LẬP TRÌNH PYTHON CHO PHÂN TÍCH DỮ LIỆU VÀ HỌC MÁY

Bài 6: Phân tích và xử lý dữ liệu với Pandas - 01

AI Academy Vietnam

Nội dung bài 6

1. Giới thiệu Pandas
2. Tạo đối tượng cơ bản trong Pandas
   - Series
   - Dataframe
3. Đọc dữ liệu từ các nguồn khác nhau
4. Quan sát và truy xuất dữ liệu trong DataFrame
5. Replacing Values, Rename Columns
6. Lọc dữ liệu trong DataFrame
7. Xác định các đặc trưng thống kê của tập dữ liệu
8. Giá trị duy nhất (Unique)
9. Phân tích Time series data (Tiếp cận từ bài toán thực tế)

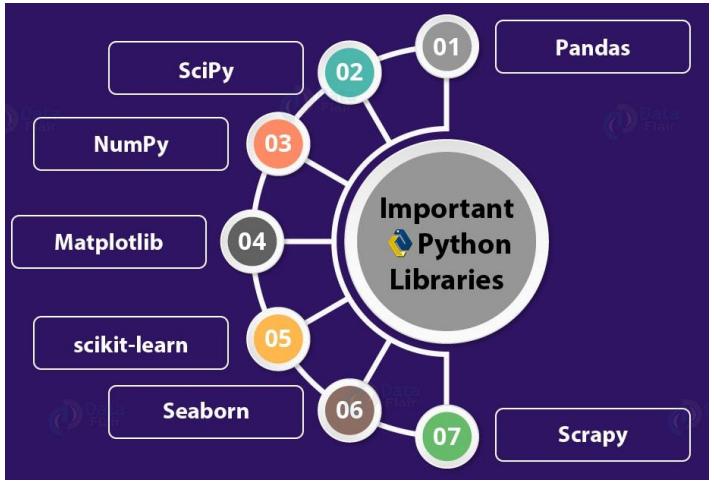
1. Giới thiệu Pandas

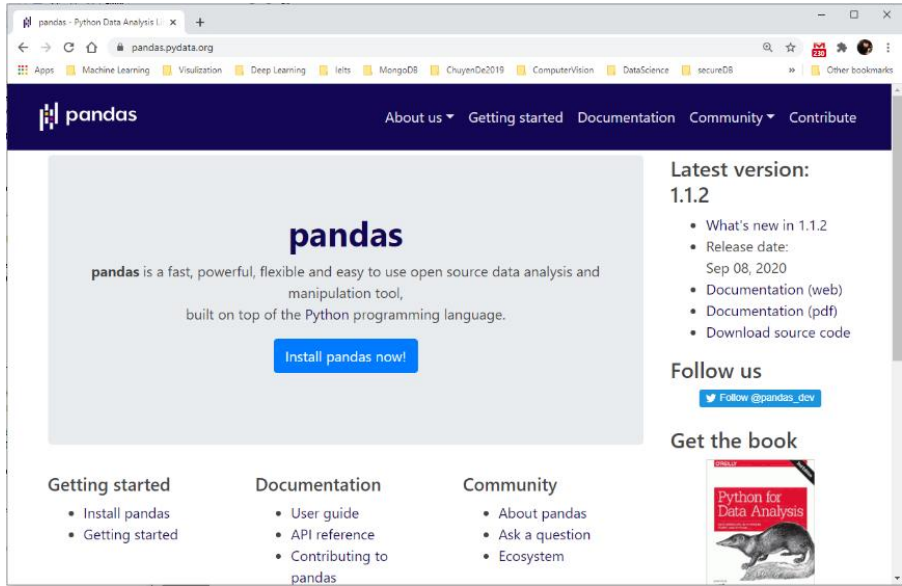
1. Giới thiệu

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.

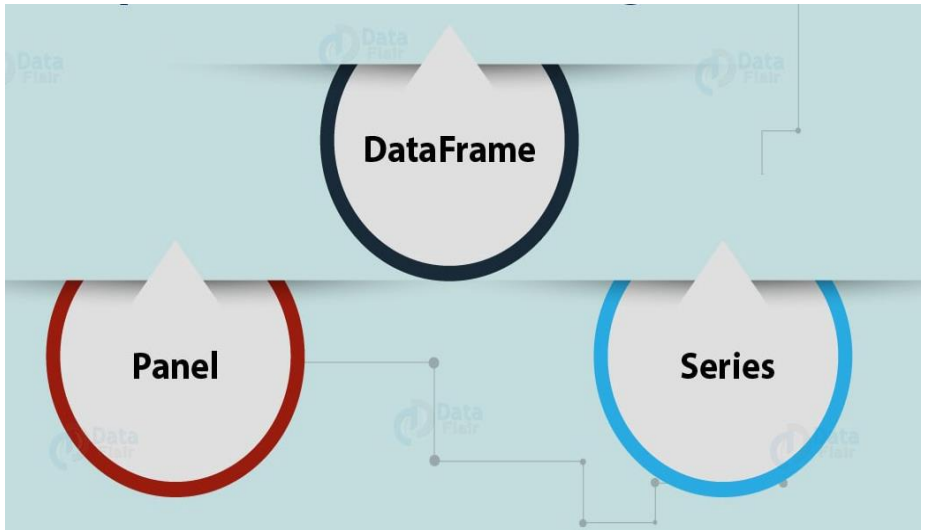
https://pandas.pydata.org/



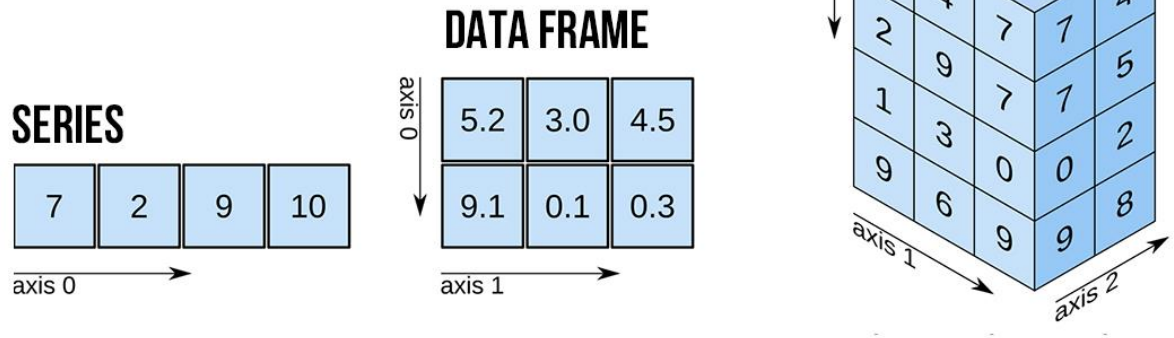


1. Giới thiệu Pandas

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



Trong ba kiểu dữ liệu, DataFrame là kiểu dữ liệu được sử dụng rộng rãi nhất.



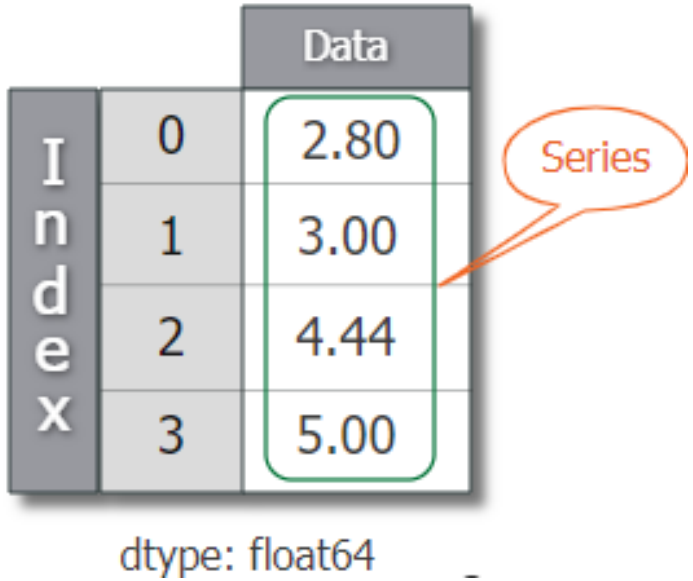

```
1  #Kiểm tra phiên bản của thư viện Pandas
2  import pandas as pd
3  print('Version Pandas: ',pd.__version__)
```

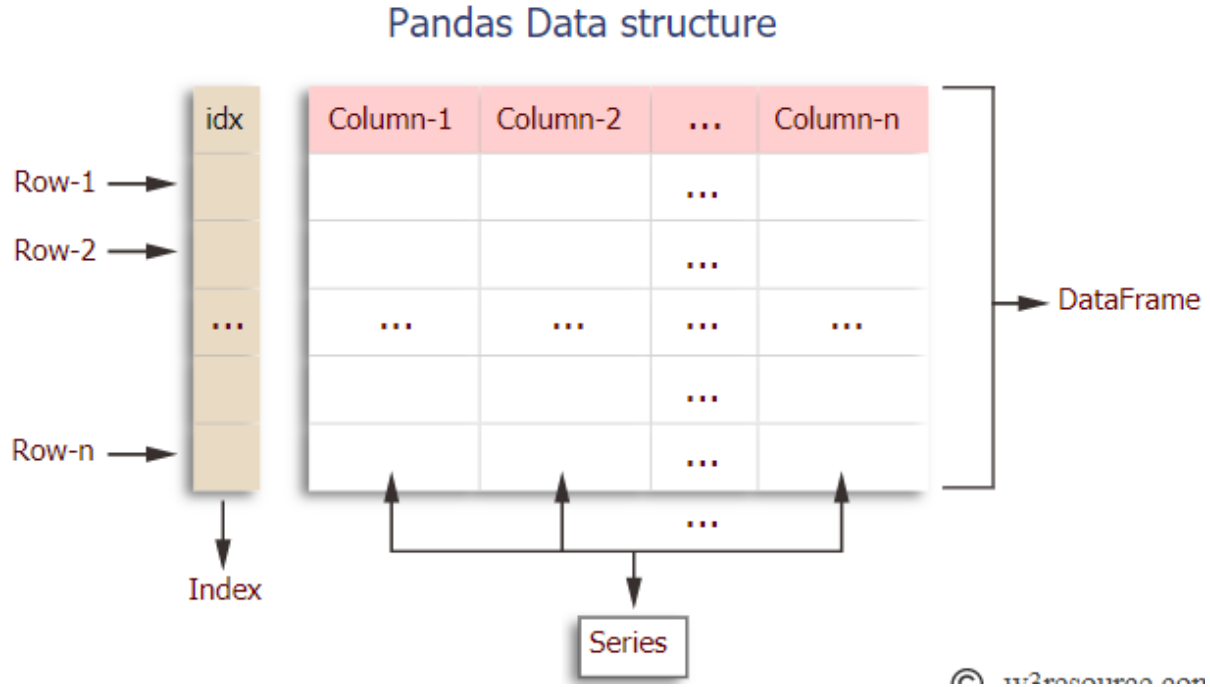
```
Version Pandas:  1.1.1
```

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.





