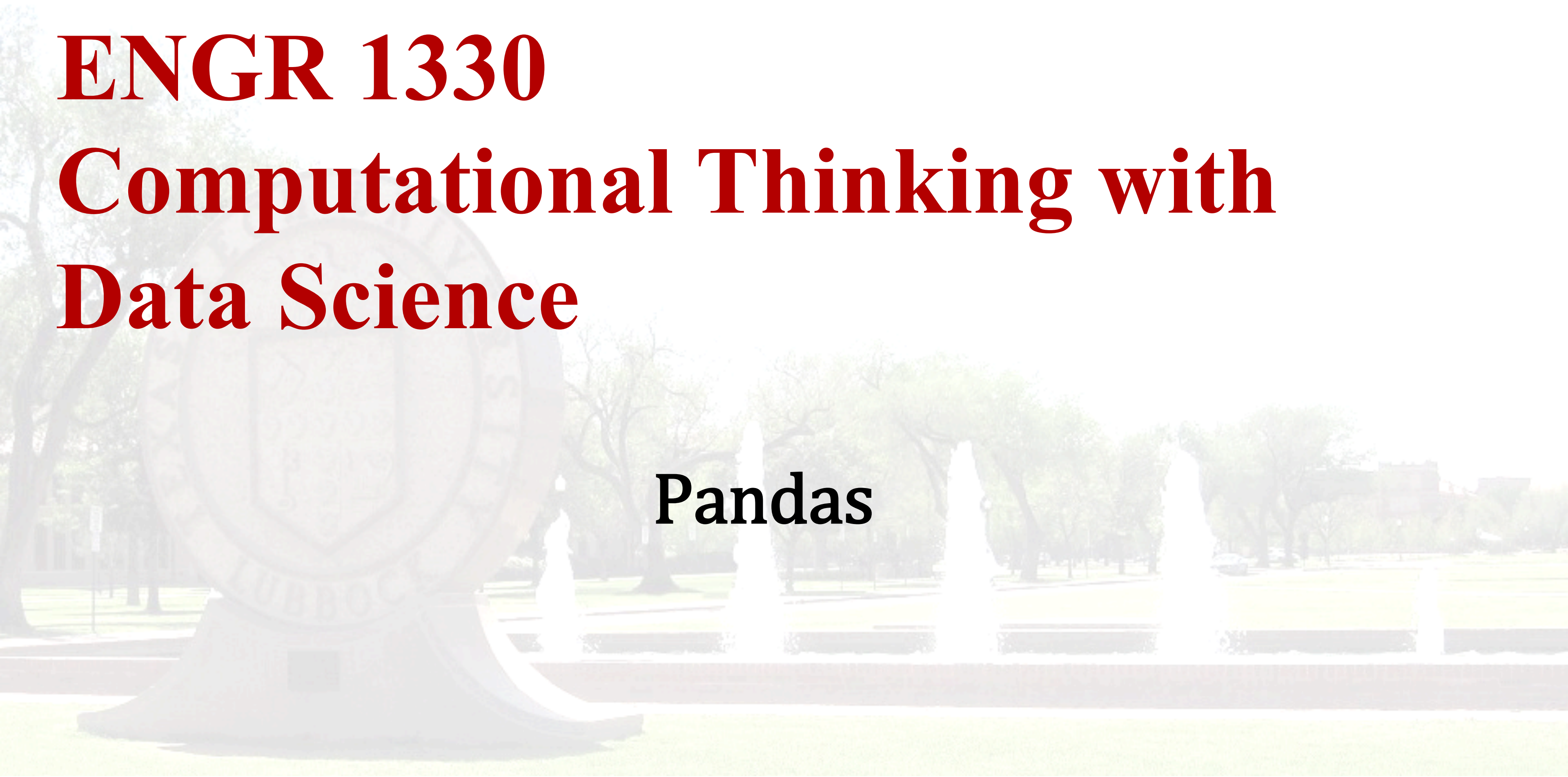


ENGR 1330 Computational Thinking with Data Science

Pandas

Topic Outline

- Pandas library

  ✓ Data representation: Dataframes

  ✓ Data operations: Indexing, summarizing statistics, aggregation, grouping, filling and dropping values, and read/write files

