# Contents

# 6. PANDAS – DATAFREAM – INTRODUCTION

## 1. DataFrame

- ✓ A Data frame is a two-dimensional data structure.
- ✓ Data frame is just like a table.
- ✓ Data frame contains rows and columns.

## 1.1. DataFrame is a pre-defined class

- ✓ **D**ata**F**rame is a pre-defined class in pandas library.

## Example

| Emp_No | Name | Salary |
|---|---|---|
| 101 | Ranjan | 10000 |
| 102 | Akshay | 20000 |
| 103 | Daniel | 30000 |
| 104 | Veeru | 40000 |

**Columns and Rows are**

- ✓ Columns are
  - First column name is : Emp_No
  - Second column name is : Name
  - Third column name is : Salary
- ✓ Rows are
  - First row data is : 101 Abhi 10000
  - Second row data is : 102 Akshay 20000
  - Third row data is : 103 Daniel 10000
  - Forth row data is : 104 Veeru 10000

## 2. Create DataFrame

- ✓ DataFrame is a predefined class in pandas.
- ✓ We can create DataFrame in different ways like below,
    - Empty DataFrame
    - By using single list
    - By using nested list
    - By using dictionary
    - with another DataFrame
    - Loading files(real time approach)

## Generally

- ✓ In real time when we load existing file then it returns DataFrame

## 2.1. Create an Empty DataFrame

✓ We can create an empty DataFrame

Program Name: Creating empty DataFrame
demo1.py

```
import pandas as pd

df = pd.DataFrame()
print(df)
print(type(df))
```

Output

```
Empty DataFrame
Columns: []
Index: []

<class 'pandas.core.frame.DataFrame'>
```

## 2.2. Create a DataFrame by using list

- ✓ We can create DataFrame by using single list.
  - If we are using single list then it's a single column DataFrame
  - If we are using list of lists(nested lists) then it's multiple columns DataFrame

Program Name: Creating DataFrame by using single list
demo2.py

```
import pandas as pd

a = [10, 20, 30, 40, 50, 60, 70, 80, 90]
df = pd.DataFrame(a)

print(df)
print(type(df))
```

Output

```
    0
0  10
1  20
2  30
3  40
4  50
5  60
6  70
7  80
8  90
<class 'pandas.core.frame.DataFrame'>
```

## Note

- ✓ From the output, DataFrame created with single column.
- ✓ Here column name is Zero, we can customise this as well.

## Note on index

- ✓ If no index is passed, then by default, index will be range(n), where n is the array length

Program Name: Creating DataFrame by using single list
demo3.py

```
import pandas as pd

names = ["Ranjan", "Sagar", "Daniel", "Prasad", "Kumari",
"Pravallika", "Arjun", "Akshay"]
df = pd.DataFrame(names)

print(df)
```

Output

```
            0
0      Ranjan
1       Sagar
2      Daniel
3      Prasad
4      Kumari
5  Pravallika
6       Arjun
7      Akshay
```

Program Name: Creating single column DataFrame and checking length
demo4.py

```
import pandas as pd

names = ["Ranjan", "Sagar", "Daniel", "Prasad", "Kumari",
"Pravallika", "Arjun", "Akshay"]
df = pd.DataFrame(names)

print(df)
print()
print("The length is:", len(df))
```

Output

```
           0
0      Ranjan
1       Sagar
2      Daniel
3      Prasad
4      Kumari
5  Pravallika
6       Arjun
7      Akshay

The length is: 8
```

## 2.3. Creating a DataFrame by using list of lists

- ✓ We can create DataFrame with list of lists (nested list).
- ✓ If we are using list of lists then it create a DataFrame with multiple columns.

Program Name: Creating DataFrame by using list of lists
demo5.py

```
import pandas as pd

details = [
        ["Ranjan", 11],
        ["Sagar", 12],
        ["Daniel", 13],
        ["Prasad", 14],
        ["Kumari", 15],
        ["Pravallika", 16],
        ["Arjun", 17],
        ["Akshay", 18]
]

df = pd.DataFrame(details)

print(df)
```

Output

```
            0   1
0      Ranjan  11
1       Sagar  12
2      Daniel  13
3      Prasad  14
4      Kumari  15
5  Pravallika  16
6       Arjun  17
7      Akshay  18
```

**Note**

- ✓ From the output, DataFrame created with two columns.
- ✓ Here column names are 0 and 1 and we can customise this as well.