2.1 Series

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

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

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

```
#Mỗi một đối tượng series bao gồm 2 thành phần
#1. Values
#2. index

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

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

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

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

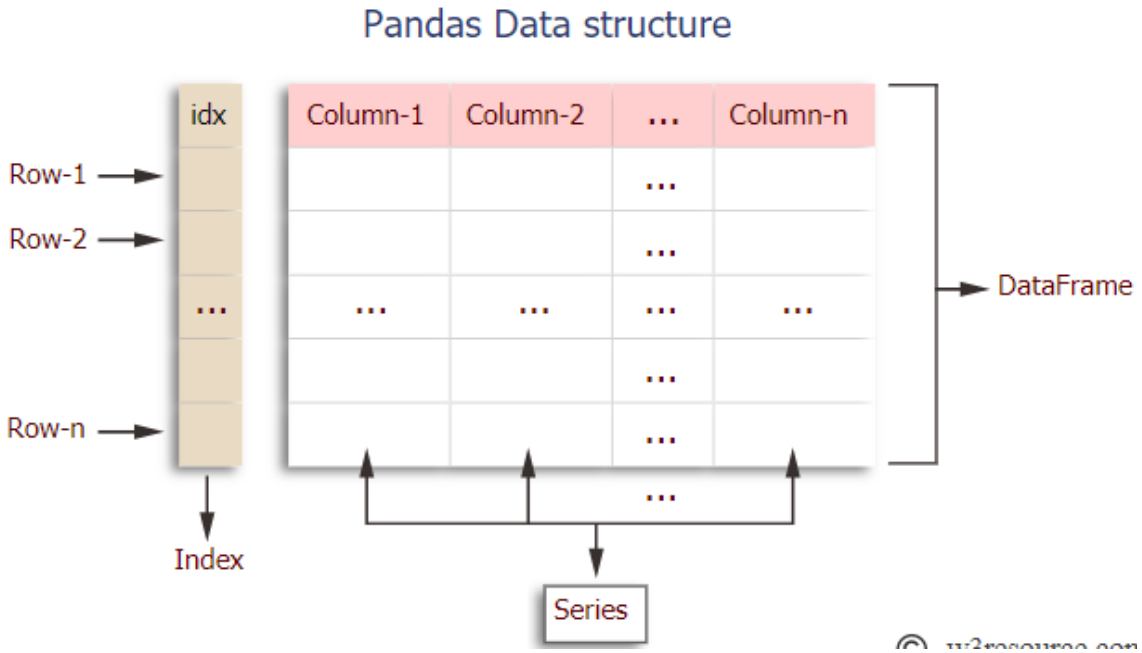
```
print('Values:', data.values)
print('Indices:', data.index)
```

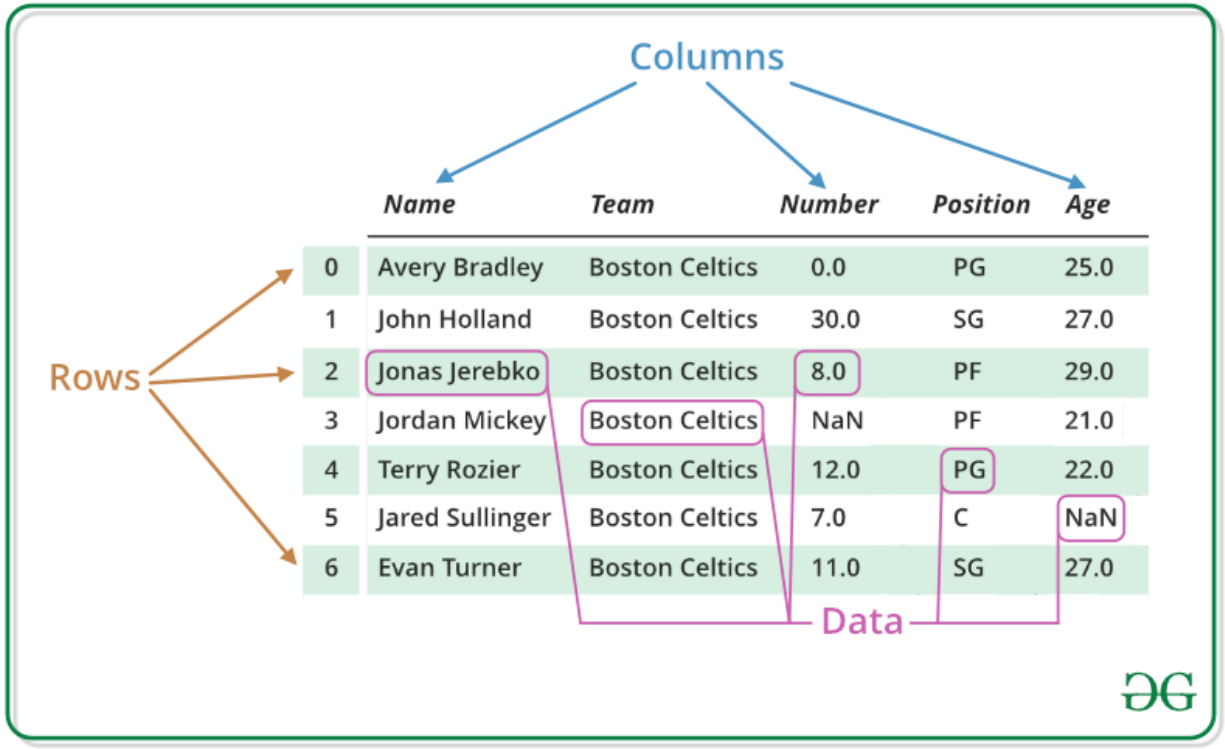
```
Values: [0.25 0.5  0.75 1.  ]
Indices: Index(['a', 'b', 'c', 'd'], dtype='object')
```

https://pandas.pydata.org/docs/reference/api/pandas.Series.html

2.2 DataFrame

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





2.2 DataFrame

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

```
#Tạo một DataFrame từ một biến Dict
#Chỉ số được tạo mặc định từ 0
data_dict = {
    'apples': [3, 2, 0, 1],
    'oranges': [0, 3, 7, 2]}

purchases = pd.DataFrame(data_dict)
purchases
```

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

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

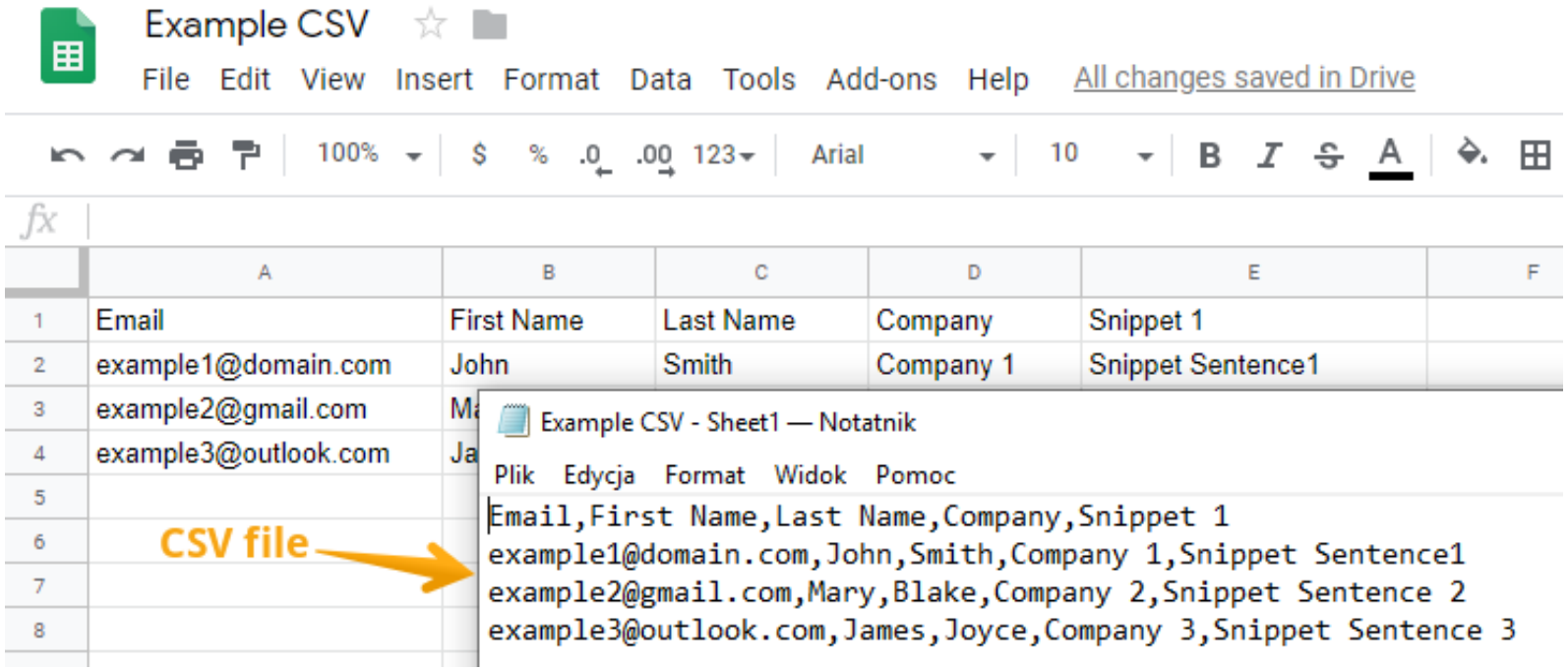
	apples	oranges
June	3	0
Robert	2	3
Lily	0	7
David	1	2

https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.html

3. Đọc dữ liệu từ các nguồn khác nhau (CSV, Text, Excel)

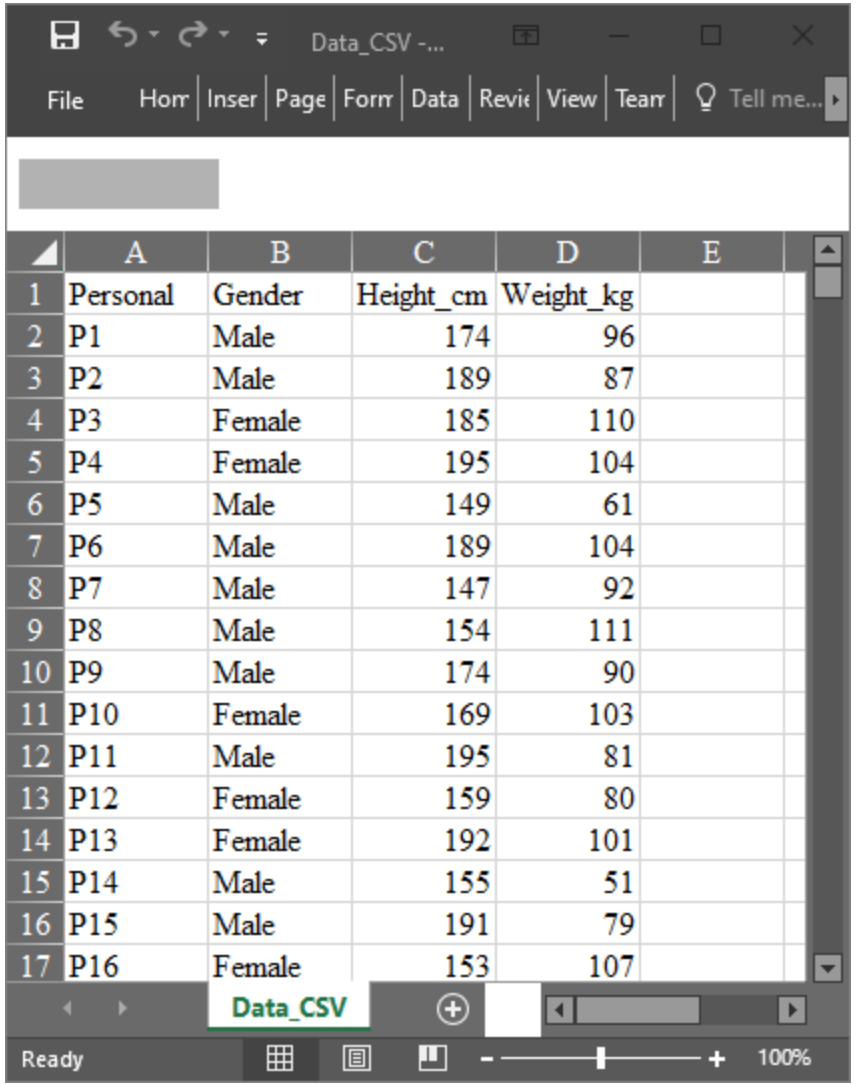
3.1 Đọc file CSV, Text

- CSV là một định dạng dữ liệu văn bản đơn giản có tên đầy đủ là Comma Separated Values. Với định dạng CSV này, các giá trị được chia tách với nhau bởi các dấu phẩy. Định dạng CSV phổ biến bởi vì chúng có tính tương thích cao, dễ dàng di chuyển từ phần mềm này sang phần mềm khác để sử dụng mà không lo gặp các xung đột.
- Tài liệu CSV cũng làm một trong những tài liệu phổ biến trên thế giới với khả năng lưu trữ nhỏ nhẹ.



3.1 Đọc file CSV, Text

Sử dụng phương thức read_csv() đọc dữ liệu từ file .CSV



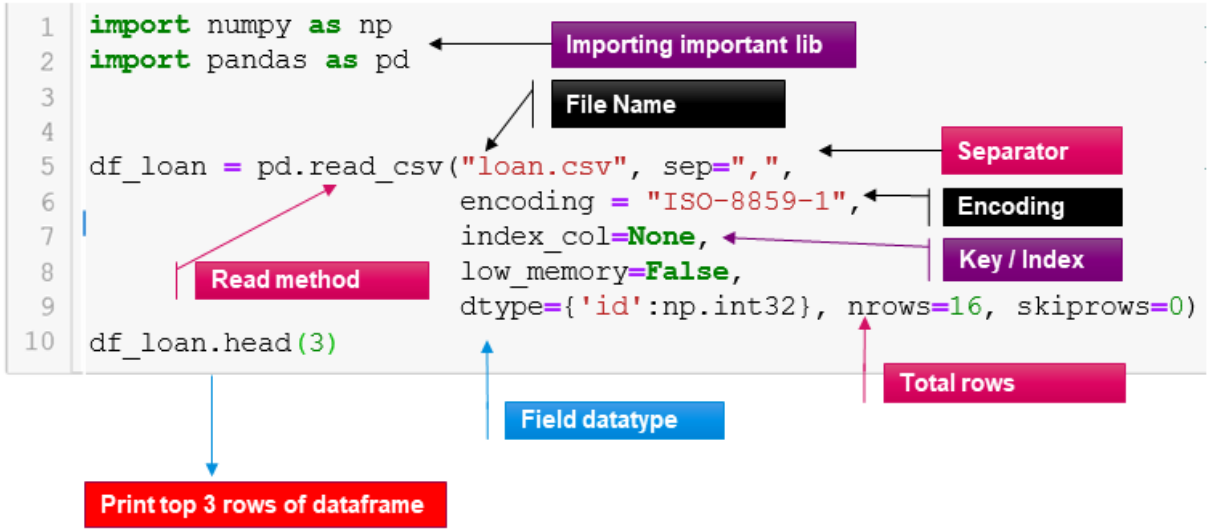
	A	B	C	D	E
1	Personal	Gender	Height_cm	Weight_kg	
2	P1	Male	174	96	
3	P2	Male	189	87	
4	P3	Female	185	110	
5	P4	Female	195	104	
6	P5	Male	149	61	
7	P6	Male	189	104	
8	P7	Male	147	92	
9	P8	Male	154	111	
10	P9	Male	174	90	
11	P10	Female	169	103	
12	P11	Male	195	81	
13	P12	Female	159	80	
14	P13	Female	192	101	
15	P14	Male	155	51	
16	P15	Male	191	79	
17	P16	Female	153	107	

