



Bài giảng môn học:

Kỹ nghệ tri thức và học máy

Chương 2: Thu thập và chuẩn bị dữ liệu (phần 03)

Đặng Văn Nam

dangvannam@humg.edu.vn

1. Xóa cột/hàng trong Dataframe
2. Xử lý hàng trùng lặp (Duplicate rows)
3. Sắp xếp trong DataFrame (Sort)
4. Nhóm các hàng trong DataFrame dựa vào giá trị
5. Áp dụng hàm cho các phần tử trong DataFrame (Apply)
6. Ghép nối các DataFrame (Concatenating)
7. Trộn các DataFrame (Merging)
8. Ví dụ xử lý dữ liệu

1. Xóa cột/hàng trong DataFrame

1.1 Xóa cột trong một DataFrame

- `df.drop([column_name], axis=1|'columns', inplace=True|False)`: Để xóa 1 cột hoặc nhiều cột trong một DataFrame.
- Lưu ý khi sử dụng tham số `inplace = True` → Áp dụng thay đổi cho chính DataFrame hiện tại

`df.drop(['Q', 'R'], axis=1)`

columns

	P	Q	R	S
0	0	1	2	3
1	4	5	6	7
2	8	9	10	11

after drop
new dataframe

	P	S
0	0	3
1	4	7
2	8	11

to be dropped
column Q and R

[axis 1 represents columns]

© w3resource.com

```
1 #Xử dụng .drop(axis=1/columns) để xóa cột
2 #Xóa một số cột trong df_loan
3 df_loan1 = df_loan.drop(['annual_inc',
4                           'dti', 'delinq_2yrs',
5                           'revol_util',
6                           'longest_credit_length',
7                           'verification_status'], axis=1)
8 df_loan1.info()
```

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

```
RangeIndex: 163987 entries, 0 to 163986
```

```
Data columns (total 9 columns):
```

#	Column	Non-Null Count	Dtype
0	loan_amnt	163987 non-null	int64
1	term	163987 non-null	object
2	int_rate	163987 non-null	float64
3	emp_length	158183 non-null	float64
4	home_ownership	163987 non-null	object
5	purpose	163987 non-null	object
6	addr_state	163987 non-null	object
7	total_acc	163958 non-null	float64
8	bad_loan	163987 non-null	int64

```
dtypes: float64(3), int64(2), object(4)
```

```
memory usage: 11.3+ MB
```

1.1 Xóa cột trong một DataFrame

- **`df.drop(df.columns[index], axis=1|columns)`**: Để xóa 1 cột hoặc nhiều cột trong một DataFrame trong trường hợp DataFrame không có tên cột, sử dụng chỉ số cột.

```
1 #Xóa cột trong một DataFrame sử dụng chỉ số cột
2 df_loan2 = df_loan.drop(df_loan.columns[[5,8,9,10,13,14]],
3                          axis='columns')
4 df_loan2.info()
```

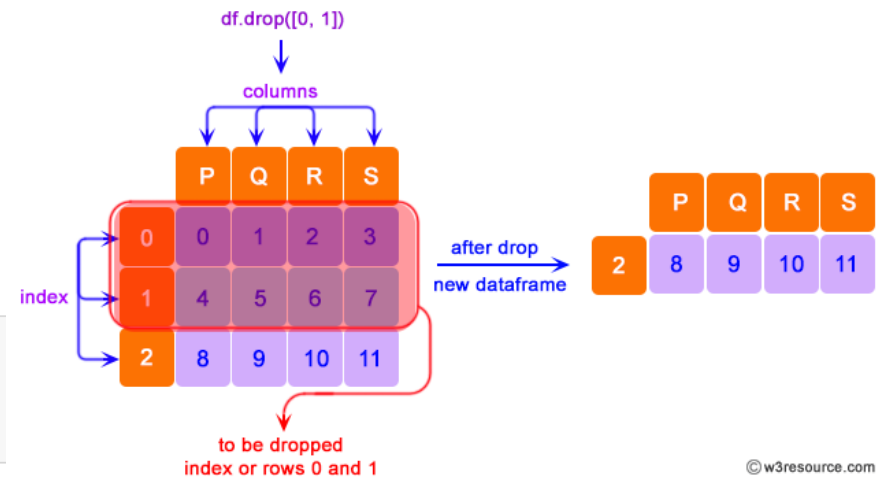
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 163987 entries, 0 to 163986
Data columns (total 9 columns):
#   Column                Non-Null Count  Dtype
---  -
0   loan_amnt             163987 non-null  int64
1   term                  163987 non-null  object
2   int_rate              163987 non-null  float64
3   emp_length            158183 non-null  float64
4   home_ownership        163987 non-null  object
5   purpose               163987 non-null  object
6   addr_state            163987 non-null  object
7   total_acc             163958 non-null  float64
8   bad_loan              163987 non-null  int64
dtypes: float64(3), int64(2), object(4)
memory usage: 11.3+ MB
```

1.2 Xóa hàng trong một DataFrame

Xóa hàng theo index:

- **df.drop([index rows], axis=0):** Để xóa 1 hàng hoặc nhiều hàng trong một DataFrame theo index của hàng.
- Tham số axis = 0 (Default)

```
1 #Xóa hàng trong một DataFrame
2 #Xóa hàng có index: 3,9
3 df_loan2.drop([3,9],inplace=True)
4 df_loan2.head(10)
```



© w3resource.com

	loan_amnt	term	int_rate	emp_length	home_ownership	purpose	addr_state	total_acc	bad_loan
0	5000	36 months	10.65	10.0	RENT	credit_card	AZ	9.0	0
1	2500	60 months	15.27	0.0	RENT	car	GA	4.0	1
2	2400	36 months	15.96	10.0	RENT	small_business	IL	10.0	0
4	5000	36 months	7.90	3.0	RENT	wedding	AZ	12.0	0
5	3000	36 months	18.64	9.0	RENT	car	CA	4.0	0
6	5600	60 months	21.28	4.0	OWN	small_business	CA	13.0	1
7	5375	60 months	12.69	0.0	RENT	other	TX	3.0	1
8	6500	60 months	14.65	5.0	OWN	debt_consolidation	AZ	23.0	0
10	9000	36 months	13.49	0.0	RENT	debt_consolidation	VA	9.0	1
11	3000	36 months	9.91	3.0	RENT	credit_card	IL	11.0	0

1.2 Xóa hàng trong một DataFrame

Xóa hàng theo điều kiện (filter data):

```
1 #Loại bỏ tất cả các dòng dữ liệu có addr_state = CA
2 df_loan3 = df_loan1[df_loan1.addr_state!='CA']
3 df_loan3
```

	loan_amnt	term	int_rate	emp_length	home_ownership	purpose	addr_state	total_acc	bad_loan
0	5000	36 months	10.65	10.0	RENT	credit_card	AZ	9.0	0
1	2500	60 months	15.27	0.0	RENT	car	GA	4.0	1
2	2400	36 months	15.96	10.0	RENT	small_business	IL	10.0	0
4	5000	36 months	7.90	3.0	RENT	wedding	AZ	12.0	0
7	5375	60 months	12.69	0.0	RENT	other	TX	3.0	1
...
163982	15000	60 months	12.39	3.0	MORTGAGE	credit_card	OK	34.0	0
163983	20000	36 months	14.99	10.0	OWN	home_improvement	VA	18.0	0
163984	12825	36 months	17.14	6.0	MORTGAGE	debt_consolidation	TX	24.0	0
163985	27650	60 months	21.99	0.0	RENT	credit_card	NY	20.0	0
163986	17000	60 months	15.99	10.0	MORTGAGE	debt_consolidation	PA	28.0	0

135285 rows × 9 columns

2. Xử lý hàng trùng lặp

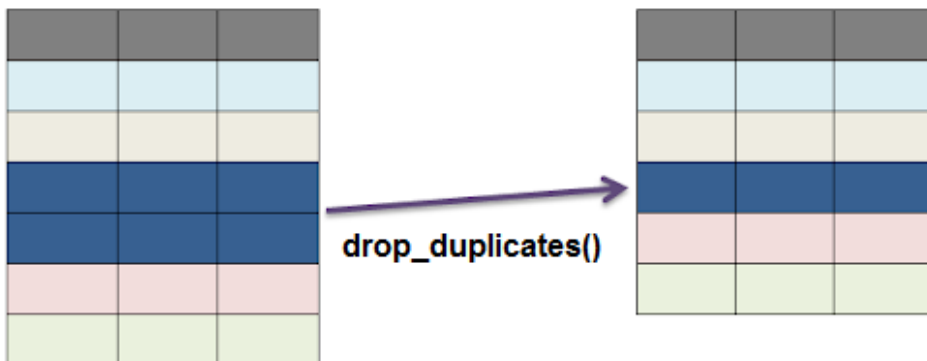
2. Xử lý các hàng trùng lặp

Pandas drop_duplicates() Function Syntax

