Computational Physics

Topic 01 : Exploratory Data Analysis

Lecture 01 : Top X pandas commands

Dr Kieran Murphy

Department of Science, WIT. (kmurphy@wit.ie)

Autumn Semester, 2021

Outline

- Reading data formats
- Computing descriptive statistics
- Processing data by filtering and grouping

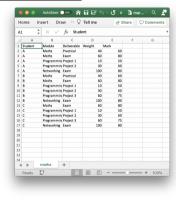
Part I

Introduction

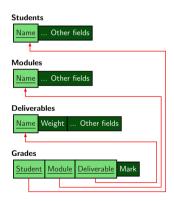
Minimal Dataset

To better understand the various pandas operations we are going use a tiny* dataset based on (fictional) student results. (marks.csv)





... or database schema ...



...like to know ...

- Student performance weighted mark on each module, missing deliverables etc.
- Module performance number of attempts and average mark.
- Deliverable performance number of attempts and average mark, predictor of overall module grade, etc.

^{*}Dataset is small enough that you can verify operation results by hand.

Termonology

df.head(1000)						
	Student	Module	Deliverable	Weight	Mark	
0	Α	Maths	Practical	40	60	
1	Α	Maths	Exam	60	80	
2	Α	Programming	Project 1	10	50	
3	Α	Programming	Project 2	30	60	
4	Α	Networking	Lab Work	100	80	
5	В	Maths	Practical	40	60	
6	В	Maths	Exam	60	80	
7	В	Programming	Project 1	10	50	
8	В	Programming	Project 2	30	60	
9	В	Programming	Project 3	60	75	
10	В	Networking	Project	100	80	
11	С	Maths	Exam	60	80	
12	С	Programming	Project 1	10	50	
13	С	Programming	Project 2	30	60	
14	С	Programming	Project 3	60	75	
15	С	Networking	Lab Work	100	80	

- A DataFrame is a table of data values.
 - df = pd.read_csv("marks.csv")
- A Series is a list of data values typically columns in a dataframe. We can access an individual column using
 - df.Deliverable (dot notation
 - df["Deliverable"] (dict notation
 - df.iloc[:,2] (numpy, index notation)
- The index is a special column whose values can be used to access rows — rather using row number.
 - The default index is equal to the row number.

Termonology

df.head(1000)					
	Student	Module	Deliverable	Veight	Mark
0	Α	Maths	Practical	40	60
1	Α	Maths	Exam	60	80
2	Α	Programming	Project 1	10	50
3	Α	Programming	Project 2	30	60
4	Α	Networking	Lab Work	100	80
5	В	Maths	Practical	40	60
6	В	Maths	Exam	60	80
7	В	Programming	Project 1	10	50
8	В	Programming	Project 2	30	60
9	В	Programming	Project 3	60	75
10	В	Networking	Project	100	80
11	С	Maths	Exam	60	80
13 C Programming		Programming	Project 1	10	50
		Programming	Project 2	30	60
		Programming	Project 3	60	75
15	С	Networking	Lab Work	100	80

- A DataFrame is a table of data values.
 - df = pd.read_csv("marks.csv")
- A Series is a list of data values typically columns in a dataframe. We can access an individual column using

```
• df.Deliverable (dot notation)
```

- df["Deliverable"] (dict notation)
- df.iloc[:,2] (numpy, index notation)
- The index is a special column whose values can be used to access rows — rather using row number.
 - The default index is equal to the row number.

Termonology

df.	df.head(1000)							
	Student	Module	Deliverable	Weight	Mark			
0	Α	Maths	Practical	40	60			
1	А	Maths	Exam	60	80			
2	А	Programming	Project 1	10	50			
3	А	Programming	Project 2	30	60			
4	А	Networking	Lab Work	100	80			
5	В	Maths	Practical	40	60			
6	В	Maths	Exam	60	80			
7	В	Programming	Project 1	10	50			
8	В	Programming	Project 2	30	60			
9	В	Programming	Project 3	60	75			
10	В	Networking	Project	100	80			
11	С	Maths	Exam	60	80			
12	С	Programming	Project 1	10	50			
13	С	Programming	Project 2	30	60			
14	С	Programming	Project 3	60	75			
15	С	Networking	Lab Work	100	80			
	•							

- A DataFrame is a table of data values.
 - df = pd.read_csv("marks.csv")
- A Series is a list of data values typically columns in a dataframe. We can access an individual column using

```
• df.Deliverable (dot notation)
```

- df["Deliverable"] (dict notation)
- df.iloc[:,2] (numpy, index notation)
- The index is a special column whose values can be used to access rows — rather using row number.
 - The default index is equal to the row number.

Part II

Input and Output

Reading data from a CSV file

Pandas supports a huge variety of input/output formats so best approach is to focus on what is needed to process the given data and verify input. Our marks dataset is in CSV format so we start with

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
plt.style.use('seaborn-darkgrid')
```

and input using

```
df = pd.read_csv('marks.csv', sep=',')
print(df.shape)
df.head()
```

Reading data from a CSV file

Pandas supports a huge variety of input/output formats so best approach is to focus on what is needed to process the given data and verify input. Our marks dataset is in CSV format so we start with

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
plt.style.use('seaborn-darkgrid')
```

and input using

```
df = pd.read_csv('marks.csv', sep=',')
print(df.shape)
df.head()
```

Reading data from a CSV file

Pandas supports a huge variety of input/output formats so best approach is to focus on what is needed to process the given data and verify input. Our marks dataset is in CSV format so we start with

```
import numpy as np
  import pandas as pd
  import matplotlib.pvplot as plt
  plt.stvle.use('seaborn-darkgrid')
                                                                               (16. 5)
                             Always verify input by checking dataset
                                                                                 Student
and input using
                             dimensions and looking at some rows,
                                                                                         Maths
                                                                                                Exam
  df = pd.read csv('marks.csv', sep=',')
                                                                                    A Programming
                                                                                               Project 1
  print(df.shape)
                                                                                               Project 2
                                                                                              Lab Work
  df.head()
```

Datatypes

Pandas data types:

- object used for text or mixed numeric and non-numeric values.
- int64 integer values,
 - Does not support missing values, so an int column containing at least one missing value will automatically be converted to float.
- float64 floating point numbers.
- bool True/False values
- datetime64 date and time values
- category Finite (typically small) list of text values

category Time (typically small) hist of text value

Student object
Module object
Deliverable object
Weight int64
Mark int64
dtype: object

df.dtypes

Regularly verifying datatypes is vital[†]:

- Operations differ based on datatype, eg, '+' concatenate strings but adds numerical values.
- datatype can change based on results, eg, int converts to float due to missing values.

[†]Google "Detecting Excel's gene auto-conversions."

Datatypes

Pandas data types:

- object used for text or mixed numeric and non-numeric values.
- int64 integer values,
 - Does not support missing values, so an int column containing at least one missing value will automatically be converted to float.
- float64 floating point numbers.
- bool True/False values
- datetime64 date and time values
- category Finite (typically small) list of text values

df.dtypes

Student object
Module object
Deliverable object
Weight int64
Mark int64
dtype: object

Regularly verifying datatypes is vital[†]:

- Operations differ based on datatype, eg, '+' concatenate strings but adds numerical values.
- datatype can change based on results, eg, int converts to float due to missing values.

[†]Google "Detecting Excel's gene auto-conversions."