```python
import pandas as pd
path = 'Data_Excersice\CSV\Data_CSV.csv'
#Sử dụng phương thức read_csv
data = pd.read_csv(path)
#Hiển thị thông tin biến Data
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 500 entries, 0 to 499
Data columns (total 4 columns):
 #   Column     Non-Null Count  Dtype
---  ------     --------------  -----
 0   Personal   500 non-null    object
 1   Gender     500 non-null    object
 2   Height_cm  500 non-null    int64
 3   Weight_kg  500 non-null    int64
dtypes: int64(2), object(2)
memory usage: 15.8+ KB
```

3.1 Đọc file CSV, Text

Sử dụng phương thức read_csv() có rất nhiều tham số khác nhau để thiết lập cách thức đọc file .csv



ID	funding_round_type	funded_at	raised_amount_usd	name	category_list	status	country_code
1	VENTURE	1/8/2014	0	DERON	NAN	OPERATING	NaN
2	SEED	1/1/2015	18192	ASYS	CONSUMER ELECTRONICS	OPERATING	USA
3	GRANT	1/10/2013	14851	Ä HIZMETLERI TIC	CONSUMER GOODS	OPERATING	NaN

```
Arguments

read_csv(filepath_or_buffer, sep=',',
         delimiter=None, header='infer',
         names=None, index_col=None, usecols=None,
         squeeze=False, prefix=None,
         mangle_dupe_cols=True, dtype=None,
         engine=None, converters=None,
         true_values=None, false_values=None,
         skipinitialspace=False, skiprows=None,
         nrows=None, na_values=None,
         keep_default_na=True, na_filter=True,
         verbose=False, skip_blank_lines=True,
         parse_dates=False,
         infer_datetime_format=False,
         keep_date_col=False, date_parser=None,
         dayfirst=False, iterator=False,
         chunksize=None, compression='infer',
         thousands=None, decimal='.',
         lineterminator=None, quotechar='"',
         quoting=0, escapechar=None, comment=None,
         encoding=None, dialect=None,
         tupleize_cols=False,
         error_bad_lines=True,
         warn_bad_lines=True, skipfooter=0,
         skip_footer=0, doublequote=True,
         delim_whitespace=False,
         as_recarray=False, compact_ints=False,
         use_unsigned=False, low_memory=True,
         buffer_lines=None, memory_map=False,
         float_precision=None)
```

https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.read_csv.html

3.1 Đọc file CSV, Text

Vd1: sử dụng tham số index_col để thiết lập cột index khi đọc file csv

```
#Sử dụng phương thức read_csv()
#Tham số: Thiết lập cột index là cột Personal
data1 = pd.read_csv(path,
                    index_col=0)
data1.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 500 entries, P1 to P500
Data columns (total 3 columns):
 #   Column     Non-Null Count  Dtype
---  ------     --------------  -----
 0   Gender     500 non-null    object
 1   Height_cm  500 non-null    int64
 2   Weight_kg  500 non-null    int64
dtypes: int64(2), object(1)
memory usage: 15.6+ KB
```

```
#Hiển thị dữ liệu 5 dòng đầu tiên
data1.head()
```

Personal	Gender	Height_cm	Weight_kg
P1	Male	174	96
P2	Male	189	87
P3	Female	185	110
P4	Female	195	104
P5	Male	149	61

3.1 Đọc file CSV, Text

Vd2: Thiết lập tham số chỉ đọc 100 dòng đầu tiên và dữ liệu trong 2 cột Height_cm, Weight_kg

```
#Sử dụng phương thức read_csv()
#Thiết lập số hàng, cột muốn đọc dữ liệu
data2 = pd.read_csv(path,
                    nrows=100,
                    usecols=['Height_cm', 'Weight_kg'])
data2.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 100 entries, 0 to 99
Data columns (total 2 columns):
 #   Column     Non-Null Count  Dtype
---  ------     --------------  -----
 0   Height_cm  100 non-null    int64
 1   Weight_kg  100 non-null    int64
dtypes: int64(2)
memory usage: 1.7 KB
```

```
#Hiển thị dữ liệu 5 dòng đầu tiên
data2.head()
```

	Height_cm	Weight_kg
0	174	96
1	189	87
2	185	110
3	195	104
4	149	61

3.1 Đọc file CSV, Text

Vd3: Thiết lập tham số đọc dữ liệu từ dòng thứ 5 trở đi, và đặt lại tên của từng cột dữ liệu thành [‘ID’,’Sex’,’H(cm)’,W(kg)’]

```
#Thiết lập tham số đọc dữ liệu từ dòng thứ 5 trong file
#và đặt lại tên của các cột dữ liệu
data3 = pd.read_csv(path,
                    names=['ID','Sex','H(cm)','W(kg)'],
                    skiprows=5)
data3.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 496 entries, 0 to 495
Data columns (total 4 columns):
 #   Column  Non-Null Count  Dtype
---  ------  --------------  -----
 0   ID      496 non-null    object
 1   Sex     496 non-null    object
 2   H(cm)   496 non-null    int64
 3   W(kg)   496 non-null    int64
dtypes: int64(2), object(2)
memory usage: 15.6+ KB
```

```
#Hiển thị 5 dòng dữ liệu đầu tiên
data3.head()
```

	ID	Sex	H(cm)	W(kg)
0	P5	Male	149	61
1	P6	Male	189	104
2	P7	Male	147	92
3	P8	Male	154	111
4	P9	Male	174	90

3.1 Đọc file CSV, Text

Vd4: Đọc dữ liệu lưu trữ trong file Text vào biến DataFrame cũng sử dụng phương thức read_csv()

```
#Đọc dữ liệu trong file txt_Data_Diamonds.txt:
df_Diamonds = pd.read_csv('Data_Excercise/txt_Data_Diamonds.txt',
                          names=['Weight(carat)','Price(USD)'],
                          sep='\t', #mặc định sep=','
                          header=None)

df_Diamonds
```

	Weight(carat)	Price(USD)
0	0.23	484
1	0.31	942
2	0.20	345
3	1.02	4459

Thực hành 1

Thực hành 1

Yêu cầu 1.1: Học viên đọc dữ liệu dạng CSV lưu trong file csv_Data_Loan.csv với các tham số mặc định

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
1	loan_amnt	term	int_rate	emp_length	home_ownership	annual_inc	purpose	addr_state	dti	delinq_2yrs	revol_util	total_acc	bad_loan	longest_cre	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

Data_Loan

Ready

100%

Thực hành 1

Yêu cầu 1.2: Đọc dữ liệu từ file Data_Loan.CSV vào 2 biến DataFrame tương ứng.

- **df_number:** Chỉ chứa các cột dữ liệu số
- **df_object:** Chỉ chứa các cột dữ liệu Object

```
#Hiển thị 5 dòng dữ liệu đầu tiên của biến df_number
df_number.head()
```

	loan_amnt	int_rate	emp_length	annual_inc	dti	delinq_2yrs	revol_util	total_acc	bad_loan	longest_credit_length
0	5000	10.65	10.0	24000.0	27.65	0.0	83.7	9.0	0	26.0
1	2500	15.27	0.0	30000.0	1.00	0.0	9.4			
2	2400	15.96	10.0	12252.0	8.72	0.0	98.5			
3	10000	13.49	10.0	49200.0	20.00	0.0	21.0			
4	5000	7.90	3.0	36000.0	11.20	0.0	28.3			

```
#Hiển thị 5 dòng dữ liệu đầu tiên của biến df_object
df_object.head()
```

	term	home_ownership	purpose	addr_state	verification_status
0	36 months	RENT	credit_card	AZ	verified
1	60 months	RENT	car	GA	verified
2	36 months	RENT	small_business	IL	not verified
3	36 months	RENT	other	CA	verified
4	36 months	RENT	wedding	AZ	verified

Thực hành 1

Yêu cầu 1.3: Đọc dữ liệu nhiệt độ của 6 thành phố [Hà Nội, Vinh, Đà Nẵng, Nha Trang, TP Hồ Chí Minh, Cà Mau] từ file txt_Data_Temp.txt vào biến DataFrame tương ứng

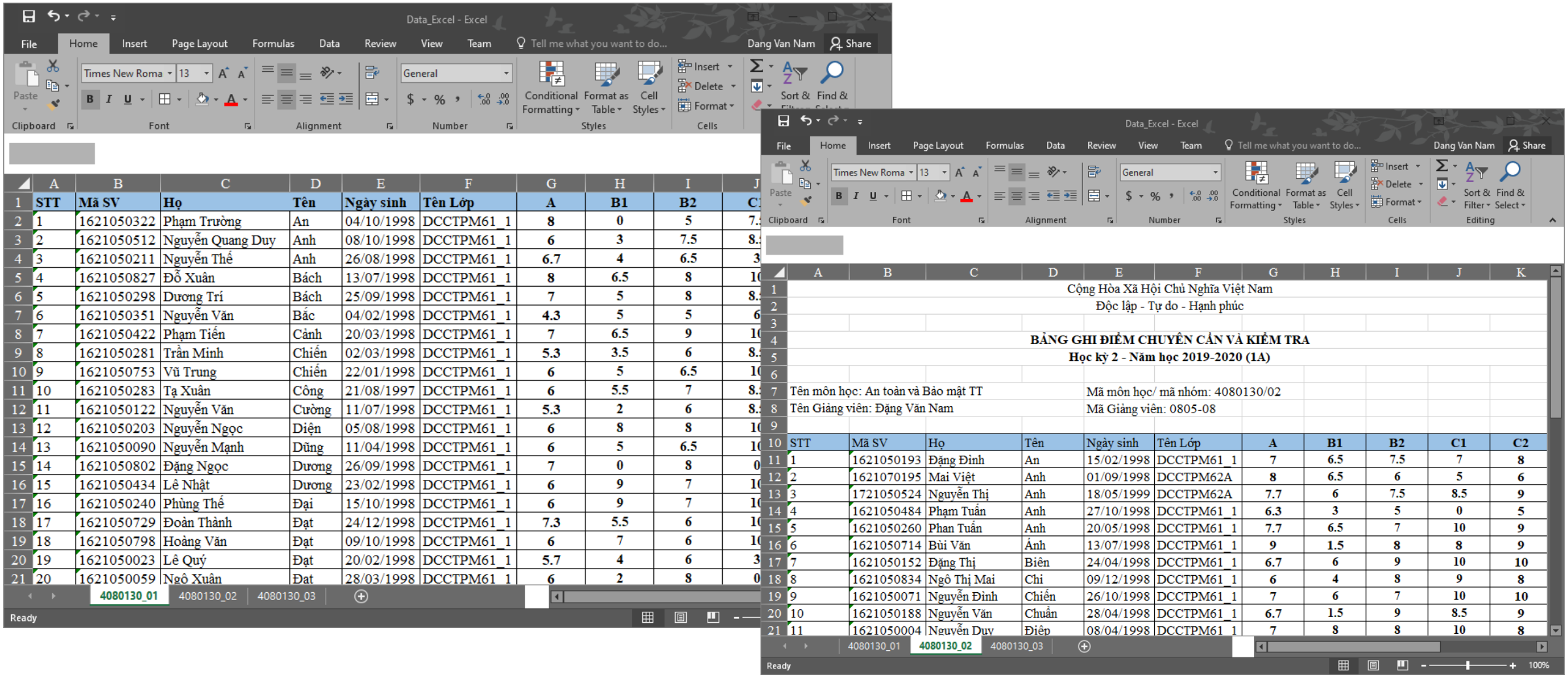
```
1 df_Temp.head()
```

	HaNoi	Vinh	DaNang	NhaTrang	HCM	CaMau
0	25.65	24.79	24.01	25.06	25.48	24.97
1	25.31	24.21	24.02	24.93	25.16	24.83
2	25.05	23.73	23.89	24.79	24.80	24.55
3	24.79	23.36	23.83	24.84	24.74	24.48
4	24.59	23.05	23.69	24.82	24.80	24.38

3.2 Đọc dữ liệu từ file Excel

3.2 Đọc file Excel

- File dữ liệu Excel demo gồm 3 sheet:



Data_Excel - Excel

STT	Mã SV	Họ	Tên	Ngày sinh	Tên Lớp	A	B1	B2	C1
1	1621050322	Phạm Trường	An	04/10/1998	DCCTPM61_1	8	0	5	7.5
2	1621050512	Nguyễn Quang Duy	Anh	08/10/1998	DCCTPM61_1	6	3	7.5	8.5
3	1621050211	Nguyễn Thế	Anh	26/08/1998	DCCTPM61_1	6.7	4	6.5	3
4	1621050827	Đỗ Xuân	Bách	13/07/1998	DCCTPM61_1	8	6.5	8	10
5	1621050298	Dương Trí	Bách	25/09/1998	DCCTPM61_1	7	5	8	8.5
6	1621050351	Nguyễn Văn	Bắc	04/02/1998	DCCTPM61_1	4.3	5	5	6
7	1621050422	Phạm Tiến	Cảnh	20/03/1998	DCCTPM61_1	7	6.5	9	10
8	1621050281	Trần Minh	Chiến	02/03/1998	DCCTPM61_1	5.3	3.5	6	8.5
9	1621050753	Vũ Trung	Chiến	22/01/1998	DCCTPM61_1	6	5	6.5	10
10	1621050283	Tạ Xuân	Công	21/08/1997	DCCTPM61_1	6	5.5	7	8.5
11	1621050122	Nguyễn Văn	Cường	11/07/1998	DCCTPM61_1	5.3	2	6	8.5
12	1621050203	Nguyễn Ngọc	Diện	05/08/1998	DCCTPM61_1	6	8	8	10
13	1621050090	Nguyễn Mạnh	Dũng	11/04/1998	DCCTPM61_1	6	5	6.5	10
14	1621050802	Đặng Ngọc	Dương	26/09/1998	DCCTPM61_1	7	0	8	0
15	1621050434	Lê Nhật	Dương	23/02/1998	DCCTPM61_1	6	9	7	10
16	1621050240	Phùng Thế	Đại	15/10/1998	DCCTPM61_1	6	9	7	10
17	1621050729	Đoàn Thành	Đạt	24/12/1998	DCCTPM61_1	7.3	5.5	6	10
18	1621050798	Hoàng Văn	Đạt	09/10/1998	DCCTPM61_1	6	7	6	10
19	1621050023	Lê Quý	Đạt	20/02/1998	DCCTPM61_1	5.7	4	6	3
20	1621050059	Ngô Xuân	Đạt	28/03/1998	DCCTPM61_1	6	2	8	0

4080130_01 | 4080130_02 | 4080130_03

Ready

Data_Excel - Excel

Cộng Hòa Xã Hội Chủ Nghĩa Việt Nam

Độc lập - Tự do - Hạnh phúc

BẢNG GHI ĐIỂM CHUYÊN CẦN VÀ KIỂM TRA

Học kỳ 2 - Năm học 2019-2020 (1A)

Tên môn học: An toàn và Bảo mật TT — Mã môn học/ mã nhóm: 4080130/02

Tên Giảng viên: Đặng Văn Nam — Mã Giảng viên: 0805-08

STT	Mã SV	Họ	Tên	Ngày sinh	Tên Lớp	A	B1	B2	C1	C2
1	1621050193	Đặng Đình	An	15/02/1998	DCCTPM61_1	7	6.5	7.5	7	8
2	1621070195	Mai Việt	Anh	01/09/1998	DCCTPM62A	8	6.5	6	5	6
3	1721050524	Nguyễn Thị	Anh	18/05/1999	DCCTPM62A	7.7	6	7.5	8.5	9
4	1621050484	Phạm Tuấn	Anh	27/10/1998	DCCTPM61_1	6.3	3	5	0	5
5	1621050260	Phan Tuấn	Anh	20/05/1998	DCCTPM61_1	7.7	6.5	7	10	9
6	1621050714	Bùi Văn	Ánh	13/07/1998	DCCTPM61_1	9	1.5	8	8	9
7	1621050152	Đặng Thị	Biên	24/04/1998	DCCTPM61_1	6.7	6	9	10	10
8	1621050834	Ngô Thị Mai	Chi	09/12/1998	DCCTPM61_1	6	4	8	9	8
9	1621050071	Nguyễn Đình	Chiến	26/10/1998	DCCTPM61_1	7	6	7	10	10
10	1621050188	Nguyễn Văn	Chuẩn	28/04/1998	DCCTPM61_1	6.7	1.5	9	8.5	9
11	1621050004	Nguyễn Duy	Điệp	08/04/1998	DCCTPM61_1	7	8	8	10	8

4080130_01 | 4080130_02 | 4080130_03

Ready

3.2 Đọc file Excel

- Sử dụng phương thức **pd.read_excel()** để đọc dữ liệu từ file excel.
  - Lưu ý 2 tham số sheetname=‘’ xác định sheet muốn đọc dữ liệu (Mặc định là sheet đầu tiên)

