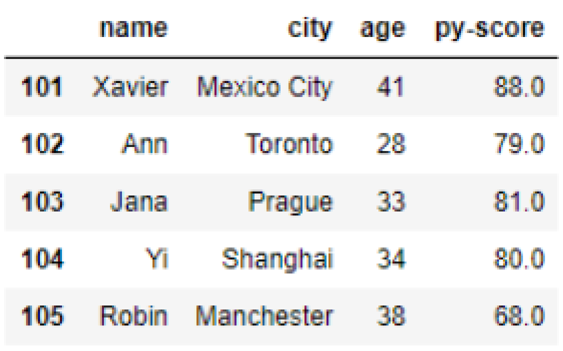
Unit 4 Assignment

I) Implement the program to create the dataframe from the below data



	name	city	age	py-score
101	Xavier	Mexico City	41	88.0
102	Ann	Toronto	28	79.0
103	Jana	Prague	33	81.0
104	Yi	Shanghai	34	80.0
105	Robin	Manchester	38	68.0

In [1]:

```python
import pandas as pd
```

In [2]:

```python
data = {
    'name': ['Xavier', 'Ann', 'Jana', "Yi", "Robin"],
    'city': ['Mexico City', "Toronto", "Prague", "Shanghai", "Manchester"],
    'age': [41, 28, 33, 34, 38],
    'py-score': [88.0, 79.0, 81.0, 80.0, 68.0]
}
df = pd.DataFrame(data, index=[101, 102, 103, 104, 105])
df
```

Out[2]:

```
       name         city  age  py-score
101  Xavier  Mexico City   41      88.0
102     Ann      Toronto   28      79.0
103    Jana       Prague   33      81.0
104      Yi     Shanghai   34      80.0
105   Robin   Manchester   38      68.0
```

1.Display only first two rows

In [3]:

```
df.head(2)
```

Out[3]:

	name	city	age	py-score
101	Xavier	Mexico City	41	88.0
102	Ann	Toronto	28	79.0

2.Display only last two rows

In [4]:

```
df.tail(2)
```

Out[4]:

	name	city	age	py-score
104	Yi	Shanghai	34	80.0
105	Robin	Manchester	38	68.0

3.Extract the py-score of Toronto column.

In [5]:

```
df.loc[102] # df.loc[102] and not df.iloc[102]
df.loc[102]['py-score']

# or

df1 = df.set_index('city')
df1['py-score']['Toronto']
```

Out[5]:

```
79.0
```

4.display of loc of last row

In [6]:

```
df.loc[105]
```

Out[6]:

```
name              Robin
city         Manchester
age                  38
py-score           68.0
Name: 105, dtype: object
```

5.Calculate mean, min, max, standard deviation.

In [7]:

```
df[['age', 'py-score']].mean(), df[['age', 'py-score']].min()
# if we didn't mention proper columns then it'll show future deprication warn
```

Out[7]:

```
(age         34.8
 py-score    79.2
 dtype: float64,
 age         28.0
 py-score    68.0
 dtype: float64)
```

In [8]:

```
df[['age', 'py-score']].max(), df[['age', 'py-score']].std()
```

Out[8]:

```
(age         41.0
 py-score    88.0
 dtype: float64,
 age         4.969909
 py-score    7.190271
 dtype: float64)
```

6.Print the basic stats using describe() method

In [9]:

```
df.describe()
```

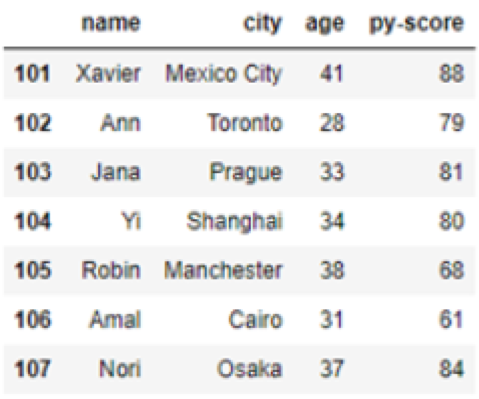
Out[9]:

	age	py-score
count	5.000000	5.000000
mean	34.800000	79.200000
std	4.969909	7.190271
min	28.000000	68.000000
25%	33.000000	79.000000
50%	34.000000	80.000000
75%	38.000000	81.000000
max	41.000000	88.000000