Datatypes

Pandas data types:

- object used for text or mixed numeric and non-numeric values.
- int64 integer values,
 - Does not support missing values, so an int column containing at least one missing value will automatically be converted to float.
- float64 floating point numbers.
- bool True/False values
- datetime64 date and time values
- category Finite (typically small) list of text values

df.dtypes

Student object
Module object
Deliverable object
Weight int64
Mark int64
dtype: object

Regularly verifying datatypes is vital[†]:

- Operations differ based on datatype, eg, '+' concatenate strings but adds numerical values.
- datatype can change based on results, eg, int converts to float due to missing values.

[†]Google "Detecting Excel's gene auto-conversions."

Datatypes — Converting

We will deal with modifying and creating new columns later, but while we are on datatypes, we will look at changing datatypes . . .

Using the Series function astype

```
df["Weight"] = df["Weight"].astype('float')
df["Weight"].dtype

dtype('float64')
```

- New datatype is required argument 'int', 'float', 'str', 'object', 'category', etc.
- Simple, but fragile if data conversion is possible.

or using pandas function to_numeric

```
df["Weight"] = pd.to_numeric(df["Weight"])
df["Weight"].dtype

dtype('float64')
```

- More powerful, can specify what to do in cases where the conversion fails etc
- Have functions to_numeric, to_datetime, and to_timedelta.

Datatypes — Converting

We will deal with modifying and creating new columns later, but while we are on datatypes, we will look at changing datatypes . . .

Using the Series function astype

```
df["Weight"] = df["Weight"].astype('float')
df["Weight"].dtype

dtype('float64')
```

- New datatype is required argument 'int', 'float', 'str', 'object', 'category', etc.
- Simple, but fragile if data conversion is possible.

or using pandas function to_numeric

```
df["Weight"] = pd.to_numeric(df["Weight"])
df["Weight"].dtype
dtype('float64')
```

- More powerful, can specify what to do in cases where the conversion fails etc
- Have functions to_numeric, to_datetime, and to_timedelta.

Datatypes — Converting

We will deal with modifying and creating new columns later, but while we are on datatypes, we will look at changing datatypes...

Using the Series function astype

```
df["Weight"] = df["Weight"].astype('float')
df["Weight"].dtype

dtype('float64')
```

- New datatype is required argument 'int', 'float', 'str', 'object', 'category', etc.
- Simple, but fragile if data conversion is possible.

```
or using pandas function to_numeric
```

```
df["Weight"] = pd.to_numeric(df["Weight"])
df["Weight"].dtype

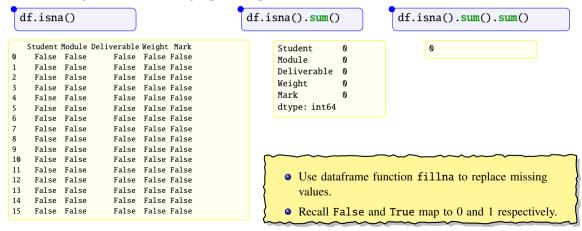
dtype('float64')
```

- More powerful, can specify what to do in cases where the conversion fails etc
- Have functions to_numeric, to_datetime, and to_timedelta.

Missing Values

Identifying and dealing with missing values is critical step in data preparation. What should you do? delete rows containing missing values? or impute then?

Here we will just look at identifying missing values.



Output

Saving dataframe to CSV is straightforward (I rarely include the (default) index when saving datasets).

```
df.to_csv('marks_2.csv', index=False)
```

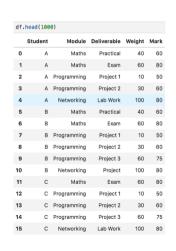
- CSV has become the default file format in Data Mining application especially for 'informal' datasets.
- Other formats are better for speed and resulting file size and for saving meta data not supported by CSV (such as columns datatypes, category information, etc).

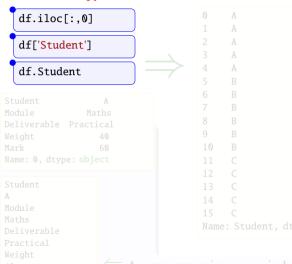
Part III

Filtering

Selecting individual rows/columns results in a series

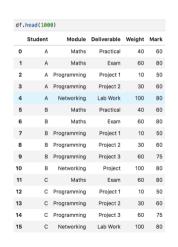
Columns can accessed using dot, dict and numpy index notation.

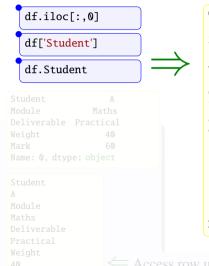


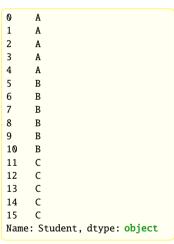


Selecting individual rows/columns results in a series

Columns can accessed using dot, dict and numpy index notation.



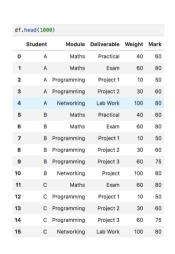


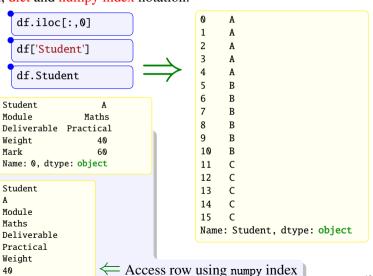


Selecting individual rows/columns results in a series

Mark

Columns can accessed using dot, dict and numpy index notation.





Head and Tail

Commands head and tail return the first and last n rows (default n = 5) of a dataframe/series.

	Student	Module	Deliverable	Weight	Mark
0	A	Maths	Practical	40	60
1	A	Maths	Exam	60	80
2	A	Programming	Project 1	10	50
3	A	Programming	Project 2	30	60
4	A	Networking	Lab Work	100	80
5	В	Maths	Practical	40	60
6	В	Maths	Exam	60	80
7	В	Programming	Project 1	10	50
8	В	Programming	Project 2	30	60
9	В	Programming	Project 3	60	75
10	В	Networking	Project	100	80
11	C	Maths	Exam	60	80
12	С	Programming	Project 1	10	50
13	C	Programming	Project 2	30	60
14	C	Programming	Project 3	60	75
15	C	Networking	Lab Work	100	80





Head and Tail

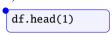
Commands head and tail return the first and last n rows (default n = 5) of a dataframe/series.

	Student	Module	Deliverable	Weight	Mark
0	A	Maths	Practical	40	60
1	A	Maths	Exam	60	80
2	A	Programming	Project 1	10	50
3	A	Programming	Project 2	30	60
4	A	Networking	Lab Work	100	80
5	В	Maths	Practical	40	60
6	В	Maths	Exam	60	80
7	В	Programming	Project 1	10	50
8	В	Programming	Project 2	30	60
9	В	Programming	Project 3	60	75
10	В	Networking	Project	100	80
11	С	Maths	Exam	60	80
12	С	Programming	Project 1	10	50
13	С	Programming	Project 2	30	60
14	С	Programming	Project 3	60	75
15	C	Networking	Lab Work	100	80



	Student	Module	Deliverable	Weight	Mark
0	A	Maths	Practical	40	60
1	A	Maths	Exam	60	80
2	A	Programming	Project 1	10	50
3	A	Programming	Project 2	30	60
4	A	Networking	Lab Work	100	80





Studer	nt Module	Deliverable	Weight	Mark
0 A	Maths	Practical	40	60

df.tail(1)

Head and Tail

Commands head and tail return the first and last n rows (default n = 5) of a dataframe/series.