```
drop_duplicates(self, subset=None, keep="first", inplace=False)
```

- **subset:** Subset takes a column or list of column label for identifying duplicate rows. By default, all the columns are used to find the duplicate rows.
- **keep:** allowed values are {'first', 'last', False}, default 'first'. If 'first', duplicate rows except the first one is deleted. If 'last', duplicate rows except the last one is deleted. If False, all the duplicate rows are deleted.
- **inplace:** if True, the source DataFrame itself is changed. By default, source DataFrame remains unchanged and a new DataFrame instance is returned.

Drop Duplicate Pandas



- `df.duplicates()`: để tìm kiếm các hàng trùng lặp

2. Xử lý các hàng trùng lặp

- `df.drop_duplicates()`: với các tham số mặc định sẽ thực hiện
 - Xóa hết các hàng dữ liệu trùng lặp nhau ở tất cả các cột
 - Giữ lại hàng đầu tiên trùng lặp

Name Age Score

0	Alisa	26	85.0	3
1	raghu	23	31.0	2
2	jodha	23	55.0	1
3	jodha	23	55.0	
4	raghu	23	31.0	2
5	Cathrine	24	77.0	
6	Alisa	26	85.0	3
7	Bobby	24	63.0	
8	Bobby	22	42.0	
9	Alisa	26	85.0	3
10	raghu	23	31.0	2
11	Cathrine	24	NaN	

```

1 #Trường hợp 1:
2 #Sử dụng df.drop_duplicates() với các tham số mặc định
3 #--> giữ lại hàng trùng lặp đầu tiên
4 df1 = df.drop_duplicates()
5 df1

```

Name Age Score

0	Alisa	26	85.0
1	raghu	23	31.0
2	jodha	23	55.0
5	Cathrine	24	77.0
7	Bobby	24	63.0
8	Bobby	22	42.0
11	Cathrine	24	NaN

2. Xử lý các hàng trùng lặp

- `df.drop_duplicates(keep='last')`:
 - Xóa hết các hàng dữ liệu trùng lặp nhau ở tất cả các cột
 - Giữ lại hàng trùng lặp cuối cùng

Name Age Score

0	Alisa	26	85.0	3
1	raghu	23	31.0	2
2	jodha	23	55.0	1
3	jodha	23	55.0	1
4	raghu	23	31.0	2
5	Cathrine	24	77.0	
6	Alisa	26	85.0	3
7	Bobby	24	63.0	
8	Bobby	22	42.0	
9	Alisa	26	85.0	3
10	raghu	23	31.0	2
11	Cathrine	24	NaN	

```

1 #Trường hợp 2:
2 #Sử dụng df.drop_duplicates()
3 #với các tham số keep='last'
4 #Giữ lại các hàng trùng lặp cuối cùng
5 df2=df.drop_duplicates(keep='last')
6 df2

```

Name Age Score

3	jodha	23	55.0
5	Cathrine	24	77.0
7	Bobby	24	63.0
8	Bobby	22	42.0
9	Alisa	26	85.0
10	raghu	23	31.0
11	Cathrine	24	NaN

2. Xử lý các hàng trùng lặp

- `df.drop_duplicates(keep=False)`:
 - Xóa hết các hàng dữ liệu trùng lặp nhau ở tất cả các cột, chỉ giữ lại các hàng dữ liệu không trùng lặp

Name Age Score

0	Alisa	26	85.0	3
1	raghu	23	31.0	2
2	jodha	23	55.0	1
3	jodha	23	55.0	1
4	raghu	23	31.0	2
5	Cathrine	24	77.0	
6	Alisa	26	85.0	3
7	Bobby	24	63.0	
8	Bobby	22	42.0	
9	Alisa	26	85.0	3
10	raghu	23	31.0	2
11	Cathrine	24	NaN	

```

1 #Trường hợp 3:
2 #Sử dụng df.drop_duplicates()
3 #với các tham số keep=False
4 #Xóa hết các hàng trùng lặp khỏi df
5 df3=df.drop_duplicates(keep=False)
6 df3

```

Name Age Score

5	Cathrine	24	77.0
7	Bobby	24	63.0
8	Bobby	22	42.0
11	Cathrine	24	NaN

2. Xử lý các hàng trùng lặp

- `df.drop_duplicates([name columns], keep='first' | 'last' | False)`:
 - Xóa các hàng dữ liệu trùng lặp nhau ở các cột được chỉ định

01

```
1 #Trường hợp 4:
2 #Sử dụng df.drop_duplicates()
3 #Loại bỏ các hàng trùng nhau theo cột Name
4 df4=df.drop_duplicates(['Name'],keep='first')
5 df4
```

Name Age Score

0 Alisa 26 85.0

1 raghu 23 31.0

2 jodha 23 55.0

3 jodha 23 55.0

4 raghu 23 31.0

5 Cathrine 24 77.0

6 Alisa 26 85.0

7 Bobby 24 63.0

8 Bobby 22 42.0

9 Alisa 26 85.0

10 raghu 23 31.0

11 Cathrine 24 NaN

Name Age Score

0 Alisa 26 85.0

1 raghu 23 31.0

2 jodha 23 55.0

5 Cathrine 24 77.0

7 Bobby 24 63.0

```
1 #Trường hợp 5:
2 #Sử dụng df.drop_duplicates()
3 #Loại bỏ các hàng trùng nhau theo cột Name, Age
4 df5=df.drop_duplicates(['Name','Age'],
5                         keep='first')
6 df5
```

Name Age Score

0 Alisa 26 85.0

1 raghu 23 31.0

2 jodha 23 55.0

5 Cathrine 24 77.0

7 Bobby 24 63.0

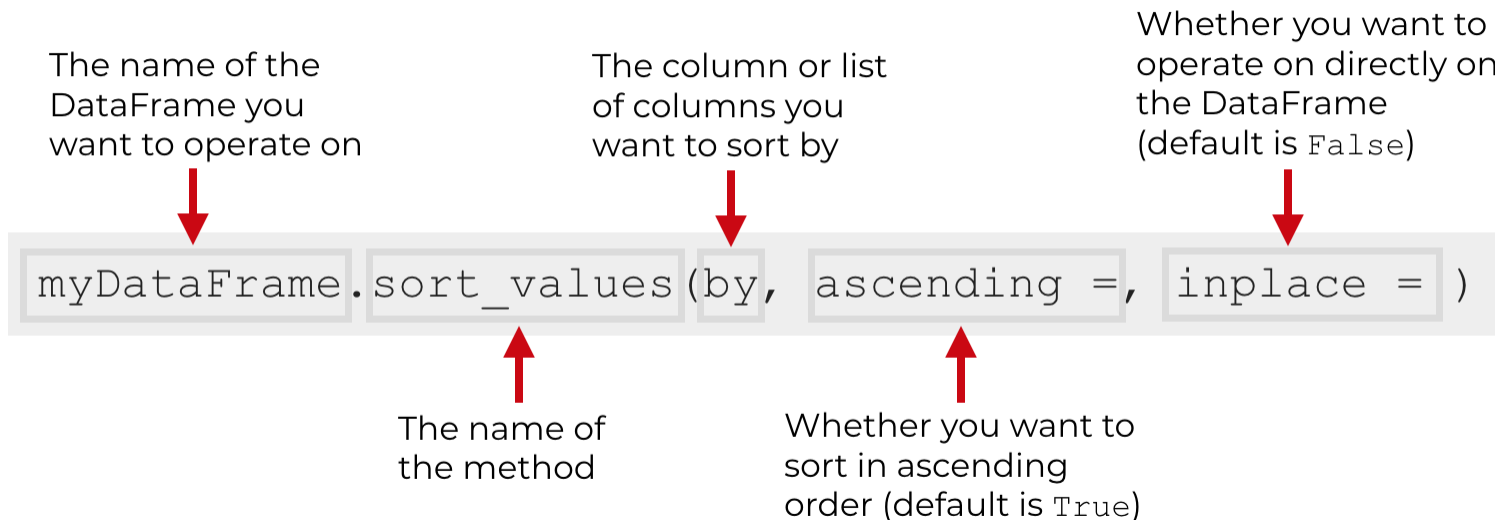
8 Bobby 22 42.0

02

3. Sắp xếp dữ liệu trong DataFrame

3. Sắp xếp dữ liệu trong DataFrame

- **df.sort_values()**: sắp xếp dữ liệu trong DataFrame theo giá trị của các cột, tăng dần (**ascending = True**)-default hoặc giảm dần (**ascending=False**)
- Lưu ý khi sử dụng tham số **inplace = True**



- **df.sort_index()**: sắp xếp dữ liệu trong DataFrame theo index

3. Sắp xếp dữ liệu trong DataFrame

- Ví dụ:

	Name	Age	Score
0	Alisa	26	89
1	Bobby	27	87
2	Cathrine	25	67
3	Madonna	24	55
4	Rocky	31	47
5	Sebastian	27	72
6	Jaquiline	25	76
7	Rahul	33	79
8	David	42	44
9	Andrew	32	92
10	Ajay	51	99
11	Teresa	47	69
12	Madonna	38	73

```
1 #Trường hợp 1:
2 #Sắp xếp dữ liệu Dataframe theo cột Score
3 #Mặc định là sắp xếp tăng dần
4 df.sort_values(by='Name')
```

	Name	Age	Score
10	Ajay	51	99
0	Alisa	26	89
9	Andrew	32	92
1	Bobby	27	87
2	Cathrine	25	67
8	David	42	44
6	Jaquiline	25	76
3	Madonna	24	55
12	Madonna	38	73

3. Sắp xếp dữ liệu trong DataFrame

- Ví dụ:

	Name	Age	Score
0	Alisa	26	89
1	Bobby	27	87
2	Cathrine	25	67
3	Madonna	24	55
4	Rocky	31	47
5	Sebastian	27	72
6	Jaquiline	25	76
7	Rahul	33	79
8	David	42	44
9	Andrew	32	92
10	Ajay	51	99
11	Teresa	47	69
12	Madonna	38	73