```
1  import pandas as pd
2  path_excel = 'Data_Excersice\Data_Excel.xlsx'
3  #Đọc dữ liệu từ file excel
4  data_ex = pd.read_excel(path_excel)
5  data_ex.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 66 entries, 0 to 65
Data columns (total 11 columns):
 #   Column     Non-Null Count  Dtype
---  ------     --------------  -----
 0   STT        66 non-null     int64
 1   Mã SV      66 non-null     int64
 2   Họ         66 non-null     object
 3   Tên        66 non-null     object
 4   Ngày sinh  66 non-null     object
 5   Tên Lớp    66 non-null     object
 6   A          66 non-null     float64
 7   B1         66 non-null     float64
 8   B2         66 non-null     float64
 9   C1         66 non-null     float64
 10  C2         66 non-null     float64
dtypes: float64(5), int64(2), object(4)
memory usage: 5.8+ KB
```

```
1  #Hiển thị 5 dòng dữ liệu đầu tiên
2  data_ex.head()
```

	STT	Mã SV	Họ	Tên	Ngày sinh	Tên Lớp	A	B1	B2	C1	C2
0	1	1621050322	Phạm Trường	An	04/10/1998	DCCTPM61_1	8.0	0.0	5.0	7.5	8.0
1	2	1621050512	Nguyễn Quang Duy	Anh	08/10/1998	DCCTPM61_1	6.0	3.0	7.5	8.5	9.0
2	3	1621050211	Nguyễn Thế	Anh	26/08/1998	DCCTPM61_1	6.7	4.0	6.5	3.0	5.0
3	4	1621050827	Đỗ Xuân	Bách	13/07/1998	DCCTPM61_1	8.0	6.5	8.0	10.0	9.0
4	5	1621050298	Dương Trí	Bách	25/09/1998	DCCTPM61_1	7.0	5.0	8.0	8.5	9.0

3.2 Đọc file Excel

- Một vài tham số quan trọng trong phương thức **pd.read_excel()** để đọc dữ liệu từ file excel.

Argument	Description
io	A string containing the pathname of the given Excel file.
sheet_name	The Excel sheet name, or sheet number, of the data you want to import. The sheet number can be an integer where 0 is the first sheet, 1 is the second, etc. If a list of sheet names/numbers are given, then the output will be a dictionary of DataFrames. The default is to read all the sheets and output a dictionary of DataFrames.
header	Row number to use for the list of column labels. The default is 0, indicating that the first row is assumed to contain the column labels. If the data does not have a row of column labels, None should be used.
names	A separate Python list input of column names. This option is None by default. This option is the equivalent of assigning a list of column names to the columns attribute of the output DataFrame.
index_col	Specifies which column should be used for row indices. The default option is None, meaning that all columns are included in the data, and a range of numbers is used as the row indices.
usecols	An integer, list of integers, or string that specifies the columns to be imported into the DataFrame. The default is to import all columns. If a string is given, then Pandas uses the standard Excel format to select columns (e.g. "A:C,F,G" will import columns A, B, C, F, and G).
skiprows	The number of rows to skip at the top of the Excel sheet. Default is 0. This option is useful for skipping rows in Excel that contain explanatory information about the data below it.

https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.read_excel.html

3.2 Đọc file Excel

- Sử dụng phương thức **pd.read_excel()** với một số tham số cơ bản.

```
#Ví dụ:
#Đọc dữ liệu tại sheet đầu tiên,
#Chỉ lấy dữ liệu cột Mã SV và các cột điểm
#Thiết lập cột đầu tiên làm index
data_ex1 = pd.read_excel(path_excel,
                         sheet_name=0,
                         usecols=[1,6,7,8,9,10],
                         index_col=0)
data_ex1.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 66 entries, 1621050322 to 1621050013
Data columns (total 5 columns):
 #   Column  Non-Null Count  Dtype
---  ------  --------------  -----
 0   A       66 non-null     float64
 1   B1      66 non-null     float64
 2   B2      66 non-null     float64
 3   C1      66 non-null     float64
 4   C2      66 non-null     float64
dtypes: float64(5)
memory usage: 3.1 KB
```

```
#Hiển thị 5 dòng dữ liệu đầu tiên
data_ex1.head()
```

Mã SV	A	B1	B2	C1	C2
1621050322	8.0	0.0	5.0	7.5	8.0
1621050512	6.0	3.0	7.5	8.5	9.0
1621050211	6.7	4.0	6.5	3.0	5.0
1621050827	8.0	6.5	8.0	10.0	9.0
1621050298	7.0	5.0	8.0	8.5	9.0

3.2 Đọc file Excel

- Sử dụng phương thức **pd.read_excel()** với một số tham số cơ bản.
  - Đọc dữ liệu sheet 2 [‘4080130_02’], từ dòng 9.

```
#Ví dụ 3:
#Đọc dữ liệu tại sheet 2,từ dòng 9
data_ex3 = pd.read_excel(path_excel,
                         sheet_name='4080130_02',
                         skiprows=9)
data_ex3.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 39 entries, 0 to 38
Data columns (total 11 columns):
 #   Column     Non-Null Count  Dtype
---  ------     --------------  -----
 0   STT        39 non-null     int64
 1   Mã SV      39 non-null     int64
 2   Họ         39 non-null     object
 3   Tên        39 non-null     object
 4   Ngày sinh  39 non-null     object
 5   Tên Lớp    39 non-null     object
 6   A          39 non-null     float64
 7   B1         39 non-null     float64
 8   B2         39 non-null     float64
 9   C1         39 non-null     float64
 10  C2         39 non-null     float64
dtypes: float64(5), int64(2), object(4)
memory usage: 3.5+ KB
```

```
#Hiển thị 5 dòng dữ liệu đầu tiên
data_ex3.head()
```

	STT	Mã SV	Họ	Tên	Ngày sinh	Tên Lớp	A	B1	B2	C1	C2
0	1	1621050193	Đặng Đình	An	15/02/1998	DCCTPM61_1	7.0	6.5	7.5	7.0	8.0
1	2	1621070195	Mai Việt	Anh	01/09/1998	DCCTPM62A	8.0	6.5	6.0	5.0	6.0
2	3	1721050524	Nguyễn Thị	Anh	18/05/1999	DCCTPM62A	7.7	6.0	7.5	8.5	9.0
3	4	1621050484	Phạm Tuấn	Anh	27/10/1998	DCCTPM61_1	6.3	3.0	5.0	0.0	5.0
4	5	1621050260	Phan Tuấn	Anh	20/05/1998	DCCTPM61_1	7.7	6.5	7.0	10.0	9.0

3.2 Đọc file Excel

- Sử dụng phương thức **pd.read_excel()** với một số tham số cơ bản.
  - Đọc dữ liệu sheet 3 [‘4080130_03’], không có dòng header

```
#Ví dụ 4
#Đọc dữ liệu từ sheet: '4080130_03'
#Dữ liệu không chứa dòng header
data_ex4 = pd.read_excel(path_excel,
                         sheet_name='4080130_03',
                         header=None)
data_ex4.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 39 entries, 0 to 38
Data columns (total 11 columns):
 #   Column  Non-Null Count  Dtype
---  ------  --------------  -----
 0   0       39 non-null     int64
 1   1       39 non-null     int64
 2   2       39 non-null     object
 3   3       39 non-null     object
 4   4       39 non-null     object
 5   5       39 non-null     object
 6   6       39 non-null     float64
 7   7       39 non-null     float64
 8   8       39 non-null     float64
 9   9       39 non-null     float64
 10  10      39 non-null     float64
dtypes: float64(5), int64(2), object(4)
memory usage: 3.5+ KB
```

```
#Hiển thị 5 dòng dữ liệu đầu tiên
data_ex4.head()
```

	0	1	2	3	4	5	6	7	8	9	10
0	1	1621050041	Đào Tuấn	Anh	22/10/1998	DCCTPM61_1	6.7	9.0	5.5	8.5	8.0
1	2	1621050262	Vũ Thị Lan	Anh	26/09/1998	DCCTPM61_1	6.7	7.0	9.0	8.5	6.0
2	3	1621050083	Trịnh Như	Bình	06/04/1998	DCCTPM61_1	7.3	8.5	9.5	10.0	9.0
3	4	1621050113	Trần Văn	Cương	19/06/1998	DCCTPM61_1	5.7	5.0	6.0	10.0	5.0
4	5	1621050384	Nguyễn Sỹ	Dũng	02/10/1998	DCCTPM61_1	7.0	0.0	7.5	8.5	9.0

3.2 Đọc file Excel

- Sử dụng phương thức **pd.read_excel()** với một số tham số cơ bản.
  - Đọc dữ liệu sheet 3 [‘4080130_03’],
  - Không có dòng header
  - Chỉ lấy dữ liệu cột 1,6,7,8,910
  - Đặt tên cho các cột lần lượt là [‘Mã SV’, ‘A’, ‘B1’,’B2’,’C1’,’C2’]
  - Thiết lập cột đầu tiên làm Index

