



Bài giảng môn học:

**Kỹ nghệ tri thức và học máy**

# Chương 2: Thu thập và chuẩn bị dữ liệu (phần 02)

Đặng Văn Nam

dangvannam@humg.edu.vn

1. Giới thiệu
2. Tạo đối tượng cơ bản trong Pandas
  - Series
  - Dataframe
3. Quan sát và truy xuất dữ liệu trong DataFrame
4. Replacing Values, Rename Columns
5. Lọc dữ liệu trong DataFrame
6. Xác định các tham số thống kê: Sum, Cumsum, Min, Max, Mean, Median, Std
7. Giá trị duy nhất (Unique)
8. Time series data

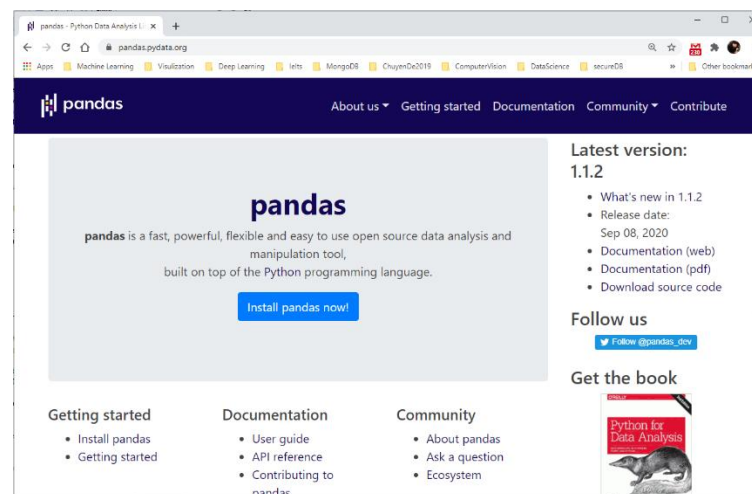
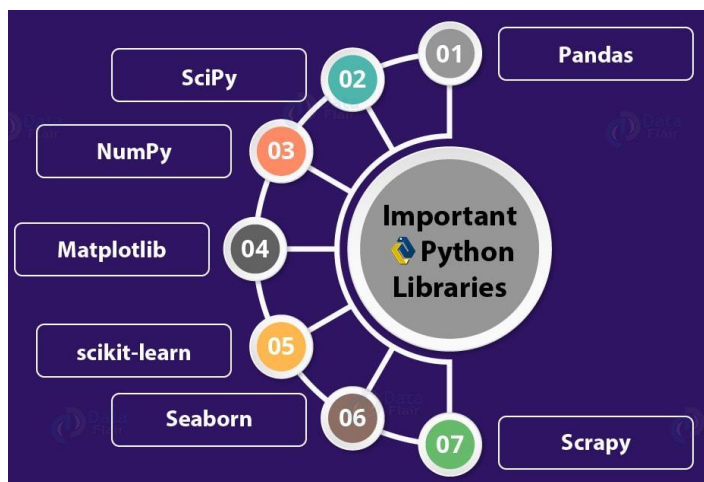
# 1. Giới thiệu

# 1. Giới thiệu

**Pandas** là một thư viện mã nguồn mở được xây dựng dựa trên NumPy, sử dụng để thao tác và phân tích dữ liệu. Với Pandas chúng ta có thể:

- Xử lý tập dữ liệu khác nhau về định dạng: chuỗi thời gian, bảng không đồng nhất, ma trận dữ liệu
- Import dữ liệu từ nhiều nguồn khác nhau như CSV, DB/SQL...
- Xử lý vô số phép toán cho tập dữ liệu: subsetting, slicing, filtering, merging, groupBy, re-ordering, and re-shaping,..
- Xử lý dữ liệu mất mát theo mong muốn.
- Xử lý, phân tích dữ liệu tốt như mô hình hoá và thống kê.
- Tích hợp tốt với các thư viện khác của python.

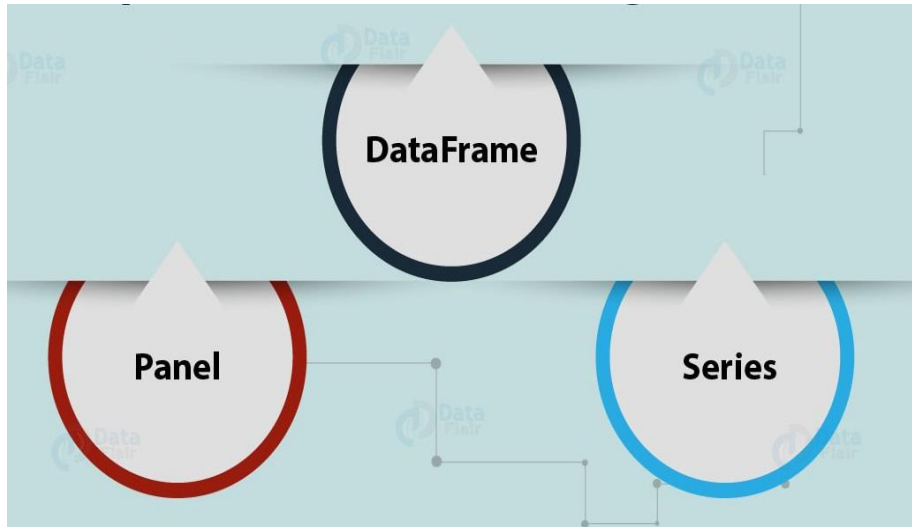
<https://pandas.pydata.org/>



The screenshot shows the official pandas website. The header includes the pandas logo and navigation links: "About us", "Getting started", "Documentation", "Community", and "Contribute". The main content area features the pandas logo, a description of pandas as a fast, powerful, flexible, and easy-to-use open source data analysis and manipulation tool, and a prominent "Install pandas now!" button. On the right side, there is a section for the "Latest version: 1.1.2", including release date (Sep 08, 2020) and links to documentation, source code, and a book. The footer contains sections for "Getting started", "Documentation", and "Community", each with links to various resources.

# 1. Giới thiệu

Pandas làm việc thông qua 3 đối tượng Series, **DataFrame**, Panel



**SERIES**

7	2	9	10
---	---	---	----

axis 0 →

**DATA FRAME**

axis 0 ↓	5.2	3.0	4.5
	9.1	0.1	0.3

axis 1 →

**PANEL**

		2		
axis 0 ↓	1	4	7	4
	2	9	7	5
	1	3	7	2
	9	6	0	8
		9	9	

axis 1 → axis 2 →

```
1 #Kiểm tra phiên bản của thư viện Pandas
2 import pandas as pd
3 print('Version Pandas: ',pd.__version__)
```