	Student	Module	Deliverable	Weight	Mark
0	A	Maths	Practical	40	60
1	A	Maths	Exam	60	80
2	A	Programming	Project 1	10	50
3	A	Programming	Project 2	30	60
4	A	Networking	Lab Work	100	80
5	В	Maths	Practical	40	60
6	В	Maths	Exam	60	80
7	В	Programming	Project 1	10	50
8	В	Programming	Project 2	30	60
9	В	Programming	Project 3	60	75
10	В	Networking	Project	100	80
11	С	Maths	Exam	60	80
12	С	Programming	Project 1	10	50
13	С	Programming	Project 2	30	60
14	С	Programming	Project 3	60	75
15	С	Networking	Lab Work	100	80



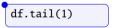
	Student	Module	Deliverable	Weight	Marl
0	A	Maths	Practical	40	60
1	A	Maths	Exam	60	80
2	A	Programming	Project 1	10	50
3	A	Programming	Project 2	30	60
4	A	Networking	Lab Work	100	80



	Student	Module	Deliverable	Weight	Mark
11	С	Maths	Exam	60	80
12	С	Programming	Project 1	10	50
13	С	Programming	Project 2	30	60
14	С	Programming	Project 3	60	75
15	С	Networking	Lab Work	100	80



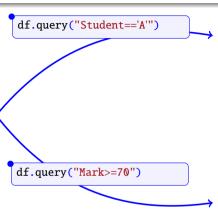
	Student	Module	Deliverable	Weight	Mark
(0 A	Maths	Practical	40	60



	Student	Module	Deliverable	Weight	Mark
15	С	Networking	Lab Work	100	80

Query — on a single-column criteria

	Student	Module	Deliverable	Weight	Mar
0	A	Maths	Practical	40	60
1	A	Maths	Exam	60	80
2	A	Programming	Project 1	10	50
3	A	Programming	Project 2	30	60
4	A	Networking	Lab Work	100	80
5	В	Maths	Practical	40	60
6	В	Maths	Exam	60	80
7	В	Programming	Project 1	10	50
8	В	Programming	Project 2	30	60
9	В	Programming	Project 3	60	75
10	В	Networking	Project	100	80
11	С	Maths	Exam	60	80
12	С	Programming	Project 1	10	50
13	C	Programming	Project 2	30	60
14	С	Programming	Project 3	60	75
15	С	Networking	Lab Work	100	80



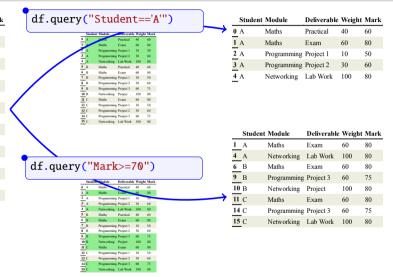
Query — on a single-column criteria

	Student	Module	Deliverable	Weight	Mar
0	A	Maths	Practical	40	60
1	A	Maths	Exam	60	80
2	A	Programming	Project 1	10	50
3	A	Programming	Project 2	30	60
4	A	Networking	Lab Work	100	80
5	В	Maths	Practical	40	60
6	В	Maths	Exam	60	80
7	В	Programming	Project 1	10	50
8	В	Programming	Project 2	30	60
9	В	Programming	Project 3	60	75
10	В	Networking	Project	100	80
11	C	Maths	Exam	60	80
12	С	Programming	Project 1	10	50
13	C	Programming	Project 2	30	60
14	С	Programming	Project 3	60	75
15	C	Networking	Lab Work	100	80

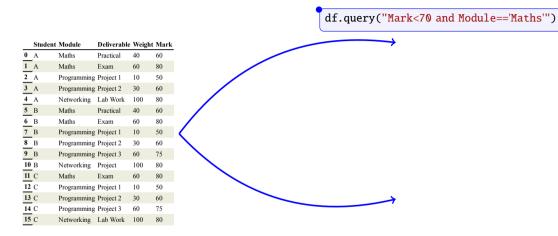


Query — on a single-column criteria

	Student	Module	Deliverable	Weight	Marl
0	A	Maths	Practical	40	60
1	A	Maths	Exam	60	80
2	A	Programming	Project 1	10	50
3	A	Programming	Project 2	30	60
4	A	Networking	Lab Work	100	80
5	В	Maths	Practical	40	60
6	В	Maths	Exam	60	80
7	В	Programming	Project 1	10	50
8	В	Programming	Project 2	30	60
9	В	Programming	Project 3	60	75
10	В	Networking	Project	100	80
11	C	Maths	Exam	60	80
12	С	Programming	Project 1	10	50
13	C	Programming	Project 2	30	60
14	С	Programming	Project 3	60	75
15	C	Networking	Lab Work	100	80

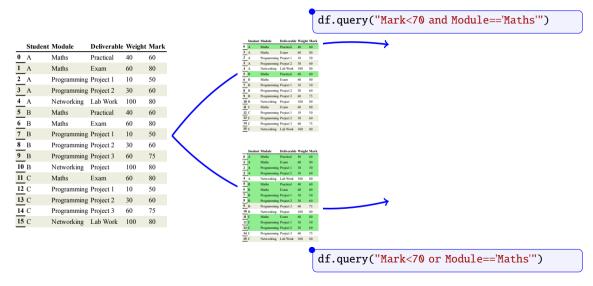


Query — on multiple columns (using python ogical operators)



df.query("Mark<70 or Module=='Maths'")</pre>

Query — on multiple columns (using python ogical operators)



Query — on multiple columns (using python ogical operators)

df.query("Mark<70 and Module=='Maths'")

Maths

Maths

	Student	Module	Deliverable	Weight	Mark
0	A	Maths	Practical	40	60
1	A	Maths	Exam	60	80
2	A	Programming	Project 1	10	50
3	A	Programming	Project 2	30	60
4	A	Networking	Lab Work	100	80
5	В	Maths	Practical	40	60
6	В	Maths	Exam	60	80
7	В	Programming	Project 1	10	50
8	В	Programming	Project 2	30	60
9	В	Programming	Project 3	60	75
10	В	Networking	Project	100	80
11	С	Maths	Exam	60	80
12	С	Programming	Project 1	10	50
13	С	Programming	Project 2	30	60
14	C	Programming	Project 3	60	75
15	С	Networking	Lab Work	100	80