```
1 #Trường hợp 2:  
2 #Sắp xếp dữ liệu Dataframe theo cột Score  
3 #Giá trị giảm dần  
4 df.sort_values(by='Score',ascending=False)
```

	Name	Age	Score
10	Ajay	51	99
9	Andrew	32	92
0	Alisa	26	89
1	Bobby	27	87
7	Rahul	33	79
6	Jaquiline	25	76
12	Madonna	38	73
5	Sebastian	27	72

3. Sắp xếp dữ liệu trong DataFrame

- Trường hợp sắp xếp nhiều cột, sẽ thực hiện sắp xếp theo thứ tự các cột từ trái sang phải:

	Name	Age	Score
0	Alisa	26	89
1	Bobby	27	87
2	Cathrine	25	67
3	Madonna	24	55
4	Rocky	31	47
5	Sebastian	27	72
6	Jaqueline	25	76
7	Rahul	33	79
8	David	42	44
9	Andrew	32	92
10	Ajay	51	99
11	Teresa	47	69
12	Madonna	38	73

```
1 #Trường hợp 3:  
2 #Sắp xếp dữ liệu Dataframe theo cột Name, Score  
3 #Giá trị tăng dần  
4 df.sort_values(by=['Name', 'Score'])
```

	Name	Age	Score
10	Ajay	51	99
0	Alisa	26	89
9	Andrew	32	92
1	Bobby	27	87
2	Cathrine	25	67
8	David	42	44
6	Jaqueline	25	76
3	Madonna	24	55
12	Madonna	38	73

3. Sắp xếp dữ liệu trong DataFrame

- Ví dụ:

	Name	Age	Score
0	Alisa	26	89
1	Bobby	27	87
2	Cathrine	25	67
3	Madonna	24	55
4	Rocky	31	47
5	Sebastian	27	72
6	Jaquiline	25	76
7	Rahul	33	79
8	David	42	44
9	Andrew	32	92
10	Ajay	51	99
11	Teresa	47	69
12	Madonna	38	73

```
1 #Trường hợp 4:
2 #Sắp xếp dữ liệu Dataframe theo cột Name, Score
3 #Giá trị cột Name tăng dần
4 #Giá trị cột Score giảm dần
5 df.sort_values(by=['Name','Score'],
6               ascending=[True,False])
```

	Name	Age	Score
10	Ajay	51	99
0	Alisa	26	89
9	Andrew	32	92
1	Bobby	27	87
2	Cathrine	25	67
8	David	42	44
6	Jaquiline	25	76
12	Madonna	38	73
3	Madonna	24	55
7	Rahul	33	79
4	Rocky	31	47

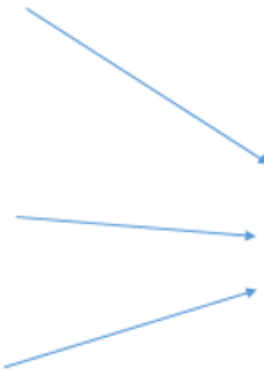
4. Nhóm dữ liệu (groupby)

4.1 Groupby values

- `df.groupby()`: Góm nhóm các giá trị trong một DataFrame bởi các cột được chỉ định.
- Kết hợp với `sum()`, `mean()`, `max()`, `min()` để xác định các thông số theo từng nhóm.

ID	Value
1	50.30
1	123.30
1	132.90
2	50.30
2	123.30
2	132.90
2	88.90
3	50.30
3	123.30

ID	Value
1	306.50
2	395.40
3	173.60



4.1 Groupby values

- Sử dụng phương thức groupby():

	Name	Exam	Subject	Score
0	Alisa	Semester 1	Mathematics	62
1	Bobby	Semester 1	Mathematics	47
2	Cathrine	Semester 1	Mathematics	55
3	Alisa	Semester 1	Science	74
4	Bobby	Semester 1	Science	31
5	Cathrine	Semester 1	Science	77
6	Alisa	Semester 2	Mathematics	85
7	Bobby	Semester 2	Mathematics	63
8	Cathrine	Semester 2	Mathematics	42
9	Alisa	Semester 2	Science	67
10	Bobby	Semester 2	Science	89
11	Cathrine	Semester 2	Science	81

```

1 #Trường hợp 1:
2 #Nhóm theo tên sinh viên (Name)
3 #Thực hiện tính điểm trung bình Score
4 df['Score'].groupby([df['Name']]).mean()

```

```

Name
Alisa      72.00
Bobby      57.50
Cathrine   63.75
Name: Score, dtype: float64

```

```

1 #Trường hợp 2:
2 #Nhóm dữ liệu theo tên sinh viên (Name)
3 #và Bài kiểm tra (Exam)
4 #sau đó thực hiện tính tổng
5 df['Score'].groupby([df['Name'],
6                      df['Exam']]).sum()

```

```

Name      Exam
Alisa      Semester 1    136
           Semester 2    152
Bobby      Semester 1     78
           Semester 2    152
Cathrine    Semester 1    132
           Semester 2    123
Name: Score, dtype: int64

```

01

02

Thực hành 1

Thực hành 1



Yêu cầu 1.1:

- Đọc dữ liệu từ file **Data_Patient.csv** vào biến kiểu dataframe: `df_patient` với cột đầu tiên (`id`) là cột chỉ số (`index_col`). Hiển thị 10 dòng dữ liệu đầu tiên.

Yêu cầu 1.2:

- Xóa cột dữ liệu có tên 'Thalassemia' và áp dụng thay đổi lên chính `df_patient`.

```
<class 'pandas.core.frame.DataFrame'>
Index: 300 entries, Patient_01 to Patient_300
Data columns (total 7 columns):
#   Column                Non-Null Count  Dtype
---  -
0   Age                   300 non-null   int64
1   Gender                300 non-null   object
2   Type                  295 non-null   object
3   Blood_pressure        300 non-null   int64
4   Cholesterol            300 non-null   int64
5   Heartbeat             300 non-null   int64
6   Result                300 non-null   int64
```


Thực hành 1



Yêu cầu 1.3:

- A) Tạo **df_patient1** bằng cách loại bỏ đi 100 dòng dữ liệu đầu tiên từ df_patient.

	Age	Gender	Type	Blood_pressure	Cholesterol	Heartbeat	Result
id							
Patient_101	34	Male	Typical angina	118	182	174	0
Patient_102	57	Female	Asymptomatic	128	303	159	0
Patient_103	71	Female	Non-anginal pain	110	265	130	0
Patient_104	49	Male	Non-anginal pain	120	188	139	1
Patient_105	54	Male	Atypical angina	108	309	156	0

- B) Tạo **df_patient2** bằng cách loại bỏ đi các dòng dữ liệu có thuộc tính Type = 'Non-anginal pain' và nhịp tim > 187 từ df_patient.

	Age	Gender	Type	Blood_pressure	Cholesterol	Heartbeat	Result
id							
Patient_54	44	Male	Atypical angina	130	219	188	0
Patient_112	52	Male	Typical angina	118	186	190	0
Patient_132	29	Male	Atypical angina	130	204	202	0
Patient_186	42	Male	Non-anginal pain	120	240	194	0
Patient_188	54	Male	Atypical angina	192	283	195	1
Patient_225	34	Female	Atypical angina	118	210	192	0

Thực hành 1



Yêu cầu 1.4:

- A) Sắp xếp lại dữ liệu cho `df_patient` theo chiều giảm dần của `index`, áp dụng thay đổi trực tiếp lên DataFrame này.
- B) Tạo **`df_patient3`** bằng cách sắp xếp dữ liệu theo thuộc tính `Gender` tăng dần, Nếu trùng giá trị `Gender` thì sắp xếp theo thuộc tính `Age` giảm dần.

```
1 df_patient3.iloc[90:100]
```

	Age	Gender	Type	Blood_pressure	Cholesterol	Heartbeat	Result
id							
Patient_277	39	Female	Non-anginal pain	138	220	152	0
Patient_222	39	Female	Non-anginal pain	94	199	179	0
Patient_210	37	Female	Non-anginal pain	120	215	170	0
Patient_117	35	Female	Asymptomatic	138	183	182	0
Patient_225	34	Female	Atypical angina	118	210	192	0
Patient_161	77	Male	Asymptomatic	125	304	162	1
Patient_258	70	Male	Atypical angina	156	245	143	0
Patient_170	70	Male	Non-anginal pain	160	269	112	1
Patient_155	70	Male	Asymptomatic	130	322	109	1
Patient_136	70	Male	Asymptomatic	145	174	125	1

Thực hành 1



Yêu cầu 1.5:

- A) Nhóm bệnh nhân theo thuộc tính Gender và tìm tuổi **lớn nhất, nhỏ nhất, trung bình** của bệnh nhân theo giới tính.
- B) Nhóm bệnh nhân theo thuộc tính Gender và Type và tìm tuổi **lớn nhất, nhỏ nhất, trung bình** của bệnh nhân theo giới tính và loại đau ngực.

1) Thống kê tuổi cao nhất theo giới tính:

```
Gender
Female    76
Male      77
Name: Age, dtype: int64
```

2) Thống kê tuổi thấp nhất theo giới tính:

```
Gender
Female    34
Male      29
Name: Age, dtype: int64
```

3) Thống kê tuổi trung bình theo giới tính:

```
Gender
Female    55.736842
Male      53.912195
Name: Age, dtype: float64
```

1) Thống kê tuổi cao nhất theo giới tính và loại:

```
Gender  Type
Female  Asymptomatic    71
        Atypical angina  74
        Non-anginal pain  76
        Typical angina   69
Male    Asymptomatic    77
        Atypical angina  70
        Non-anginal pain  70
        Typical angina   69
Name: Age, dtype: int64
```

Yêu cầu 1.6:

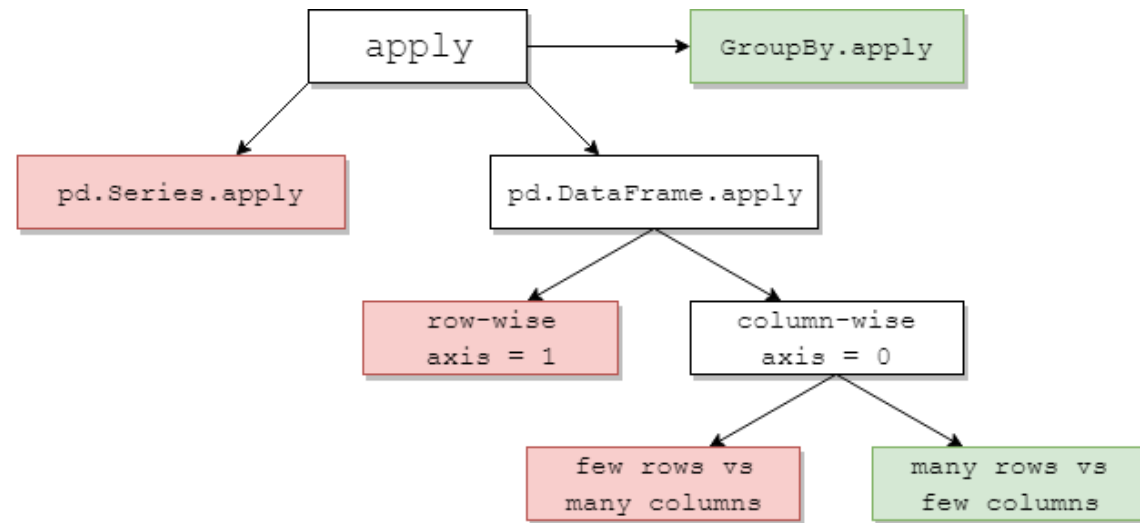
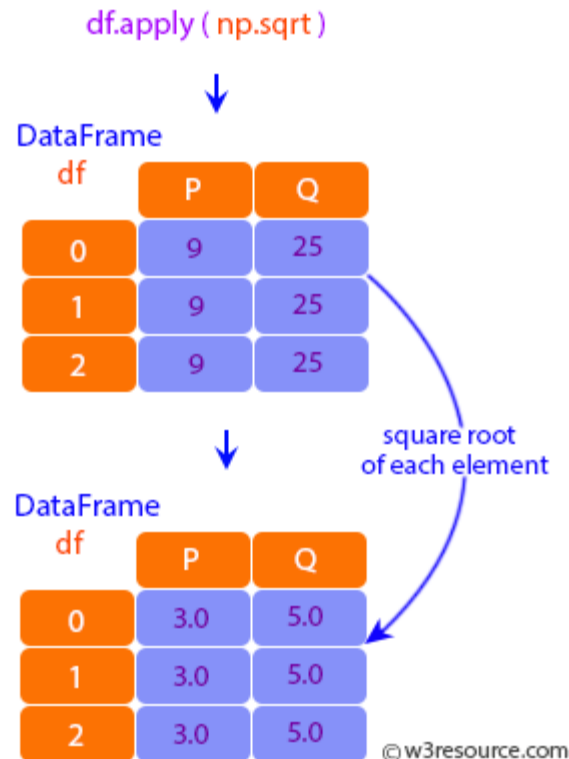
- Sử dụng `df_patient.reset_index(inplace=True)` để bỏ cột index. Sau đó thực hiện xóa các bệnh nhân có giá trị trong cột id trùng nhau, giữ lại bệnh nhân có id trùng nhau đầu tiên, áp dụng cho chính dataframe hiện tại.

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 292 entries, 0 to 299
Data columns (total 8 columns):
#   Column                Non-Null Count  Dtype
---  -
0   id                    292 non-null   object
1   Age                   292 non-null   int64
2   Gender                292 non-null   object
3   Type                  287 non-null   object
4   Blood_pressure        292 non-null   int64
5   Cholesterol           292 non-null   int64
6   Heartbeat             292 non-null   int64
7   Result                292 non-null   int64
dtypes: int64(5), object(3)
```

5. `apply(function)`

5 .apply(func)

- df.apply(func): Thực hiện thao tác func áp dụng cho từng cột riêng lẻ trong DataFrame, hoặc cho nhiều cột



5 .apply(func)

- Áp dụng hàm cho các phần tử trong một cột dữ liệu của DataFrame: Viết hoa các giá trị trong cột Name (3 Cách thực hiện)

	Name	Score_Math	Score_Science
0	william	66	89
1	Mason	57	87
2	ella	75	67
3	jackson	44	55
4	lincoln	31	47
5	aubrey	67	72
6	Hudson	85	76
7	christian	33	79
8	Sawyer	42	44
9	silas	62	92
10	Bennett	51	93
11	kingston	47	69

```

1 #Thực hiện: Viết hoa tên học sinh
2 #Cách 1:
3 def upcase(x):
4     return x.upper()
5
6 df['Name'] = df['Name'].apply(upcase)
7 df

```

01

	Name	Score_Math	Score_Science
0	WILLIAM	66	89
1	MASON	57	87
2	ELLA	75	67

```

1 #Cách 2:
2 df['Name'] = df['Name'].apply(lambda x:x.upper())
3 df

```

02

	Name	Score_Math	Score_Science
0	WILLIAM	66	89
1	MASON	57	87
2	ELLA	75	67

```

1 #Cách 3:
2 df['Name'] = df['Name'].str.upper()
3 df

```

03

	Name	Score_Math	Score_Science
0	WILLIAM	66	89
1	MASON	57	87
2	ELLA	75	67

5 .apply(func)

- Áp dụng hàm cho các phần tử trong nhiều cột dữ liệu của DataFrame:
- Thực hiện tính điểm cho từng học sinh theo công thức:
 - **Point = (Score_Math*2 + Score_Science)/3**

	Name	Score_Math	Score_Science
0	william	66	89
1	Mason	57	87
2	ella	75	67
3	jackson	44	55
4	lincoln	31	47
5	aubrey	67	72
6	Hudson	85	76
7	christian	33	79
8	Sawyer	42	44
9	silas	62	92
10	Bennett	51	93
11	kingston	47	69

```

1 #Điểm trung bình = (score_math*2 + score_science)/3
2 #Viết hàm tính điểm trung bình
3 def mean_point(point1,point2):
4     return round((point1*2+point2)/3,1)

```

```

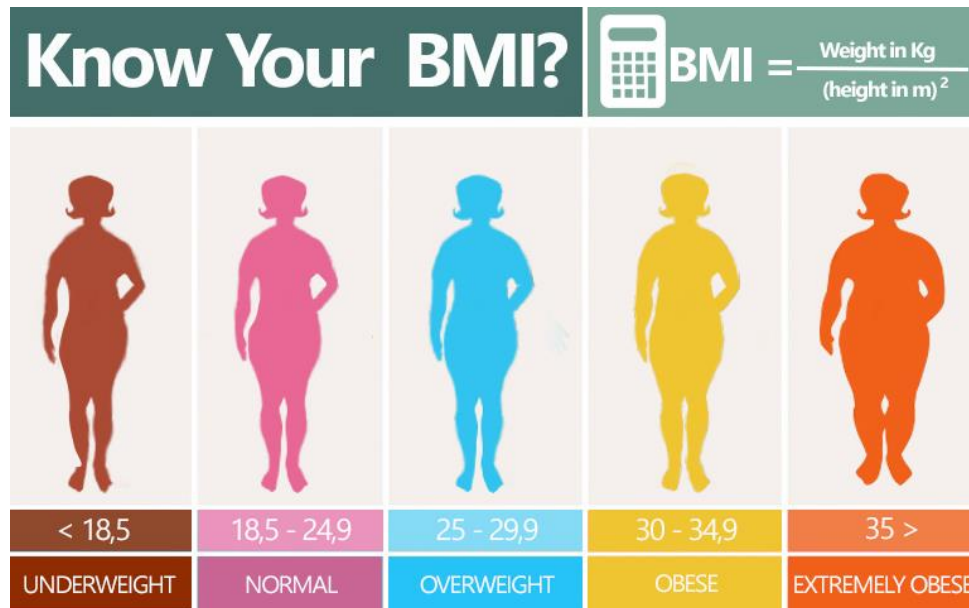
1 #Tạo một cột Point tính điểm của từng học sinh
2 df['Point'] = df.apply(lambda row: mean_point(row['Score_Math'],
3                                               row['Score_Science']),
4                       axis=1)
5 df

```

	Name	Score_Math	Score_Science	Point
0	WILLIAM	66	89	73.7
1	MASON	57	87	67.0
2	ELLA	75	67	72.3
3	JACKSON	44	55	47.7
4	LINCOLN	31	47	36.3

5 .apply(func)

- Áp dụng viết các hàm để tính chỉ số BMI, và phân loại dựa theo chỉ số tính được trên tập dữ liệu csv_Data_BMI.csv