Version Pandas: 1.1.1

## 2. Series, DataFrame trong Pandas

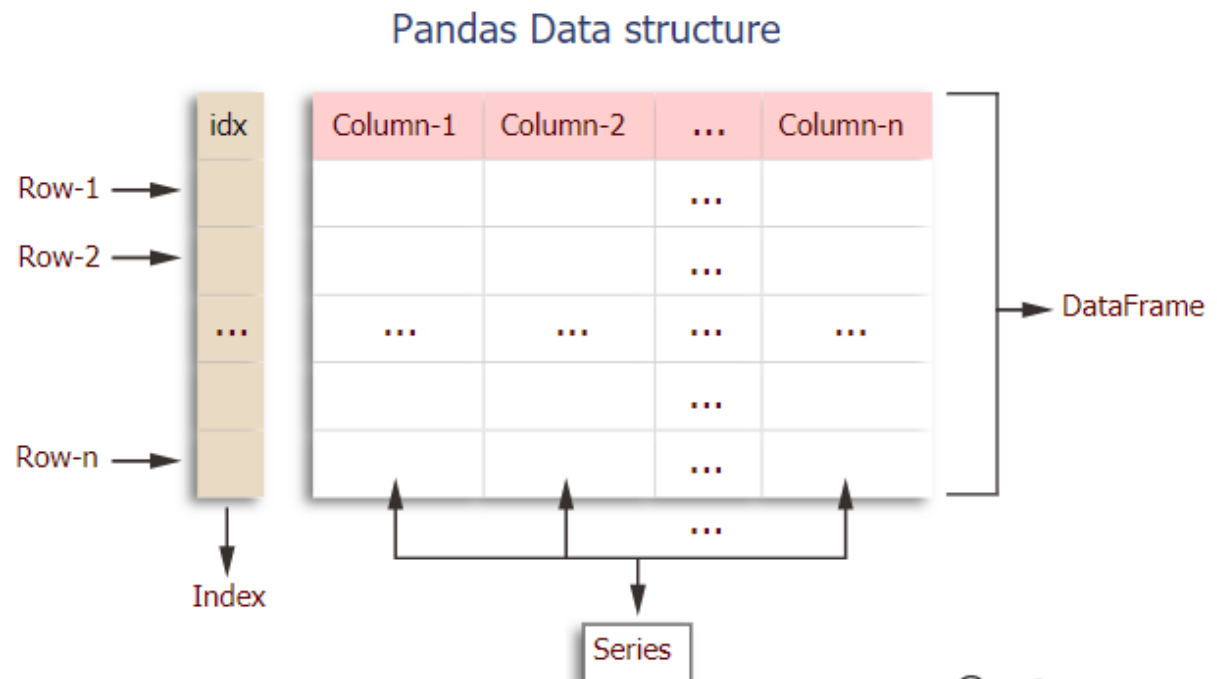
# 2.1 Series

- Series** là mảng một chiều (1D) giống như kiểu vector trong Numpy, hay như một cột của một bảng, nhưng nó bao gồm thêm một bảng đánh index.

		Data
Index	0	2.80
	1	3.00
	2	4.44
	3	5.00

dtype: float64

Series



# 2.1 Series

- Tạo Series sử dụng phương thức;
  - `pd.Series(data, index, dtype, name)`

```
1 #Tạo một đối tượng series
2 #index mặc định đánh số từ 0
3 data = pd.Series([2.8, 3, 4.44, 5])
4 data
```

```
0    2.80
1    3.00
2    4.44
3    5.00
dtype: float64
```

```
1 #Mỗi một đối tượng series bao gồm 2 thành phần
2 #1. Values
3 #2. index
4
5 print('Values:', data.values)
6 print('Indices:', data.index)
```

```
Values: [2.8  3.   4.44 5. ]
Indices: RangeIndex(start=0, stop=4, step=1)
```

```
1 #Tạo một đối tượng series với index thiết lập
2 data = pd.Series([1.25, 2, 3.5, 4.75, 8.0],
3                  index=['a', 'b', 'c', 'd', 'k'])
4 data
```

```
a    1.25
b    2.00
c    3.50
d    4.75
k    8.00
dtype: float64
```

```
1 print('Values:', data.values)
2 print('Indices:', data.index)
```

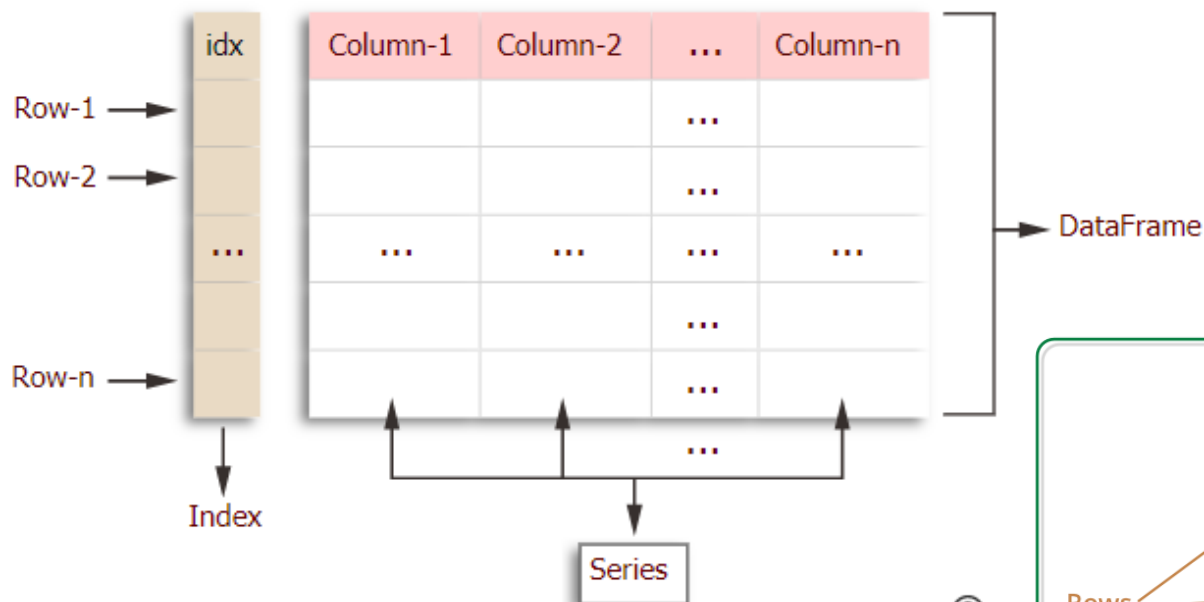
```
Values: [0.25 0.5  0.75 1. ]
Indices: Index(['a', 'b', 'c', 'd'], dtype='object')
```



## 2.2 DataFrame

**DataFrame:** Cấu trúc dạng bảng 2D, kích thước có thể thay đổi được. Dữ liệu một cột là đồng nhất nhưng có thể không đồng nhất giữa các cột

