

Thao tác với tệp dữ liệu sử dụng Pandas

Giảng viên: TS. **Nguyễn Văn Quyết**

Email: quyetict@gmail.com

- Khái quát về đọc và ghi dữ liệu trên Pandas
- Đọc và ghi dữ liệu tệp CSV
- Đọc và ghi dữ liệu tệp JSON
- Đọc và ghi dữ liệu tệp XML

1. Khái quát về đọc và ghi dữ liệu với tệp trên Pandas

- Pandas cung cấp các hàm đọc và ghi dữ liệu từ các đối tượng pandas.
- Hàm đọc dữ liệu từ tệp có dạng `read_loaitệp()` trả về một DataFrame:
 - Ví dụ: Hàm `read_excel()` đọc dữ liệu từ tệp .xlsx và trả về một dataframe.
- Hàm ghi dữ liệu từ tệp có dạng `DataFrame.to_loaitệp()` sẽ ghi một DataFrame dưới dạng định dạng tệp.
 - Ví dụ: Hàm `DataFrame.to_excel()` sẽ ghi dữ liệu từ DataFrame thành một file .xlsx.

Khái niệm về tệp csv

- CSV, viết tắt của comma-separated values, là một loại tệp text sử dụng dấu phẩy “,” để chia cắt giữa các giá trị.
- Mỗi dòng của tệp là một bản ghi (record) của dữ liệu.
- Mỗi bản ghi có thể gồm nhiều trường, với dấu phẩy “,” được dùng để chia cắt các trường.
- Tệp csv được dùng để ghi dữ liệu dạng bảng dưới dạng text.

```

COVID-19 Dataset.csv
1 Continent,Country,Last_Updated_Date>Total_Cases>Total_Deaths>Total_Cases_Per_Million>Total_Deaths_Per_Million
2 Asia,Afghanistan,11-10-2021,155540,7228,3904.565,181.447,,,2369625,828601,,39835428,54.422,18.6,2.58
3 Europe,Albania,11-10-2021,175163,2777,60970.074,966.608,,,1795351,962281,833070,2872934,104.871,38,1
4 Africa,Algeria,11-10-2021,204695,5855,4587.864,131.229,,,,,44616626,17.348,29.1,6.211,3.857,13913.8
5 Europe,Andorra,11-10-2021,15307,130,197882.462,1680.585,215733,0.054,,,77354,163.755,,,,
6 Africa,Angola,11-10-2021,61580,1629,1814.72,48.006,,,4074677,2863708,1210969,33933611,23.89,16.8,2.4
7 North America,Anguilla,01-10-2021,,,,,,18704,9548,9156,15125,,,,,
8 North America,Antigua and Barbuda,11-10-2021,3750,93,37983.146,941.982,,,98840,54435,44405,98728,231
9 South America,Argentina,11-10-2021,5266275,115491,115473.741,2532.374,24156096,0.019,54115894,302652
10 Asia,Armenia,11-10-2021,272957,5575,91962.678,1878.288,1770914,0.102,514241,344029,170212,2968128,16
11 North America,Aruba,10-10-2021,,,,,,157541,81871,75670,107195,584.8,41.2,13.085,7.452,35973.781
12 Oceania,Australia,11-10-2021,131415,1461,5095.932,56.654,39943525,0.012,31020482,17808025,13212457,2
13 Europe,Austria,11-10-2021,762538,11106,84322.894,1228.122,90377770,0.005,10957619,5790455,5483648,96
14 Asia,Azerbaijan,11-10-2021,492790,6677,48202.428,653.113,4952737,0.087,8997289,4889404,4107885,10223
15 North America,Bahamas,11-10-2021,21580,590,54369.46,1486.468,147633,0.178,224531,129121,100098,39691
16 Asia,Bahrain,11-10-2021,275734,1390,157715.946,795.06,6612314,0.004,2669575,1168806,1129802,1748295,
17 Asia,Bangladesh,11-10-2021,1562958,27699,9398.227,166.557,9907321,0.035,54798986,36580037,18218949,1
18 North America,Barbados,11-10-2021,11132,98,38692.007,340.623,,,253554,141820,111734,287708,664.463,3
  