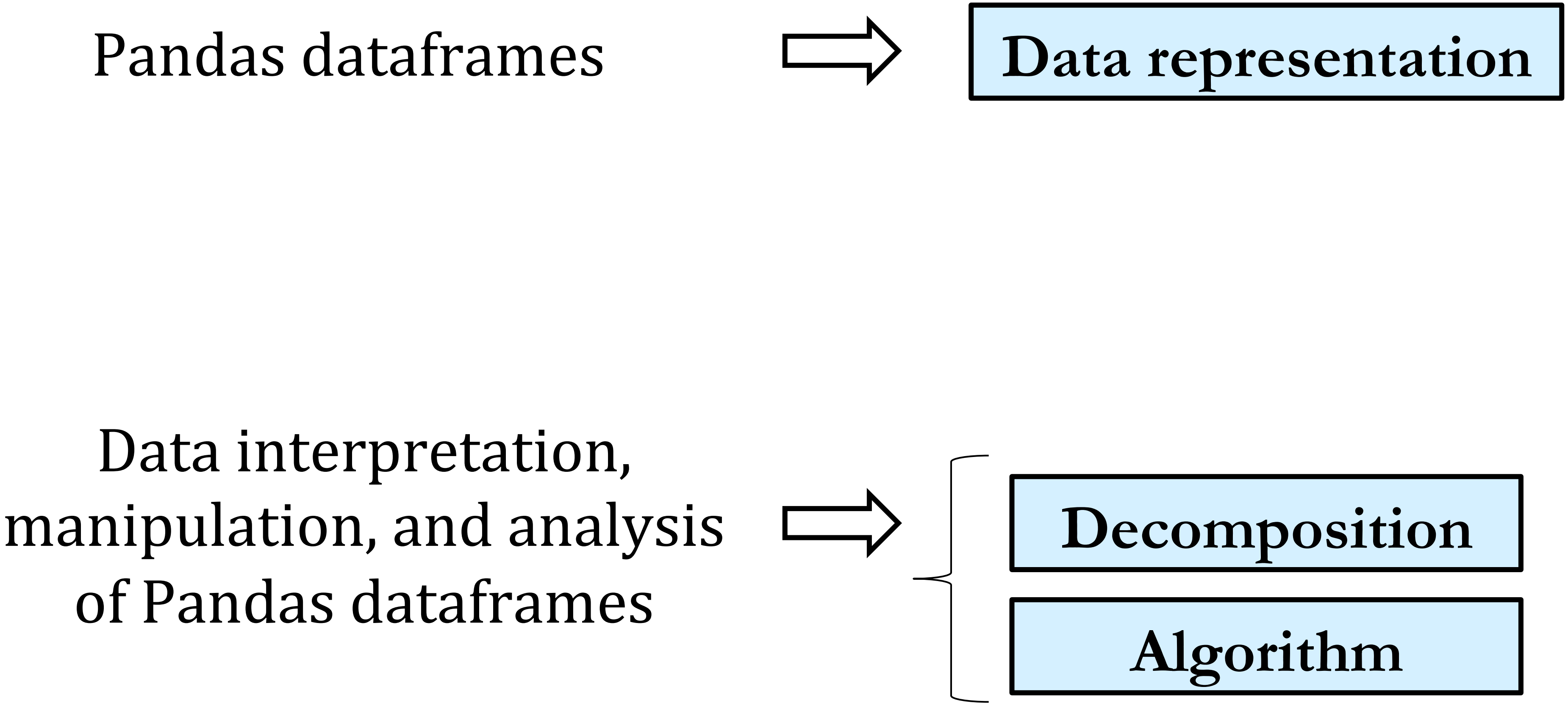
Objectives

- To be able to represent data in the form of dataframes via the Pandas library

- To be able to access and manipulate data within a dataframe

- To be able to obtain basic statistical measures of data within a dataframe

Computational Thinking Concepts



Pandas dataframes → **Data representation**

Data interpretation, manipulation, and analysis of Pandas dataframes → **Decomposition**, **Algorithm**

Pandas

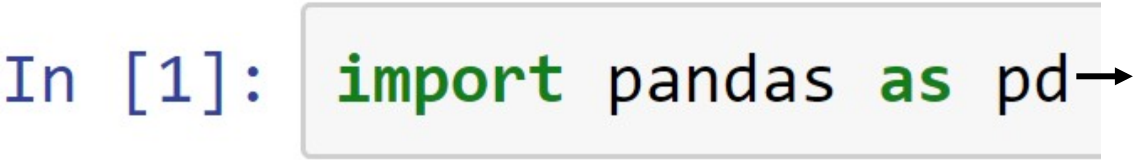
- Pandas: Derived from the term ‘Panel Data’

- Dataframe: 2-dimensional mutable and heterogenous tabular data structure

  ✓ Provides rich data structures and functions designed to make working with data fast, easy, and expressive

Pandas Dataframes

- Creating a dataframe:



```
In [1]: import pandas as pd
```

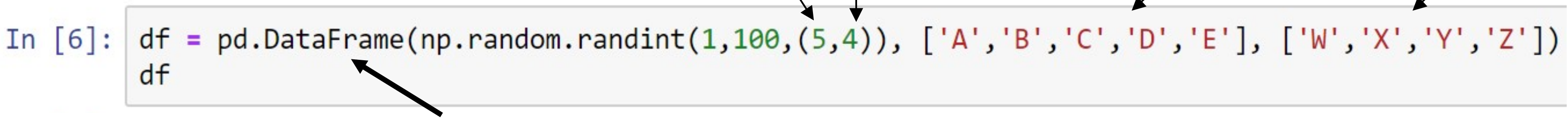
Importing Pandas library

columns

rows

index

columns



```
In [6]: df = pd.DataFrame(np.random.randint(1,100,(5,4)), ['A','B','C','D','E'], ['W','X','Y','Z'])
        df
```