T						
A Mades Isam 60 12 A Postgrammer Propost 2 30 56 A Postgrammer	_	Student				
2 A Degenerate Project 10 16 2 A Negenerate Project 10 16 4 4 A Nelsouchig Lab Week 10 16 4 4 A Nelsouchig Lab Week 10 16 4 3 B Mode Project 10 5 5 2 B Degenerate Project 10 5 6 7 3 B Project 10 7 6 7 3 B Project 10 6 7 1 3 C Project 10 6 7 1 1 6 7 1		A	Maths	Practical	40	60
A Notice-step Left Windows L	1	A	Maths	Exam	60	80
A Notice-step Left Windows L	2	A	Programming	Project I	10	50
Number Presented 10 10 10 10 10 10 10 1	3	A	Programming	Project 2	30	60
Marks Exam 60 10	4		Networking	Lab Work	100	80
Department Project 10 5 5	5		Maths	Practical	40	60
1			Maths	Exam	60	80
	7	В	Programming	Project I	10	50
			Programming	Project 2	30	60
	9	В	Programming	Project 3	60	75
Dec Department Project 10 15 15 15 15 15 15 15	10	В	Networking	Project	100	80
December Properties 20	11	C	Maths	Exam	60	80
December Project 40 7 1 1 1 1 1 1 1 1 1	12	C	Programming	Project I	10	50
Section Column	13	C	Programming.	Project 2	30	60
Subset Velocide Deliverate Weight	14	C	Programming	Project 3	60	75
A. Maha Presson 40 10 11 12 13 14 15 15 15 15 15 15 15	15	C	Networking	Lab Work	100	80
Mark Exam 61						
A Mahs Exam 60 16 A Pospurmer Project 10 A Pospurmer Project 20 A Pospurmer Project 20 A Netteroll, Leb Vest 60 B Mahs Pacical 60 B Pospurmer Project 20 B Pospurmer Project 60 B Netteroll, Project 60 B Neteroll, Project 60 B Netteroll, Project 60 B Neteroll, Project 60 B Netteroll, Project 60 B Netterol		Student	Medule	Deliverable	Weight	Mark
A New November Project 2 10	0	Student				Mark 60
A Note coding Lab Work 100 16		Student A A	Maths	Practical	40	
Nation Procised 40 64 6 11 Males Procised 40 64 6 11 10 10 10 10 10 10	1	Student A A A	Maths Maths	Practical Exam	40 60	60
6 B Marla Exim 60 84 7 B Pogarining Project 1 10 55 8 B Pogarining Project 2 30 66 9 B Pogarining Project 30 60 75 10 B Networking Project 100 88 11 C Marlos Exim 60 88 12 C Progarining Project 2 30 66 13 C Progarining Project 3 60 75	1 2	A A A A	Maths Maths Programming	Practical Exam Project 1	40 60 10	60 80
1 Programming Project 10 56	1 2	A A A	Maths Maths Programming Programming	Practical Exam Project 1 Project 2	40 60 10 30	60 80 50
B Programming Project 2 30 66 75 9 B Programming Project 3 60 75 10 B Networking Project 1 10 86 11 C Mathe Exam 60 83 12 C Programming Project 1 10 55 13 C Programming Project 2 30 66 14 C Programming Project 3 60 75	3 4 5	A A A A	Maths Maths Programming Programming Networking	Practical Exam Project I Project 2 Lab Work	40 60 10 30 100	60 80 50 60
5 B Programming Project 3 60 75	3 4 5	A A A A B	Maths Maths Programming Programming Networking Maths	Practical Exam Project 1 Project 2 Lab Work Practical	40 60 10 30 100 40	60 80 50 60 80
10 B Networking Project 100 80 11 C Maths Exam 60 81 12 C Programming Project 10 50 13 C Programming Project 30 60 75 14 C Programming Project 30 60 75 14 C Programming Project 30 60 75 15 C Programming Project 30 60 75 16 C Programming Project 30 60 75 17 C Programming Project 30 60 75 18 C Programming Project 30 60 75 60 75 75 75 75 75 75 75 7	3 4 5	A A A A B B	Maths Maths Programming Programming Networking Maths Maths	Practical Exam Project I Project 2 Lab Work Practical Exam	40 60 10 30 100 40 60	60 80 50 60 80
II C Maths Exam 60 80 12 C Programming Project 1 10 50 13 C Programming Project 2 30 60 14 C Programming Project 3 60 75 75 75 75 75 75 75 7	1 2 3 4 5 6	A A A A B B	Maths Maths Programming Programming Networking Maths Maths Programming	Practical Exam Project 1 Project 2 Lab Work Practical Exam Project 1	40 60 10 30 100 40 60	60 80 50 60 80 60 80
12 C Programming Project 1 10 50 13 C Programming Project 2 30 60 14 C Programming Project 3 60 75	1 2 3 4 5 6 7 8	A A A A B B B	Maths Maths Programming Programming Networking Maths Maths Programming Programming	Practical Exam Project 1 Project 2 Lab Work Practical Exam Project 1 Project 2	40 60 10 30 100 40 60 10 30	60 80 50 60 80 60 80 50
13 C Programming Project 2 30 60 14 C Programming Project 3 60 75	1 2 3 4 5 6 7 8	A A A A B B B B	Maths Maths Programming Programming Networking Maths Maths Programming Programming Programming	Practical Exam Project 1 Project 2 Lab Work Practical Exam Project 1 Project 2 Project 2 Project 3	40 60 10 30 100 40 60 10 30 60	60 80 50 60 80 60 80 50
14 C Programming Project 3 60 75	1 2 3 4 5 6 7 8 9	A A A A B B B B B	Maths Maths Programming Programming Networking Maths Maths Programming Programming Programming Networking	Practical Exam Project 1 Project 2 Lab Work Practical Exam Project 1 Project 2 Project 3 Project 3	40 60 10 30 100 40 60 10 30 60	60 80 50 60 80 60 80 50 60
	1 2 3 4 5 6 7 8 9	A A A A B B B B B B	Maths Maths Programming Programming Networking Maths Maths Programming Programming Programming Networking Maths	Practical Exam Project 1 Project 2 Lab Work Practical Exam Project 1 Project 2 Project 3 Project Exam	40 60 10 30 100 40 60 10 30 60 100 60	60 80 50 60 80 60 80 50 60 75
15 C Networking Lab Work 100 90	1 2 3 4 5 6 7 8 9 10	A A A A B B B B B C	Maths Maths Programming Programming Networking Maths Maths Programming Programming Networking Maths Programming Programming Networking Maths Programming	Practical Exam Project 1 Project 2 Lab Work Practical Exam Project 1 Project 2 Project 3 Project 3 Project 1 Exam Project 1	40 60 10 30 100 40 60 10 30 60 100 60 100	60 80 50 60 80 60 80 50 60 75 80
	1 2 3 4 5 6 7 8 9 10 11 12 13	A A A A B B B B B C C	Maths Maths Programming Programming Networking Maths Maths Programming Programming Networking Maths Programming Networking Programming Programming Programming Programming Programming Programming Programming Programming	Project 1 Project 1 Project 2 Lab Work Practical Exam Project 1 Project 3 Project 3 Project 1 Exam	40 60 10 30 100 40 60 10 30 60 100 60 100 30	60 80 50 60 80 60 80 50 60 75 80 80 50

	Student	Module	Deliverable	Weight	Mark
0	A	Maths	Practical	40	60
1	A	Maths	Exam	60	80
2	A	Programming	Project 1	10	50
3	A	Programming	Project 2	30	60
5	В	Maths	Practical	40	60
6	В	Maths	Exam	60	80
7	В	Programming	Project 1	10	50
8	В	Programming	Project 2	30	60
11	C	Maths	Exam	60	80
12	C	Programming	Project 1	10	50
13	C	Programming	Project 2	30	60