```
5  data_ex41 = pd.read_excel(path_excel,
6                            sheet_name='4080130_03',
7                            header=None,
8                            usecols=[1,6,7,8,9,10],
9                            names=['Mã SV','A','B1','B2','C1','C2'],
10                           index_col=0)
11 data_ex41.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 39 entries, 1621050041 to 1621050034
Data columns (total 5 columns):
 #   Column  Non-Null Count  Dtype
---  ------  --------------  -----
 0   A       39 non-null     float64
 1   B1      39 non-null     float64
 2   B2      39 non-null     float64
 3   C1      39 non-null     float64
 4   C2      39 non-null     float64
dtypes: float64(5)
memory usage: 1.8 KB
```

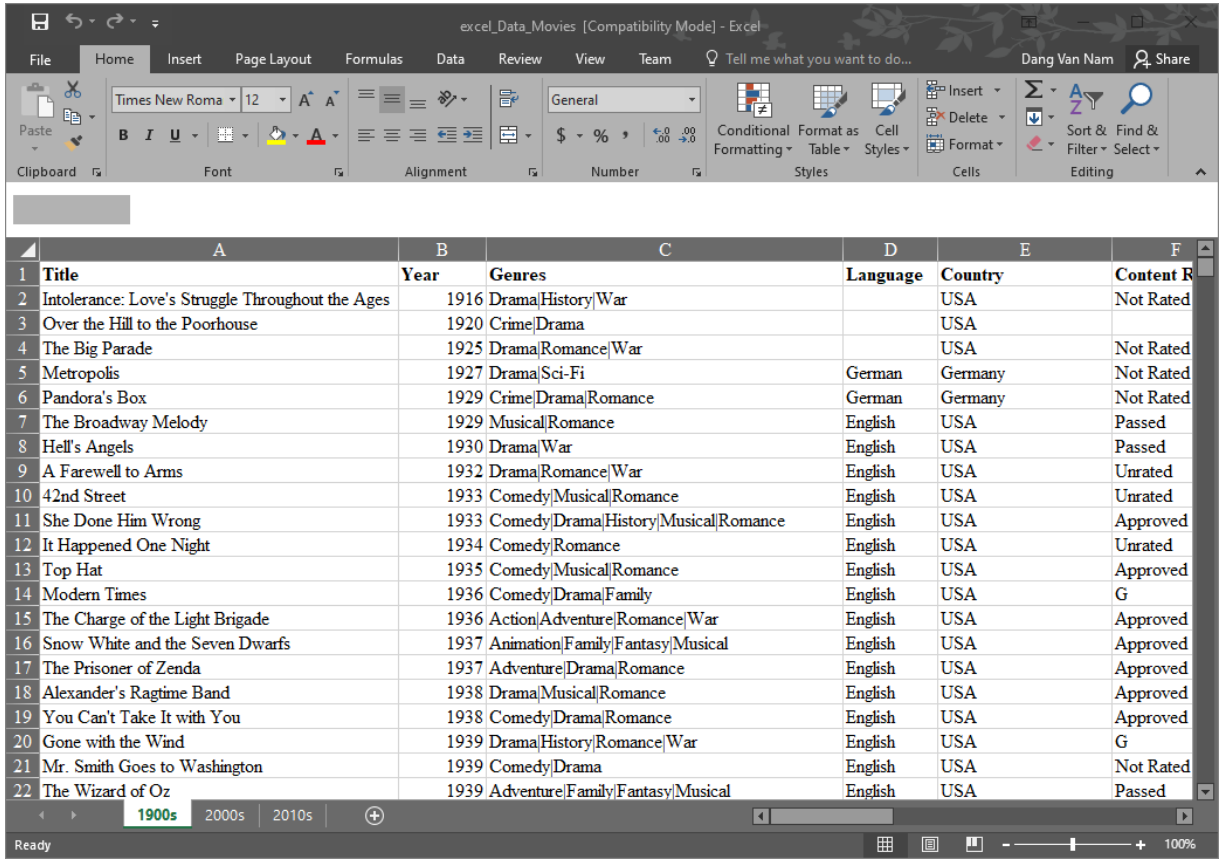
```
1 #Hiển thị 5 dòng dữ liệu đầu tiên
2 data_ex41.head()
```

Mã SV	A	B1	B2	C1	C2
1621050041	6.7	9.0	5.5	8.5	8.0
1621050262	6.7	7.0	9.0	8.5	6.0
1621050083	7.3	8.5	9.5	10.0	9.0
1621050113	5.7	5.0	6.0	10.0	5.0
1621050384	7.0	0.0	7.5	8.5	9.0

Thực hành 2

Thực hành 2

Yêu cầu: Học viên đọc dữ liệu dạng excel lưu trong file excel_Data_Movies.xls theo từng sheet, Mỗi sheet chỉ lấy các cột dữ liệu: Title -Year - Genres - Country - Director - User Votes - IMDB Score. Sau đó tổng hợp thành một Dataframe và lưu ra file .CSV



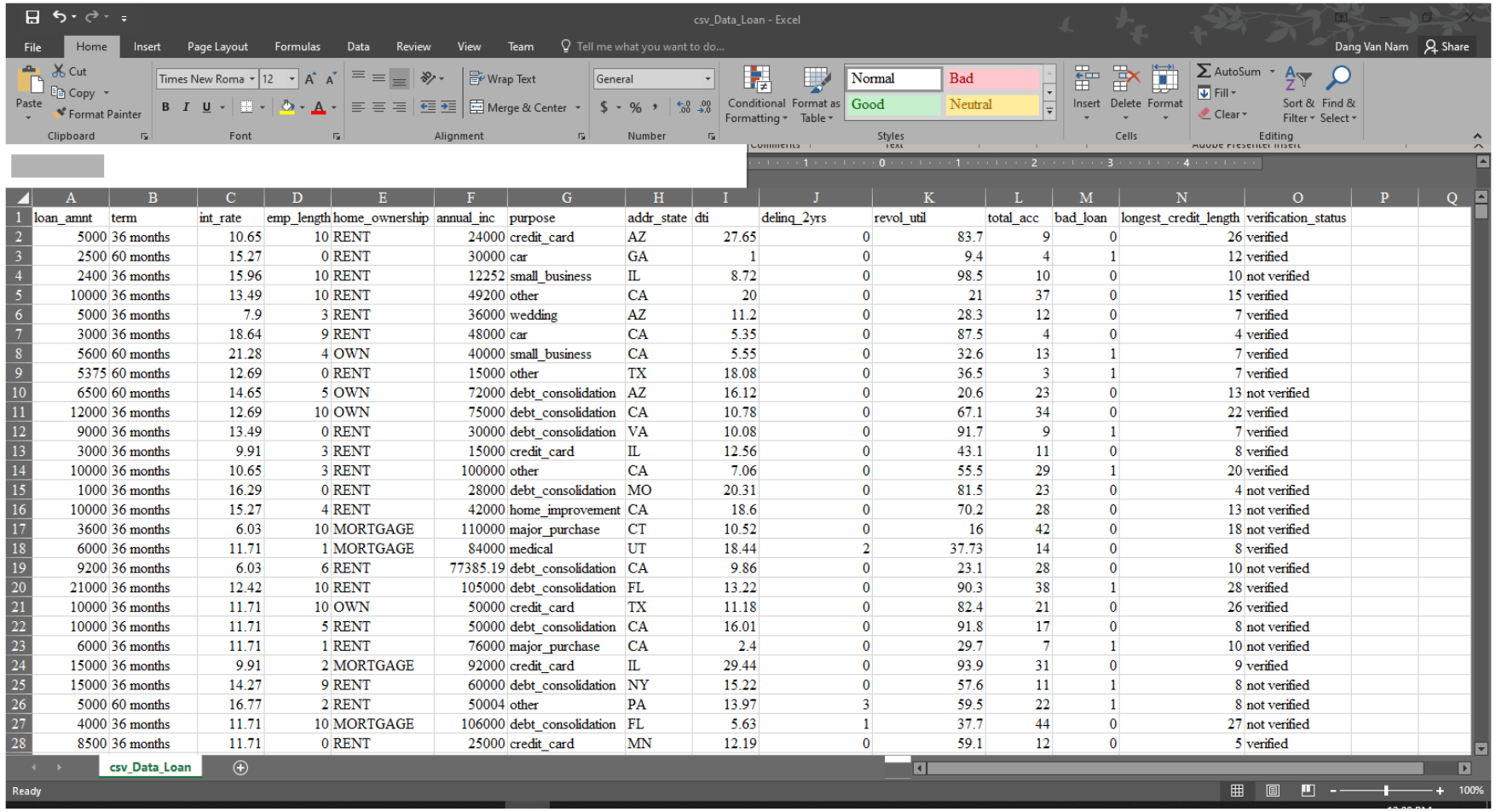
excel_Data_Movies [Compatibility Mode] - Excel

	A	B	C	D	E	F
1	Title	Year	Genres	Language	Country	Content R
2	Intolerance: Love's Struggle Throughout the Ages	1916	Drama|History|War		USA	Not Rated
3	Over the Hill to the Poorhouse	1920	Crime|Drama		USA	
4	The Big Parade	1925	Drama|Romance|War		USA	Not Rated
5	Metropolis	1927	Drama|Sci-Fi	German	Germany	Not Rated
6	Pandora's Box	1929	Crime|Drama|Romance	German	Germany	Not Rated
7	The Broadway Melody	1929	Musical|Romance	English	USA	Passed
8	Hell's Angels	1930	Drama|War	English	USA	Passed
9	A Farewell to Arms	1932	Drama|Romance|War	English	USA	Unrated
10	42nd Street	1933	Comedy|Musical|Romance	English	USA	Unrated
11	She Done Him Wrong	1933	Comedy|Drama|History|Musical|Romance	English	USA	Approved
12	It Happened One Night	1934	Comedy|Romance	English	USA	Unrated
13	Top Hat	1935	Comedy|Musical|Romance	English	USA	Approved
14	Modern Times	1936	Comedy|Drama|Family	English	USA	G
15	The Charge of the Light Brigade	1936	Action|Adventure|Romance|War	English	USA	Approved
16	Snow White and the Seven Dwarfs	1937	Animation|Family|Fantasy|Musical	English	USA	Approved
17	The Prisoner of Zenda	1937	Adventure|Drama|Romance	English	USA	Approved
18	Alexander's Ragtime Band	1938	Drama|Musical|Romance	English	USA	Approved
19	You Can't Take It with You	1938	Comedy|Drama|Romance	English	USA	Approved
20	Gone with the Wind	1939	Drama|History|Romance|War	English	USA	G
21	Mr. Smith Goes to Washington	1939	Comedy|Drama	English	USA	Not Rated
22	The Wizard of Oz	1939	Adventure|Family|Fantasy|Musical	English	USA	Passed

4. Quan sát và truy cập dữ liệu trong DataFrame

4.1 Quan sát dữ liệu

- Đọc file dữ liệu mẫu: **csv_Data_loan**
- Đây là file dữ liệu cho biết thông tin về các khoản vay cho các mục đích khác nhau của người dùng Mỹ.



csv_Data_Loan - Excel

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
1	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		

4.1 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

4.1 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

4.2 Truy cập dữ liệu

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

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

	addr_state
0	AZ
1	GA
2	IL
3	CA
4	AZ

```
#Truy xuất dữ liệu theo cột
#Chỉ lấy dữ liệu của 3 cột: loan_amnt, int_rate, purpose
df_loan1 = df_loan[['loan_amnt','int_rate','purpose']]
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

4.2 Truy cập dữ liệu

- **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)**

```
#Sử dụng .iloc truy xuất dữ liệu như với Numpy
#Truy xuất 10 dòng dữ liệu từ [10 --> 20) tất cả các cột
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	l
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	

4.2 Truy cập dữ liệu

- **df.loc[[name_index],[name_col]]:** Truy cập tới dữ liệu của hàng và cột qua **name_index, name_column**

```
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

4.2 Truy cập dữ liệu

Type	Notes
`df[val]`	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)
`df.loc[val]`	Selects single row or subset of rows from the DataFrame by label
`df.loc[:, val]`	Selects single column or subset of columns by label
`df.loc[val1, val2]`	Select both rows and columns by label
`df.iloc[where]`	Selects single row or subset of rows from the DataFrame by integer position
`df.iloc[:, where]`	Selects single column or subset of columns by integer position
`df.iloc[where_i, where_j]`	Select both rows and columns by integer position
`df.at[label_i, label_j]`	Select a single scalar value by row and column label
`df.iat[i, j]`	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

5. Replacing Values, Rename columns

5.1 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

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

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

5.1 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

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

	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

02

```
#Thay thế nhiều giá trị trong DataFrame
#RENT --> MORTGAGE
#car --> small_business
df_new.replace({'RENT':'MORTGAGE',
                '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

5.1 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':'Virgrinia'}
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

5.2 Rename Columns

- **df.rename(): thay thổi tên cột trong DataFrame**

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

```
#Đổi tên cột sang viết hoa
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 3

Thực hành 3

Mô tả file dữ liệu: Data_Patient.csv

- File dữ liệu chứa thông tin của 300 bệnh nhân bị chứng đau ngực
- 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

Data_Patient

Ready 100%

Thực hành 3

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 | 6: Khiếm khuyết cố định | 7: Kiếm 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)

Thực hành 3

Yêu cầu 3.1:

- Đọc dữ liệu từ file Data_Patient.csv vào biến kiểu dataframe: df_patient với cột id 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 Dataframe như sau:

  - 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

Thực hành 3

Yêu cầu 3.2:

- 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 3

Yêu cầu 3.3:

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

Yêu cầu 3.4:

- 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 3

Yêu cầu 3.5:

- 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**

id	Age	Gender	Type	Blood_pressure	Cholesterol	Heartbeat	Thalassemia	Result
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

6. Filter Data

6. Filter Data

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

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

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

01

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

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

02

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

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

03

6. Filter Data

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

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

	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

01

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

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

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

03

6. Filter Data

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

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

7. Tính toán đặc trưng thống kê trong DataFrame

7. Đặc trưng thống kê

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

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

```
Cân nặng lớn nhất: 160 (kg)
Cân nặng nhỏ nhất: 50 (kg)
```

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

```
Chiều cao trung bình: 169.944 (cm)
Trung vị: 170.5 (cm)
```

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

```
sdt của chiều cao: 16.37526067959376
sdt của cân nặng: 32.38260746964435
```

8. Xác định giá trị duy nhất (Unique)

8. 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.