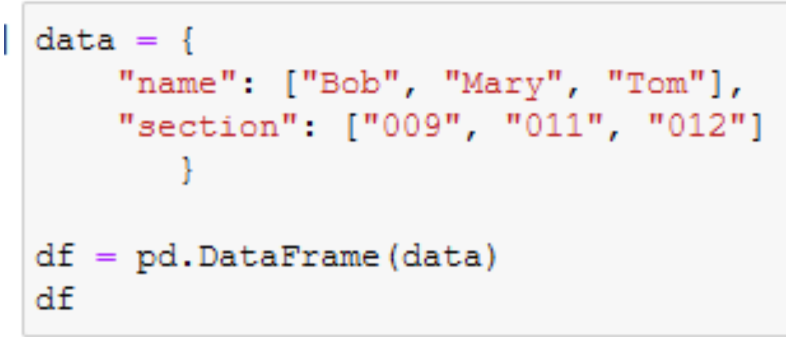
Function to create a Pandas dataframe

- What will be the shape of the above 2D Pandas dataframe?

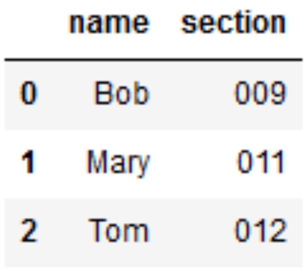
(Demo)

Pandas Dataframes

- Creating a dataframe: from dictionary



```
data = {
    "name": ["Bob", "Mary", "Tom"],
    "section": ["009", "011", "012"]
       }

df = pd.DataFrame(data)
df
```



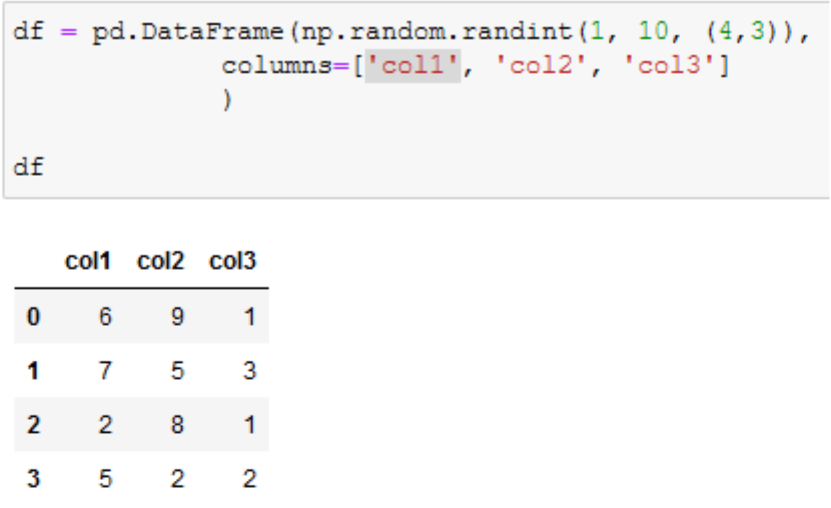
	name	section
0	Bob	009
1	Mary	011
2	Tom	012

Dictionary’s keys

Auto indexing

Dataframes: Indexing, slicing

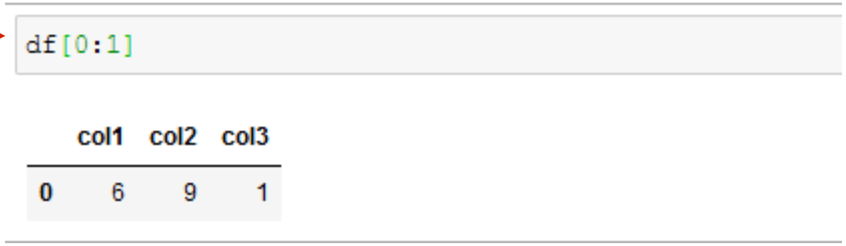
Selecting rows and all columns: same as indexing in list.



```
df = pd.DataFrame(np.random.randint(1, 10, (4,3)),
                  columns=['col1', 'col2', 'col3']
                  )

df
```

	col1	col2	col3
0	6	9	1
1	7	5	3
2	2	8	1
3	5	2	2

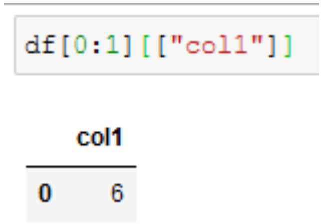
df[start:end:step]



```
df[0:1]
```

	col1	col2	col3
0	6	9	1

Selecting rows and some columns: include in a list of names of selecting columns.