Pandas Data structure

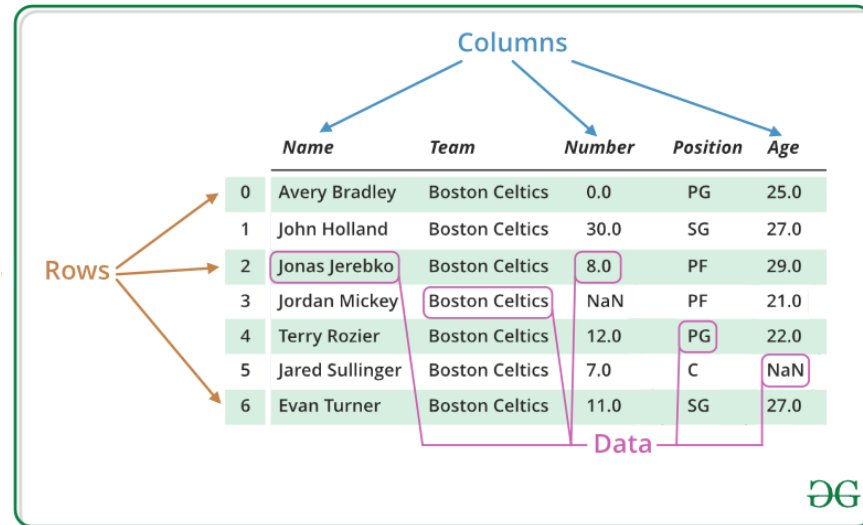


Columns

	Name	Team	Number	Position	Age
0	Avery Bradley	Boston Celtics	0.0	PG	25.0
1	John Holland	Boston Celtics	30.0	SG	27.0
2	Jonas Jerebko	Boston Celtics	8.0	PF	29.0
3	Jordan Mickey	Boston Celtics	NaN	PF	21.0
4	Terry Rozier	Boston Celtics	12.0	PG	22.0
5	Jared Sullinger	Boston Celtics	7.0	C	NaN
6	Evan Turner	Boston Celtics	11.0	SG	27.0

Rows

Data



# 2.2 DataFrame

- Tạo DataFrame sử dụng phương thức;
  - `pd.DataFrame(data, index, columns, dtype)`

```

1  #Tạo một DataFrame từ một biến Dict
2  #Chỉ số được tạo mặc định từ 0
3  data_dict = {
4      'apples': [3, 2, 0, 1],
5      'oranges': [0, 3, 7, 2]}
6
7  purchases = pd.DataFrame(data_dict)
8  purchases

```

	apples	oranges
0	3	0
1	2	3
2	0	7
3	1	2

```

1  #Tạo DataFrame với index thiết lập
2  purchases = pd.DataFrame(data_dict,
3                          index=['June', 'Robert', 'Lily', 'David'])
4  purchases

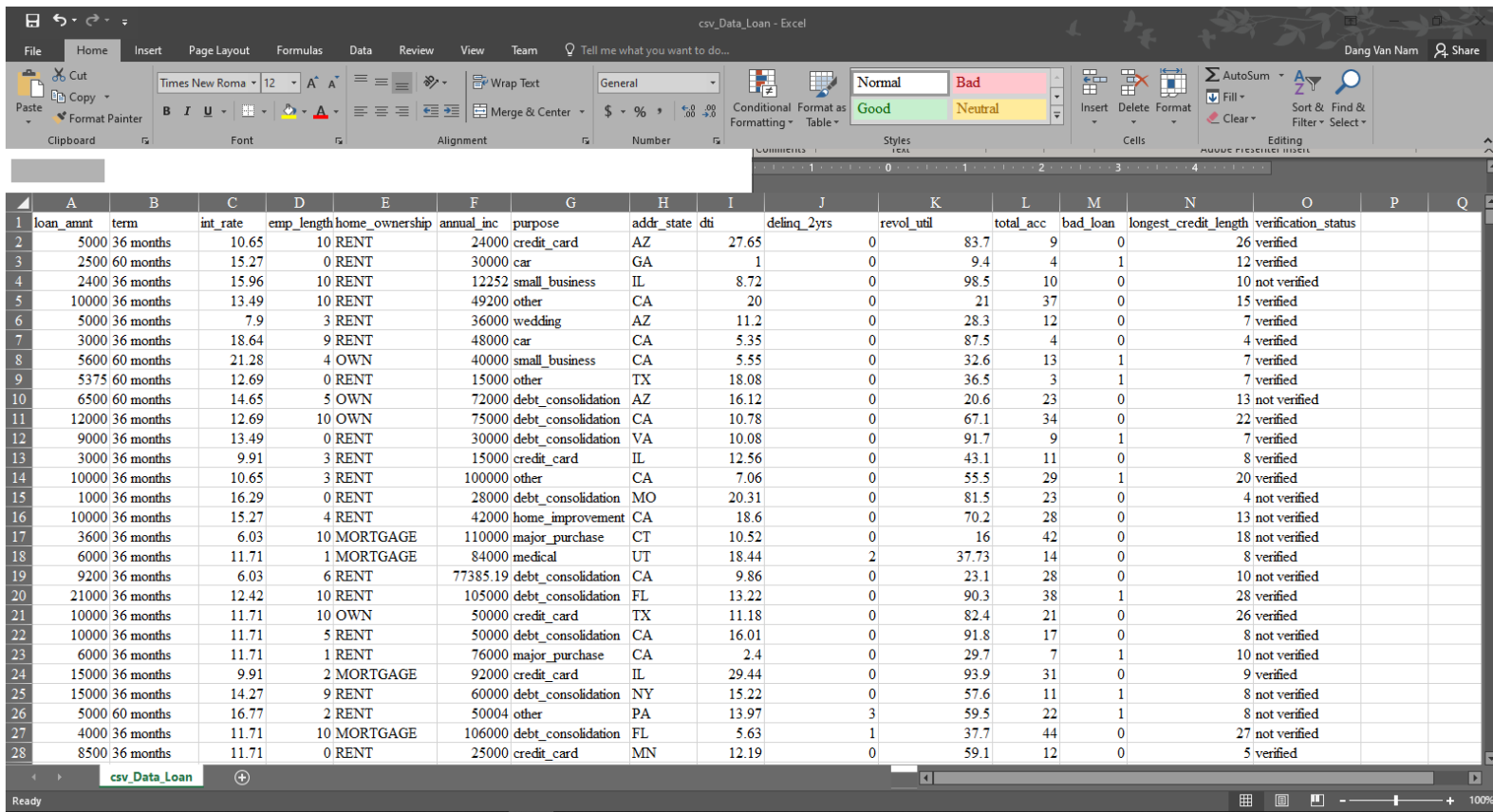
```

	apples	oranges
June	3	0
Robert	2	3
Lily	0	7
David	1	2

### 3. Quan sát và truy cập dữ liệu trong DataFrame

# 3.1 Quan sát dữ liệu

- Đọc file dữ liệu mẫu: **csv\_Data\_loan** (Bài 06)
- Đây là file dữ liệu cho biết thông tin về các khoản vay cho các mục đích khác nhau của người dùng Mỹ.



