager_axis(axis),
    3935     verify=True,
    3936     convert_indices=convert_indices,
    3937 )
    3938 return self._constructor(new_data).__finalize__(self, method="take")

File ~\anaconda3\Lib\site-packages\pandas\core\internals\managers.py:960,
in BaseBlockManager.take(self, indexer, axis, verify, convert_indices)
    958 n = self.shape[axis]
    959 if convert_indices:
-> 960     indexer = maybe_convert_indices(indexer, n, verify=verify)
    962 new_labels = self.axes[axis].take(indexer)
    963 return self.reindex_indexer(
    964     new_axis=new_labels,
    965     indexer=indexer,
    (...)
    968     copy=None,
    969 )

File ~\anaconda3\Lib\site-packages\pandas\core\indexers\utils.py:284, in m
aybe_convert_indices(indices, n, verify)
    282 mask = (indices >= n) | (indices < 0)
    283 if mask.any():
-> 284     raise IndexError("indices are out-of-bounds")
    285 return indices

```

IndexError: indices are out-of-bounds

```
In [43]: visa_df.take([2,3,8],axis=1)

# in python index start with zero
# 2 means===== 3rd column
# 3 means===== 4th column
# 8 means===== 9th column
```

```
Out[43]:
```

	education_of_employee	has_job_experience	prevailing_wage
0	High School	N	592.2029
1	Master's	Y	83425.6500
2	Bachelor's	N	122996.8600
3	Bachelor's	N	83434.0300
4	Master's	Y	149907.3900
...
25475	Bachelor's	Y	77092.5700
25476	High School	Y	279174.7900
25477	Master's	Y	146298.8500
25478	Master's	Y	86154.7700
25479	Bachelor's	Y	70876.9100

25480 rows × 3 columns

```
In [44]: visa_df.take([2,3,12],axis=1)
```

it give error because 12 means 13th column, which are not present

```

-----
-
IndexError                                Traceback (most recent call last)
Cell In[44], line 1
----> 1 visa_df.take([2,3,12],axis=1)

File ~\anaconda3\Lib\site-packages\pandas\core\generic.py:3909, in NDFrame
.take(self, indices, axis, **kwargs)
    3833 """
    3834 Return the elements in the given *positional* indices along an axis.
    3835
    3836 (...)
    3904 3      lion  mammal      80.5
    3905 """
    3907 nv.validate_take((), kwargs)
-> 3909 return self._take(indices, axis)

File ~\anaconda3\Lib\site-packages\pandas\core\generic.py:3932, in NDFrame
._take(self, indices, axis, convert_indices)
    3924     if (
    3925         axis == 0
    3926         and indices.ndim == 1
    3927         and using_copy_on_write()
    3928         and is_range_indexer(indices, len(self))
    3929     ):
    3930         return self.copy(deep=None)
-> 3932 new_data = self._mgr.take(
    3933     indices,
    3934     axis=self._get_block_manager_axis(axis),
    3935     verify=True,
    3936     convert_indices=convert_indices,
    3937 )
    3938 return self._constructor(new_data).__finalize__(self, method="take")

File ~\anaconda3\Lib\site-packages\pandas\core\internals\managers.py:960,
in BaseBlockManager.take(self, indexer, axis, verify, convert_indices)
    958 n = self.shape[axis]
    959 if convert_indices:
-> 960     indexer = maybe_convert_indices(indexer, n, verify=verify)
    962 new_labels = self.axes[axis].take(indexer)
    963 return self.reindex_indexer(
    964     new_axis=new_labels,
    965     indexer=indexer,
    966     (...)
    967     copy=None,
    968 )

File ~\anaconda3\Lib\site-packages\pandas\core\indexers\utils.py:284, in m
aybe_convert_indices(indices, n, verify)
    282     mask = (indices >= n) | (indices < 0)
    283     if mask.any():
-> 284         raise IndexError("indices are out-of-bounds")
    285 return indices

```

IndexError: indices are out-of-bounds

```
In [45]: visa_df.take([2,3,8],axis=0)  # now rows will come
```

```
Out[45]:
```

	case_id	continent	education_of_employee	has_job_experience	requires_job_training	no_
2	EZYV03	Asia	Bachelor's	N		Y
3	EZYV04	Asia	Bachelor's	N		N
8	EZYV09	Asia	Bachelor's	N		N

task – 2

```
In [ ]: # i want 150,300,450 rows from 5,8,12
```

```
In [47]: visa_df.take([4,7,11],axis=1).take([150,300,450])
```

```
Out[47]:
```

	requires_job_training	region_of_employment	case_status
150	N	West	Denied
300	N	Midwest	Certified
450	Y	West	Denied

iloc – loc