Student Module Deliverable Weight Mark

Practical

Practical

60

60

df.query("Mark<70 or Module=='Maths'")</pre>

Query — on multiple columns (using pandas ogical operators)

df.query("(Mark<70) & (Module=='Maths')")</pre>

Maths

Maths

	Student	Module	Deliverable	Weight	Mark
0	A	Maths	Practical	40	60
1	A	Maths	Exam	60	80
2	A	Programming	Project 1	10	50
3	A	Programming	Project 2	30	60
4	A	Networking	Lab Work	100	80
5	В	Maths	Practical	40	60
6	В	Maths	Exam	60	80
7	В	Programming	Project 1	10	50
8	В	Programming	Project 2	30	60
9	В	Programming	Project 3	60	75
10	В	Networking	Project	100	80
11	C	Maths	Exam	60	80
12	С	Programming	Project 1	10	50
13	С	Programming	Project 2	30	60
14	C	Programming	Project 3	60	75
15	С	Networking	Lab Work	100	80

	Student	Module	Deliverable	Weight	Mark
0	A	Maths	Practical	40	60
1	A	Maths	Exam	60	80
2	A	Programming	Project I	10	50
3	A	Programming.	Project 2	30	60
4	A	Networking	Lab Work	100	80
5	В	Maths	Practical	40	60
6	В	Maths	Exam	60	80
7	В	Programming.	Project I	10	50
8	В	Programming.	Project 2	30	60
9	В	Programming.	Project 3	60	75
10	В	Networking	Project	100	80
11	C	Maths	Exam	60	80
	C	Programming	Project I	10	50
		Bearing	Project 2	30	60
	C				
		Programming	Project 3	60	75
13			Project 3 Lab Work	100	75 80
13	c c	Programming		100	80
13	c c	Programming Networking	Lab Work	100	80
13	c c	Programming Networking Medule	Lab Week Deliverable	100 Weight	80 Mark
13 14 15	c c	Programming Networking Medule Maths Maths	Lab Work Deliverable Practical Exam	100 Weight	Mark 60
13 14 15	c c	Programming Networking Medule Maths	Lab Work Deliverable Practical Exam Project I	100 Weight 40 60	80 Mark 60 80
13 14 15	c c	Programming Networking Medule Maths Maths Programming	Lab Work Deliverable Practical Exam Project I	Weight 40 60	Mark 60 80 50
13 14 15 1 2 3 4 5	C C Student A A A	Programming Networking Medule Maths Maths Programming Programming	Lab Work Deliverable Practical Exam Project 1 Project 2	Weight 40 60 10 30	Mark 60 80 50
13 14 15 1 2 3	C C Student A A A A	Programming Networking Medule Maths Maths Programming Programming Networking	Lab Work Deliverable Practical Exam Project 1 Project 2 Lab Work	Weight 40 60 10 30	Mark 60 80 50 60
13 14 15 1 2 3 4 5	C C Student A A A A A	Programming Networking Medule Maths Maths Programming Programming Networking Maths	Lab Work Deliverable Practical Exam Project 1 Project 2 Lab Work Practical Exam	Weight 40 60 10 30 100 40	Mark 60 80 50 60 80
13 14 15 1 2 3 4 5	C C Student A A A A A B B	Programming Networking Medule Maths Maths Programming Programming Networking Maths Maths	Lab Work Deliverable Practical Exam Project 1 Project 2 Lab Work Practical Exam Project 1	Weight 40 60 10 30 100 40 60	Mark 60 80 50 60 80 80
13 14 15 6 7 8	Student A A A A A B B B	Programming Networking Medule Marks Marks Programming Marks Marks Programming Programming Programming Programming Programming Programming Programming Programming	Deliverable Practical Exam Project 1 Project 2 Lab Work Practical Exam Project 1 Project 1 Project 2	Weight 40 60 10 30 100 40 60 10	Mark 60 80 50 60 80 60 80
13 14 15 1 2 3 4 5	C C Student A A A A B B B B B B	Programming Networking Medule Maths Marhs Programming Networking Maths Maths Programming Programming Programming Programming Programming	Deliverable Practical Exam Project 1 Project 2 Lab Work Practical Exam Project 1 Project 1 Project 2	Weight 40 60 10 30 100 40 60 10 30	Mark 60 80 50 60 80 60 80 60
13 14 15 15 6 7 8 9	C C Student A A A A B B B B B B	Programming Medule Maths Maths Maths Programming Networking Maths Maths Programming Programming Programming Programming Programming Programming Programming Programming Programming	Deliverable Practical Exam Project 1 Project 2 Lab Work Practical Exam Project 1 Project 2 Project 2 Project 3	Weight 40 60 10 30 100 40 60 10 30 60	Mark 60 80 50 60 80 60 80 50
13 14 15 15 6 7 8 9	C C Student A A A A B B B B B C C	Programming Networking Medule Maths Maths Maths Networking Maths Programming Networking Maths Programming Networking Networking Networking Networking	Lab Work Deliverable Practical Exam Project 1 Project 2 Lab Work Exam Project 1 Project 1 Project 2 Project 3 Project 3 Project 3	Weight 40 60 10 30 100 40 60 10 30 60 100	Mark 60 80 50 60 80 50 60 80 50 60 75
13 14 15 15 6 7 8 9	C C Student A A A A B B B B C C C	Programming Networking Medule Maths Maths Programming Networking Maths Maths Programming Programming Programming Programming Programming Networking Maths	Deliverable Practical Exam Project 1 Project 2 Lab Work Practical Exam Project 1 Project 1 Project 2 Project 3 Project 3 Project 3 Project 1 Exam Project 1 Exam Project 1	Weight 40 60 10 30 100 40 60 10 30 60 100 60	Mark 60 80 50 60 80 50 60 80 75 80 80 80 80 80 80 80 80 80 80 80 80 80
13 14 15 15 6 7 8 9 10 11 12	C C Student A A A A B B B B B C C C C C C C	Programming Medule Maths Maths Programming Maths Programming Programming Programming Programming Programming Programming Programming Programming Programming Maths Maths Programming Programming Maths Programming Maths Programming Maths	Deliverable Practical Exam Project 2 Lab Work Practical Exam Project 1 Project 2 Project 3 Project 3 Project 1 Exam Project 1 Project 1 Project 1 Project 1 Project 2	Weight 40 60 10 30 100 40 60 10 30 60 100 60 100	Mark 60 80 50 60 80 50 60 80 50 60 80 50 60 80 50 60 80 50 60 80 50 60 80 50 60 80 80 80 80 80 80 80 80 80 80 80 80 80

	Student	Module	Deliverable	Weight	Mark
0	A	Maths	Practical	40	60
1	A	Maths	Exam	60	80
2	A	Programming	Project 1	10	50
3	A	Programming	Project 2	30	60
5	В	Maths	Practical	40	60
6	В	Maths	Exam	60	80
7	В	Programming	Project 1	10	50
8	В	Programming	Project 2	30	60
11	C	Maths	Exam	60	80
12	С	Programming	Project 1	10	50
13	C	Programming	Project 2	30	60

Student Module Deliverable Weight Mark

Practical

Practical

60

60

df.query("(Mark<70) | (Module=='Maths')")</pre>

Filtering using 10c

	the		ackets —	

		Student	Module	Deliverable	Weight	Marl
	0	A	Maths	Practical	40	60
į	1	A	Maths	Exam	60	80
	2	A	Programming	Project 1	10	50
	3	A	Programming	Project 2	30	60
	4	A	Networking	Lab Work	100	80
ĺ	5	В	Maths	Practical	40	60
	6	В	Maths	Exam	60	80
ĺ	7	В	Programming	Project 1	10	50
	8	В	Programming	Project 2	30	60
ĺ	9	В	Programming	Project 3	60	75
	10	В	Networking	Project	100	80
	11	С	Maths	Exam	60	80
	12	С	Programming	Project 1	10	50
	13	С	Programming	Project 2	30	60
	14	C	Programming	Project 3	60	75
į	15	С	Networking	Lab Work	100	80

df.loc[ROW_SELECTION, COL_SELECTION] where row and columns selection can be

- Single values: row number or column name
- An integer list for rows or list of column names
- A boolean list for logical indexing of rows
- A colon to indicate every row/column

```
df.loc[df.Module=="Maths", ["Student", "Mark"]]
```

Filtering using 10c

Note the square (not round) brackets — think of loc as array indexing not a function call.