	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
	loan_amnt	term	int_rate	emp_length	home_ownership	annual_inc	purpose	addr_state	dti	delinq_2yrs	revol_util	total_acc	bad_loan	longest_credit_length	verification_status		
2	5000	36 months	10.65	10	RENT	24000	credit_card	AZ	27.65	0	83.7	9	0	26	verified		
3	2500	60 months	15.27	0	RENT	30000	car	GA	1	0	9.4	4	1	12	verified		
4	2400	36 months	15.96	10	RENT	12252	small_business	IL	8.72	0	98.5	10	0	10	not verified		
5	10000	36 months	13.49	10	RENT	49200	other	CA	20	0	21	37	0	15	verified		
6	5000	36 months	7.9	3	RENT	36000	wedding	AZ	11.2	0	28.3	12	0	7	verified		
7	3000	36 months	18.64	9	RENT	48000	car	CA	5.35	0	87.5	4	0	4	verified		
8	5600	60 months	21.28	4	OWN	40000	small_business	CA	5.55	0	32.6	13	1	7	verified		
9	5375	60 months	12.69	0	RENT	15000	other	TX	18.08	0	36.5	3	1	7	verified		
10	6500	60 months	14.65	5	OWN	72000	debt_consolidation	AZ	16.12	0	20.6	23	0	13	not verified		
11	12000	36 months	12.69	10	OWN	75000	debt_consolidation	CA	10.78	0	67.1	34	0	22	verified		
12	9000	36 months	13.49	0	RENT	30000	debt_consolidation	VA	10.08	0	91.7	9	1	7	verified		
13	3000	36 months	9.91	3	RENT	15000	credit_card	IL	12.56	0	43.1	11	0	8	verified		
14	10000	36 months	10.65	3	RENT	100000	other	CA	7.06	0	55.5	29	1	20	verified		
15	1000	36 months	16.29	0	RENT	28000	debt_consolidation	MO	20.31	0	81.5	23	0	4	not verified		
16	10000	36 months	15.27	4	RENT	42000	home_improvement	CA	18.6	0	70.2	28	0	13	not verified		
17	3600	36 months	6.03	10	MORTGAGE	110000	major_purchase	CT	10.52	0	16	42	0	18	not verified		
18	6000	36 months	11.71	1	MORTGAGE	84000	medical	UT	18.44	2	37.73	14	0	8	verified		
19	9200	36 months	6.03	6	RENT	77385.19	debt_consolidation	CA	9.86	0	23.1	28	0	10	not verified		
20	21000	36 months	12.42	10	RENT	105000	debt_consolidation	FL	13.22	0	90.3	38	1	28	verified		
21	10000	36 months	11.71	10	OWN	50000	credit_card	TX	11.18	0	82.4	21	0	26	verified		
22	10000	36 months	11.71	5	RENT	50000	debt_consolidation	CA	16.01	0	91.8	17	0	8	not verified		
23	6000	36 months	11.71	1	RENT	76000	major_purchase	CA	2.4	0	29.7	7	1	10	not verified		
24	15000	36 months	9.91	2	MORTGAGE	92000	credit_card	IL	29.44	0	93.9	31	0	9	verified		
25	15000	36 months	14.27	9	RENT	60000	debt_consolidation	NY	15.22	0	57.6	11	1	8	not verified		
26	5000	60 months	16.77	2	RENT	50004	other	PA	13.97	3	59.5	22	1	8	not verified		
27	4000	36 months	11.71	10	MORTGAGE	106000	debt_consolidation	FL	5.63	1	37.7	44	0	27	not verified		
28	8500	36 months	11.71	0	RENT	25000	credit_card	MN	12.19	0	59.1	12	0	5	verified		

## 3.1 Quan sát dữ liệu

---

- **df.info()** : Hiển thị thông tin chi tiết biến DataFrame
- **df.head(n)**: Hiển thị n dòng đầu tiên của biến df (default = 5)
- **df.tail(n)** : Hiển thị n dòng cuối cùng biến df (default = 5)
- **df.shape** : Hiển thị kích thước (rows x columns) của biến df
- **df.columns**: Tên các cột trong biến df
- **df.isnull()** : Kiểm tra dữ liệu rỗng trong biến df
- **df.isnull().sum()** : Tính tổng các dòng dữ liệu null trong df
- **df.count()** : Tổng số dòng dữ liệu không null trong df
- **df.size** : Số phần tử của biến df (=rows x columns)
- **df.dtypes** : Kiểu dữ liệu của từng columns trong df

## 3.1 Quan sát dữ liệu

- **df.describe()** : Một số đặc trưng thống kê của biến df
  - Tham số include = 'O': thống kê các cột có kiểu dữ liệu Object
  - Tham số include = 'all': Thống kê tất cả các cột trong df

```
1 #Quan sát một số đặc trưng thống kê của df
2 #Thống kê các cột dữ liệu Object
3 df_loan.describe(include='O')
```

	term	home_ownership	purpose	addr_state	verification_status
count	163987	163987	163987	163987	163987
unique	2	6	14	50	2
top	36 months	MORTGAGE	debt_consolidation	CA	verified
freq	129950	79714	93261	28702	104832

## 3.2 Truy cập dữ liệu trong DataFrame

- `df[['Col1', 'Col2', 'Col3']]`: Chỉ truy cập dữ liệu của các cột có tên **Col1, Col2, Col3** trong dataframe `df`

```
1 #Truy xuất dữ liệu theo cột
2 #Lấy dữ liệu của một cột
3 df_state = df_loan[['addr_state']]
4 df_state.head()
```

	addr_state
0	AZ
1	GA
2	IL
3	CA
4	AZ

```
1 #Truy xuất dữ liệu theo cột
2 #Chỉ lấy dữ liệu của 3 cột: loan_amnt, int_rate, purpose
3 df_loan1 = df_loan[['loan_amnt', 'int_rate', 'purpose']]
4 df_loan1.head()
```

	loan_amnt	int_rate	purpose
0	5000	10.65	credit_card
1	2500	15.27	car
2	2400	15.96	small_business
3	10000	13.49	other
4	5000	7.90	wedding

## 3.2 Truy cập dữ liệu trong DataFrame

- `df.iloc[[index_row],[index_col]]`: Truy cập tới dữ liệu của hàng và cột qua **chỉ số index\_row, index\_col (tương tự như với Numpy)**

```
1 #Sử dụng .iloc truy xuất dữ liệu như với Numpy
2 #Truy xuất 10 dòng dữ liệu từ [10 --> 20) tất cả các cột
3 df_loan.iloc[10:20,:]
```

	loan_amnt	term	int_rate	emp_length	home_ownership	annual_inc	purpose	addr_state	dti	delinq_2yrs	revol_util	total_acc	bad_loan
10	9000	36 months	13.49	0.0	RENT	30000.00	debt_consolidation	VA	10.08	0.0	91.70	9.0	1
11	3000	36 months	9.91	3.0	RENT	15000.00	credit_card	IL	12.56	0.0	43.10	11.0	0
12	10000	36 months	10.65	3.0	RENT	100000.00	other	CA	7.06	0.0	55.50	29.0	1
13	1000	36 months	16.29	0.0	RENT	28000.00	debt_consolidation	MO	20.31	0.0	81.50	23.0	0
14	10000	36 months	15.27	4.0	RENT	42000.00	home_improvement	CA	18.60	0.0	70.20	28.0	0
15	3600	36 months	6.03	10.0	MORTGAGE	110000.00	major_purchase	CT	10.52	0.0	16.00	42.0	0



