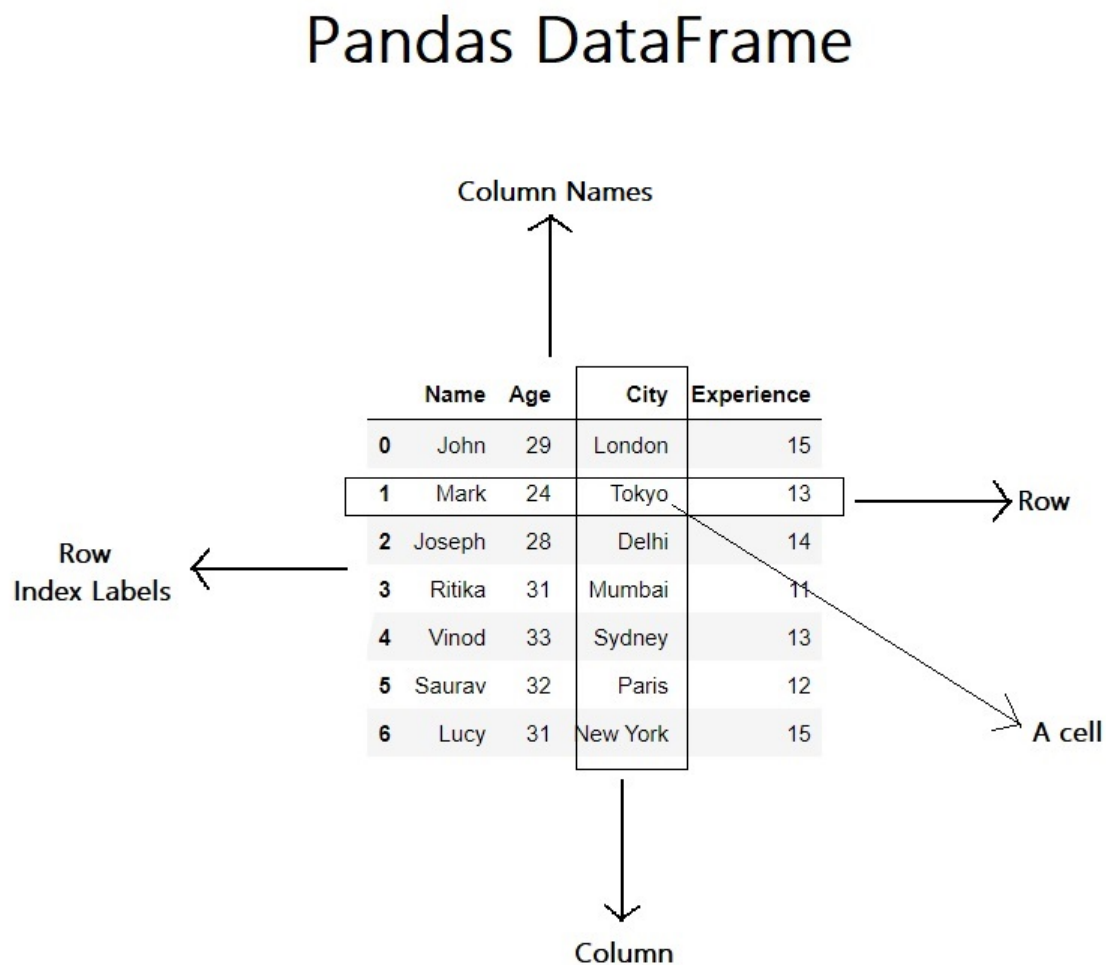


## ▼ Pandas DataFrame

- A DataFrame is a two-dimensional data structure with heterogeneous data.
- A DataFrame is made up of a collection of Series.
- Pandas DataFrame consists of three principal components, the data, rows, and columns.

Figure 1:



Each row as has an index label associated with it and each column has a column name associated with it.

We can select and process individual rows, columns or cells in DataFrame.