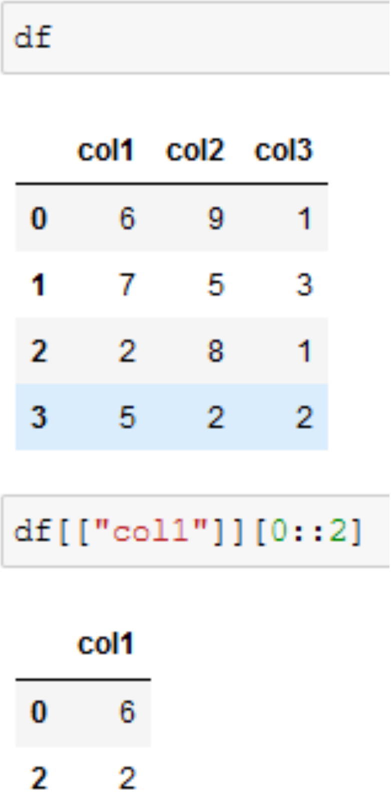


```
df[0:1][["col1"]]
```

	col1
0	6

Dataframes: Indexing, slicing

Select columns first then rows later.



```
df
```

	col1	col2	col3
0	6	9	1
1	7	5	3
2	2	8	1
3	5	2	2

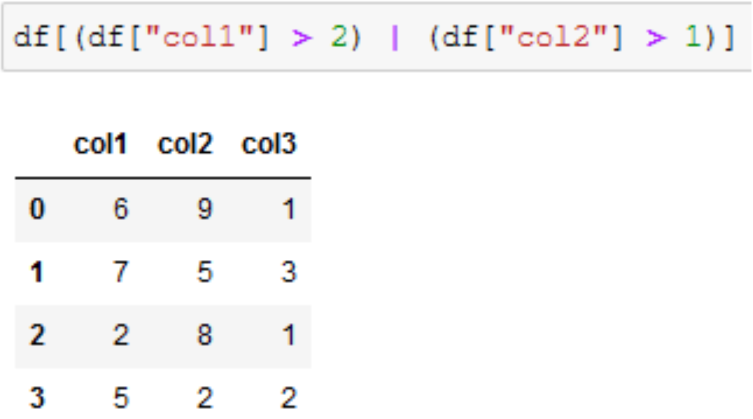
```
df[["col1"]][0::2]
```

	col1
0	6
2	2

Indexing with conditions

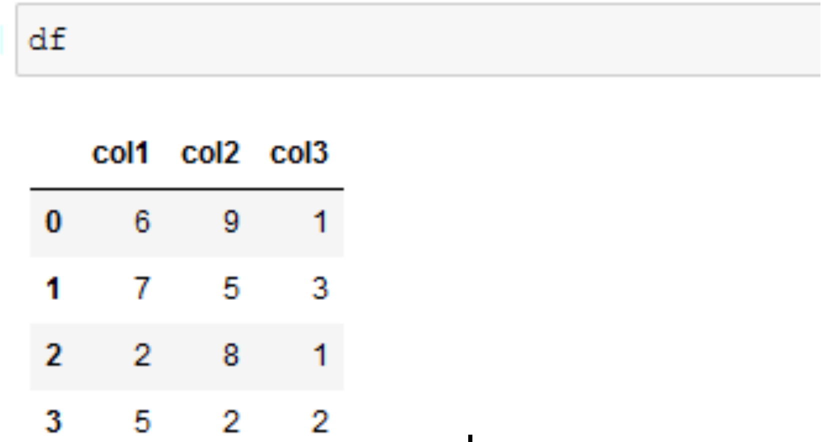
Operator	Meaning
&	Both must true
|	Either condition true

or



```
df[(df["col1"] > 2) | (df["col2"] > 1)]
```

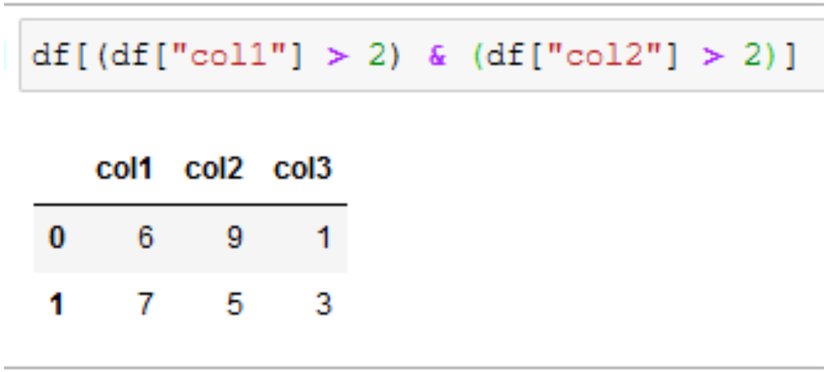
```
   col1  col2  col3
0     6     9     1
1     7     5     3
2     2     8     1
3     5     2     2
```



```
df
```

```
   col1  col2  col3
0     6     9     1
1     7     5     3
2     2     8     1
3     5     2     2
```

and



```
df[(df["col1"] > 2) & (df["col2"] > 2)]
```

```
   col1  col2  col3
0     6     9     1
1     7     5     3
```

Dataframes: Basic Operations

- Functions to do basic operations on Pandas dataframes

✓ df.head( ): Returns first 5 rows of a dataframe

✓ df.info( ): Returns information such as number of rows and columns about a dataframe

✓ df.describe( ): Returns basic statistical measures of a dataframe

(Demo)