## 3.2 Truy cập dữ liệu trong DataFrame

- `df.loc[[name_index],[name_col]]`: Truy cập tới dữ liệu của hàng và cột qua **name\_index, tên cột name\_col**

```
1 #Truy cập từ dòng 20 đến dòng 25 của df
2 #chỉ lấy dữ liệu 4 cột: loan_amnt, home_ownership, purpose, addr_state
3 df_loan.loc[20:25,['loan_amnt','home_ownership','purpose','addr_state']]
```

	loan_amnt	home_ownership	purpose	addr_state
20	10000	RENT	debt_consolidation	CA
21	6000	RENT	major_purchase	CA
22	15000	MORTGAGE	credit_card	IL
23	15000	RENT	debt_consolidation	NY
24	5000	RENT	other	PA
25	4000	MORTGAGE	debt_consolidation	FL

## 3.2 Truy cập dữ liệu trong DataFrame

Type	Notes
<code>df[val]</code>	Select single column or sequence of columns from the DataFrame; special case conveniences: boolean array (filter rows), slice (slice rows), or boolean DataFrame (set values based on some criterion)
<code>df.loc[val]</code>	Selects single row or subset of rows from the DataFrame by label
<code>df.loc[:, val]</code>	Selects single column or subset of columns by label
<code>df.loc[val1, val2]</code>	Select both rows and columns by label
<code>df.iloc[where]</code>	Selects single row or subset of rows from the DataFrame by integer position
<code>df.iloc[:, where]</code>	Selects single column or subset of columns by integer position
<code>df.iloc[where_i, where_j]</code>	Select both rows and columns by integer position
<code>df.at[label_i, label_j]</code>	Select a single scalar value by row and column label
<code>df.iat[i, j]</code>	Select a single scalar value by row and column position (integers)
reindex method	Select either rows or columns by labels
get_value, set_value methods	Select single value by row and column label

## 4. Replacing Values, Rename columns

# 4.1 Replacing Values

- Thay thế 1 giá trị trong Dataframe, thực hiện tương tự như với Numpy. Sử dụng **.loc**; **.iloc** để xác định phần tử cần cập nhật, thay đổi giá trị

	loan_amnt	home_ownership	purpose	addr_state
0	5000	RENT	credit_card	AZ
1	2500	RENT	car	GA
2	2400	RENT	small_business	IL
3	10000	RENT	other	CA
4	5000	RENT	wedding	AZ
5	3000	RENT	car	CA
6	5600	OWN	small_business	CA
7	5375	RENT	other	TX
8	6500	OWN	debt_consolidation	AZ
9	12000	OWN	debt_consolidation	CA
10	9000	RENT	debt_consolidation	VA

```

1 #Thay thế giá trị purpose: credit_card--> wedding
2 #của index đầu tiên
3 df_new.loc[0,'purpose'] = 'wedding'
4 df_new

```

	loan_amnt	home_ownership	purpose	addr_state
0	5000	RENT	wedding	AZ
1	2500	RENT	car	GA
2	2400	RENT	small_business	IL
3	10000	RENT	other	CA

```

1 #Thay thế giá trị thuộc tính loan_amnt: 2400 --> 8800
2 #của index = 2
3 df_new.iloc[2,0] = 8800
4 df_new

```

	loan_amnt	home_ownership	purpose	addr_state
0	5000	RENT	wedding	AZ
1	2500	RENT	car	GA
2	8800	RENT	small_business	IL
3	10000	RENT	other	CA

## 4.1 Replacing Values

- df.replace():** Thay thế các giá trị trong toàn bộ DataFrame. (tham số inplace=True|False áp dụng thay đổi cho dataframe hiện tại hay không?).

	loan_amnt	home_ownership	purpose	addr_state
0	5000	RENT	credit_card	AZ
1	2500	RENT	car	GA
2	2400	RENT	small_business	IL
3	10000	RENT	other	CA
4	5000	RENT	wedding	AZ
5	3000	RENT	car	CA
6	5600	OWN	small_business	CA
7	5375	RENT	other	TX
8	6500	OWN	debt_consolidation	AZ
9	12000	OWN	debt_consolidation	CA
10	9000	RENT	debt_consolidation	VA

```

1 #Khi muốn thay đổi áp dụng lên DataFrame hiện tại
2 #Thiết lập tham số inplace=True
3 df_new.replace({'RENT':'MORTGAGE',
4                 'car':'small_business'}, inplace=True)
5 df_new

```

```

1 #Thay thế nhiều giá trị trong DataFrame
2 #RENT --> MORTGAGE
3 #car --> small_business
4 df_new.replace({'RENT':'MORTGAGE',
5                 'car':'small_business'})

```

01

	loan_amnt	home_ownership	purpose	addr_state
0	5000	MORTGAGE	wedding	AZ
1	2500	MORTGAGE	small_business	GA
2	8800	MORTGAGE	small_business	IL
3	10000	MORTGAGE	other	CA
4	5000	MORTGAGE	wedding	AZ
5	3000	MORTGAGE	small_business	CA
6	5600	OWN	small_business	CA
7	5375	MORTGAGE	other	TX
8	6500	OWN	debt_consolidation	AZ
9	12000	OWN	debt_consolidation	CA
10	9000	MORTGAGE	debt_consolidation	VA

02

	loan_amnt	home_ownership	purpose	addr_state
0	5000	MORTGAGE	credit_card	AZ
1	2500	MORTGAGE	small_business	GA
2	2400	MORTGAGE	small_business	IL

## 4.1 Replacing Values

- **df.replace():** Thay thế các giá trị theo từng cột (tham số inplace=True|False áp dụng thay đổi cho dataframe hiện tại hay không?).

	loan_amnt	home_ownership	purpose	addr_state
0	5000	RENT	credit_card	AZ
1	2500	RENT	car	GA
2	2400	RENT	small_business	IL
3	10000	RENT	other	CA
4	5000	RENT	wedding	AZ
5	3000	RENT	car	CA
6	5600	OWN	small_business	CA
7	5375	RENT	other	TX
8	6500	OWN	debt_consolidation	AZ
9	12000	OWN	debt_consolidation	CA
10	9000	RENT	debt_consolidation	VA

```

1  #Thay thế tên viết tắt bằng tên đầy đủ.
2  state_name={'AZ':'Arizona',
3             'GA':'Georgia',
4             'IL':'Illinois',
5             'CA':'California',
6             'TX':'Texas',
7             'VA':'Virginia'}
8  #Trong cột addr_state
9  df_new['addr_state'].replace(state_name,inplace=True)
10 df_new

```

	loan_amnt	home_ownership	purpose	addr_state
0	5000	RENT	credit_card	Arizona
1	2500	RENT	car	Georgia
2	2400	RENT	small_business	Illinois
3	10000	RENT	other	California
4	5000	RENT	wedding	Arizona
5	3000	RENT	car	California

## 4.2 Rename Columns

- `df.rename()`: thay đổi tên cột trong DataFrame

```
1 #Muốn áp dụng thay đổi vào trực tiếp biến df, sử dụng inplace=True
2 df_new.rename(columns={'loan_amnt':'Số tiền vay',
3                        'home_ownership':'Tình trạng nhà ở',
4                        'purpose': 'Mục đích vay tiền',
5                        'addr_state':'Địa chỉ'}, inplace=True)
6 df_new.head()
```