Figure 2:

		Name	Score	Attempts	Qualify
0	Anastasia	12.5	1	yes	
1	Dima	9.0	3	no	
2	Katherine	16.5	2	yes	
3	James	NaN	3	no	
4	Emily	9.0	2	no	

## Pandas DataFrame

### How to Create DataFrame?

- The Pandas module provides a function `Dataframe()`, which accepts a data as the argument and returns a `DataFrame` object containing the given elements.

#### Syntax:

```
pandas.DataFrame(data, index, columns, dtype)
```

#### Parameters:

**data:** data takes various forms like ndarray, series, map, lists, dict, constants and also another DataFrame.

**columns:** For column labels

**index:** For the row labels, the Index to be used for frame (optional)

**dtype:** Data type of each column (optional)

**name:** str (optional)

- In DataFrame() the data can be:
  1. A Python list, tuple or dictionary or list of dictionaries
  2. A NumPy ndarray
  3. A Series
  4. Another DataFrame
  5. CSV Files

Note:

- *These are called Methods of creating DataFrames.*

*Note: In the real world, a Pandas DataFrame will be created by loading the datasets from existing storage, storage can be SQL Database, CSV file, and Excel file.*

## ▼ Creating an empty DataFrame

```
#import the pandas library and aliasing as pd
import pandas as pd

d = pd.DataFrame()

print(d)
```

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

## Different Methods of Creating DataFrame:

### ▼ Method 1: Creating DataFrame using a single tuple.

```
# import pandas as pd
import pandas as pd

# tuple
lst = ('Python', 'For', 'Data', 'Science', 'AI', 'ML', 'DL')

# Calling DataFrame constructor on list
df = pd.DataFrame(lst)
print(df)
```

```
      0
0  Python
1    For
2   Data
3 Science
4     AI
5     ML
6     DL
```

```
# import pandas as pd
import pandas as pd

# list
lst = ['Python', 'For', 'Data', 'Science', 'AI', 'ML', 'DL']

# Calling DataFrame constructor on list
df = pd.DataFrame(lst)
print(df)
```

	0
0	Python
1	For
2	Data
3	Science
4	AI
5	ML
6	DL

## ▼ Creating DataFrame using a list of list.

```
import pandas as pd

#list of lists
data = [['Alex',10],['Bob',12],['Clarke',13]]

df = pd.DataFrame(data,columns=['Name','Age'])

print(df)
```

	Name	Age
0	Alex	10
1	Bob	12
2	Clarke	13

## ▼ Creating a Dataframe using dictionary of lists

```
import pandas as pd

# initialise data of lists.
data = {'Name':['Tom', 'nick', 'krish', 'jack'], 'Age':[20, 21, 19, 18]}

# Create DataFrame
df = pd.DataFrame(data)

# Print the output.
print(df)
```

	Name	Age
0	Tom	20
1	nick	21
2	krish	19
3	jack	18

```
import pandas as pd
# Create a dictionary of lists
employees = { 'Name': ['John', 'Mark', 'Joseph', 'Ritika', 'Vinod', 'Saurav', 'Lucy'],
              'Age': [29, 24, 28, 31, 33, 32, 31],
              'City': ['London', 'Tokyo', 'Delhi', 'Mumbai', 'Sydney', 'Paris', 'New York'],
              'Experience': [15, 13, 14, 11, 13, 12, 15]}

# Create a Pandas DataFrame from a Dictionaries
df = pd.DataFrame(employees)

# Display the DataFrame
print(df)
```

	Name	Age	City	Experience
0	John	29	London	15
1	Mark	24	Tokyo	13
2	Joseph	28	Delhi	14
3	Ritika	31	Mumbai	11
4	Vinod	33	Sydney	13
5	Saurav	32	Paris	12
6	Lucy	31	New York	15

## Method 2: Creating DataFrame using a NumPy Array.

Import the Pandas and Numpy modules.