Dataframes: Basic Aggregation

- Functions to do basic operations on Pandas dataframes

✓ df.mean(.): mean of rows or columns

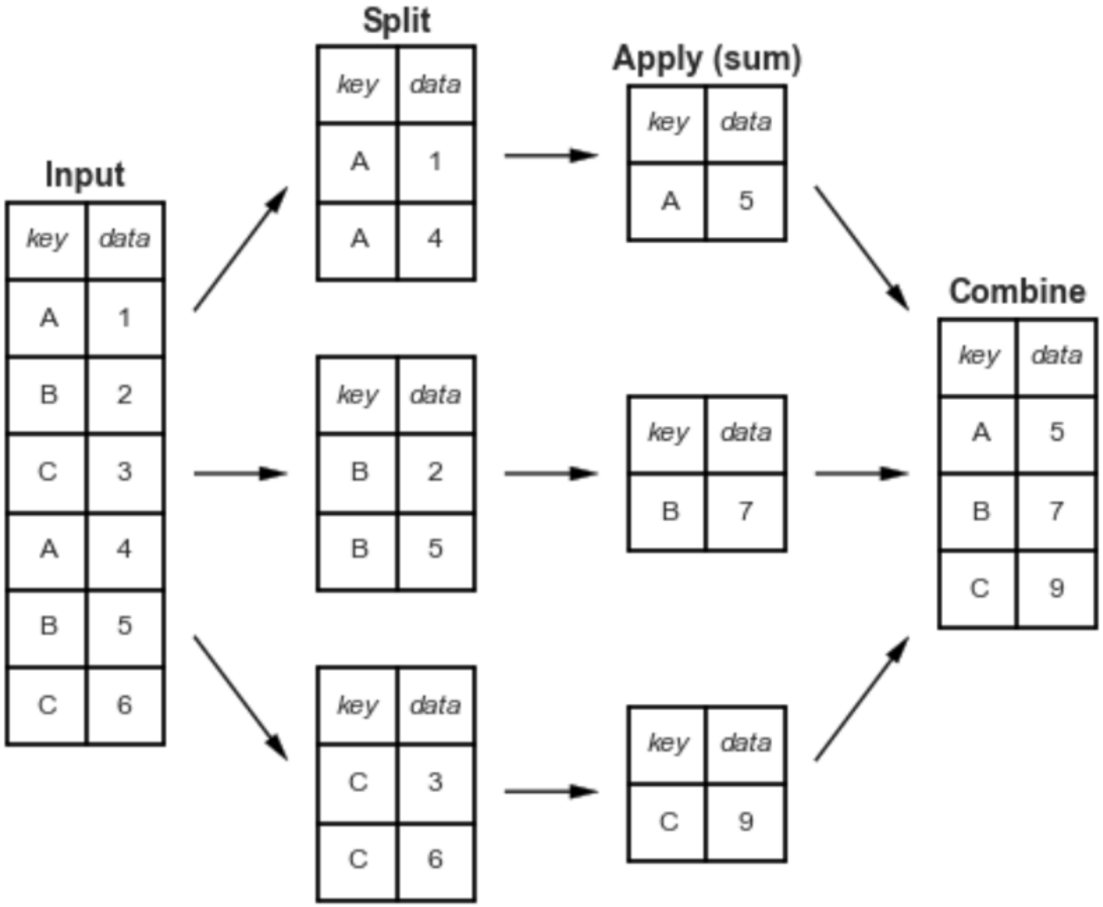
✓ df.min(.): min element of rows or columns

✓ df.max(.): max element of rows or columns

✓ df.sum(.): Returns the sum of rows or columns

(Demo)

Aggregation with Grouping



Input

key	data
A	1
B	2
C	3
A	4
B	5
C	6

Split

key	data
A	1
A	4

key	data
B	2
B	5

key	data
C	3
C	6

Apply (sum)