	Student	Module	Deliverable	Weight	Mark
0	A	Maths	Practical	40	60
1	A	Maths	Exam	60	80
2	A	Programming	Project 1	10	50
3	A	Programming	Project 2	30	60
4	A	Networking	Lab Work	100	80
5	В	Maths	Practical	40	60
6	В	Maths	Exam	60	80
7	В	Programming	Project 1	10	50
8	В	Programming	Project 2	30	60
9	В	Programming	Project 3	60	75
10	В	Networking	Project	100	80
11	С	Maths	Exam	60	80
12	С	Programming	Project 1	10	50
13	С	Programming	Project 2	30	60
14	С	Programming	Project 3	60	75
15	C	Networking	Lab Work	100	80

df.loc[ROW_SELECTION, COL_SELECTION]
where row and columns selection can be

- Single values: row number or column name
- An integer list for rows or list of column names
- A boolean list for logical indexing of rows
- A colon to indicate every row/column

	Student	Module	Deliverable	Weight	Mark
D	Α	Maths	Practical	40	60
ı	Α	Maths	Exam	60	80
2	Α	Programming	Project 1	10	50
3	A	Programming	Project 2	30	60
4	A	Networking	Lab Work	100	80
5	В	Maths	Practical	40	60
6	В	Maths	Exam	60	80
7	В	Programming	Project 1	10	50
8	В	Programming	Project 2	30	60
9	В	Programming	Project 3	60	75
10	В	Networking	Project	100	80
11	C	Maths	Exam	60	80
12	C	Programming	Project 1	10	50
13	C	Programming	Project 2	30	60
14	C	Programming	Project 3	60	75
15	C	Networking	Lab Work	100	80

df.loc[df.Module=="Maths", ["Student", "Mark"]]

Filtering using 10c

Note the square (not round) brackets — think of loc as array indexing not a function call.

	Student	Module	Deliverable	Weight	Mark
0	A	Maths	Practical	40	60
1	A	Maths	Exam	60	80
2	A	Programming	Project 1	10	50
3	A	Programming	Project 2	30	60
4	A	Networking	Lab Work	100	80
5	В	Maths	Practical	40	60
6	В	Maths	Exam	60	80
7	В	Programming	Project 1	10	50
8	В	Programming	Project 2	30	60
9	В	Programming	Project 3	60	75
10	В	Networking	Project	100	80
11	C	Maths	Exam	60	80
12	С	Programming	Project 1	10	50
13	C	Programming	Project 2	30	60
14	С	Programming	Project 3	60	75
15	C	Networking	Lab Work	100	80

df.loc[ROW_SELECTION, COL_SELECTION]
where row and columns selection can be

- Single values: row number or column name
- An integer list for rows or list of column names
- A boolean list for logical indexing of rows
- A colon to indicate every row/column

Α	Maths	Practical	40	60
Α	Maths	Exam	60	80
Α	Programming	Project 1	10	50
A	Programming	Project 2	30	60
Α	Networking	Lab Work	100	80
В	Maths	Practical	40	60
В	Maths	Exam	60	80
В	Programming	Project 1	10	50
В	Programming	Project 2	30	60
В	Programming	Project 3	60	75
В	Networking	Project	100	80
С	Maths	Exam	60	80
С	Programming	Project 1	10	50
C	Programming	Project 2	30	60
С	Programming	Project 3	60	75
C	Networking	Lab Work	100	80
	A A A B B	A Maths A Programming A Programming A Networking B Maths B Programming B Programming B Programming C Maths C Programming C Programming C Programming	A Mults Exam A Programming Project 1 A Programming Project 2 A Networking Lab Work B Mults Exam B Programming Project 1 B Programming Project 1 B Programming Project 1 B Programming Project 2 B Programming Project C C Programming Project C	A Maths Exam 60 A Programming Project 1 10 A Programming Project 2 10 A Networking Lab Work 100 B Maths Function 60 B Maths Exam 60 B Programming Project 2 10 B Programming Project 1 10 B Programming Project 1 00 C Maths Exam 60 C Programming Project 1 00 C Programming Project 2 30 C Programming Project 2 30 C Programming Project 2 30 C Programming Project 3 30

	Student	Mark
0	A	60
1	A	80
5	В	60
6	В	80
11	С	80

df.loc[df.Module=="Maths", ["Student", "Mark"]]

More complicated example

I prefer to define define row selection criteria, and the column list and order, separately to the loc statement.

	Student	Module	Deliverable	Weight	Mark
0	A	Maths	Practical	40	60
1	A	Maths	Exam	60	80
2	A	Programming	Project 1	10	50
3	A	Programming	Project 2	30	60
4	A	Networking	Lab Work	100	80
5	В	Maths	Practical	40	60
6	В	Maths	Exam	60	80
7	В	Programming	Project 1	10	50
8	В	Programming	Project 2	30	60
9	В	Programming	Project 3	60	75
10	D	Matruarkina	Decidat	100	90

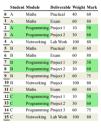
```
criteria = ((df.Mark<50) & (df.Module=='Maths')) | ((df.Mark<70) & (df.Module!='Maths'))
columns = ['Module', 'Student', 'Mark']</pre>
```

df.loc[criteria, columns]

More complicated example

I prefer to define define row selection criteria, and the column list and order, separately to the loc statement.

	Student	Module	Deliverable	Weight	Mark
0	A	Maths	Practical	40	60
1	A	Maths	Exam	60	80
2	A	Programming	Project 1	10	50
3	A	Programming	Project 2	30	60
4	A	Networking	Lab Work	100	80
5	В	Maths	Practical	40	60
6	В	Maths	Exam	60	80
7	В	Programming	Project 1	10	50
8	В	Programming	Project 2	30	60
9	В	Programming	Project 3	60	75
10	D	Matworkina	Decidat	100	90



```
criteria = ((df.Mark<50) & (df.Module=='Maths')) | ((df.Mark<70) & (df.Module!='Maths'))
columns = ['Module', 'Student', 'Mark']
df.loc[criteria, columns]</pre>
```

More complicated example

I prefer to define define row selection criteria, and the column list and order, separately to the loc statement.