```

Đọc dữ liệu với tệp csv

- Hàm `read_csv()` được dùng để đọc dữ liệu từ tệp csv, hàm trả về một đối tượng `DataFrame`.
- Các tham số cơ bản của hàm `read_csv()`:
 - **filepath_or_buffer**: đường dẫn tới file cần đọc.
 - **sep**: phân tử chia cắt các giá trị giữa các trường, mặc định “,”.
 - **header**: số dòng được chọn làm header.
 - **dtype**: sử dụng để định nghĩa kiểu dữ liệu cho toàn bảng hoặc từng cột.

Ví dụ: Đọc dữ liệu file titanic.csv

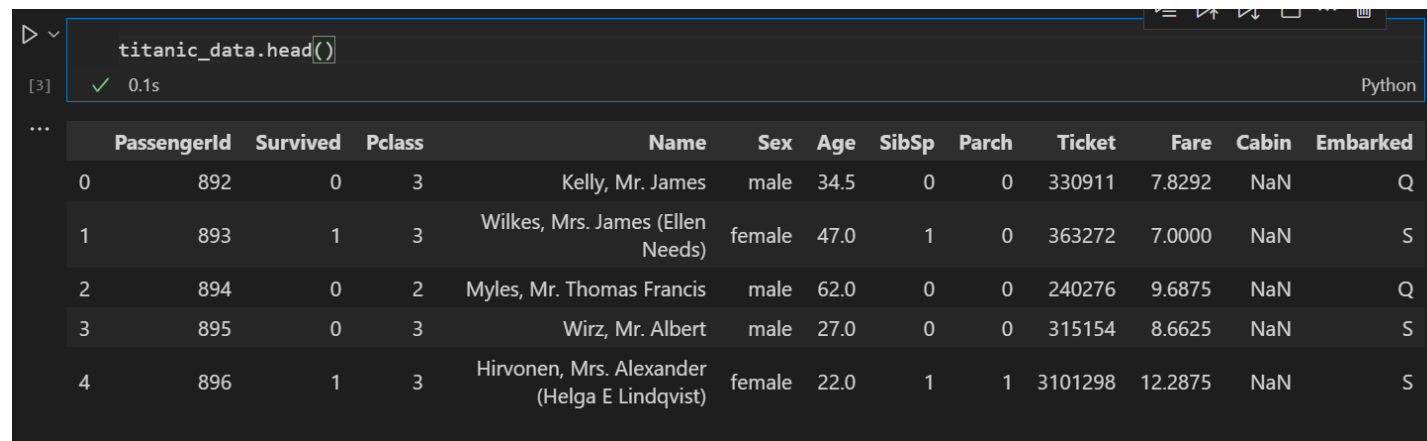
- Tải file theo đường link sau: [Titanic dataset | Kaggle](#)
- Đổi tên file thành *titanic.csv*
- Đọc dữ liệu từ file dùng pandas:

```
[1] import pandas as pd
    ✓ 6.8s

[2] titanic_data=pd.read_csv('titanic.csv')
    ✓ 0.2s
```

Ví dụ: Đọc dữ liệu file titanic.csv

- Xem qua dữ liệu của DataFrame bằng hàm head()
 - Hàm head() sẽ hiển thị 5 bản ghi đầu tiên của DataFrame.



```
titanic_data.head()
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	892	0	3	Kelly, Mr. James	male	34.5	0	0	330911	7.8292	NaN	Q
1	893	1	3	Wilkes, Mrs. James (Ellen Needs)	female	47.0	1	0	363272	7.0000	NaN	S
2	894	0	2	Myles, Mr. Thomas Francis	male	62.0	0	0	240276	9.6875	NaN	Q
3	895	0	3	Wirz, Mr. Albert	male	27.0	0	0	315154	8.6625	NaN	S
4	896	1	3	Hirvonen, Mrs. Alexander (Helga E Lindqvist)	female	22.0	1	1	3101298	12.2875	NaN	S

Ví dụ: Đọc dữ liệu file titanic.csv

- Có thể sử dụng đường link trực tuyến đến file thay vì tải trực tiếp về máy:

- Sử dụng đường link sau để đọc dữ liệu thay vì tải trực tiếp về máy:

<https://raw.githubusercontent.com/datasciencedojo/datasets/master/titanic.csv>