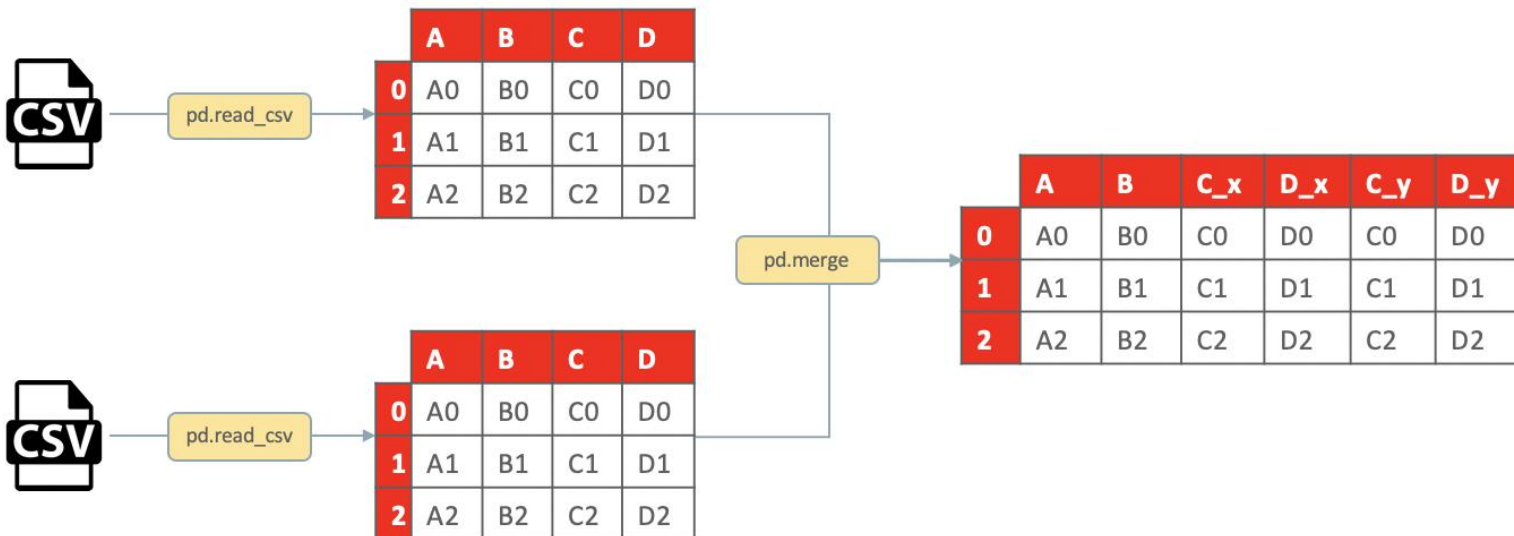


	Gender	Height_cm	Weight_kg	BMI
Personal				
P1	Male	174	96	31.7
P2	Male	189	87	24.4
P3	Female	185	110	32.1
P4	Female	195	104	27.4
P5	Male	149	61	27.5

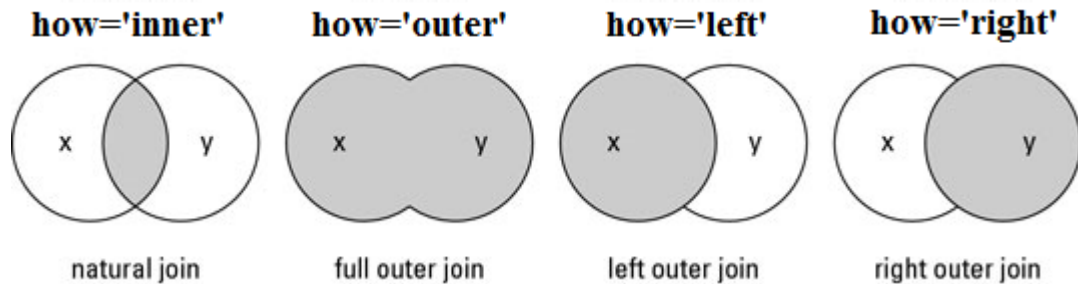
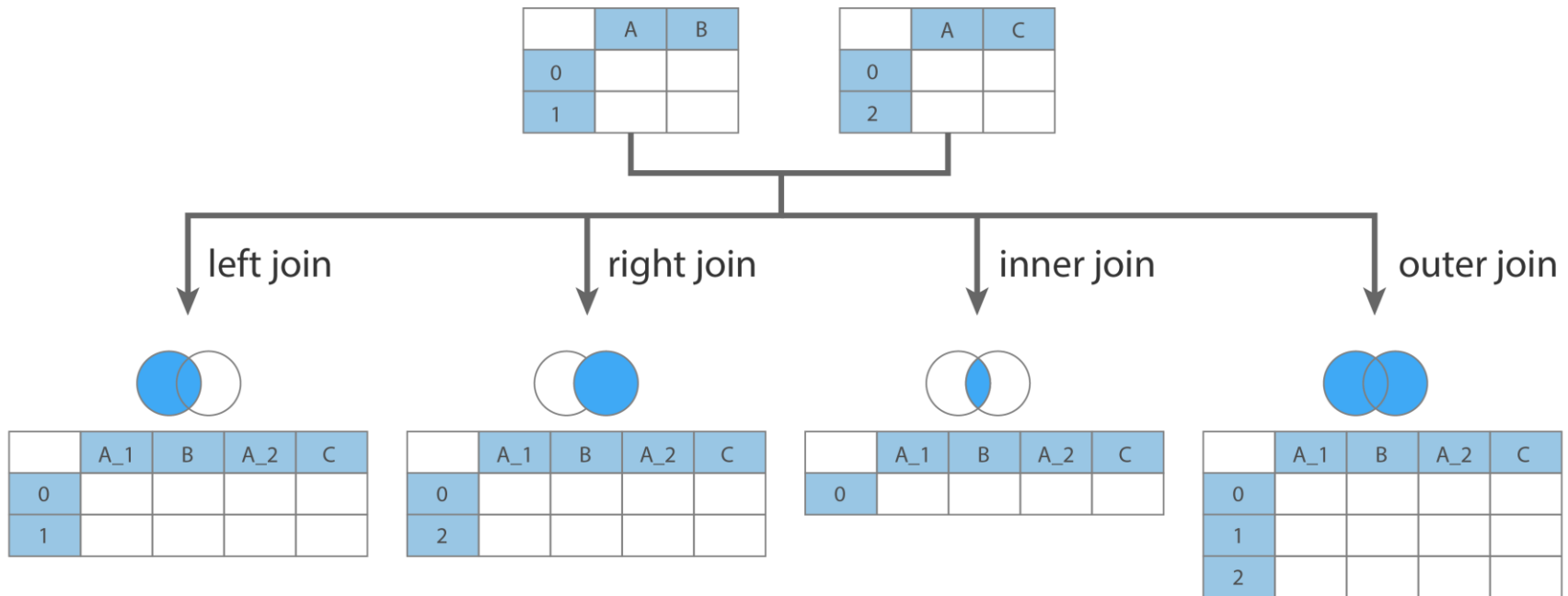
6. Trộn các DataFrame (Merge)

6 Trộn các DataFrame

- `pd.merge(left_df, right_df, on='key', how='left' | 'right' | 'inner' | 'outer')`: Thực hiện trộn 2 DataFrame lại với nhau.
 - Left_df: DataFrame 1
 - Right_df: DataFrame2
 - On: Tên cột dùng để nối dữ liệu giữa 2 DataFrame (tên cột phải có ở trong cả 2 DataFrame 1, 2)
 - How: Cách thức trộn dữ liệu [left, right, outer, inner (default)]



6 Trộn các DataFrame



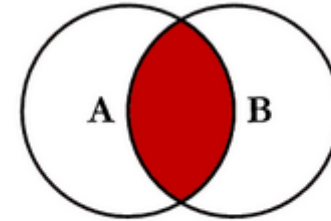
6 Trộn các DataFrame (inner)

	Customer_id	Product
0	1	Oven
1	2	Oven
2	3	Oven
3	4	Television
4	5	Television
5	6	Television

df1

	Customer_id	State
0	2	California
1	4	California
2	6	Texas
3	7	New York
4	8	Indiana

df2



Inner Join

```

1 #Trường hợp 1:
2 #Inner join DataFrame
3 inner_join_df= pd.merge(df1, df2,
4                           on='Customer_id',
5                           how='inner')
6 inner_join_df

```

	Customer_id	Product	State
0	2	Oven	California
1	4	Television	California
2	6	Television	Texas

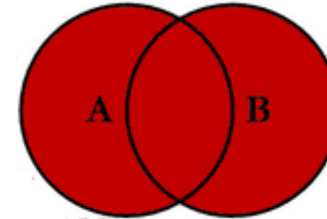
6 Trộn các DataFrame (outer)

	Customer_id	Product
0	1	Oven
1	2	Oven
2	3	Oven
3	4	Television
4	5	Television
5	6	Television

df1

	Customer_id	State
0	2	California
1	4	California
2	6	Texas
3	7	New York
4	8	Indiana

df2



Outer Join

```

1 #Trường hợp 2:
2 #Outer join DataFrame
3 inner_join_df= pd.merge(df1, df2,
4                           on='Customer_id',
5                           how='outer')
6 inner_join_df