In [10]:

```
# df['py-score'] = df['py-score'].astype(int)
# df
# this can be used to write the data (pyscore col) in int and
# then convert it to float
```

II) Implement a program, to read the CSV file for the given snapshot and perform the given operations:



	name	city	age	py-score
101	Xavier	Mexico City	41	88
102	Ann	Toronto	28	79
103	Jana	Prague	33	81
104	Yi	Shanghai	34	80
105	Robin	Manchester	38	68
106	Amal	Cairo	31	61
107	Nori	Osaka	37	84

In [11]:

```
data = {
    'name': ['Xavier', 'Ann', 'Jana', "Yi", "Robin", "Amal", "Nori"],
    'city': ['Mexico City', "Toronto", "Prague", "Shanghai", "Manchester", "Cairo", "Osaka"
    'age': [41, 28, 33, 34, 38, 31, 37],
    'py-score': [88, 79, 81, 80, 68, 61, 84]
}
df = pd.DataFrame(data, index=list(range(101, 108)))
df
```

Out[11]:

```
       name         city  age  py-score
101  Xavier  Mexico City   41        88
102     Ann      Toronto   28        79
103    Jana       Prague   33        81
104      Yi     Shanghai   34        80
105   Robin   Manchester   38        68
106    Amal        Cairo   31        61
107    Nori        Osaka   37        84
```

a. Display the index and columns

In [12]:

```
df.index, df.columns
```

Out[12]:

```
(Int64Index([101, 102, 103, 104, 105, 106, 107], dtype='int64'),
 Index(['name', 'city', 'age', 'py-score'], dtype='object'))
```

b. modify the labels 101 to 107 to 10 to 17.

In [13]:

```
# wrong as it'll search for [11, 12... 18] all these as columns and replace
# that as the index
# df = df.set_index(list(range(11, 18)))
# df

# correct, it'll modify the old index values with these new ones
df = df.set_index(pd.Index(list(range(11, 18))))
df
```

Out[13]:

	name	city	age	py-score
11	Xavier	Mexico City	41	88
12	Ann	Toronto	28	79
13	Jana	Prague	33	81
14	Yi	Shanghai	34	80
15	Robin	Manchester	38	68
16	Amal	Cairo	31	61
17	Nori	Osaka	37	84

c. Display the shape, size, ndimensional values

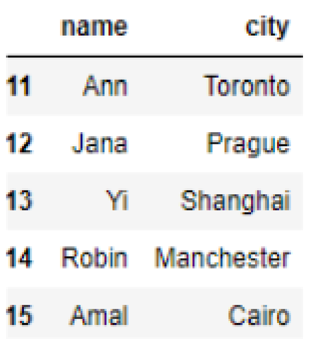
In [14]:

```
df.shape, df.size, df.ndim  # DOUBT
```

Out[14]:

```
((7, 4), 28, 2)
```

d. Implement NumPy slicing of array to get the output displayed below:



	name	city
11	Ann	Toronto
12	Jana	Prague
13	Yi	Shanghai
14	Robin	Manchester
15	Amal	Cairo