Create a Numpy array.

Create list of index values and column values for the DataFrame.

Create the DataFrame.

Display the DataFrame.

```
# importing the modules
import pandas as pd
import numpy as np

# creating the Numpy array
array = np.array([[ 'Aditya', 20], [ 'Samruddhi', 15],
                  [ 'Rohan', 21], [ 'Anantha', 20],
                  [ 'Abhinandan', 21]])

# creating a list of index names
index_values = [ 'A', 'B', 'C', 'D', 'E' ]

# creating a list of column names
column_values = [ 'Names', 'Age' ]

# creating the dataframe
df = pd.DataFrame(data = array,
                  index = index_values,
                  columns = column_values)

# displaying the dataframe
print(df)
```

	Names	Age
A	Aditya	20
B	Samruddhi	15
C	Rohan	21
D	Anantha	20
E	Abhinandan	21

## Method 3: Creating DataFrame using a Series.

```
#importing pandas library
import pandas as pd

#Creating a list
author = ['Jitender', 'Purnima', 'Arpit', 'Jyoti']

#Creating a Series by passing list variable to Series() function
auth_series = pd.Series(author)

#Printing Series
print(auth_series)
```

```
0    Jitender
1    Purnima
2     Arpit
3     Jyoti
dtype: object
```

### Creating Dataframe from multiple Series

```
#Importing Pandas library
import pandas as pd

#Creating two lists
author = ['Jitender', 'Purnima', 'Arpit', 'Jyoti']
article = [210, 211, 114, 178]

#Creating two Series by passing lists
auth_series = pd.Series(author)
article_series = pd.Series(article)

#Creating a dictionary by passing Series objects as values
frame = { 'Author': auth_series, 'Article': article_series }

#Creating DataFrame by passing Dictionary
result = pd.DataFrame(frame)

#Printing elements of Dataframe
print(result)
```

	Author	Article
0	Jitender	210
1	Purnima	211
2	Arpit	114
3	Jyoti	178

## Method 4: Creating DataFrame using another DataFrame.

Create New DataFrame Using Multiple Columns from Old DataFrame

Syntax:

```
new_df = old_df[['col1','col2']].copy()
```

```
new = result[['Author']].copy()
```

```
print(new)
```

```
      Author
0  Jitender
1  Purnima
2    Arpit
3    Jyoti
```

## Method 5: Creating Dataframe from .csv files

Pandas module provides a function `read_csv()` to create a DataFrame from .csv files.

It takes the csv file path or name as argument and imports the content of a csv file into a Dataframe object.

```
import pandas as pd
```

```
df = pd.read_csv ('/content/drive/MyDrive/Automobile.csv')
```

```
print(df)
```

	company	body-style	length	engine	mileage	sunroof
0	alfa-romero	convertible	168.8	dohc	21	No
1	alfa-romero	hatchback	171.2	dohc	19	No
2	audi	sedan	176.6	ohc	24	Yes
3	audi	sedan	176.6	None	18	Yes
4	audi	sedan	177.3	ohc	19	Yes
5	audi	wagon	192.7	ohc	19	Yes
6	bmw	sedan	176.8	ohc	23	Yes
7	bmw	sedan	176.8	ohc	23	Yes
8	bmw	sedan	176.8	ohc	21	Yes
9	bmw	sedan	189.0	ohc	16	Yes
10	bmw	sedan	193.8	ohc	16	Yes
11	bmw	sedan	197.0	ohc	15	Yes
12	chevrolet	hatchback	141.1	ohc	47	No
13	chevrolet	hatchback	155.9	ohc	38	No
14	chevrolet	sedan	158.8	ohc	38	No
15	dodge	hatchback	157.3	ohc	31	No
16	dodge	hatchback	157.3	ohc	31	No