```
titanic_data=pd.read_csv('https://raw.githubusercontent.com/datasciencedojo/datasets/master/titanic.csv')
```

[5] ✓ 0.6s Python

```
titanic_data.head()
```

[6] ✓ 0.5s Python

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	S
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	C
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2.	7.9250	NaN	S

Ví dụ: Đọc dữ liệu file titanic.csv

- Theo mặc định, hàng đầu tiên của file sẽ được lấy làm tên của cột. Tuy nhiên trong nhiều trường hợp, tên cột mặc định trong file có thể không phù hợp. Ta có thể điều chỉnh tên cột bằng cách điều chỉnh thuộc tính `names`:

```
col_names=["Id",
           "Survived",
           "Passenger Class",
           "Full Name",
           "Gender",
           "Age",
           'SibSp',
           'Parch',
           'Ticket Number',
           'Price', 'Cabin',
           'Station' ]

[17] ✓ 0.4s

titanic_data=pd.read_csv('titanic.csv',names=col_names)
titanic_data

[20] ✓ 0.8s
```



	Id	Survived	Passenger Class	Full Name	Gender
0	PassengerId	Survived	Pclass	Name	Sex
1	892	0	3	Kelly, Mr. James	male
2	893	1	3	Wilkes, Mrs. James (Ellen Needs)	female
3	894	0	2	Myles, Mr. Thomas Francis	male
4	895	0	3	Wirz, Mr. Albert	male
...
414	1305	0	3	Spector, Mr. Woolf	male
415	1306	1	1	Oliva y Ocana, Dona. Fermina	female
416	1307	0	3	Saether, Mr.	male

Ví dụ: Đọc dữ liệu file titanic.csv

- Ta thấy rằng sau khi thêm header thì dòng đầu tiên của file sẽ trở thành một bản ghi trong DataFrame. Để loại bỏ dòng này, ta sử dụng tham số `skiprows`.

```
titanic_data=pd.read_csv('titanic.csv',names=col_names, skiprows=[0])
titanic_data
```

[21] ✓ 0.1s Python

...

	Id	Survived	Passenger Class	Full Name	Gender	Age	SibSp	Parch	Ticket Number	Price	Cabin	Station
0	892	0	3	Kelly, Mr. James	male	34.5	0	0	330911	7.8292	NaN	Q
1	893	1	3	Wilkes, Mrs. James (Ellen Needs)	female	47.0	1	0	363272	7.0000	NaN	S
2	894	0	2	Myles, Mr. Thomas Francis	male	62.0	0	0	240276	9.6875	NaN	Q
3	895	0	3	Wirz, Mr. Albert	male	27.0	0	0	315154	8.6625	NaN	S

Ví dụ: Đọc dữ liệu file titanic.csv

- Người dùng có thể loại bỏ header bằng cách đặt `header=None` và `skiprows=[0]`

```

titanic_data=pd.read_csv('titanic.csv',header=None, skiprows=[0])
titanic_data

```

[22] ✓ 0.1s

	0	1	2	3	4	5	6	7	8	9	10	11
0	892	0	3	Kelly, Mr. James	male	34.5	0	0	330911	7.8292	NaN	Q
1	893	1	3	Wilkes, Mrs. James (Ellen Needs)	female	47.0	1	0	363272	7.0000	NaN	S
2	894	0	2	Myles, Mr. Thomas Francis	male	62.0	0	0	240276	9.6875	NaN	Q
3	895	0	3	Wirz, Mr. Albert	male	27.0	0	0	315154	8.6625	NaN	S
4	896	1	3	Hirvonen, Mrs. Alexander (Helga E Lindqvist)	female	22.0	1	1	3101298	12.2875	NaN	S
...
413	1305	0	3	Spector, Mr. Woolf	male	NaN	0	0	A.5. 3236	8.0500	NaN	S

Ví dụ 2: Ghi dữ liệu tệp DataFrame ra tệp csv

- Giả sử ta có DataFrame như sau:

```
In [23]: import pandas as pd
students=pd.DataFrame([[ 'Đặng Thành Nam', '101191A'],
                        [ 'Lê Văn Long', '101192']],
                      columns=['full_name', 'class_id'])
students
```

```
Out[23]:
```

	full_name	class_id
0	Đặng Thành Nam	101191A
1	Lê Văn Long	101192

Ví dụ 2: Ghi dữ liệu từ tệp DataFrame ra tệp csv

- Để ghi dữ liệu vào tệp ta sử dụng hàm `to_csv()`.

```
In [2]: import pandas as pd
students=pd.DataFrame([['Đặng Thành Nam', '101191A'],
                        ['Lê Văn Long', '101192']],
                        columns=['full_name', 'class_id'])
students
```

Out[2]:

	full_name	class_id
0	Đặng Thành Nam	101191A
1	Lê Văn Long	101192

```
In [3]: students.to_csv("students.csv")
```

- Hàm csv được tạo sẽ có dạng sẽ được mặc định có thêm cột index là địa chỉ các bản ghi.

```
1 ,full_name,class_id
2 0,Đặng Thành Nam,101191A
3 1,Lê Văn Long,101192
4
```

Loại bỏ cột Index khi tạo tệp

- Để loại bỏ cột index, ta cho tham số Index = False:

```
: students.to_csv("students.csv", index=False)
```



1	full_name,class_id
2	Đặng Thành Nam,101191A
3	Lê Văn Long,101192
4	

Điều chỉnh kí tự ngăn cách các giá trị

- Giá trị mặc định ngăn cách các giá trị là dấu “,” ta có thể điều chỉnh thông qua tham số sep.

```
students.to_csv("students.csv", index=False, sep=";")
```



```
1 full_name;class_id  
2 Đặng Thành Nam;101191A  
3 Lê Văn Long;101192  
4
```


Xử lý dữ liệu khuyết thiếu

- Dữ liệu khuyết thiếu trong pandas sẽ không được ghi ra tệp, tuy nhiên ta có thể thay thế dữ liệu khuyết thiếu bằng các giá trị mong muốn thông qua tham số `na_rep`.

```
: students.to_csv("students.csv", index=False, na_rep="unknown")
```



```
1 full_name,class_id
2 Đặng Thành Nam,101191A
3 Lê Văn Long,101192
4 Nguyễn Hoài Nam,unknown
5
```

File JSON là gì?

- JSON, viết tắt của JavaScript Object Notation, là một kiểu dữ liệu mở trong JavaScript. Kiểu dữ liệu này bao gồm chủ yếu là text, có thể đọc được theo dạng cặp "thuộc tính - giá trị".
- JSON là một kiểu dữ liệu trung gian, chủ yếu được dùng để vận chuyển thông tin giữa các thành phần của một chương trình

```
{
  "firstName": "John",
  "lastName": "Smith",
  "isAlive": true,
  "age": 27,
  "address": {
    "streetAddress": "21 2nd Street",
    "city": "New York",
    "state": "NY",
    "postalCode": "10021-3100"
  },
  "phoneNumbers": [
    {
      "type": "home",
      "number": "212 555-1234"
    },
    {
      "type": "office",
      "number": "646 555-4567"
    }
  ],
  "children": [],
  "spouse": null
}
```

Tạo JSON data với nested dictionary

- Có thể sử dụng từ điển lồng (nested dictionary) để tạo dữ liệu JSON trong Python. Mỗi item ở vòng ngoài sẽ tương ứng với một cột trong JSON file.
- Mỗi từ khóa (key) tại mỗi cột là tên cột và mỗi giá trị tại từ điển bên trong sẽ là giá trị tại mỗi hàng tương ứng với cột.