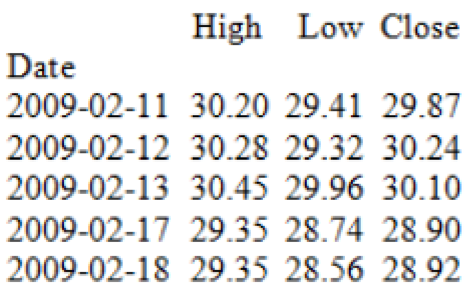
In [16]:

```
df = df[['name', 'city']].loc[11:15] # iloc[0:5]
# in loc 15 is INCLUSIVE
# loc and not iloc here else iloc[0:5]

# DOES NOT WORK IN PANDAS
# df.loc[14], df.loc[15] = df.loc[15], df.loc[14]
# because they are not copy but views, so first df.loc[15] val is give to
# df.loc[14] and then df.loc[14] i.e nothing but val of df.loc[15] itself
# is given to df.loc[15]

df.loc[14], df.loc[15] = df.loc[15].copy(), df.loc[14].copy()
# here on the right side first 2 new values are created i.e their copies
# are created (first the right side is completely executed) and then the
# assignment begins,...

df
```

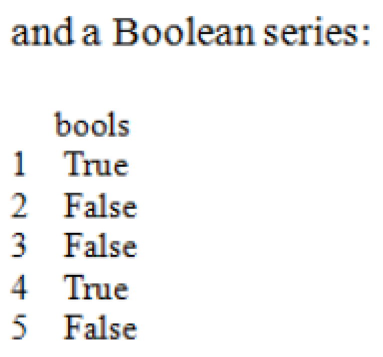
Out[16]:

```
      name        city
11  Xavier  Mexico City
12     Ann      Toronto
13    Jana       Prague
14      Yi     Shanghai
15   Robin   Manchester
```

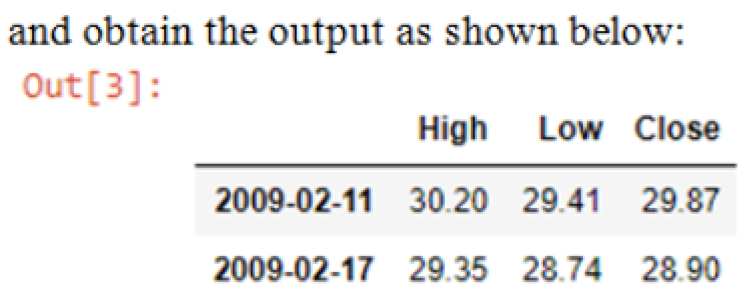
III) Implement a program to create a Data Frame which contains data given



Date	High	Low	Close
2009-02-11	30.20	29.41	29.87
2009-02-12	30.28	29.32	30.24
2009-02-13	30.45	29.96	30.10
2009-02-17	29.35	28.74	28.90
2009-02-18	29.35	28.56	28.92


and a Boolean series:

	bools
1	True
2	False
3	False
4	True
5	False


and obtain the output as shown below:

Out[3]:

	High	Low	Close
2009-02-11	30.20	29.41	29.87
2009-02-17	29.35	28.74	28.90

In [23]:

```
data = {
    "Date": [
        '2009-02-11',
        '2009-02-12',
        '2009-02-13',
        '2009-02-17',
        '2009-02-18'
    ],
    "High": [30.20, 30.38, 30.45, 29.35, 29.35],
    "Low": [29.41, 29.32, 29.96, 28.74, 28.56],
    "Close": [29.87, 30.24, 30.10, 28.90, 28.92]
}

df = pd.DataFrame(data)
df.set_index("Date", inplace=True)
df
```

Out[23]:

	High	Low	Close
Date			
2009-02-11	30.20	29.41	29.87
2009-02-12	30.38	29.32	30.24
2009-02-13	30.45	29.96	30.10
2009-02-17	29.35	28.74	28.90
2009-02-18	29.35	28.56	28.92

In [25]:

```
s = pd.Series([True, False, False, True, False], index=[1, 2, 3, 4, 5])
s
```

Out[25]:

```
1     True
2    False
3    False
4     True
5    False
dtype: bool
```

In [26]:

```
df.iloc[s.values]
```

Out[26]:

	High	Low	Close
Date			
2009-02-11	30.20	29.41	29.87
2009-02-17	29.35	28.74	28.90

In [ ]:

```

```