01

	Số tiền vay	Tình trạng nhà ở	Mục đích vay tiền	Địa chỉ
0	5000	RENT	credit_card	AZ
1	2500	RENT	car	GA
2	2400	RENT	small_business	IL
3	10000	RENT	other	CA
4	5000	RENT	wedding	AZ

```
1 #Đổi tên cột sang viết hoa
2 df_new.rename(str.upper, axis='columns')
```

02

	SỐ TIỀN VAY	TÌNH TRẠNG NHÀ Ở	MỤC ĐÍCH VAY TIỀN	ĐỊA CHỈ
0	5000	RENT	credit_card	AZ
1	2500	RENT	car	GA
2	2400	RENT	small_business	IL
3	10000	RENT	other	CA
4	5000	RENT	wedding	AZ

# Thực hành 1



# Thực hành 1

## Mô tả file dữ liệu: Data\_Patient.csv

- File dữ liệu chứa thông tin của 300 bệnh nhân bị bệnh tim mạch
- Mỗi dòng ứng với thông tin của một bệnh nhân, bao gồm 9 thuộc tính

	A	B	C	D	E	F	G	H	I
1	id	feature_1	feature_2	feature_3	feature_4	feature_5	feature_6	feature_7	feature_8
2	Patient_01	63	Male	Typical angina	145	233	150	6	0
3	Patient_02	67	Male	Asymptomatic	160	286	108	3	1
4	Patient_03	67	Male	Asymptomatic	120	229	129	7	1
5	Patient_04	37	Male	Non-anginal pain	130	250	187	3	0
6	Patient_05	41	Female	Atypical angina	130	204	172		0
7	Patient_06	56	Male	Atypical angina	120	236	178	3	0
8	Patient_07	62	Female	Asymptomatic	140	268	160	3	1
9	Patient_08	57	Female	Asymptomatic	120	354	163	3	0
10	Patient_09	63	Male	Asymptomatic	130	254	147	7	1
11	Patient_10	53	Male	Asymptomatic	140	203	155	7	1
12	Patient_11	57	Male	Asymptomatic	140	192	148	6	0
13	Patient_12	56	Female	Atypical angina	140	294	153	3	0
14	Patient_13	56	Male	Non-anginal pain	130	256	142	6	1
15	Patient_14	44	Male	Atypical angina	120	263	173	7	0
16	Patient_15	52	Male	Non-anginal pain	172	199	162	7	0
17	Patient_16	57	Male	Non-anginal pain	150	168	174	3	0
18	Patient_17	48	Male	Atypical angina	110	229	168	7	1
19	Patient_18	54	Male	Asymptomatic	140	239	160	3	0
20	Patient_19	48	Female	Non-anginal pain	130	275	139	3	0
21	Patient_20	49	Male	Atypical angina	130	266	171	3	0

# Thực hành 1

---

**Chi tiết thông tin của một bệnh nhân như sau:**

- **id:** Mã của bệnh nhân (số)
- **Feature\_1:** Tuổi của bệnh nhân (số)
- **Feature\_2:** Giới tính của bệnh nhân (chuỗi: Male – Female)
- **Feature\_3:** Cho biết loại triệu chứng đau ngực mà bệnh nhân này mắc phải, với 4 giá trị: (Typical angina, Atypical angina, Non-anginal pain, Asymptomatic)
- **Feature\_4:** Huyết áp của bệnh nhân – đơn vị: mmhg (số)
- **Feature\_5:** Chỉ số cholesterol của bệnh nhân – đơn vị: mg/dl (số)
- **Feature\_6:** Thông số nhịp tim của bệnh nhân – đơn vị: lần/phút (số)
- **Feature\_7:** Chỉ số Thalassemia của bệnh nhân chỉ gồm 3 giá trị (3: Bình thường | 4: Khiếm khuyết cố định | 7: Kiểm khuyết có thể đảo ngược)
- **Feature\_8:** Cho biết bệnh nhân có bị bệnh tim hay không? (0: Không bị bệnh tim mạch | 1: Bị bệnh tim mạch)

## Yêu cầu 1.1:

- Đọc dữ liệu từ file `Data_Patient.csv` vào biến kiểu dataframe: `df_patient` với cột `feature_1` là cột chỉ số (`index_col`)
- Hiển thị thông tin tổng quan của tập dữ liệu
- Hiển thị thông tin của 10 bệnh nhân đầu tiên và 5 bệnh nhân cuối cùng của tập dữ liệu.
- Đặt lại tên các cột dữ liệu trong `df_patient`
  - `Feature_1` → `Age`
  - `Feature_2` → `Gender`
  - `Feature_3` → `Type`
  - `Feature_4` → `Blood_pressure`
  - `Feature_5` → `Cholesterol`
  - `Feature_6` → `Heartbeat`
  - `Feature_7` → `Thalassemia`
  - `Feature_8` → `Result`



## Yêu cầu 1.2:

- Sử dụng phương thức `.describe()` cho biết:
  - Thuộc tính Age:
    - Tuổi của bệnh nhân trẻ nhất
    - Tuổi của bệnh nhân già nhất
  - Thuộc tính Cholesterol:
    - Cholesterol trung bình của các bệnh nhân
    - Độ lệch chuẩn của giá trị này trong toàn bộ tập dữ liệu
  - Bao nhiêu bệnh nhân giới tính nam (Male)
  - Có bao nhiêu giá trị khác nhau của thuộc tính Type.  
Giá trị xuất hiện nhiều nhất là giá trị nào, bao nhiêu lần

	Gender	Type
count	300	295
unique	2	4
top	Male	Asymptomatic
freq	205	139

# Thực hành 1

## Yêu cầu 1.3:

- Cho biết những cột nào trong dữ liệu có chứa missing data và số lượng missing là bao nhiêu?

## Yêu cầu 1.4:

- Hiển thị thông tin của các bệnh nhân:
  - Bệnh nhân có index: **Patient\_100; Patient\_150; Patient\_200**
  - Bệnh nhân ở vị trí 255 đến 260, với 3 thuộc tính: Age, Gender và Result**

id	Age	Gender	Type	Blood_pressure	Cholesterol	Heartbeat	Thalassemia	Result
Patient_100	45	Male	Asymptomatic	115	260	185	3.0	0
Patient_150	52	Male	Typical angina	152	298	178	7.0	0
Patient_200	50	Female	Asymptomatic	110	254	159	3.0	0

id	Age	Gender	Result
Patient_255	42	Female	0
Patient_256	67	Female	0
Patient_257	76	Female	0
Patient_258	70	Male	0
Patient_259	57	Male	1
Patient_260	44	Female	0

# Thực hành 1

## Yêu cầu 1.5:

- Thay đổi giá trị cho thuộc tính Gender: **Male** → **0**, **Female** → **1**
- Thay đổi giá trị cho thuộc tính Result: **0** → **No**, **1** → **Yes**
- Cập nhật giá trị thuộc tính Thalassemia của bệnh nhân có index: **Patient\_05** bằng giá trị **4.0**

	Age	Gender	Type	Blood_pressure	Cholesterol	Heartbeat	Thalassemia	Result
id								
Patient_01	63	0	Typical angina	145	233	150	6.0	No
Patient_02	67	0	Asymptomatic	160	286	108	3.0	Yes
Patient_03	67	0	Asymptomatic	120	229	129	7.0	Yes
Patient_04	37	0	Non-anginal pain	130	250	187	3.0	No
Patient_05	41	1	Atypical angina	130	204	172	4.0	No