```
In [15]: patients = {
    "Name": {"0": "John", "1": "Nick", "2": "Ali", "3": "Joseph"},
    "Gender": {"0": "Male", "1": "Male", "2": "Female", "3": "Male"},
    "Nationality": {"0": "UK", "1": "French", "2": "USA", "3": "Brazil"},
    "Age": {"0": 10, "1": 25, "2": 35, "3": 29}
}
patients

Out[15]: {'Name': {'0': 'John', '1': 'Nick', '2': 'Ali', '3': 'Joseph'},
'Gender': {'0': 'Male', '1': 'Male', '2': 'Female', '3': 'Male'},
'Nationality': {'0': 'UK', '1': 'French', '2': 'USA', '3': 'Brazil'},
'Age': {'0': 10, '1': 25, '2': 35, '3': 29}}
```



	Name	Gender	Nationality	Age
0	John	Male	UK	10
1	Nick	Male	French	25
2	Ali	Female	USA	35
3	Joseph	Male	Brazil	29

Tạo dữ liệu JSON với danh sách các từ điển (List of Dictionaries)

- JSON có thể được tạo từ một danh sách bao gồm các từ điển có cùng số phần tử với nhau.

```
In [16]: cars = [  
    {"Name": "Honda", "Price": 10000, "Model": 2005, "Power": 1300},  
    {"Name": "Toyota", "Price": 12000, "Model": 2010, "Power": 1600},  
    {"Name": "Audi", "Price": 25000, "Model": 2017, "Power": 1800},  
    {"Name": "Ford", "Price": 28000, "Model": 2009, "Power": 1200},  
]
```

Ghi dữ liệu JSON tới file JSON trong Python

- Với từ điển lồng và danh sách các từ điển, ta có thể lưu trữ dữ liệu vào file json thông qua thư viện json và hàm dump():

```
In [20]: import json
          with open('patients.json', 'w') as f:
              json.dump(patients, f)

          with open('cars.json', 'w') as f:
              json.dump(cars, f) |
```

Đọc JSON file bằng Pandas

- Ta sử dụng hàm `read_json()` để đọc dữ liệu từ file json và lấy về DataFrame tương ứng.

```
In [22]: patients_df=pd.read_json("patients.json")
patients_df
```

Out[22]:

	Name	Gender	Nationality	Age
0	John	Male	UK	10
1	Nick	Male	French	25
2	Ali	Female	USA	35
3	Joseph	Male	Brazil	29

```
In [23]: cars_df=pd.read_json("cars.json")
cars_df
```

Out[23]:

	Name	Price	Model	Power
0	Honda	10000	2005	1300
1	Toyota	12000	2010	1600
2	Audi	25000	2017	1800
3	Ford	28000	2009	1200

Ghi dữ liệu tới file JSON thông qua Pandas

- Tương tự như với csv, DataFrame cũng có thể được lưu trữ dưới dạng file JSON thông qua hàm `to_json()`.

```
In [26]: cars_df.to_json("cars1.json")
```



```
1 {"Name":{"0":"Honda","1":"Toyota","2":"Audi","3":"Ford"},"Price":{"0":10000,"1":12000,"2":25000,"3":28000},"Model":  
  {"0":2005,"1":2010,"2":2017,"3":2009},"Power":{"0":1300,"1":1600,"2":1800,"3":1200}}
```

XML là gì?

- XML, viết tắt của Extensible Markup Language (ngôn ngữ đánh dấu có thể mở rộng), là ngôn ngữ đánh dấu với mục đích chung do [W3C](#) đề nghị, để tạo ra các ngôn ngữ đánh dấu khác.
- Mục đích chính của XML là đơn giản hóa việc chia sẻ dữ liệu giữa các hệ thống khác nhau, đặc biệt là các hệ thống được kết nối với Internet.
- XML cung cấp một phương tiện dùng văn bản (text) để mô tả thông tin và áp dụng một cấu trúc kiểu cây cho thông tin đó.