```

	Customer_id	Product	State
0	1	Oven	NaN
1	2	Oven	California
2	3	Oven	NaN
3	4	Television	California
4	5	Television	NaN
5	6	Television	Texas
6	7	NaN	New York
7	8	NaN	Indiana

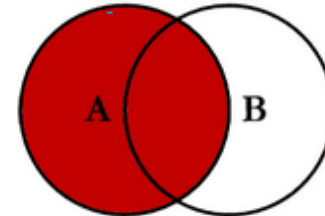
6 Trộn các DataFrame (left)

	Customer_id	Product
0	1	Oven
1	2	Oven
2	3	Oven
3	4	Television
4	5	Television
5	6	Television

df1

	Customer_id	State
0	2	California
1	4	California
2	6	Texas
3	7	New York
4	8	Indiana

df2



Left join

```

1 #Trường hợp 3:
2 #Left join DataFrame
3 inner_join_df= pd.merge(df1, df2,
4                           on='Customer_id',
5                           how='left')
6 inner_join_df

```

	Customer_id	Product	State
0	1	Oven	NaN
1	2	Oven	California
2	3	Oven	NaN
3	4	Television	California
4	5	Television	NaN
5	6	Television	Texas

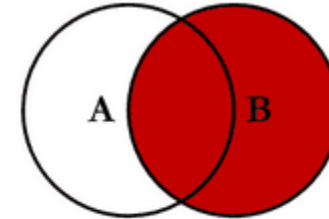
6 Trộn các DataFrame (right)

	Customer_id	Product
0	1	Oven
1	2	Oven
2	3	Oven
3	4	Television
4	5	Television
5	6	Television

df1

	Customer_id	State
0	2	California
1	4	California
2	6	Texas
3	7	New York
4	8	Indiana

df2



Right Join

```

1 #Trường hợp 4:
2 #Right join DataFrame
3 inner_join_df= pd.merge(df1, df2,
4                           on='Customer_id',
5                           how='right')
6 inner_join_df

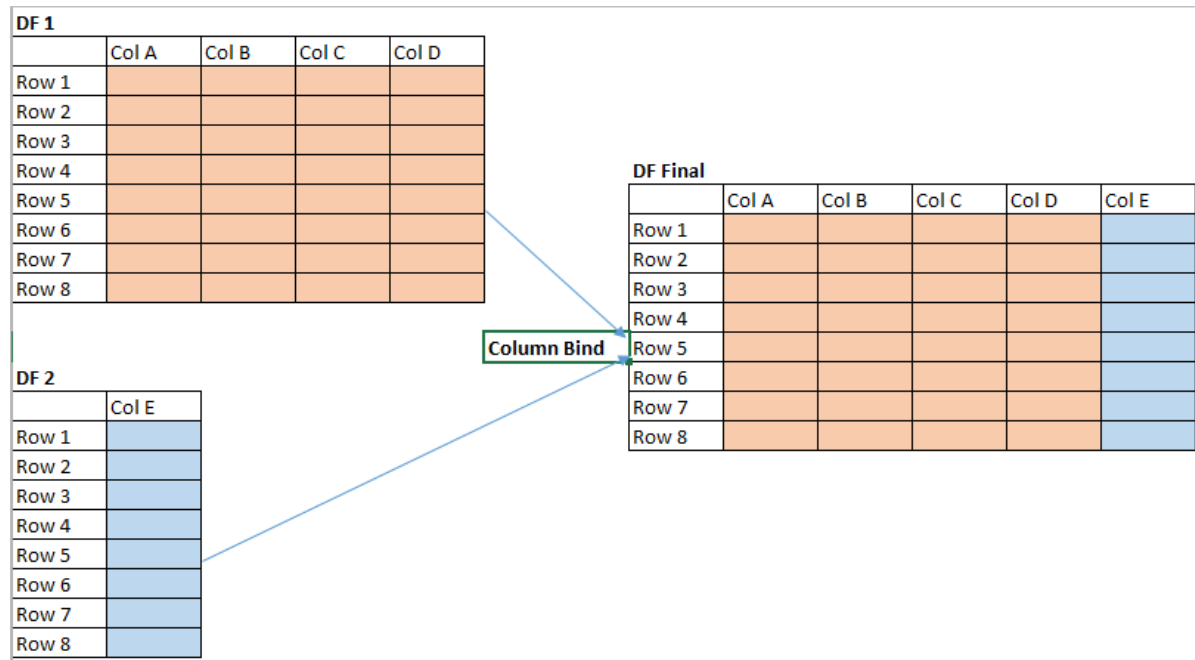
```

	Customer_id	Product	State
0	2	Oven	California
1	4	Television	California
2	6	Television	Texas
3	7	NaN	New York
4	8	NaN	Indiana

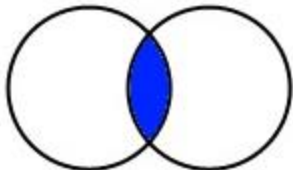
7. Nối các DataFrame (concat, append)

6.1 Nối các DataFrame theo cột

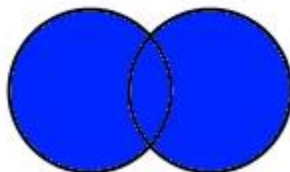
- `pd.concat([df1,df2], axis=1, join='inner'|'outer')`: Thực hiện ghép nối các DataFrame lại với nhau theo cột



INNER JOIN



FULL OUTER JOIN

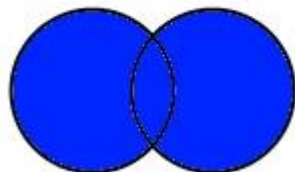


6.1 Nối các DataFrame theo cột

	Name	Score1	Score2
0	Alisa	62	89
1	Bobby	47	87
2	Cathrine	55	67
3	Madonna	74	55
4	Rocky	31	47
5	Sebastian	77	72
6	Jaquiline	85	76
7	Rahul	63	79
8	David	42	44

01

FULL OUTER JOIN



	Name	Score3
0	Alisa	56
1	Bobby	86
2	Cathrine	77
3	Madonna	45
4	Rocky	73
5	Sebastian	62
6	Jaquiline	74

02

- `pd.concat([df1,df2], axis=1)`: sử dụng các tham số mặc định

```

1 #Trường hợp 1:
2 #Mặc định join='outer'
3 df_concat1 = pd.concat([df1, df2], axis=1)
4 df_concat1

```

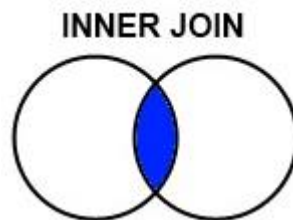
	Name	Score1	Score2	Name	Score3
0	Alisa	62	89	Alisa	56.0
1	Bobby	47	87	Bobby	86.0
2	Cathrine	55	67	Cathrine	77.0
3	Madonna	74	55	Madonna	45.0
4	Rocky	31	47	Rocky	73.0
5	Sebastian	77	72	Sebastian	62.0
6	Jaquiline	85	76	Jaquiline	74.0
7	Rahul	63	79	NaN	NaN
8	David	42	44	NaN	NaN

6.1 Nối các DataFrame theo cột

	Name	Score1	Score2
0	Alisa	62	89
1	Bobby	47	87
2	Cathrine	55	67
3	Madonna	74	55
4	Rocky	31	47
5	Sebastian	77	72
6	Jaquiline	85	76
7	Rahul	63	79
8	David	42	44

	Name	Score3
0	Alisa	56
1	Bobby	86
2	Cathrine	77
3	Madonna	45
4	Rocky	73
5	Sebastian	62
6	Jaquiline	74

01



02

- `pd.concat([df1,df2], axis=1, join='inner')`: Sử dụng tham số join

```

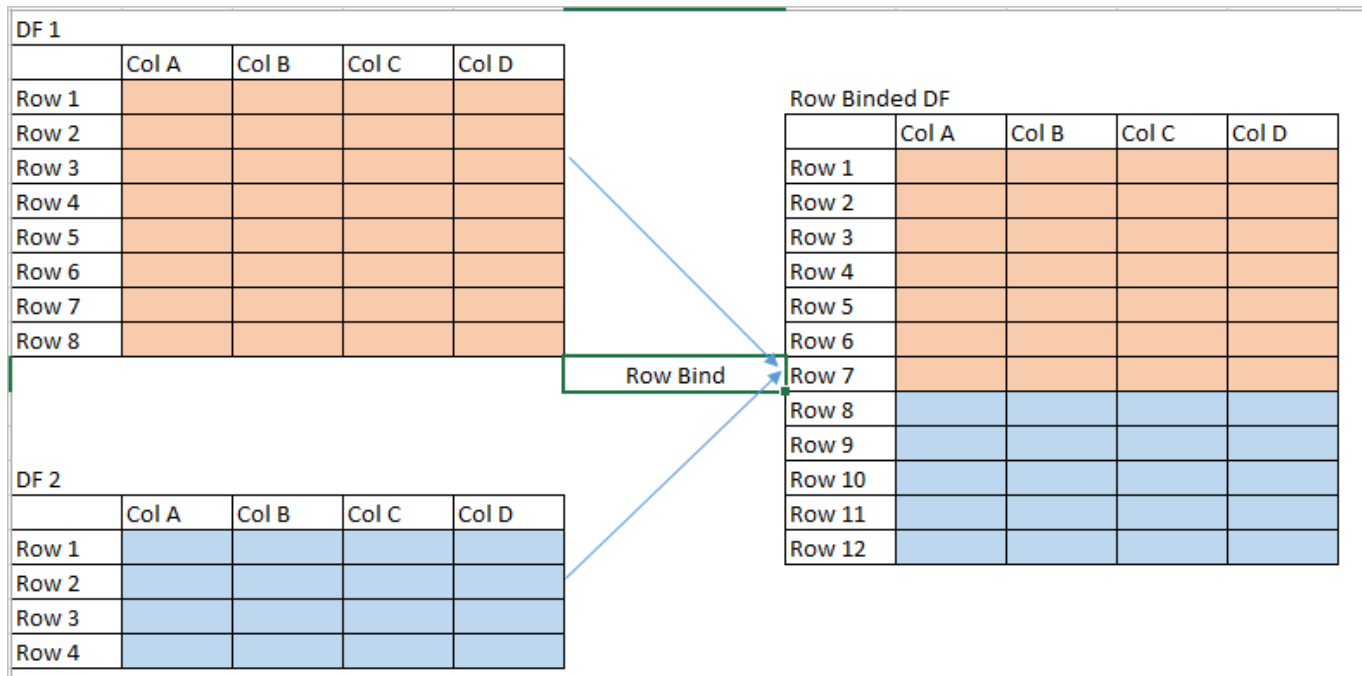
1 #Trường hợp 2:
2 #Tham số join='inner'
3 df_concat2 = pd.concat([df1, df2],
4                          axis=1,
5                          join='inner')
6 df_concat2