key	data
A	5

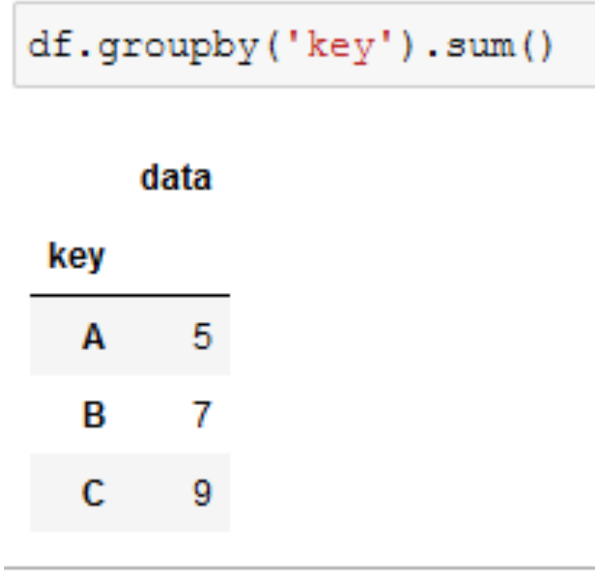
key	data
B	7

key	data
C	9

Combine

key	data
A	5
B	7
C	9

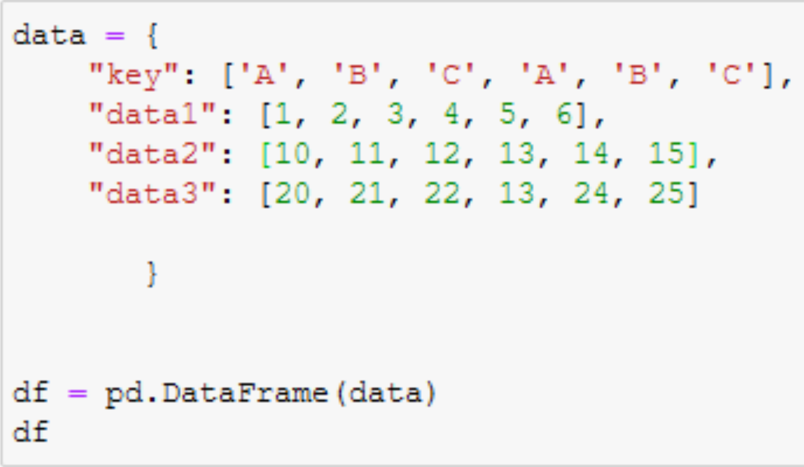
Grouping all columns



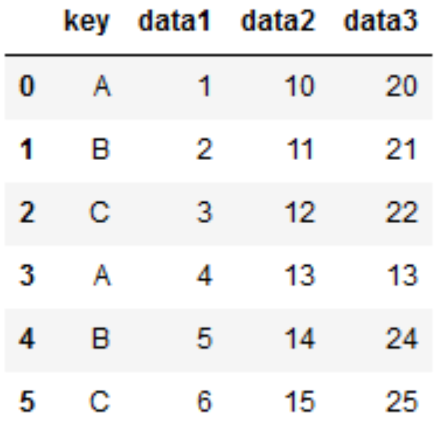
```
df.groupby('key').sum()
```

key	data
A	5
B	7
C	9

Aggregation with Grouping

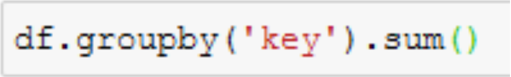


```
data = {
    "key": ['A', 'B', 'C', 'A', 'B', 'C'],
    "data1": [1, 2, 3, 4, 5, 6],
    "data2": [10, 11, 12, 13, 14, 15],
    "data3": [20, 21, 22, 13, 24, 25]

      }


df = pd.DataFrame(data)
df
```

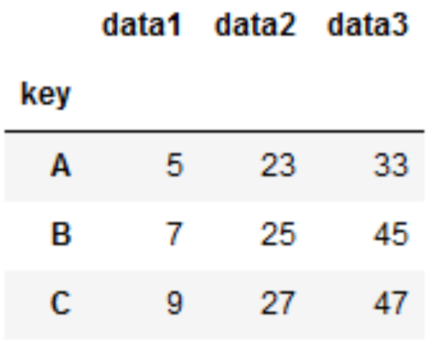


	key	data1	data2	data3
0	A	1	10	20
1	B	2	11	21
2	C	3	12	22
3	A	4	13	13
4	B	5	14	24
5	C	6	15	25

Grouping all columns

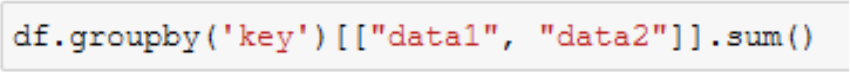


```
df.groupby('key').sum()
```

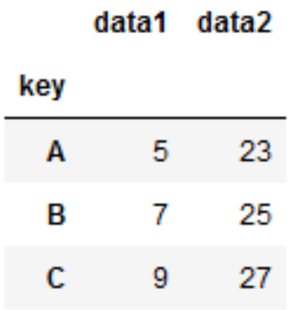


	data1	data2	data3
key			
A	5	23	33
B	7	25	45
C	9	27	47

Grouping for selected columns



```
df.groupby('key')[["data1", "data2"]].sum()
```

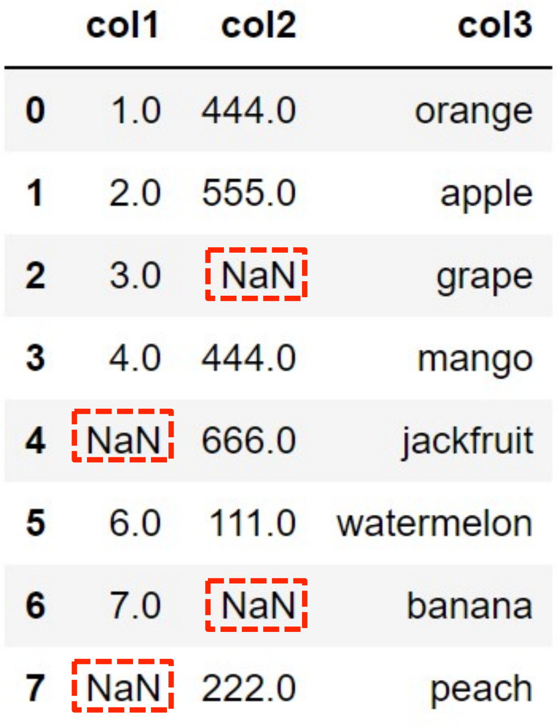


	data1	data2
key		
A	5	23
B	7	25
C	9	27

Dataframes: Dropping values

- Often, the data will consist of missing values 'NaN'