```
<?xml version="1.0" encoding="UTF-8"?>
<note>
  <to>Tove</to>
  <from>Jani</from>
  <heading>Reminder</heading>
  <body>Don't forget me this weekend!</body>
</note>
```


XPath là gì?

- XPath là đường dẫn tới các element trong XML.



```
<?xml version="1.0" encoding="UTF-8"?>
<bookstore>
  <book category="cooking">
    <title lang="en">Everyday Italian</title>
    <author>Giada De Laurentiis</author>
    <year>2005</year>
    <price>30.00</price>
  </book>
```

XPath Expression

/bookstore/book[1]

/bookstore/book[last()]

/bookstore/book[last()-1]

/bookstore/book[position()<3]

//title[@lang]

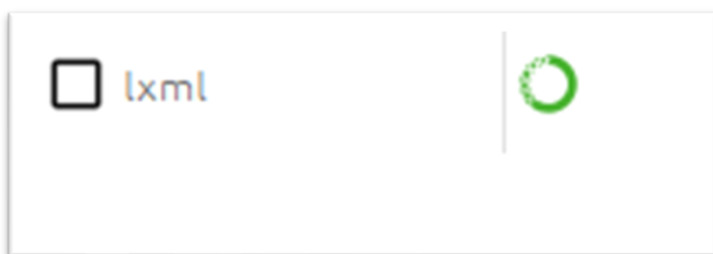
//title[@lang='en']

/bookstore/book[price>35.00]

/bookstore/book[price>35.00]/title

Đọc dữ liệu file XML với file

- Tương tự như với csv và json, pandas có hàm `read_xml` để thiết kế để đọc dữ liệu từ file xml, người dùng cần tải package `lxml`, nếu không phải định nghĩa bộ chuyển đổi với tham số `parser="etree"` để giúp pandas có thể biết được bộ chuyển đổi nào sẽ sử dụng để chuyển đổi file xml sang cấu trúc mà pandas có thể hiểu được.



```
In [30]: df = pd.read_xml(xml, parser='etree')
          df

Out[30]:
```

Đọc dữ liệu từ xml

- Giả sử ta có cấu trúc xml như sau, hàm `read_xml()` được dùng để đọc dữ liệu từ xml sang DataFrame.

```
>>> xml = '''<?xml version='1.0' encoding='utf-8'?>
... <data>
...   <row shape="square" degrees="360" sides="4.0"/>
...   <row shape="circle" degrees="360"/>
...   <row shape="triangle" degrees="180" sides="3.0"/>
... </data>'''
```

```
In [27]: xml = '''<?xml version='1.0' encoding='utf-8'?>
<data xmlns="http://example.com">
  <row>
    <shape>square</shape>
    <degrees>360</degrees>
    <sides>4.0</sides>
  </row>
  <row>
    <shape>circle</shape>
    <degrees>360</degrees>
    <sides/>
  </row>
  <row>
    <shape>triangle</shape>
    <degrees>180</degrees>
    <sides>3.0</sides>
  </row>
</data>'''
```

```
In [31]: df = pd.read_xml(xml)
df
```

Out[31]:

	shape	degrees	sides
0	square	360	4.0
1	circle	360	NaN
2	triangle	180	3.0

Sử dụng Xpath để đọc dữ liệu XML trong Pandas

- Bằng cách điều chỉnh tham số xpath trong hàm read_xml, ta có thể điều chỉnh các element nào sẽ được ghi vào DataFrame.