17	honda	wagon	157.1	ohc	30	No
18	honda	sedan	175.4	ohc	24	No
19	honda	sedan	169.1	ohc	25	No
20	isuzu	sedan	170.7	ohc	24	No
21	isuzu	sedan	155.9	ohc	38	No
22	isuzu	sedan	155.9	ohc	38	No
23	jaguar	sedan	199.6	dohc	15	Yes
24	jaguar	sedan	199.6	dohc	15	Yes
25	jaguar	sedan	191.7	ohcv	13	Yes
26	mazda	hatchback	159.1	ohc	30	Yes
27	mazda	hatchback	159.1	ohc	31	Yes
28	mazda	hatchback	159.1	ohc	31	Yes
29	mazda	hatchback	169.0	rotor	17	Yes
30	mazda	sedan	175.0	ohc	31	Yes
31	mercedes-benz	sedan	190.9	ohc	22	Yes
32	mercedes-benz	wagon	190.9	ohc	22	Yes
33	mercedes-benz	sedan	208.1	ohcv	14	Yes
34	mercedes-benz	hardtop	199.2	ohcv	14	Yes
35	mitsubishi	hatchback	157.3	ohc	37	Yes
36	mitsubishi	hatchback	157.3	ohc	31	Yes
37	mitsubishi	sedan	172.4	ohc	25	Yes
38	mitsubishi	sedan	172.4	ohc	25	Yes
39	nissan	sedan	165.3	ohc	45	No
40	nissan	sedan	165.3	ohc	31	No
41	nissan	sedan	165.3	ohc	31	No
42	nissan	wagon	170.2	ohc	31	No
43	nissan	sedan	184.6	ohcv	19	No
44	porsche	hardtop	168.9	ohcf	17	Yes
45	porsche	convertible	168.9	ohcf	17	Yes
46	porsche	hatchback	175.7	dohcv	17	Yes
47	toyota	hatchback	158.7	ohc	35	No
48	toyota	hatchback	158.7	ohc	31	No
49	toyota	hatchback	158.7	ohc	31	No
50	toyota	wagon	169.7	ohc	31	No
51	toyota	wagon	169.7	ohc	27	No
52	toyota	wagon	169.7	ohc	27	No
53	toyota	wagon	187.8	dohc	19	No
54	volkswagen	sedan	171.7	ohc	37	Yes
55	volkswagen	sedan	171.7	ohc	27	Yes
56	volkswagen	sedan	171.7	ohc	37	Yes

```
import pandas as pd
df = pd.read_csv ('/content/drive/MyDrive/nba.csv')
print(df)
```

	Name	Team	Number	Position	Age	Height	Weight	\
0	Avery Bradley	Boston Celtics	0.0	PG	25.0	6-2	180.0	
1	Jae Crowder	Boston Celtics	99.0	SF	25.0	6-6	235.0	
2	John Holland	Boston Celtics	30.0	SG	27.0	6-5	205.0	
3	R.J. Hunter	Boston Celtics	28.0	SG	22.0	6-5	185.0	
4	Jonas Jerebko	Boston Celtics	8.0	PF	29.0	6-10	231.0	
..	...	...	...	...	...	...	...	
453	Shelvin Mack	Utah Jazz	8.0	PG	26.0	6-3	203.0	
454	Raul Neto	Utah Jazz	25.0	PG	24.0	6-1	179.0	
455	Tibor Pleiss	Utah Jazz	21.0	C	26.0	7-3	256.0	
456	Jeff Withey	Utah Jazz	24.0	C	26.0	7-0	231.0	
457	NaN	NaN	NaN	NaN	NaN	NaN	NaN	

	College	Salary
0	Texas	7730337.0
1	Marquette	6796117.0
2	Boston University	NaN
3	Georgia State	1148640.0
4	NaN	5000000.0



```
..      ...      ...
453      Butler  2433333.0
454      NaN    900000.0
455      NaN    2900000.0
456      Kansas  947276.0
457      NaN      NaN
```

```
[458 rows x 9 columns]
```

✓ 0s completed at 11:56