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

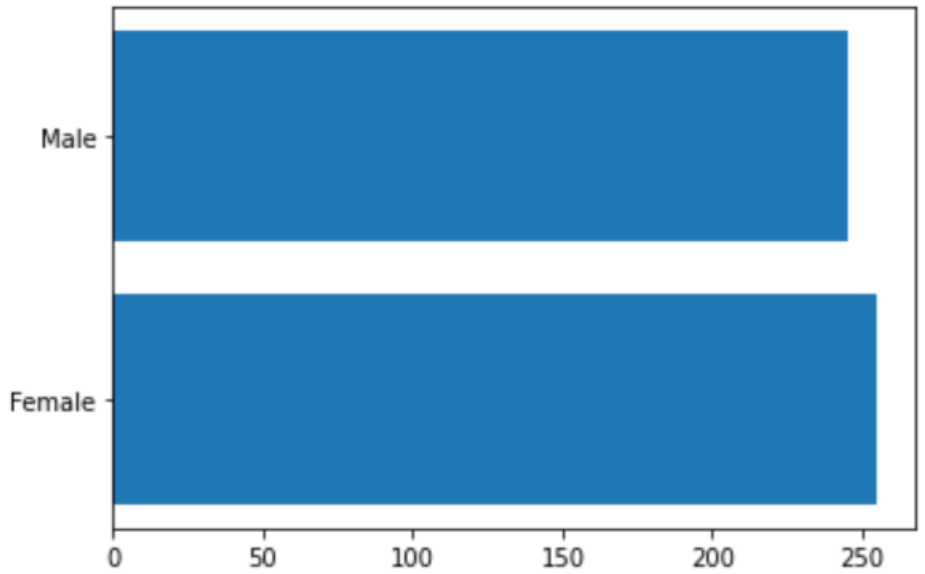
```
array(['Male', 'Female'], dtype=object)
```

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

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

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

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



Thực hành 4

Thực hành 4

Yêu cầu 4.1:

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

	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

Data_Patient

Ready 100%

- 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

Thực hành 4

Yêu cầu 4.2:

- 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
  - df_yes: danh sách những người bị bệnh đau tim

Yêu cầu 4.3:

- Lọc trong df_patient dưa 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.

Thực hành 4

Yêu cầu 4.4: 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

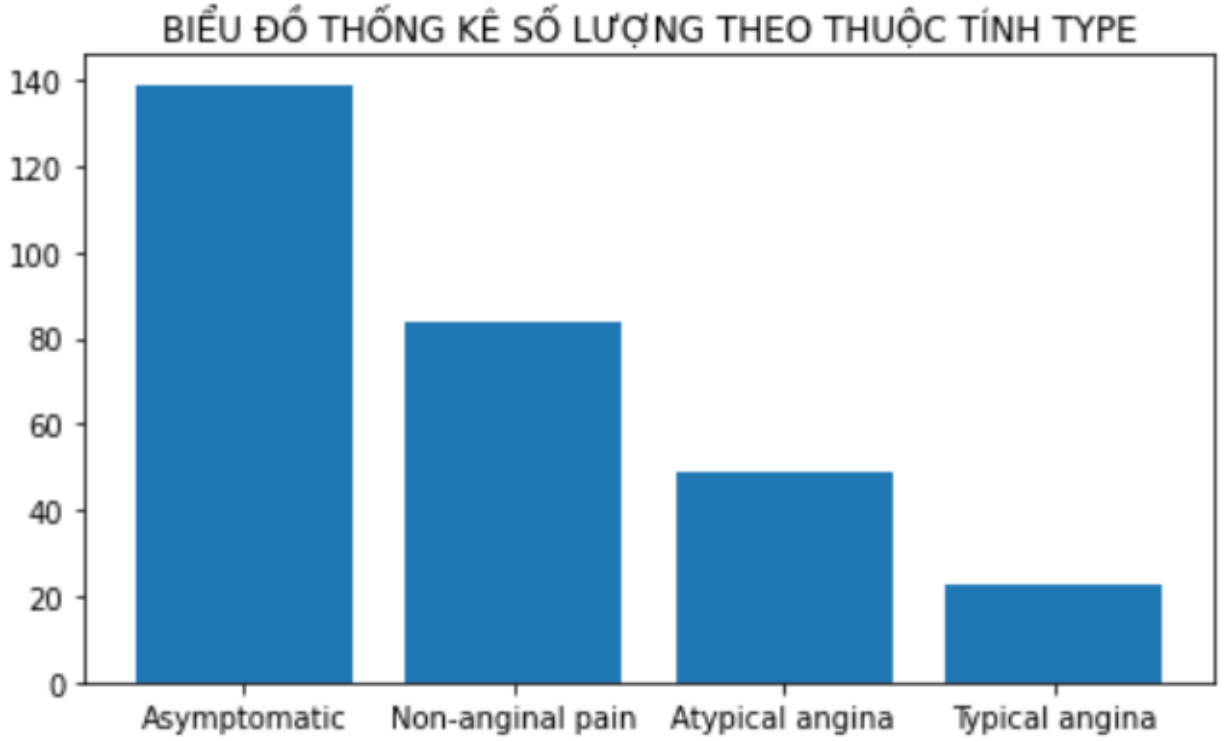
```
1. Chỉ số huyết áp:
Min:  94
Max:  200
Mean:  131.68666666666667
Median:  130.0
Std:  17.682497692285477

2. Chỉ số nhịp tim:
Min:  71
Max:  202
Mean:  149.56333333333333
Median:  152.5
Std:  22.818595118151098
```