## 5. Filter Data

# 5. Filter Data

- Để lọc dữ liệu trong DataFrame có thể sử dụng nhiều cách khác nhau

```
1 #Lọc danh sách người giới tính nam
2 #Cách 1:
3 df_male1 = df_bmi[df_bmi.Gender=='Male']
4 df_male1.head(2)
```

	Personal	Gender	Height_cm	Weight_kg
0	P1	Male	174	96
1	P2	Male	189	87

01

```
1 #Cách 2: sử dụng phương thức query
2 df_male2 = df_bmi.query('Gender=="Male"')
3 df_male2.head(2)
```

	Personal	Gender	Height_cm	Weight_kg
0	P1	Male	174	96
1	P2	Male	189	87

02

```
1 #Cách 3: sử dụng iloc
2 df_male3 = df_bmi.loc[(df_bmi.Gender=="Male")]
3 df_male3.head(2)
```

	Personal	Gender	Height_cm	Weight_kg
0	P1	Male	174	96
1	P2	Male	189	87

03



# 5. Filter Data

- Sử dụng toán tử **& (and)** - **| (or)** - **~ (not)** để kết hợp nhiều điều kiện trong khi lọc dữ liệu

```
1 #Kết hợp nhiều tiêu chí lọc dữ liệu
2 #lọc người có giới tính Femal và cân nặng dưới 70kg
3 df_p1 = df_bmi[(df_bmi.Gender == 'Female') & (df_bmi.Weight_kg<70)]
4 df_p1
```

01

	Personal	Gender	Height_cm	Weight_kg
24	P25	Female	172	67
25	P26	Female	151	64
32	P33	Female	195	65
51	P52	Female	176	54

```
1 #Kết hợp nhiều tiêu chí tìm kiếm
2 #lọc người có chiều cao > 195 cm hoặc cân nặng > 150kg
3 df_p2 = df_bmi[(df_bmi.Height_cm >195) | (df_bmi.Weight_kg>150)]
4 df_p2
```

	Personal	Gender	Height_cm	Weight_kg
28	P29	Female	163	159
29	P30	Male	179	152
34	P35	Female	157	153
36	P37	Female	197	114

02

```
1 # toán tử ~ - Not
2 df_p3 = df_bmi[~(df_bmi.Weight_kg<155)]
3 df_p3
```

	Personal	Gender	Height_cm	Weight_kg
28	P29	Female	163	159
65	P66	Female	179	158

03

## 5. Filter Data

- Sử dụng phương thức `.isin()` để kết lọc dữ liệu theo một tập hợp

```
1 #Lọc ra những người có cân nặng bằng 150, 155 và 160kg
2 # phương thức isin (tương tự như in)
3 df_p4 = df_bmi[df_bmi.Weight_kg.isin([150,155,160])]
4 df_p4
```

	Personal	Gender	Height_cm	Weight_kg
102	P103	Male	161	155
106	P107	Male	166	160
123	P124	Female	184	160
134	P135	Female	171	155
135	P136	Female	183	150

## 6. Tính toán min, max, mean, median, std, sum, cumsum

## 6. Đặc trưng thống kê trong DataFrame

- Sử dụng phương thức `.max()`, `.min()`, `.sum()`, `.mean()`, `.median()`, `.cumsum()`, `.std()` để tính các đặc trưng thống kê cho DataFrame hoặc theo từng cột.

```
1 #tìm Max, Min của thuộc tính cân nặng
2 w_max = df_bmi['Weight_kg'].max()
3 w_min = df_bmi['Weight_kg'].min()
4 print('Cân nặng lớn nhất:', w_max, '(kg)')
5 print('Cân nặng nhỏ nhất:', w_min, '(kg)')
```

Cân nặng lớn nhất: 160 (kg)

Cân nặng nhỏ nhất: 50 (kg)

```
1 #tìm độ lệch chuẩn của chiều cao, cân nặng
2 h_std = df_bmi['Height_cm'].std()
3 w_std = df_bmi['Weight_kg'].std()
4 print('sdt của chiều cao:', h_std)
5 print('sdt của cân nặng:', w_std)
```

sdt của chiều cao: 16.37526067959376

sdt của cân nặng: 32.38260746964435

```
1 #tìm Mean, Median của chiều cao
2 h_mean = df_bmi['Height_cm'].mean()
3 h_median = df_bmi['Height_cm'].median()
4 print('Chiều cao trung bình:', h_mean, '(cm)')
5 print('Trung vị:', h_median, '(cm)')
```

Chiều cao trung bình: 169.944 (cm)

Trung vị: 170.5 (cm)

## 7. Xác định giá trị duy nhất (Unique)

# 7. Unique

- **df.unique():** liệt kê danh sách các giá trị khác nhau trong một cột dữ liệu của DataFrame.
- **df.value\_counts():** Tính tổng số theo từng giá trị khác nhau trong một cột dữ liệu của DataFrame. Kết quả là một đối tượng series.

```
1 #Xác định giá trị duy nhất trong một cột
2 df_bmi['Gender'].unique()
```

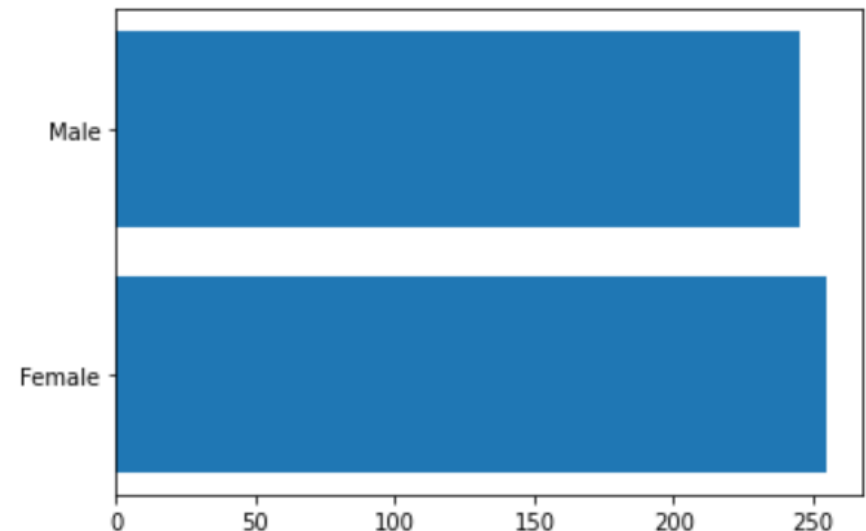
```
array(['Male', 'Female'], dtype=object)
```

```
1 #Vẽ đồ thị thể hiện kết quả
2 plt.barh(unique_gender.index, unique_gender.values)
```

```
<BarContainer object of 2 artists>
```

```
1 #Thống kê số lượng theo giá trị duy nhất
2 unique_gender = df_bmi['Gender'].value_counts()
3 unique_gender
```

```
Female    255
Male      245
Name: Gender, dtype: int64
```