```
In [ ]: # iloc will take numbers only
# this is patternt to find specific rows and columns
visa_df.iloc[<rows>,<columns>]

visa_df.iloc[start:end,start:end]
```

```
In [49]: visa_df.iloc[100:150] # if you not provide columns value that means all col
```

Out[49]:

	case_id	continent	education_of_employee	has_job_experience	requires_job_training
100	EZYV101	Asia	Master's	Y	N
101	EZYV102	Asia	Master's	Y	N
102	EZYV103	Asia	Bachelor's	Y	N
103	EZYV104	Asia	Doctorate	Y	N
104	EZYV105	Asia	Master's	Y	N
105	EZYV106	North America	Doctorate	Y	N
106	EZYV107	Asia	Bachelor's	N	N
107	EZYV108	Asia	Bachelor's	Y	N
108	EZYV109	Asia	Bachelor's	N	N
109	EZYV110	Asia	Bachelor's	Y	N
110	EZYV111	North America	High School	Y	N
111	EZYV112	Asia	Master's	Y	N
112	EZYV113	Asia	Bachelor's	Y	N
113	EZYV114	North America	High School	Y	N
114	EZYV115	Asia	Doctorate	N	N
115	EZYV116	Asia	Master's	N	N
116	EZYV117	Asia	Bachelor's	Y	N
117	EZYV118	North America	Master's	Y	Y
118	EZYV119	Asia	Bachelor's	Y	N
119	EZYV120	Asia	Master's	N	N
120	EZYV121	North America	Master's	N	N
121	EZYV122	South America	Bachelor's	N	N
122	EZYV123	Asia	Doctorate	Y	Y
123	EZYV124	North America	Master's	N	N
124	EZYV125	Asia	Master's	Y	N
125	EZYV126	North America	Master's	Y	N
126	EZYV127	Asia	High School	Y	N
127	EZYV128	North America	Bachelor's	Y	N
128	EZYV129	Asia	Bachelor's	N	N
129	EZYV130	Asia	Bachelor's	N	N
130	EZYV131	South America	High School	N	N
131	EZYV132	Asia	Bachelor's	Y	N
132	EZYV133	Asia	Doctorate	N	N

	case_id	continent	education_of_employee	has_job_experience	requires_job_training
133	EZYV134	Asia	Doctorate	Y	N
134	EZYV135	Asia	Doctorate	Y	Y
135	EZYV136	Asia	Master's	Y	N
136	EZYV137	Asia	Master's	Y	N
137	EZYV138	Asia	Master's	N	N
138	EZYV139	Asia	Bachelor's	Y	N
139	EZYV140	Asia	Master's	Y	N
140	EZYV141	Asia	Master's	N	N
141	EZYV142	Asia	Master's	Y	N
142	EZYV143	Asia	Master's	N	Y
143	EZYV144	Asia	Doctorate	Y	N
144	EZYV145	Asia	Bachelor's	N	Y
145	EZYV146	Europe	Bachelor's	Y	N
146	EZYV147	Europe	Bachelor's	Y	N
147	EZYV148	Asia	Master's	Y	N
148	EZYV149	Asia	Bachelor's	Y	N
149	EZYV150	Asia	Master's	N	N

In [50]: visa_df.iloc[50:55,2:6]

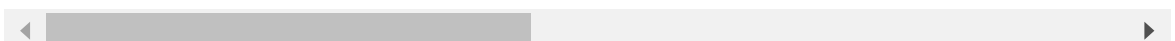
Out[50]:

	education_of_employee	has_job_experience	requires_job_training	no_of_employees
50	Bachelor's	N	N	746
51	Master's	Y	N	3129
52	Bachelor's	N	N	1647
53	High School	Y	N	2438
54	Master's	Y	N	11733

In [52]: visa_df.iloc[[100,200,300]]

Out[52]:

	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 [53]: visa_df.iloc[[100,200,300],[5,8,11]]
```

```
Out[53]:
```

	no_of_employees	prevailing_wage	case_status
100	2227	28243.79	Certified
200	3282	74441.11	Certified
300	3268	101371.21	Certified

```
In [ ]: # iloc take only number this drawback fix with loc keywords
```

loc

```
In [ ]: # list of rows
        # list of columns
```

```
In [55]: visa_df.loc[[100,200,300], 'case_status']
```

```
Out[55]: 100    Certified
         200    Certified
         300    Certified
         Name: case_status, dtype: object
```

```
In [56]: visa_df.loc[[100,200,300], ['case_status']]
```

```
Out[56]:
```

	case_status
100	Certified
200	Certified
300	Certified

```
In [58]: rows=[100,200,300]
         cols=['continent', 'case_status']
         visa_df.loc[rows,cols]
```

```
Out[58]:
```

	continent	case_status
100	Asia	Certified
200	Asia	Certified
300	Asia	Certified

len

```
In [59]: len(visa_df)    # it provide rows
```

```
Out[59]: 25480
```

isnull

- any missing values are there in the data
- is null you are asking question to computer: True or False
- if any missing value is there it is True

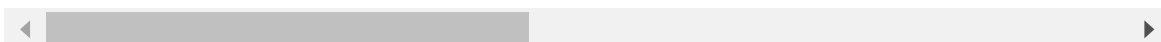
- if data present/no missing value it is False

In [60]: `visa_df.isnull()`

Out[60]:

	case_id	continent	education_of_employee	has_job_experience	requires_job_training
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
...
25475	False	False	False	False	False
25476	False	False	False	False	False
25477	False	False	False	False	False
25478	False	False	False	False	False
25479	False	False	False	False	False

25480 rows × 12 columns



In [61]: `visa_df.isnull().sum()` *# no missing value in this data set because 0*

Out[61]:

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	

info

```
In [63]: visa_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 [ ]:
```