Thực hành 4

<u>Yêu cầu 4.5:</u> 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
```



9. Phân tích dữ liệuTime series

(Tiếp cận từ bài toán thực tế)

Mô tả bài toán

Fremont Bridge

Trang web | Chỉ đường | Lưu

4,4 ★★★★½ 127 đánh giá trên Google

Cầu ở Portland, Oregon

Được dịch từ tiếng Anh - Cầu Fremont là cây cầu vòm bằng thép bắc qua sông Willamette nằm ở Portland, Oregon, Hoa Kỳ. Nó mang giao thông Interstate 405 và US 30 giữa trung tâm thành phố và Bắc Portland, nơi giao nhau với Xa lộ liên tiểu bang 5.
Wikipedia (tiếng Anh)

Xem mô tả gốc

Địa chỉ: Stadium Fwy, Portland, OR 97232, Hoa Kỳ

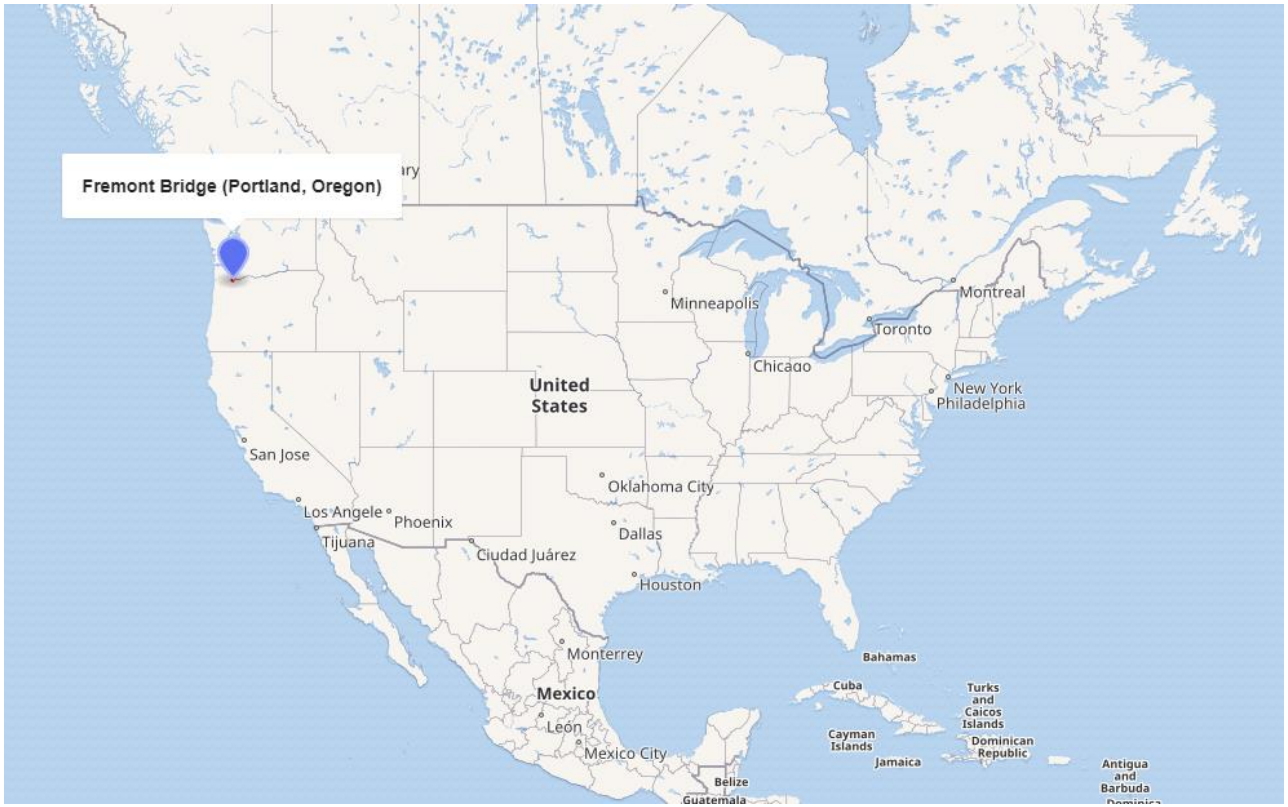
Chiều cao: 116 m

Bắt đầu xây dựng: 1968

Tổng chiều dài: 656 m

Khoảng hở bên dưới: 53 m

+ Người ta lắp đặt thiết bị để đếm số lượng xe đạp đi qua chiều phía đông và phía tây của cây cầu Fremont Bridge theo từng giờ.



Tập dữ liệu

- Tập dữ liệu là số lượng xe đạp đi qua cây cầu Fremont Bridge. Dữ liệu này được thu thập tự động thông qua các cảm biến ở 2 lối đi bộ ở phía đông và phía tây của cây cầu. Số lượng xe đạp được tổng hợp theo từng giờ.
- Tập dữ liệu lưu trữ trong file .csv, bao gồm 4 cột:
  - Date: Thời gian (ngày - giờ): 03/10/2012 00:00:00 (Kiểu thời gian) – 30/04/2025 23:00:00
  - West: Số xe đạp đi qua lối phía tây của cầu (Kiểu số nguyên)
  - East: Số xe đạp đi qua lối phía đông của cầu (Kiểu số nguyên)
  - Total: Tổng số xe đi theo cả 2 lối đông và tây (Kiểu số nguyên)

Date	West	East	Total
2012-10-03 01:00:00	4	6	10
2012-10-03 02:00:00	1	1	2
2012-10-03 03:00:00	2	3	5
2012-10-03 04:00:00	6	1	7
2012-10-03 05:00:00	21	10	31
2012-10-03 06:00:00	105	50	155

Mục tiêu

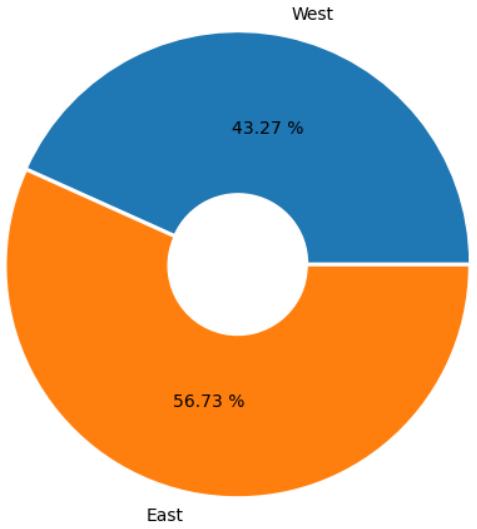
- Phân tích dữ liệu chuỗi thời gian (Time Series Data) sử dụng Pandas.
- Kết hợp với các biểu đồ để tìm ra được những Insight ẩn chứa trong tập dữ liệu.

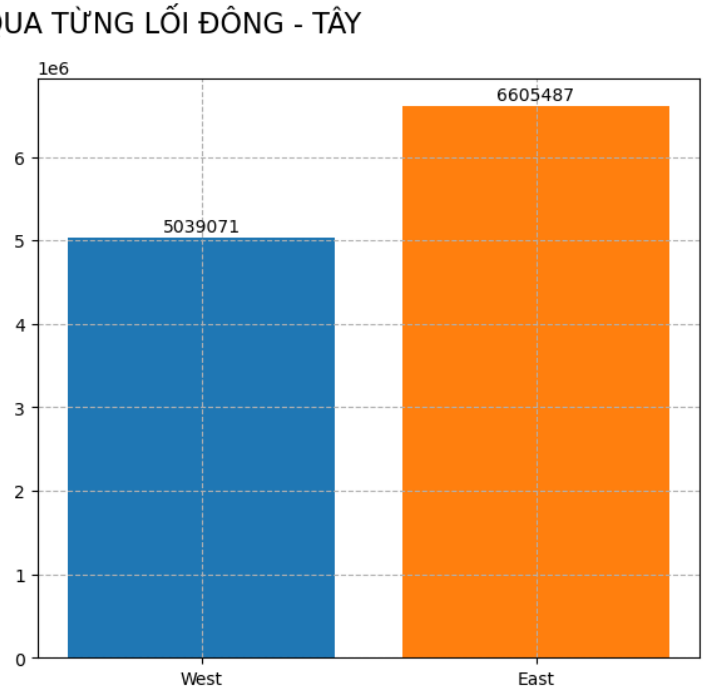
Kết Quả

Phát hiện 1:

- Trong khoảng thời gian từ 03/10/2012 đến 30/04/2025, Tổng số lượt người đi xe đạp qua cây cầu là: 11 644 658 lượt. Trung bình mỗi năm khoảng ~ 1 triệu lượt di chuyển; Trung bình có 106 lượt người đi xe đạp qua cầu mỗi giờ; 2550 lượt di chuyển mỗi ngày; --> Người dân khu vực này sử dụng phương tiện xe đạp để di chuyển là khá thường xuyên và phổ biến.
- Số lượng người đi xe đạp bên cầu phía đông - East (6 605 487 ~ 56.73%) cao hơn cầu phía Tây - West khoảng 13.5% (5 039 071 ~ 43.27%)

	West	East	Total
count	110220.000000	110220.000000	110220.000000
mean	45.718300	59.930022	105.648322
std	57.939276	82.576367	131.103362
min	0.000000	0.000000	0.000000
25%	5.000000	7.000000	13.000000
50%	26.000000	31.000000	59.000000
75%	64.000000	78.000000	145.000000
max	698.000000	850.000000	1097.000000



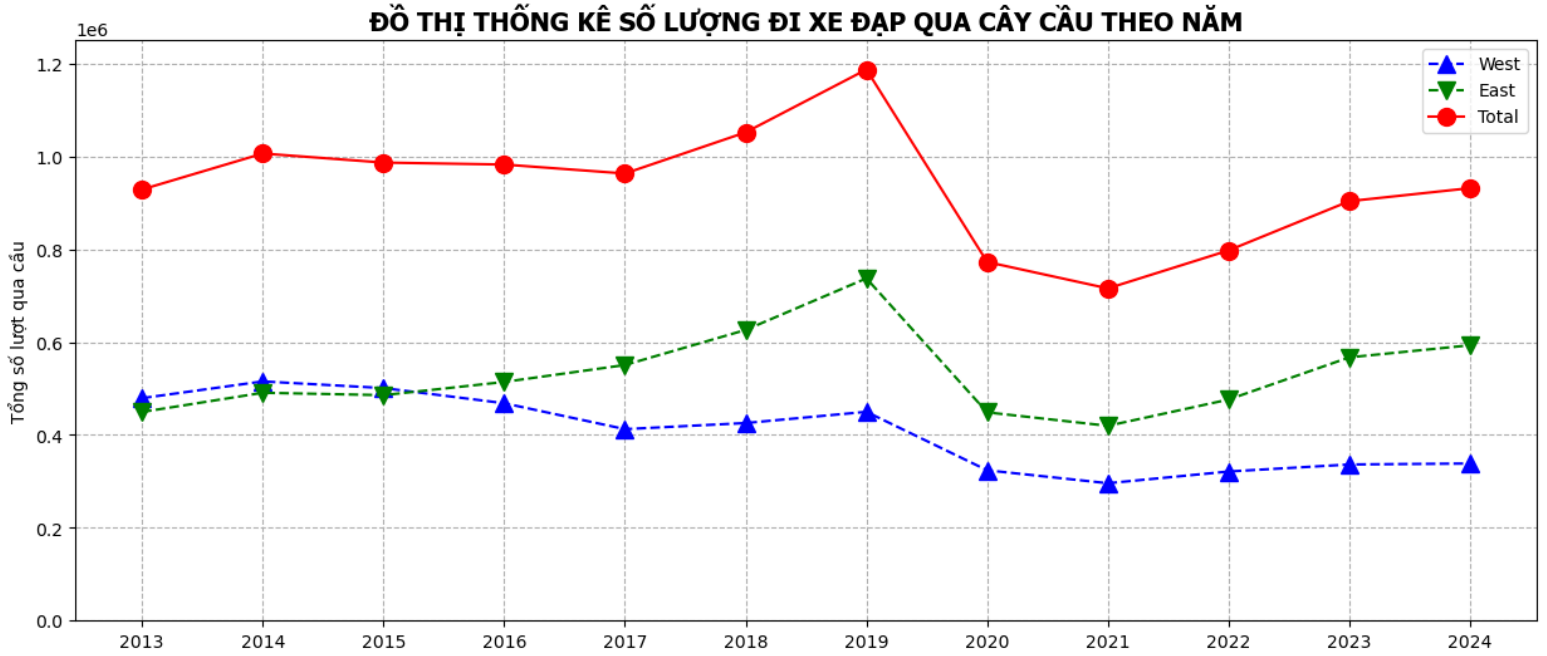


Chi tiết các bước thực hiện trong file jupyter notebook…

Kết Quả

Phát hiện 2:Từ biểu đồ thống kê tổng số xe đạp qua cầu theo năm ta thấy:

- Số lượng người đi xe đạp qua cầu Fremont có xu hướng tăng lên theo từng năm, những năm 2017, 2018, 2019 có sự gia tăng nhanh. Sau đó giảm mạnh vào năm 2020, 2021 và đang có xu hướng tăng trở lại từ năm 2022 --> Nguyên nhân?
- Giai đoạn từ năm 2013 đến năm 2015 lượng người đi xe đạp qua 2 phía là tương đương nhau. từ năm 2016 trở đi số lượt xe đạp qua lối đi phía đông nhiều hơn lối đi phía tây và cũng có xu hướng tăng nhanh trong những năm gần đây ---> Nguyên nhân?



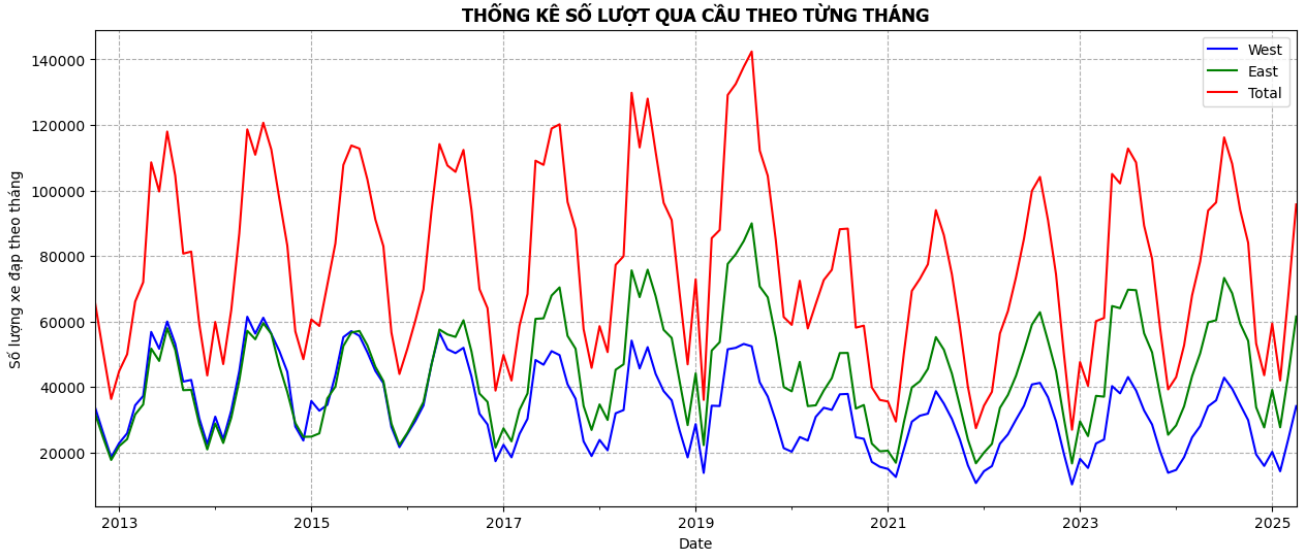
Chi tiết các bước thực hiện trong file jupyter notebook…

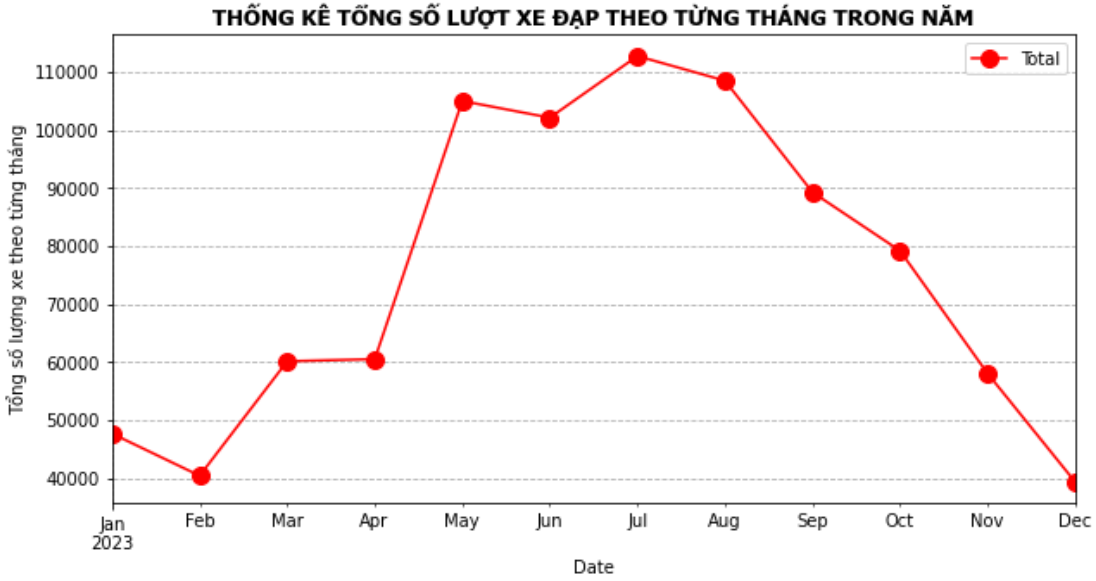
Kết Quả

Phát hiện 3: Từ biểu đồ thể hiện lượng xe đạp qua cầu theo tháng ta thấy:

- Số lượng người đi xe đạp qua cây cầu có tính thời vụ (Seasonal), lặp lại chu kỳ theo từng năm.
- Lượng người đi xe đạp nhiều hơn vào các tháng giữa năm (các tháng 5, 6, 7, 8 và 9 là các tháng có lượt người qua cây cầu lớn nhất), đầu năm (tháng 1, 2 ,3 ,4) và cuối năm (tháng 10, 11, 12) số lượng người qua cầu giảm đi đáng kể.
- Sự chênh nhau giữa tháng thấp nhất và cao nhất ~ >3 lần.