```
>>> xml = '''<?xml version='1.0' encoding='utf-8'?>
... <data>
...   <row shape="square" degrees="360" sides="4.0"/>
...   <row shape="circle" degrees="360"/>
...   <row shape="triangle" degrees="180" sides="3.0"/>
... </data>'''
```

```
>>> df = pd.read_xml(xml, xpath="//row")
```

```
>>> df
```

	shape	degrees	sides
0	square	360	4.0
1	circle	360	NaN
2	triangle	180	3.0

Ví dụ: Sử dụng Xpath để đọc dữ liệu với Pandas

- Ta copy đoạn XML tại đây: [XML and XPath \(w3schools.com\)](https://www.w3schools.com/xml/default.asp)

```
xml="""<?xml version="1.0" encoding="UTF-8"?>

<bookstore>

<book category="cooking">
  <title lang="en">Everyday Italian</title>
  <author>Giada De Laurentiis</author>
  <year>2005</year>
  <price>30.00</price>
</book>

<book category="children">
  <title lang="en">Harry Potter</title>
  <author>J K. Rowling</author>
  <year>2005</year>
  <price>29.99</price>
</book>
```

Ví dụ: Sử dụng Xpath để đọc dữ liệu với Pandas

- Lấy dữ liệu từ cuốn sách đầu tiên:

```
In [35]: df=pd.read_xml(xml, xpath="/bookstore/book[1]")  
df
```

```
Out[35]:
```

	category	title	author	year	price
0	cooking	Everyday Italian	Giada De Laurentiis	2005	30.0

Ví dụ: Sử dụng Xpath để đọc dữ liệu với Pandas

- Lấy dữ liệu là tất cả các cuốn sách có trong xml:

```
In [36]: df=pd.read_xml(xml, xpath="//book")
df
```

Out[36]:

	category	title	author	year	price
0	cooking	Everyday Italian	Giada De Laurentiis	2005	30.00
1	children	Harry Potter	J K. Rowling	2005	29.99
2	web	XQuery Kick Start	Vaidyanathan Nagarajan	2003	49.99
3	web	Learning XML	Erik T. Ray	2003	39.95

Ví dụ: Sử dụng Xpath để đọc dữ liệu với Pandas

- Lấy dữ liệu là 2 cuốn sách đầu tiên:

```
In [37]: df=pd.read_xml(xml, xpath="/bookstore/book[position()<3]")
df
```

Out[37]:

	category	title	author	year	price
0	cooking	Everyday Italian	Giada De Laurentiis	2005	30.00
1	children	Harry Potter	J K. Rowling	2005	29.99

Ghi dữ liệu từ DataFrame ra file XML

- Ta dùng hàm `to_xml()` để ghi file xml từ DataFrame:

```
>>> df = pd.DataFrame({'shape': ['square', 'circle', 'triangle'],
...                     'degrees': [360, 360, 180],
...                     'sides': [4, np.nan, 3]})
```

```
>>> df.to_xml()
<?xml version='1.0' encoding='utf-8'?>
<data>
  <row>
    <index>0</index>
    <shape>square</shape>
    <degrees>360</degrees>
    <sides>4.0</sides>
  </row>
  <row>
    <index>1</index>
    <shape>circle</shape>
    <degrees>360</degrees>
    <sides/>
  </row>
  <row>
    <index>2</index>
    <shape>triangle</shape>
    <degrees>180</degrees>
    <sides>3.0</sides>
  </row>
</data>
```

Tham số attr_cols

- Tham số attr_cols dùng để định nghĩa tên các cột sẽ lưu vào thuộc tính của các element trong file xml.

```
>>> df.to_xml()
<?xml version='1.0' encoding='utf-8'?>
<data>
  <row>
    <index>0</index>
    <shape>square</shape>
    <degrees>360</degrees>
    <sides>4.0</sides>
  </row>
  <row>
    <index>1</index>
    <shape>circle</shape>
    <degrees>360</degrees>
    <sides/>
  </row>
  <row>
    <index>2</index>
    <shape>triangle</shape>
    <degrees>180</degrees>
    <sides>3.0</sides>
  </row>
</data>
```

```
>>> df.to_xml(attr_cols=[
...     'index', 'shape', 'degrees', 'sides'
...     ])
<?xml version='1.0' encoding='utf-8'?>
<data>
  <row index="0" shape="square" degrees="360" sides="4.0"/>
  <row index="1" shape="circle" degrees="360"/>
  <row index="2" shape="triangle" degrees="180" sides="3.0"/>
</data>
```



Q & A