	Student	Module	Deliverable	Weight	Mark
0	A	Maths	Practical	40	60
1	A	Maths	Exam	60	80
2	A	Programming	Project 1	10	50
3	A	Programming	Project 2	30	60
4	A	Networking	Lab Work	100	80
5	В	Maths	Practical	40	60
6	В	Maths	Exam	60	80
7	В	Programming	Project 1	10	50
8	В	Programming	Project 2	30	60
9	В	Programming	Project 3	60	75
10	D	Matworking	Decidat	100	90

	Student	Module	Deliverable	Weight	Mark
0	Α	Maths	Practical	40	60
1	Α	Maths	Exam	60	80
2	Α	Programming	Project l	10	50
3	Λ	Programming	Project 2	30	60
4	Α	Networking	Lab Work	100	80
5	В	Maths	Practical	40	60
6	В	Maths	Exam	60	80
7	В	Programming	Project 1	10	50
8	В	Programming	Project 2	30	60
9	В	Programming	Project 3	60	75
10	В	Networking	Project	100	80
11	C	Maths	Exam	60	80
12	C	Programming	Project 1	10	50
13	C	Programming	Project 2	30	60
14	C	Programming	Project 3	60	75
15	C	Networking	Lab Work	100	80

		Module	Student	Mark
	2	Programming	A	50
•	3	Programming	A	60
	7	Programming	В	50
	8	Programming	В	60
	12	Programming	C	50
	13	Programming	C	60

```
criteria = ((df.Mark<50) & (df.Module=='Maths')) | ((df.Mark<70) & (df.Module!='Maths'))
columns = ['Module', 'Student', 'Mark']
df.loc[criteria, columns]</pre>
```

Sampling

The sample function selects a random subset of the dataframe rows.

- Either specify the number of rows (as an integer) or fraction of the data (as a float).
- Can set the seed using random_state parameter for reproducible samples.

	Module	Deliverable	Weigh	t Mark		St	udent	Module	Deliverable	Weight	M
	Maths	Practical	40	60		5 B	-	Maths	Practical	40	60
	Maths	Exam	60	80		_		wiaui3	Tactical	40	00
	Programming	Project 1	10	50		2 A		Programming	Project 1	10	50
	Programming	Project 2	30	60		9 в		Programming	Project 3	60	75
	Networking	Lab Work	100	80					Trojects	00	, ,
;	Maths	Practical	40	60							
3	Maths	Exam	60	80		S	tudent	Module	Deliverable	Weight	M
3	Programming	Project 1	10	50	df.sample(n=3)	0 A		Maths	Practical	40	60
	Programming	Project 2	30	60		_					
3	Programming	Project 3	60	75		15 C		Networking	Lab Work	100	80
	Networking	Project	100	80		6 B	3	Maths	Exam	60	80
2	Maths	Exam	60	80				With	DAMIII	00	00
2	Programming	Project 1	10	50							
2	Programming	Project 2	30	60		S	tudent	Module	Deliverabl	e Weigh	t N
2	Programming	Project 3	60	75		10 B	2	Networking	Project	100	8
2	Networking	Lab Work	100	80	7	10 1	,	U	Tioject		
						1 A		Maths	Exam	60	8
						14 (1	Programming	Project 3	60	7.

Part IV

Sorting

Sorting

A pandas dataframe has two sorting operations:

- sort_index() orders rows based on current index.
- sort_values(COLUMNS) orders rows based on single column or list of columns.

Two important modifications:

- By default, the sort order is in ascending. Set parameter ascending=False to reverse this.
- By default, a new dataframe is returned with desired sort order, set parameter inplace=True to update current dataframe instead.

21 of 28

	Student	Module	Deliverable	Weight	Marl
0	Α	Maths	Practical	40	60
1	A	Maths	Exam	60	80
2	Α	Programming	Project 1	10	50
3	Α	Programming	1	20	-60
		Networking	L df.	sort	_va
5	В	Maths	Praemeur	-10	00
6	В	Maths	Exam	60	80
7	В	Programming	Project 1	10	50
8		Programming	Project 2	30	60
9	В	Programming	Project 3	60	75
10	В	Networking	Project	100	80
11	С	Maths	Exam	60	80
12	С	Programming	Project 1	10	50
13	С	Programming	Project 2	30	60
14	С	Programming	Project 3	60	75

Sorting

A pandas dataframe has two sorting operations:

- sort_index() orders rows based on current index.
- sort_values(COLUMNS) orders rows based on single column or list of columns.

Two important modifications:

- By default, the sort order is in ascending. Set parameter ascending=False to reverse this.
- By default, a new dataframe is returned with desired sort order, set parameter inplace=True to update current dataframe instead.

Studen	t Module	Deliverable	Weight	Mark			Student	Module	Deliverable	Weight	Mar
0 A	Maths	Practical	40	60		1	A	Maths	Exam	60	80
1 A	Maths	Exam	60	80		6	В	Maths	Exam	60	80
2 A	Programming	Project 1	10	50		11	C	Maths	Exam	60	80
3 A	Programming		20	-		Δ.	-		etical	40	60
4 A	Networking	L df.	sort	_va	alues([' <mark>Module</mark> ','Deliverable'], inplace=Tı	ru	.e)		ctical	40	60
5 B	Maths	Praerieur	-10	00				тестотки в	wao Work	100	80
6 B	Maths	Exam	60	80		15	C	Networking	Lab Work	100	80
7 B	Programming	Project 1	10	50	/	10	В	Networking	Project	100	80
8 B	Programming	Project 2	30	60		2	A	Programming	Project 1	10	50
9 B	Programming	Project 3	60	75		7	В	Programming	Project 1	10	50
10 B	Networking	Project	100	80		12	C	Programming	Project 1	10	50
11 C	Maths	Exam	60	80		3	A	Programming	Project 2	30	60
12 C	Programming	Project 1	10	50		8	В	Programming	Project 2	30	60
13 C	Programming	Project 2	30	60		13	C	Programming	Project 2	30	60
14 C	Programming	Project 3	60	75		9	В	Programming	Project 3	60	75

21 of 28

Part V

Defining New Columns

Defining new columns — row-wise operation

We want to compute the weighted mark for each module for each student. Two steps:

- Create column, W_Mark, to store the weighted mark for each deliverable. This is a row by row calculation only need data in current row to compute the result.
- Create column, M_Mark, to store the module mark for each student. This is a group calculation need all rows for that student and module to compute the result.

	Student	Module	Deliverable	Weight	Mark		
1	A	Maths	Exam	60	80	•	
6	В	Maths	Exam	60	80		
11	С	Maths	Exam	60	80		
0	A	Maths	Practical	40	60		
5 4	В	Maths	Practical	40	60		
4	A	Networking	Lab Work	100	-10	CCIN Maralala de Mariala de Marala d	/ 10
15	С	Networking	Lab Work	100	ai	f[<mark>'W_Mark</mark> '] = df.Weight * df.Mark	/ TO
10	В	Networking	Project	100	80		
	A	Programming	Project 1	10	50		
7	В	Programming	Project 1	10	50		
12 3	С	Programming	Project 1	10	50		
3	A	Programming	Project 2	30	60	Need to use dist notation (not	et.
8		Programming	Project 2	30	60	Need to use dict notation (not	
13	C	Programming	Project 2	30	60	dot notation) when defining a)
9	В	Programming	Project 3	60	75	new column.	,
14	C	Programming	Project 3	60	75	new commin.	
							-

Defining new columns — row-wise operation

We want to compute the weighted mark for each module for each student. Two steps:

- Create column, W_Mark, to store the weighted mark for each deliverable. This is a row by row calculation only need data in current row to compute the result.
- Create column, M_Mark, to store the module mark for each student. This is a group calculation need all rows for that student and module to compute the result.