df =

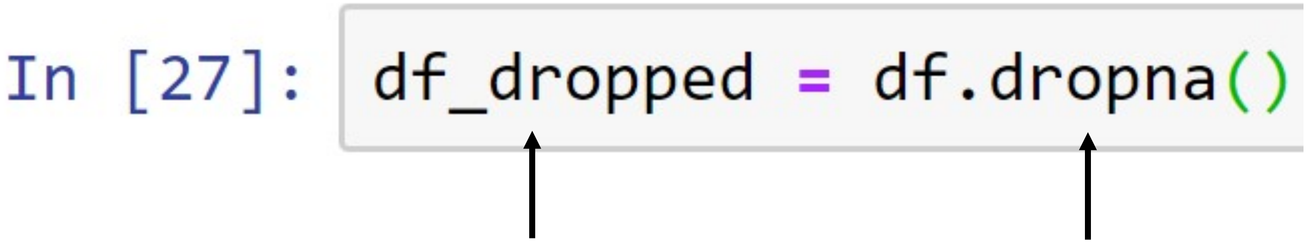


	col1	col2	col3
0	1.0	444.0	orange
1	2.0	555.0	apple
2	3.0	NaN	grape
3	4.0	444.0	mango
4	NaN	666.0	jackfruit
5	6.0	111.0	watermelon
6	7.0	NaN	banana
7	NaN	222.0	peach

- Missing values lead to problems in the data analysis process

Dataframes: Dropping values

- You can use the dropna( ) function to drop all the rows consisting of the missing values



```
In [27]: df_dropped = df.dropna()
```

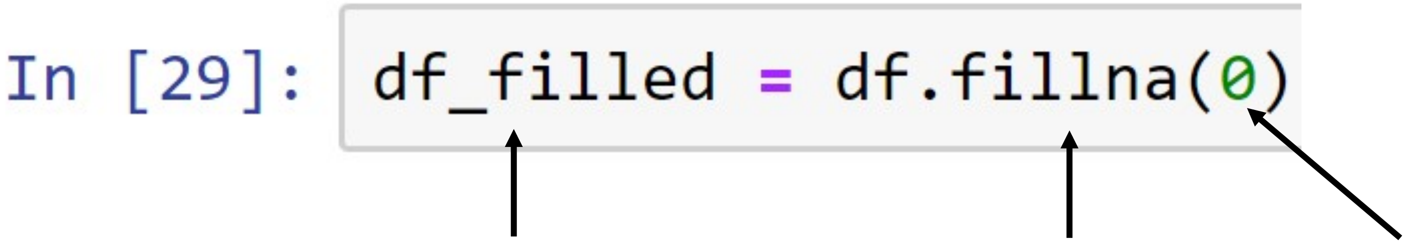
New dataframe after filling values

Function to drop values

(Demo)

Dataframes: Filling values

- You can also use the fillna( ) function to fill values (e.g. a value of ‘0’ in the place of ‘NaN’) in the place of missing values



```
In [29]: df_filled = df.fillna(0)
```

New dataframe after filling values

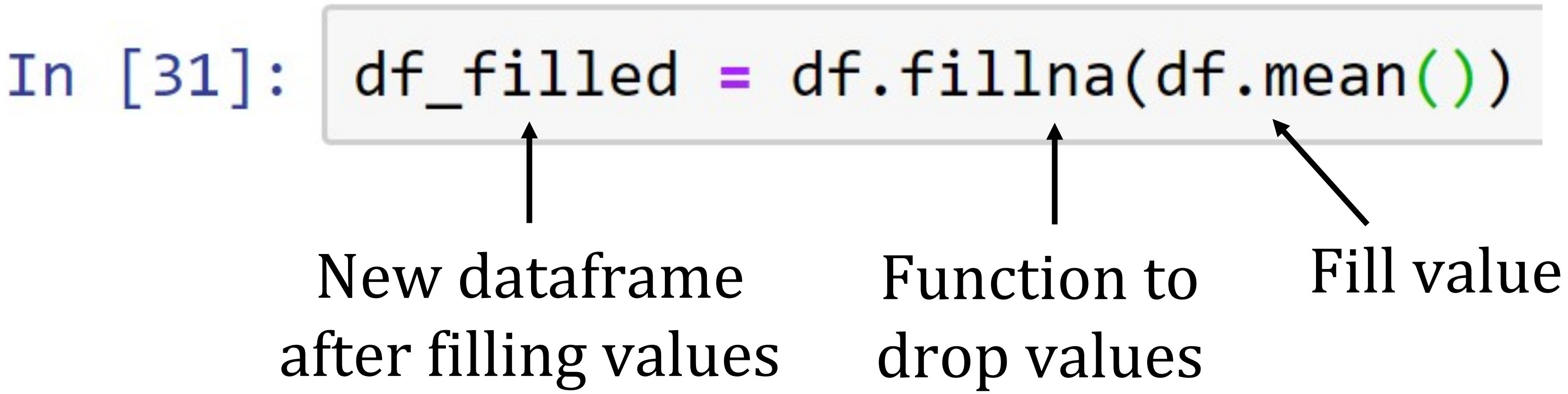
Function to drop values

Fill value

(Demo)