# Thực hành 2

# Thực hành 2

## Yêu cầu 2.1:

- Đọc dữ liệu từ file `Data_Patient.csv` vào biến kiểu dataframe: `df_patient` với cột `feature_1` là cột chỉ số (`index_col`)
- Đặt lại tên các cột dữ liệu trong Dataframe



	A	B	C	D	E	F	G	H	I
1	id	feature_1	feature_2	feature_3	feature_4	feature_5	feature_6	feature_7	feature_8
2	Patient_01	63	Male	Typical angina	145	233	150	6	0
3	Patient_02	67	Male	Asymptomatic	160	286	108	3	1
4	Patient_03	67	Male	Asymptomatic	120	229	129	7	1
5	Patient_04	37	Male	Non-anginal pain	130	250	187	3	0
6	Patient_05	41	Female	Atypical angina	130	204	172		0
7	Patient_06	56	Male	Atypical angina	120	236	178	3	0
8	Patient_07	62	Female	Asymptomatic	140	268	160	3	1
9	Patient_08	57	Female	Asymptomatic	120	354	163	3	0
10	Patient_09	63	Male	Asymptomatic	130	254	147	7	1
11	Patient_10	53	Male	Asymptomatic	140	203	155	7	1
12	Patient_11	57	Male	Asymptomatic	140	192	148	6	0
13	Patient_12	56	Female	Atypical angina	140	294	153	3	0
14	Patient_13	56	Male	Non-anginal pain	130	256	142	6	1
15	Patient_14	44	Male	Atypical angina	120	263	173	7	0
16	Patient_15	52	Male	Non-anginal pain	172	199	162	7	0
17	Patient_16	57	Male	Non-anginal pain	150	168	174	3	0
18	Patient_17	48	Male	Atypical angina	110	229	168	7	1
19	Patient_18	54	Male	Asymptomatic	140	239	160	3	0
20	Patient_19	48	Female	Non-anginal pain	130	275	139	3	0
21	Patient_20	49	Male	Atypical angina	130	266	171	3	0

Ready

- Feature\_1 → Age
- Feature\_2 → Gender
- Feature\_3 → Type
- Feature\_4 → Blood\_pressu
- Feature\_5 → Cholesterol
- Feature\_6 → Heartbeat
- Feature\_7 → Thalassemia
- Feature\_8 → Result



# Thực hành 2



## Yêu cầu 2.2:



- Lọc dữ liệu trong `df_patient` thành các DataFrame:

- `df_male`: chứa danh sách bệnh nhân Nam
- `df_female`: chứa danh sách bệnh nhân nữ
- `df_no`: danh sách những người không bị bệnh đau tim

## Yêu cầu 2.3:

- Lọc trong `df_patient` đưa ra danh sách bệnh nhân thỏa mãn yêu cầu sau:
  1. Những người bị mắc bệnh **đau tim** và trên **70 tuổi**
  2. Người có giới tính **Female**, có huyết áp trên **170 mmhg** nhưng **không bị bệnh đau tim**.
  3. Những người có triệu chứng đau ngực là **Typical angina**, giới tính **Male** và **bị bệnh đau tim**.

## Yêu cầu 2.4: Xác định:



1. Chỉ số huyết áp (**Blood\_pressure**) thấp nhất, cao nhất, trung bình, trung vị và độ lệch chuẩn của tập dữ liệu
2. Chỉ số nhịp tim (**Heartbeat**) thấp nhất, cao nhất, trung bình, trung vị và độ lệch chuẩn của tập dữ liệu

1. Chỉ số huyết áp:

Min: 94

Max: 200

Mean: 131.68666666666667

Median: 130.0

Std: 17.682497692285477

2. Chỉ số nhịp tim:

Min: 71

Max: 202

Mean: 149.56333333333333

Median: 152.5

Std: 22.818595118151098

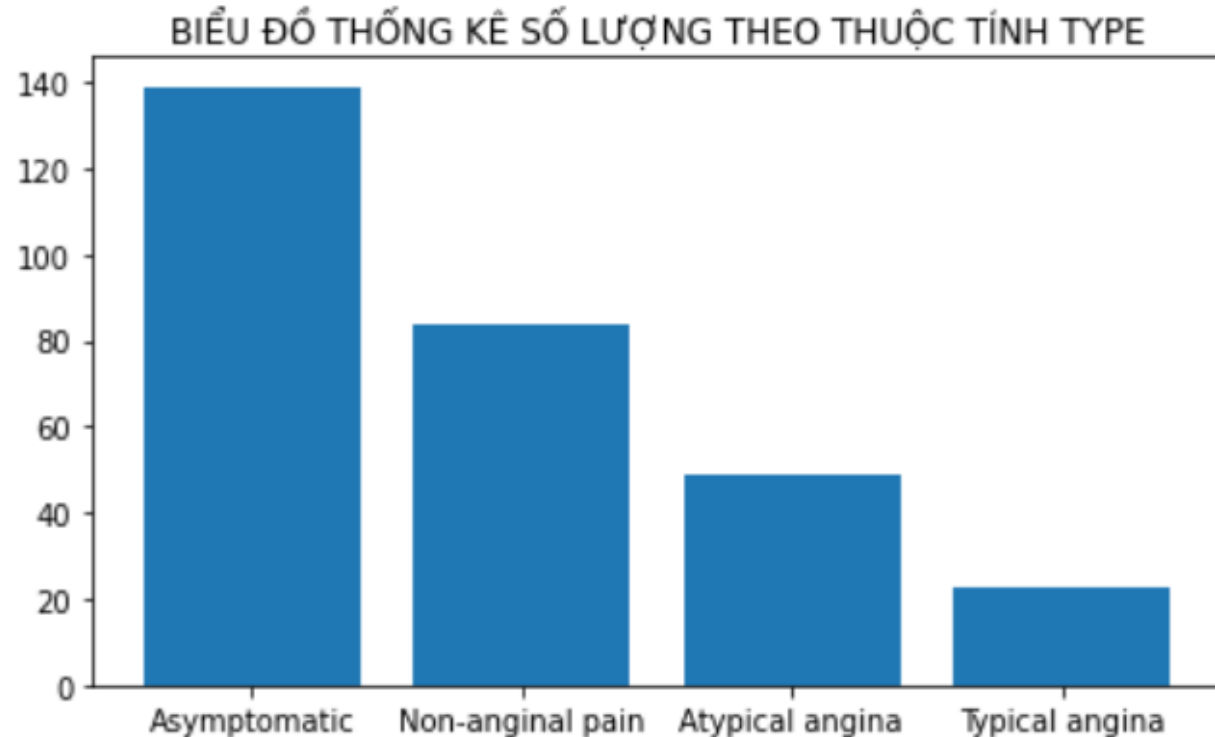


# Thực hành 2

## Yêu cầu 2.5: Xác định:



1. Số giá trị khác nhau của thuộc tính **Type**
2. Vẽ đồ thị dạng cột thể hiện kết quả thống kê số lượng theo từng giá trị khác nhau của thuộc tính Type



Asymptomatic	139
Non-anginal pain	84
Atypical angina	49
Typical angina	23