```

	Name	Score1	Score2	Name	Score3
0	Alisa	62	89	Alisa	56
1	Bobby	47	87	Bobby	86
2	Cathrine	55	67	Cathrine	77
3	Madonna	74	55	Madonna	45
4	Rocky	31	47	Rocky	73
5	Sebastian	77	72	Sebastian	62
6	Jaquiline	85	76	Jaquiline	74

6.2 Nối các DataFrame theo hàng

- **pd.concat([df1,df2], axis=0, join='inner'|'outer', ignore_index=True|False) hoặc df1.append(df2):** Thực hiện ghép nối các DataFrame lại với nhau theo hàng



6.2 Nối các DataFrame theo hàng

01

	Name	Score1	Score2	Score3
0	Alisa	62	89	56
1	Bobby	47	87	86
2	Cathrine	55	67	77
3	Madonna	74	55	45
4	Rocky	31	47	73

Trường hợp các cột cùng tên:

02

	Name	Score1	Score2	Score3
0	Andrew	32	92	67
1	Ajay	71	99	97
2	Teresa	57	69	68

- **pd.concat([df1,df2]):**

```
1 #Trường hợp 1: sử dụng concat
2 df_row = pd.concat([df1,df2])
3 df_row
```

	Name	Score1	Score2	Score3
0	Alisa	62	89	56
1	Bobby	47	87	86
2	Cathrine	55	67	77
3	Madonna	74	55	45
4	Rocky	31	47	73
0	Andrew	32	92	67
1	Ajay	71	99	97
2	Teresa	57	69	68

- **df1.append(df2):**

```
1 #Trường hợp 1: Sử dụng append()
2 df_row2 = df1.append(df2)
3 df_row2
```

	Name	Score1	Score2	Score3
0	Alisa	62	89	56
1	Bobby	47	87	86
2	Cathrine	55	67	77
3	Madonna	74	55	45
4	Rocky	31	47	73
0	Andrew	32	92	67
1	Ajay	71	99	97
2	Teresa	57	69	68

6.2 Nối các DataFrame theo hàng

01

	Name	Score1	Score2	Score3
0	Alisa	62	89	56
1	Bobby	47	87	86
2	Cathrine	55	67	77
3	Madonna	74	55	45
4	Rocky	31	47	73

Trường hợp các cột khác tên:

	Name	Score1	Score4	Score5
0	Jack	32	72	57
1	danny	71	91	72
2	vishwa	70	89	78

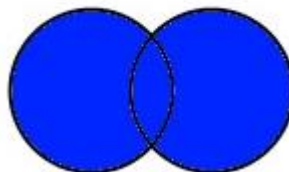
02

- `pd.concat([df1,df3])` | `df1.append(df3):`

```
1 #Trường hợp các cột khác tên
2 pd.concat([df1,df3])
```

	Name	Score1	Score2	Score3	Score4	Score5
0	Alisa	62	89.0	56.0	NaN	NaN
1	Bobby	47	87.0	86.0	NaN	NaN
2	Cathrine	55	67.0	77.0	NaN	NaN
3	Madonna	74	55.0	45.0	NaN	NaN
4	Rocky	31	47.0	73.0	NaN	NaN
0	Jack	32	NaN	NaN	72.0	57.0
1	danny	71	NaN	NaN	91.0	72.0
2	vishwa	70	NaN	NaN	89.0	78.0

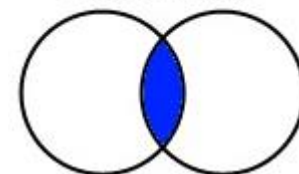
FULL OUTER JOIN



```
1 #sử dụng tham số join='inner'
2 pd.concat([df1,df3], join='inner')
```

	Name	Score1
0	Alisa	62
1	Bobby	47
2	Cathrine	55
3	Madonna	74
4	Rocky	31
0	Jack	32
1	danny	71
2	vishwa	70

INNER JOIN



Thực hành 2

Thực hành 2



Yêu cầu 2.1:

- Đọc dữ liệu từ file **Data_Point.xlsx** vào biến kiểu dataframe:
 - df_lop1 dữ liệu điểm sheet 0 (4080130_01)
 - df_lop2 dữ liệu điểm sheet 1 (4080130_02)
 - df_lop3 dữ liệu điểm sheet 2 (4080130_03)

1	Code	A	B1	B2	C1	C2
2	1621050322	8	0	5	7.5	8
3	1621050512	6	3	7.5	8.5	9
4	1621050211	6.7	4	6.5	3	5
5	1621050827	8	6.5	8	10	9
6	1621050298	7	5	8	8.5	9
7	1621050351	4.3	5	5	6	6
8	1621050422	7	6.5	9	10	10
9	1621050281	5.3	3.5	6	8.5	8
10	1621050753	6	5	6.5	10	10
11	1621050283	6	5.5	7	8.5	8
12	1621050122	5.3	2	6	8.5	8
13	1621050203	6	8	8	10	10
14	1621050090	6	5	6.5	10	8
15	1621050802	7	0	8	0	5
16	1621050434	6	9	7	10	9.5
17	1621050240	6	9	7	10	9.5

◀ ▶

4080130_01

4080130_02

4080130_03

Code

+

Xem lại Bài 06

Thực hành 2



Yêu cầu 2.2:

- Nối 3 DataFrame df_lop1, df_lop2, df_lop3 thành một DataFrame df_full chứa tất cả danh sách bảng điểm của 3 lớp

```
1 df_full.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 144 entries, 0 to 143
Data columns (total 6 columns):
 #   Column  Non-Null Count  Dtype  
---  -
 0   Code    144 non-null    int64  
 1   A        144 non-null    float64
 2   B1       144 non-null    float64
 3   B2       144 non-null    float64
 4   C1       144 non-null    float64
 5   C2       144 non-null    float64
dtypes: float64(5), int64(1)
memory usage: 6.9 KB
```

	Code	A	B1	B2	C1	C2
0	1621050322	8.0	0.0	5.0	7.5	8.0
1	1621050512	6.0	3.0	7.5	8.5	9.0
2	1621050211	6.7	4.0	6.5	3.0	5.0
3	1621050827	8.0	6.5	8.0	10.0	9.0
4	1621050298	7.0	5.0	8.0	8.5	9.0
...
139	1721050290	7.0	8.0	8.0	10.0	9.0
140	1621050162	6.3	7.0	8.5	10.0	9.0
141	1721050199	6.3	0.0	7.5	10.0	6.0
142	1621050308	0.0	5.0	0.0	10.0	9.0
143	1621050034	8.0	8.0	7.5	8.5	8.0

144 rows × 6 columns

Thực hành 2



Yêu cầu 2.3:

- Trong df_full: Tạo một cột **Diem_he10** được tính dựa vào các cột tương ứng của từng hàng dữ liệu, theo công thức sau:

$$\text{Diem_he10} = 0.6 * A + 0.3 * ((B1+B2)/2) + 0.1 * ((C1+C2)/2)$$

- Làm tròn đến 1 số sau dấu phẩy

	Code	A	B1	B2	C1	C2	Diem_he10
0	1621050322	8.0	0.0	5.0	7.5	8.0	6.3
1	1621050512	6.0	3.0	7.5	8.5	9.0	6.0
2	1621050211	6.7	4.0	6.5	3.0	5.0	6.0
3	1621050827	8.0	6.5	8.0	10.0	9.0	7.9
4	1621050298	7.0	5.0	8.0	8.5	9.0	7.0
...
139	1721050290	7.0	8.0	8.0	10.0	9.0	7.6
140	1621050162	6.3	7.0	8.5	10.0	9.0	7.1
141	1721050199	6.3	0.0	7.5	10.0	6.0	5.7
142	1621050308	0.0	5.0	0.0	10.0	9.0	1.7
143	1621050034	8.0	8.0	7.5	8.5	8.0	8.0

144 rows × 7 columns

Thực hành 2

Yêu cầu 2.4:

- Từ cột Diem_he10 trong df_full tạo một cột Diem_chu, Diem_so theo quy đổi dưới đây:

Điểm theo thang 10	Điểm theo hệ 4	
	Điểm chữ	Điểm số
Từ 9,0 đến 10,0	A ⁺	4,0
Từ 8,5 đến cận 9,0	A	3,7
Từ 8,0 đến cận 8,4	B ⁺	3,5
Từ 7,0 đến cận 7,9	B	3,0
Từ 6,5 đến cận 7,0	C ⁺	2,5
Từ 5,5 đến cận 6,5	C	2,0
Từ 5,0 đến cận 5,5	D ⁺	1,5
Từ 4,0 đến cận 5,0	D	1,0
Từ 0,0 đến cận 4,0	F	0

	Code	A	B1	B2	C1	C2	Diem_he10	Diem_chu	Diem_so
0	1621050322	8.0	0.0	5.0	7.5	8.0	6.3	C	2.0
1	1621050512	6.0	3.0	7.5	8.5	9.0	6.0	C	2.0
2	1621050211	6.7	4.0	6.5	3.0	5.0	6.0	C	2.0
3	1621050827	8.0	6.5	8.0	10.0	9.0	7.9	B	3.0
4	1621050298	7.0	5.0	8.0	8.5	9.0	7.0	B	3.0
...
139	1721050290	7.0	8.0	8.0	10.0	9.0	7.6	B	3.0
140	1621050162	6.3	7.0	8.5	10.0	9.0	7.1	B	3.0
141	1721050199	6.3	0.0	7.5	10.0	6.0	5.7	C	2.0
142	1621050308	0.0	5.0	0.0	10.0	9.0	1.7	F	0.0
143	1621050034	8.0	8.0	7.5	8.5	8.0	8.0	B+	3.5

144 rows × 9 columns

Thực hành 2



Yêu cầu 2.5:

- Tạo một DataFrame **df_diem_ok** chỉ lấy dữ liệu các cột ['Code', 'Diem_he10', 'Diem_chu', 'Diem_so'] từ df_full

```
1 df_diem_ok.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 144 entries, 0 to 143
Data columns (total 4 columns):
#   Column      Non-Null Count  Dtype
---  -
0   Code        144 non-null    int64
1   Diem_he10    144 non-null    float64
2   Diem_chu     144 non-null    object
3   Diem_so     144 non-null    float64
dtypes: float64(2), int64(1), object(1)
memory usage: 4.6+ KB
```

	Code	Diem_he10	Diem_chu	Diem_so
0	1621050322	6.3	C	2.0
1	1621050512	6.0	C	2.0
2	1621050211	6.0	C	2.0
3	1621050827	7.9	B	3.0
4	1621050298	7.0	B	3.0
...
139	1721050290	7.6	B	3.0
140	1621050162	7.1	B	3.0
141	1721050199	5.7	C	2.0
142	1621050308	1.7	F	0.0
143	1621050034	8.0	B+	3.5

144 rows × 4 columns

Thực hành 2



Yêu cầu 2.6:

- Đọc dữ liệu trong sheet: code của file excel Data_point vào DataFrame **df_code**.
- Trộn (merge) dữ liệu của df_code và df_diem_ok để ghép phách cho bảng điểm và lưu và **df_finaly**.

1 df_finaly

	Code	Name	Birth	Class	Diem_he10	Diem_chu	Diem_so
0	1421050452	Nguyễn Duy Khánh	28/03/1995	DCCTPM59_1	0.0	F	0.0
1	1421050514	Vũ Trà My	01/01/1995	DCCTPM59_1	7.6	B	3.0
2	1521020083	Tạ Văn Được	20/08/1996	DCCTPM60_1	7.1	B	3.0
3	1521050138	Nguyễn Hữu Trang	04/10/1997	DCCTPM60_1	5.2	D+	1.5
4	1521050164	Phí Đình Thành	19/05/1997	DCCTPM60_1	0.0	F	0.0
...
139	1721050290	Nguyễn Hoài Thương	15/01/1999	DCCTPM62A	7.6	B	3.0
140	1721050401	Nguyễn Đức Nguyên	20/06/1999	DCCTPM62B	7.8	B	3.0
141	1721050524	Nguyễn Thị Anh	18/05/1999	DCCTPM62A	7.5	B	3.0
142	1721050707	Nguyễn Thị Lý	21/08/1994	DCCTPM62B	9.3	A+	4.0
143	1931050001	Lưu Quang Linh	17/05/1988	LCCTCT64HN	7.4	B	3.0

144 rows × 7 columns

Lưu dữ liệu trong
DataFrame df_finaly ra
file excel:
Diem_4080130.xlsx

8. Ví dụ về xử lý dữ liệu

Ví dụ chuẩn bị và phân tích tập dữ liệu

agricultural_raw_material - Excel

File Home Insert Page Layout Formulas Data Review View Team Tell me what you want to do...

Clipboard Font Alignment Number Styles Cells Editing

H37 469.37

	A	B	C	D	E	F	G	H	I	J	K	L	M
	Month	Coarse wool Price	Coarse wool price % Change	Copra Price	Copra price % Change	Cotton Price	Cotton price % Change	Fine wool Price	Fine wool price % Change	Hard log Price	Hard log price % Change	Hard sawnwood Price	Hard sawnwood price % Change
1	Apr-90	482.34	-	236	-	1.83	-	1,071.63	-	161.2	-	549.91	-
2	May-90	447.26	-7.27%	234	-0.85%	1.89	3.28%	1,057.18	-1.35%	172.86	7.23%	491.88	-
3	Jun-90	440.99	-1.40%	216	-7.69%	1.99	5.29%	898.24	-15.03%	181.67	5.10%	495.39	-
4	Jul-90	418.44	-5.11%	205	-5.09%	2.01	1.01%	895.83	-0.27%	187.96	3.46%	485.86	-
5	Aug-90	418.44	0.00%	198	-3.41%	1.79	-10.95%	951.22	6.18%	186.13	-0.97%	487.52	-
6	Sep-90	412.18	-1.50%	196	-1.01%	1.79	0.00%	936.77	-1.52%	185.33	-0.43%	487.75	-
7	Oct-90	394.64	-4.26%	198	1.02%	1.79	0.00%	901.85	-3.73%	189.76	2.39%	505.24	-
8	Nov-90	334.5	-15.24%	236	19.19%	1.82	1.68%	888.61	-1.47%	179.02	-5.66%	511.33	-
9	Dec-90	328.24	-1.87%	237	0.42%	1.85	1.65%	870.55	-2.03%	171.13	-4.41%	499.2	-
10	Jan-91	319.47	-2.67%	233	-1.69%	1.85	0.00%	887.41	1.94%	169.19	-1.13%	497.05	-
11	Feb-91	323.23	1.18%	226	-3.00%	1.87	1.08%	596.02	-32.84%	176.93	4.57%	492.56	-
12	Mar-91	328.24	1.55%	236	4.42%	1.86	-0.53%	586.39	-1.62%	162.57	-8.12%	491.12	-
13	Apr-91	365.82	11.45%	224	-5.08%	1.83	-1.61%	596.02	1.64%	175.59	8.01%	495.44	-
14	May-91	371.88	1.66%	226	0.89%	1.82	-0.55%	721	20.97%	174.04	-0.88%	509.75	-
15	Jun-91	340.6	-8.41%	245	8.41%	1.78	-2.20%	777.51	7.84%	200.15	15.00%	480.3	-
16	Jul-91	337.48	-0.92%	303	23.67%	1.7	-4.49%	723.48	-6.95%	207.82	3.83%	480.3	-
17	Aug-91	337.22	-0.08%	299	-1.32%	1.62	-4.71%	680.64	-5.92%	210.57	1.32%	553.48	-
18	Sep-91	313.96	-6.90%	296	-1.00%	1.55	-4.32%	613.45	-9.87%	210.68	0.05%	582.79	-
19	Oct-91	308.39	-1.77%	353	19.26%	1.5	-3.23%	558.18	-9.01%	214.44	1.78%	573.05	-
20	Nov-91	307.57	-0.27%	385	9.07%	1.4	-6.67%	641.49	14.93%	195.5	-8.83%	573.89	-
21	Dec-91	295.4	-3.96%	411	6.75%	1.36	-2.86%	652.19	1.67%	200.33	2.47%	567.39	-
22	Jan-92	297.04	0.56%	488	18.73%	1.31	-3.68%	619.37	-5.03%	202.85	1.26%	577.77	-
23	Feb-92	341.89	15.10%	471	-3.48%	1.24	-5.34%	655.83	5.89%	214.04	5.52%	599.19	-
24	Mar-92	341.18	-0.21%	429	-8.92%	1.22	-1.61%	667.93	1.84%	201.96	-5.64%	602.24	-
25	Apr-92	352.12	3.21%	425	-0.93%	1.28	4.92%	667.7	-0.03%	199.67	-1.13%	616.38	-
26	May-92	367.36	4.33%	413	-2.82%	1.34	4.69%	663.07	-0.69%	200.75	0.54%	628.68	-
27	Jun-92	364.88	-0.68%	390	-5.57%	1.41	5.22%	633.89	-4.40%	205.11	2.17%	610.79	-

agricultural_raw_material

Ready

13:22 PM 2020-09-22

https://colab.research.google.com/drive/1qbuedvlz2z_yHx7jDldCtsHjywdXR8Nu?usp=sharing



Thank you!