Dataframes: Filling values

- You can also use the fillna( ) function to fill values (e.g. mean value of each column in the place of ‘NaN’) in the place of missing values



```
In [31]: df_filled = df.fillna(df.mean())
```

New dataframe after filling values

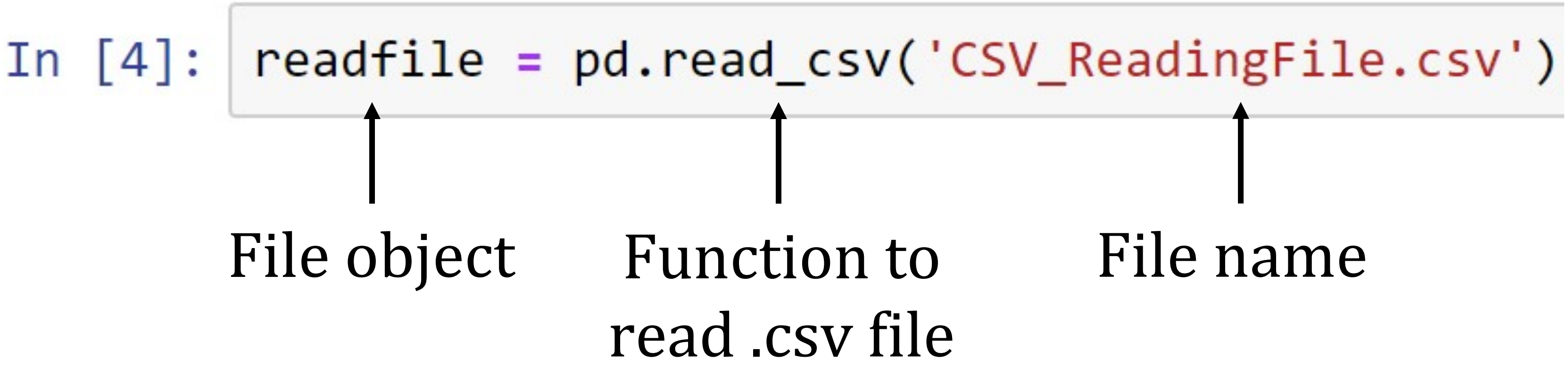
Function to drop values

Fill value

(Demo)

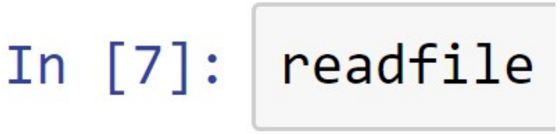
Dataframes: Reading a File

- Objective is to read the data in a ‘.csv’ (comma separated values) file and print it as a dataframe



```
In [4]: readfile = pd.read_csv('CSV_ReadingFile.csv')
```

- Printing the contents of the .csv file to the output screen

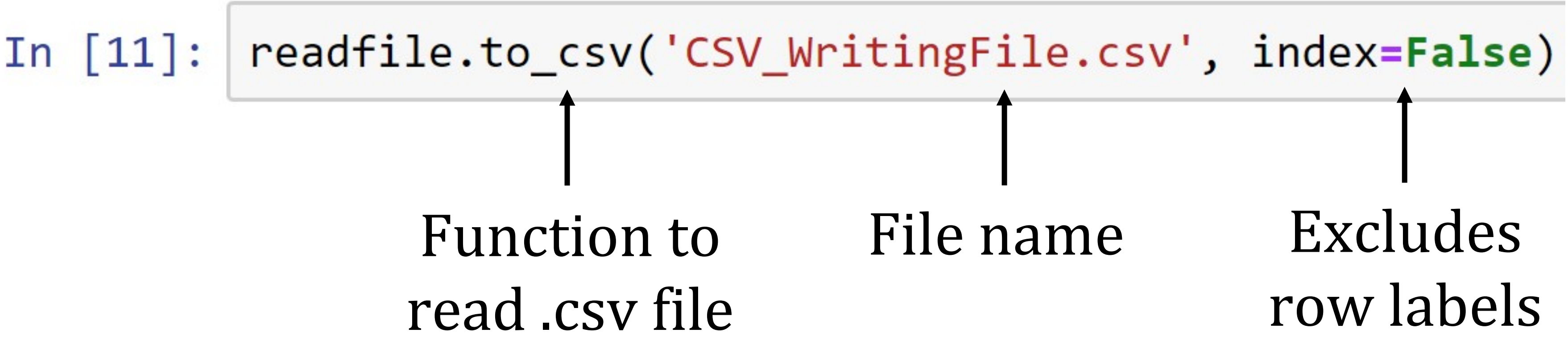


```
In [7]: readfile
```

(Demo)

Dataframes: Writing a File

- Objective is to write the data in a new '.csv' (comma separated values) file



```
In [11]: readfile.to_csv('CSV_WritingFile.csv', index=False)
```

Function to read .csv file

File name

Excludes row labels

- Note: File name that you give will first be created in the same folder where the Jupyter notebook is present

(Demo)

Summary

- Concepts of representing data in the form of Pandas dataframes are covered

- Concepts of interpreting, manipulating, and analyzing data within Pandas dataframes are covered