--> Dữ liệu có tính xu hướng (tăng dần qua các năm) và tính thời vụ (số lượng nhiều hơn vào các tháng mùa hè và ít hơn vào các tháng mùa đông và lặp lại)



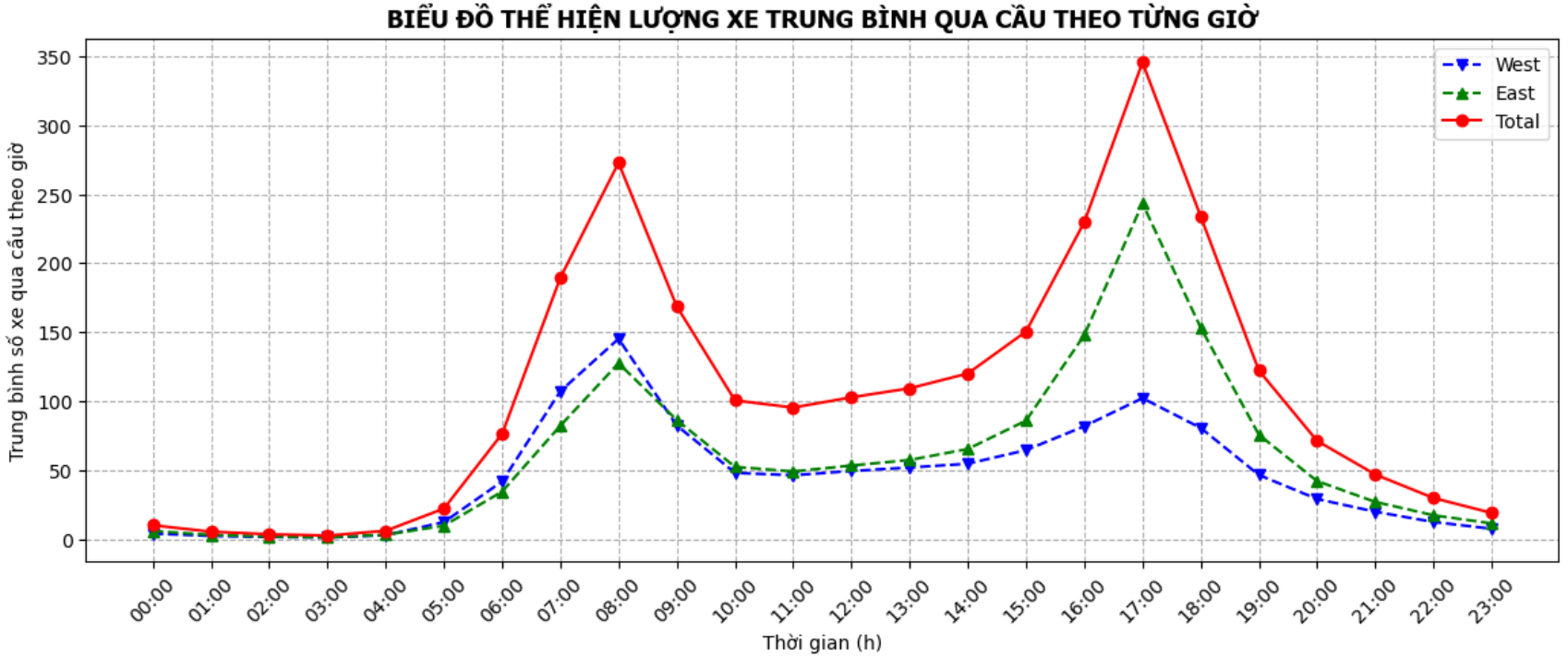


Chi tiết các bước thực hiện trong file jupyter notebook…

Kết Quả

Phát hiện 4: Từ biểu đồ thể hiện lượng xe đạp qua cầu trung bình theo giờ ta thấy:

- Lượng người đi xe đạp qua cầu chủ yếu tập trung vào khung thời gian từ 6:00 am đến 19:00pm. Cao điểm vào 7,8,9h buổi sáng | 16, 17, 18h buổi chiều
- Lượng người đi nhiều nhất vào thời điểm 8h sáng - 17h chiều.
- Thời điểm sáng sớm (0-->6h am) và tối muộn (20-->23h pm)lượng người di chuyển qua 2 phía tương đương nhau; thời điểm ban ngày từ 7am - 19pm số lượt qua cây cầu phía Đông cao hơn cây cầu phía tây, đặc biệt là khung giờ từ 15-->18h pm

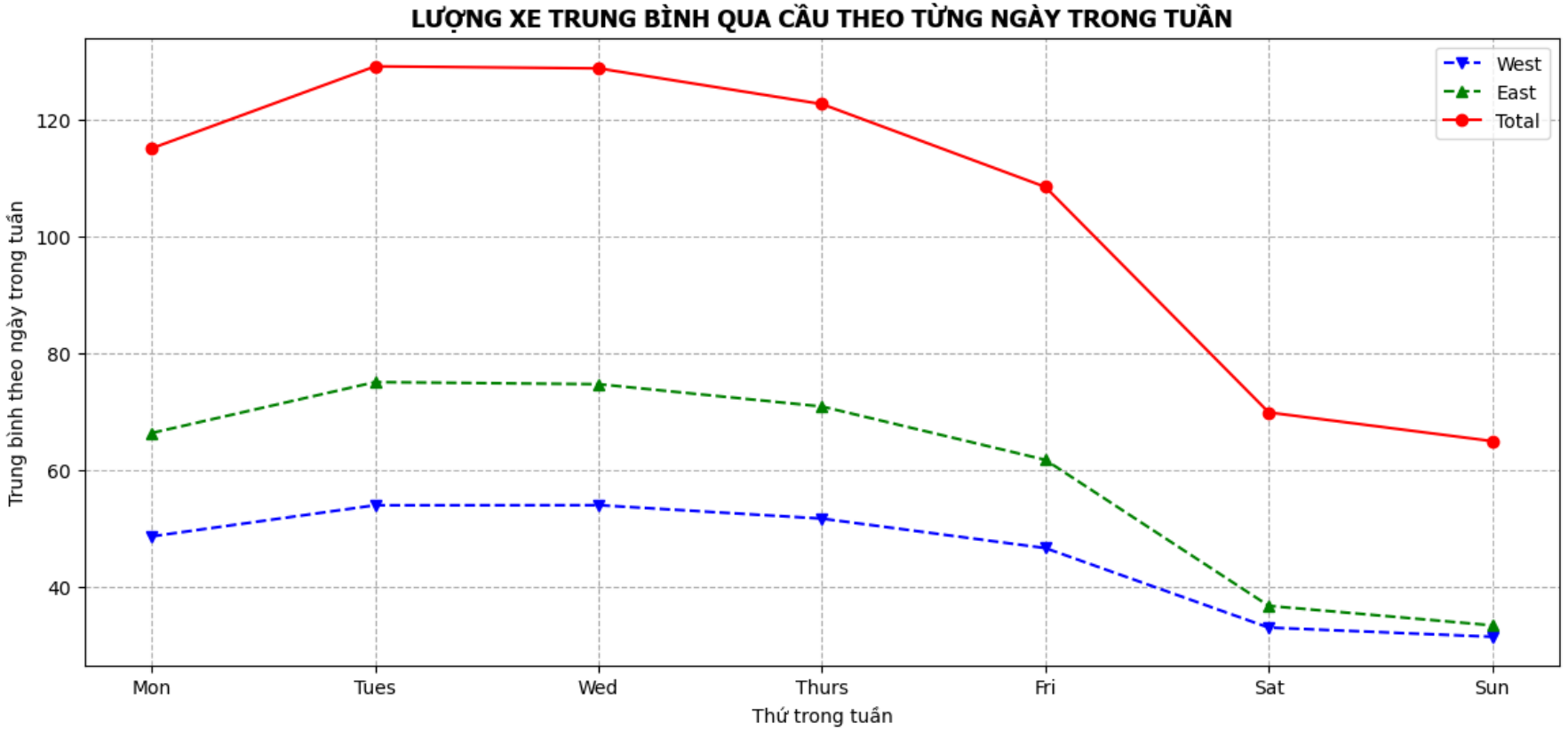


Chi tiết các bước thực hiện trong file jupyter notebook…

Kết Quả

Phát hiện 5: Từ biểu đồ thể hiện lượng xe đạp qua cầu trung bình theo ngày trong tuần:

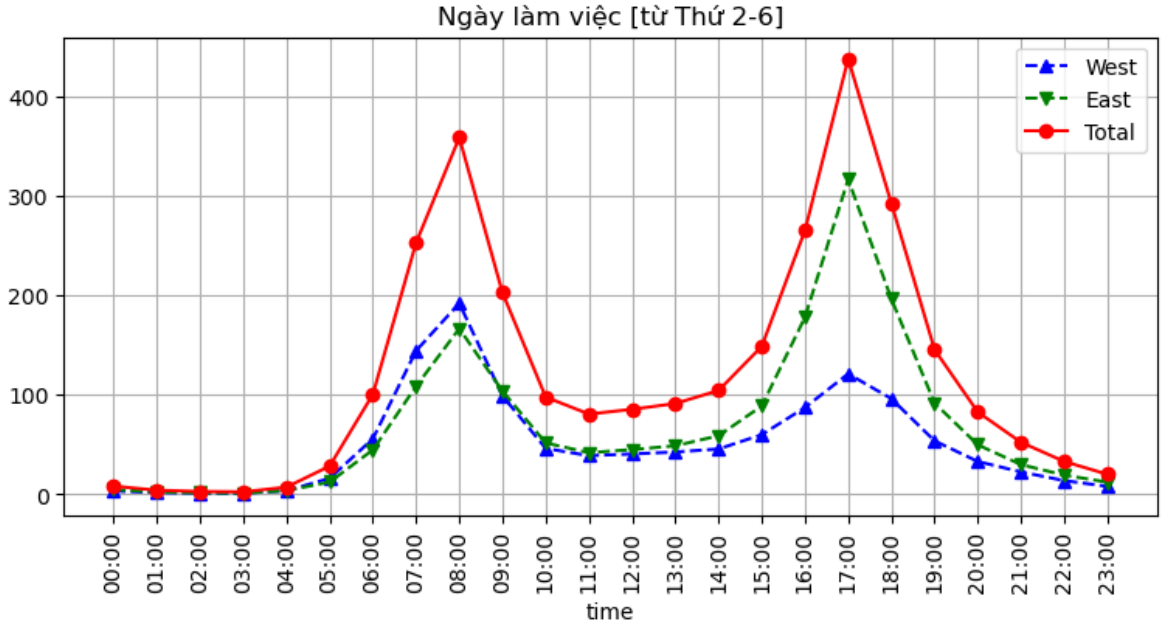
- Lượng người đi xe đạp qua cầu chủ yếu vào các ngày làm việc trong tuần [thứ 2 --> thứ 6]; Cuối tuần [Thứ 7, CN] lượng người đi qua cầu giảm đi đáng kể. Lượng người đi qua cầu ngày làm việc gấp đôi ngày cuối tuần.

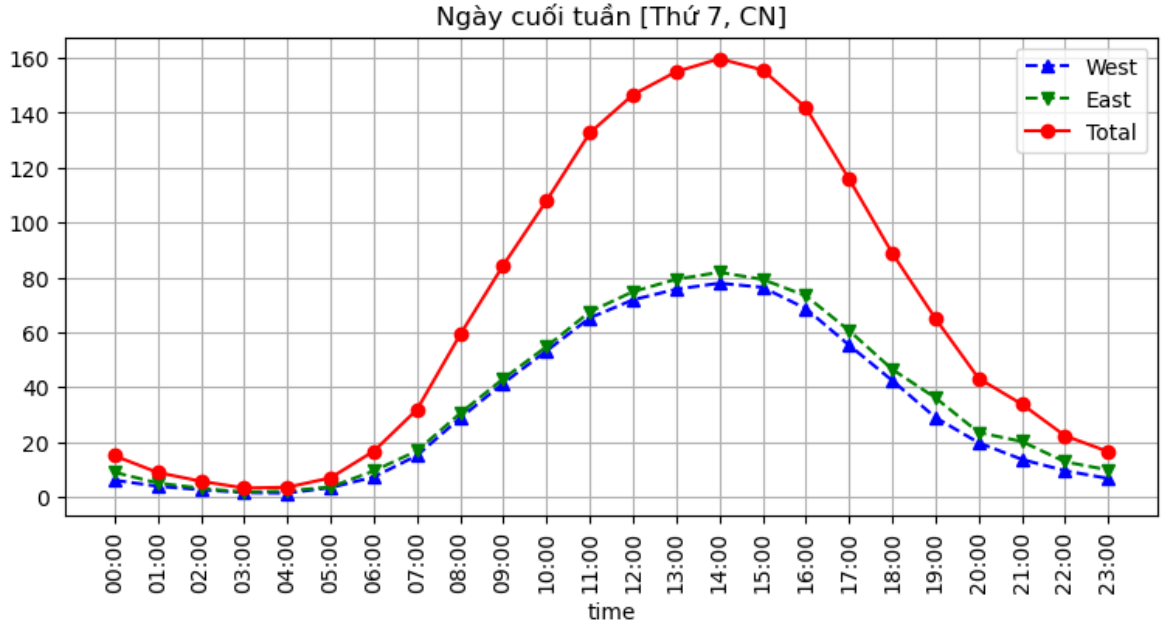


Chi tiết các bước thực hiện trong file jupyter notebook…

Kết Quả

- Lượng người đi xe đạp qua cầu chủ yếu vào các ngày làm việc trong tuần [thứ 2 --> thứ 6]; Cuối tuần [Thứ 7, CN] lượng người đi qua cầu giảm đi đáng kể. Lượng người đi qua cầu ngày làm việc gấp đôi ngày cuối tuần.
- Vào các ngày làm việc trong tuần lượng người đi xe đạp qua cầu chủ yếu tập trung vào thời điểm 7,8,9h buổi sáng | 16, 17, 18h buổi chiều. Lượng người đi nhiều nhất vào thời điểm 8(h) sáng - 17h chiều. Khung giờ từ 16--> 18h buổi chiều số lượt qua cây cầu phía Đông cao hơn rất nhiều so với người đi qua cây cầu phía Tây.
- Vào các ngày cuối tuần, người đi xe đạp chủ yếu qua cầu trong thời gian từ 12-16h; 2 ngày cuối tuần tỷ lệ lượt người qua cây cầu phía Đông và phía Tây là tương đương nhau.





Chi tiết các bước thực hiện trong file jupyter notebook…

Thực hành 5

Thực hành 5

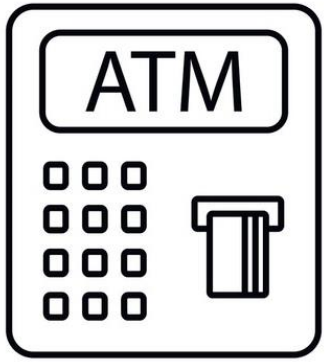
Mô tả Dữ liệu:

Tập dữ liệu lưu trữ thông tin giao dịch tại cây Big Stree ATM (đặt tại một giao lộ sầm uất của Ấn Độ) theo thời gian từng ngày, cho biết Tổng số lần giao dịch và Số tiền giao dịch mỗi ngày (rupee - INR). Từ ngày 01-01-2014 đến 31-12-2019 (6 năm). Dữ liệu bao gồm 6 thuộc tính:

- Transaction Data: Thời gian
- ATM Name: Tên cây ATM
- No Of Withdrawals: Tổng số lần KH giao dịch tại ATM trong ngày
- Amount withdrawn Card: Số tiền giao dịch sử dụng thẻ của chính Ngân hàng
- Amount withdrawn Other Card: Số tiền giao dịch sử dụng thẻ của Ngân hàng khác
- Total amount Withdrawn: Tổng số tiền KH giao dịch tại ATM trong ngày

Thực hành 5

Transaction Date	ATM Name	No Of Withdrawals	Amount withdrawn Card	Amount withdrawn Other Card	Total amount Withdrawn
2014-01-01	Big Street ATM	50	168880	369450	538330
2014-01-02	Big Street ATM	17	84640	143550	228190
2014-01-03	Big Street ATM	24	312260	49500	361760
2014-01-04	Big Street ATM	34	217730	238140	455870
2014-01-05	Big Street ATM	30	235320	224740	460060
2014-01-06	Big Street ATM	29	231340	148840	380180
2014-01-07	Big Street ATM	23	215180	122380	337560
2014-01-08	Big Street ATM	37	230480	351540	582020
2014-01-09	Big Street ATM	36	159890	242860	402750
2014-01-10	Big Street ATM	29	245020	131850	376870
2014-01-11	Big Street ATM	42	301320	168750	470070
2014-01-12	Big Street ATM	19	49710	118180	167890
2014-01-13	Big Street ATM	22	217080	124200	341280
2014-01-14	Big Street ATM	25	274590	104850	379440
2014-01-15	Big Street ATM	24	399740	77400	477140
2014-01-16	Big Street ATM	17	119480	115200	234680



Thực hành 5

Sử dụng các kỹ thuật để Phân tích tập dữ liệu, tìm ra các thông tin có ích về thói quen, hành vi rút tiền tại cây ATM; giúp Ngân hàng đánh giá hiệu quả hoạt động của cây ATM và có kế hoạch tiếp quỹ hợp lý.

Q & A
Thank you!