Studer	t Module	Deliverable	Weight	Mark										Studer	t Module	Deliverable	Weight	Mark	W_Mark
1 A	Maths	Exam	60	80										1 A	Maths	Exam	60	80	48
6 B	Maths	Exam	60	80										6 B	Maths	Exam	60	80	48
11 C	Maths	Exam	60	80										11 C	Maths	Exam	60	80	48
0 A	Maths	Practical	40	60										0 A	Maths	Practical	40	60	24
5 B	Maths	Practical	40	60										5 B	Maths	Practical	40	60	24
4 A	Networking	Lab Work	100	3.0	CEIra	M1	-17	-1.C	7/7 - d1	l. 4.	JC M	1- //	100		Networking	Lab Work	100	80	80
15 C	Networking	Lab Work	100	ai	: [.M_]	mari	<u>c</u> .]	= ar	.weigi	nt *	ai.M	ark //	100		Networking	Lab Work	100	80	80
10 B	Networking	Project	100	80	_								_	10 B	Networking	Project	100	80	80
2 A	Programming	Project 1	10	50										2 A	Programming	Project 1	10	50	5
7 B	Programming	Project 1	10	50										7 B	Programming	Project 1	10	50	5
12 C	Programming	Project 1	10	50										12 C	Programming	Project 1	10	50	5
3 A	Programming	Project 2	30	60			T	14-		4	. 4 . 4	(3 A	Programming	Project 2	30	60	18
8 B	Programming	Project 2	30	60		1	Nee	ato	use dic	no no	otation	(not		8 B	Programming	Project 2	30	60	18
13 C	Programming	Project 2	30	60) (dot	notat	ion) w	hen	definii	ng a		13 C	Programming	Project 2	30	60	18
9 B	Programming	Project 3	60	75		(colu						9 B	Programming	Project 3	60	75	45
14 C	Programming	Project 3	60	75		<u></u>	iew	Colu	~~~	~~				14 C	Programming	Project 3	60	75	45

- Create column, W_Mark, to store the weighted mark for each deliverable. This is a row by row calculation only need data in current row to compute the result.
- Create column, M_Mark, to store the module mark for each student. This is a group calculation need all rows for that student and module to compute the result.

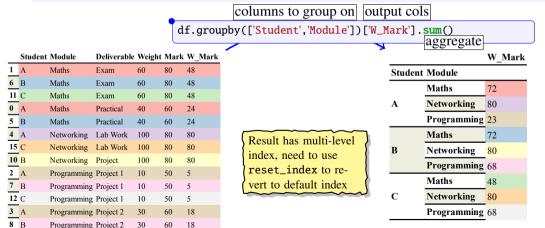
columns to group on output cols

df.groupby(['Student','Module'])['W_Mark'].sum()
aggregate

	Student	Module	Deliverable	Weight	Mark	W_Mark
1	A	Maths	Exam	60	80	48
6	В	Maths	Exam	60	80	48
11	C	Maths	Exam	60	80	48
0	A	Maths	Practical	40	60	24
5	В	Maths	Practical	40	60	24
4	A	Networking	Lab Work	100	80	80
15	С	Networking	Lab Work	100	80	80
10	В	Networking	Project	100	80	80
2	A	Programming	Project 1	10	50	5
7	В	Programming	Project 1	10	50	5
12	С	Programming	Project 1	10	50	5
3	A	Programming	Project 2	30	60	18
8	В	Programming	Project 2	30	60	18

Result has multi-level index, need to use reset_index to revert to default index

- Create column, W_Mark, to store the weighted mark for each deliverable. This is a row by row calculation only need data in current row to compute the result.
- Create column, M_Mark, to store the module mark for each student. This is a group calculation need all rows for that student and module to compute the result.



columns to group on

output cols

df.groupby(['Student','Module'])[['W_Mark']].sum().reset_index()

aggregate

	Student	Module	Deliverable	Weight	Mark	W_Mark
1	A	Maths	Exam	60	80	48
6	В	Maths	Exam	60	80	48
11	С	Maths	Exam	60	80	48
0	A	Maths	Practical	40	60	24
5	В	Maths	Practical	40	60	24
4	A	Networking	Lab Work	100	80	80
15	С	Networking	Lab Work	100	80	80
10	В	Networking	Project	100	80	80
2	A	Programming	Project 1	10	50	5
7	В	Programming	Project 1	10	50	5
12	С	Programming	Project 1	10	50	5
3	A	Programming	Project 2	30	60	18
8	В	Programming	Project 2	30	60	18
13	C	Programming	Project 2	30	60	18
9	В	Programming	Project 3	60	75	45
14	C	Programming	Project 3	60	75	45

columns to group on

output cols

df.groupby(['Student','Module'])[['W_Mark']].sum().reset_index()

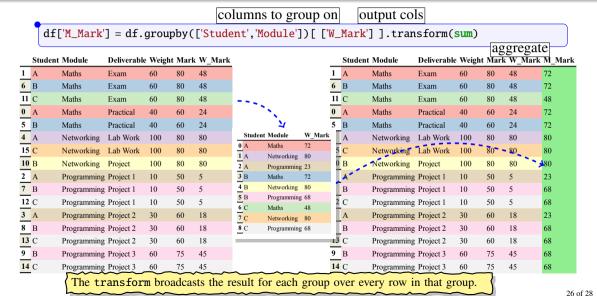
	Student	Module	Deliverable	Weight	Mark	W_Marl
1	A	Maths	Exam	60	80	48
6	В	Maths	Exam	60	80	48
11	C	Maths	Exam	60	80	48
0	A	Maths	Practical	40	60	24
5	В	Maths	Practical	40	60	24
4	A	Networking	Lab Work	100	80	80
15	C	Networking	Lab Work	100	80	80
10	В	Networking	Project	100	80	80
2	A	Programming	Project 1	10	50	5
7	В	Programming	Project 1	10	50	5
12	C	Programming	Project 1	10	50	5
3	A	Programming	Project 2	30	60	18
8	В	Programming	Project 2	30	60	18
13	C	Programming	Project 2	30	60	18
9	В	Programming	Project 3	60	75	45
14	C	Programming	Project 3	60	75	45

This is the required result and we can save this to a new dataframe. However, we often want to put this into to out original dataframe as an extra column. Only problem we have different rows so can't just assign to a new column — need to use transform function.

aggregate

	Student	Module	W_Marl
0	A	Maths	72
1	A	Networking	80
2	A	Programming	23
		Maths	72
		Networking	80
		Programming	68
6	C	Maths	48
7	C	Networking	80
8	C	Programming	68
	1 2 3 4 5 6 7	Student 0 A 1 A 2 A 3 B 4 B 5 B 6 C 7 C 8 C	1 A Networking 2 A Programming 3 B Maths 4 B Networking 5 B Programming 6 C Maths 7 C Networking

Defining new columns — group aggregate result



Part VI

Review Exercises

Review Exercises

Generate the following reports:

- Number of deliverables by each student.
- 2 List and rank deliverables by grade.
- Top 2 deliverables (by grade).
- **1** Top 2 module (by average grade).
- **5** Top 2 modules (by minimum grade).
- Modules (by minimum grade).

Harder exercises (new functions)

1 List which students missed which deliverables.

(value_counts, or groupby and count)

(sort_values, rank)

(pivot, melt)