### 2.3.1. Giving column names to DataFrame

✓ We can give column names to DataFrame.

Program Name: Creating DataFrame and giving names to columns
demo6.py

```
import pandas as pd

details = [
        ["Sagar", 20, 10000],
        ["Daniel", 16, 20000],
        ["Veeru", 24, 30000],
        ["Raju", 25, 40000],
        ["Kiran", 26, 50000],
        ["Kedar", 27, 60000],
        ["Reena", 28, 70000],
        ["Karthik", 29, 80000],
        ["Satish", 30, 90000]
]

cols = ["Name", "Age", "Salary"]

df = pd.DataFrame(details, columns = cols)

print(df)
```

Output

```
      Name  Age  Salary
0    Sagar   20   10000
1   Daniel   16   20000
2    Veeru   24   30000
3     Raju   25   40000
4    Kiran   26   50000
5    Kedar   27   60000
6    Reena   28   70000
7  Karthik   29   80000
8   Satish   30   90000
```

## 2.4. Creating a DataFrame by using dictionary

- ✓ We can create DataFrame by using dictionary
- ✓ If we are using list of lists then it create a DataFrame with multiple columns.

Program Name: Creating DataFrame by using dictionary
demo8.py

```
import pandas as pd

details = {
        "Name": ["Daniel", "Abhi", "Veeru", "Raju", "Kiran",
        "Kedar", "Reena", "Karthik", "Satish"],

        "Age": [20, 21, 23, 24, 25, 26, 27, 28, 29]
}

df = pd.DataFrame(details)

print(df)
```

Output

```
      Name  Age
0   Daniel   20
1     Abhi   21
2    Veeru   23
3     Raju   24
4    Kiran   25
5    Kedar   26
6    Reena   27
7  Karthik   28
8   Satish   29
```

## Note

- ✓ In above example Name and age considered as column names

### 2.4.1. We can customize the index values

- ✓ By default index value start from 0
- ✓ We can customise the index values in DataFrame.
- ✓ If index is passed, then the length of the index should equal to the length of the DataFrame.

Program Name: Creating DataFrame and giving index
demo9.py

```
import pandas as pd

details = [
        ["Sagar", 20, 10000],
        ["Daniel", 16, 20000],
        ["Veeru", 24, 30000],
        ["Raju", 25, 40000],
        ["Kiran", 26, 50000],
        ["Kedar", 27, 60000],
        ["Reena", 28, 70000],
        ["Karthik", 29, 80000],
        ["Satish", 30, 90000]
]

c = ["Name", "Age", "Salary"]
i = [11, 22, 33, 44, 55, 66, 77, 88, 99]

df = pd.DataFrame(details, columns = c, index = i)

print(df)
```

Output

```
        Name  Age  Salary
11     Sagar   20   10000
22    Daniel   16   20000
33     Veeru   24   30000
44      Raju   25   40000
55     Kiran   26   50000
66     Kedar   27   60000
77     Reena   28   70000
88   Karthik   29   80000
99    Satish   30   90000
```

Program Name: Creating DataFrame and giving index
demo10.py

```
import pandas as pd

details = [
        ["Sagar", 20, 10000],
        ["Daniel", 16, 20000],
        ["Veeru", 24, 30000],
        ["Raju", 25, 40000],
        ["Kiran", 26, 50000],
        ["Kedar", 27, 60000],
        ["Reena", 28, 70000],
        ["Karthik", 29, 80000],
        ["Satish", 30, 90000]
]

c = ["Name", "Age", "Salary"]
i = ["Row1", "Row2", "Row4", "Row5", "Row6", "Row7", "Row8",
"Row9", "Row10"]

df = pd.DataFrame(details, columns = c, index = i)

print(df)
```

Output

```
          Name  Age  Salary
Row1     Sagar   20   10000
Row2    Daniel   16   20000
Row4     Veeru   24   30000
Row5      Raju   25   40000
Row6     Kiran   26   50000
Row7     Kedar   27   60000
Row8     Reena   28   70000
Row9   Karthik   29   80000
Row10   Satish   30   90000
```

## 2.5. Creating DataFrame by loading the files.

- ✓ We can create DataFrame by loading files like csv, json etc.
- ✓ This we will learn more on 8th chapter.