



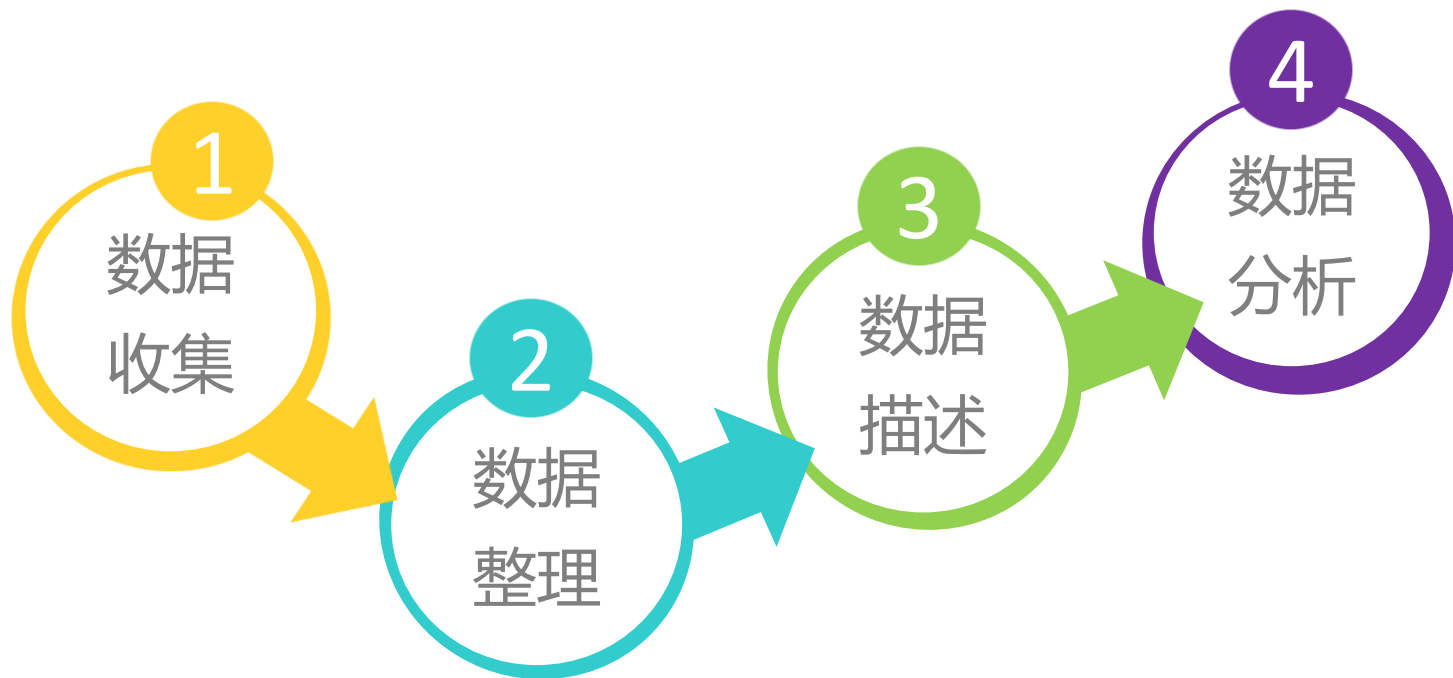
Basic data processing of Python

Python基本数据统计

Dazhuang@NJU

Department of Computer Science and Technology
Department of University Basic Computer Teaching

简单数据处理过程



用Python玩转数据

便捷数据获取

用Python获取数据

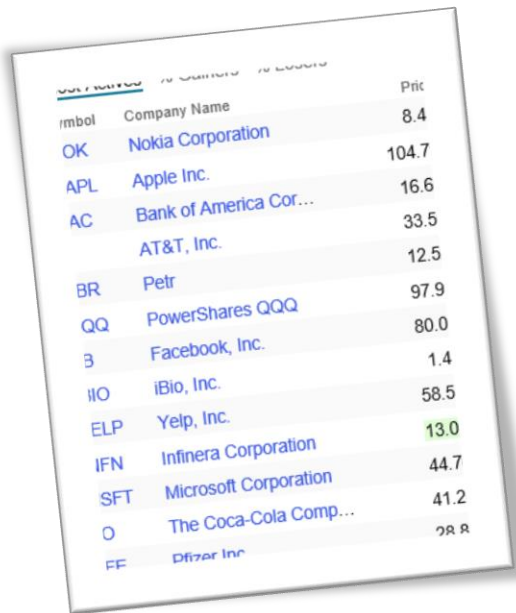
本地数据如何获取?

文件的打开，读写和关闭

- 文件打开
- 读文件
- 写文件
- 文件关闭

[illegible]

用Python获取数据



A tilted screenshot of a stock market data table. The table has three columns: 'Symbol', 'Company Name', and 'Price'. The data is as follows:

Symbol	Company Name	Price
OK	Nokia Corporation	8.4
APL	Apple Inc.	104.7
AC	Bank of America Cor...	16.6
	AT&T, Inc.	33.5
BR	Petr	12.5
QQ	PowerShares QQQ	97.9
B	Facebook, Inc.	80.0
IIO	iBio, Inc.	1.4
ELP	Yelp, Inc.	58.5
IFN	Infinera Corporation	13.0
SFT	Microsoft Corporation	44.7
O	The Coca-Cola Comp...	41.2
FF	Pfizer Inc	28.8

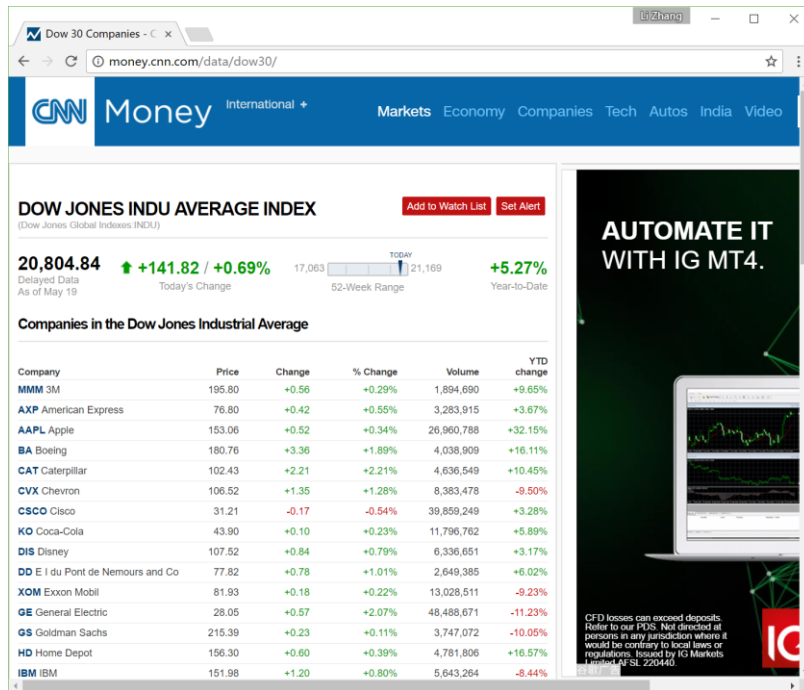
网络数据如何获取（爬取）？

抓取网页，解析网页内容

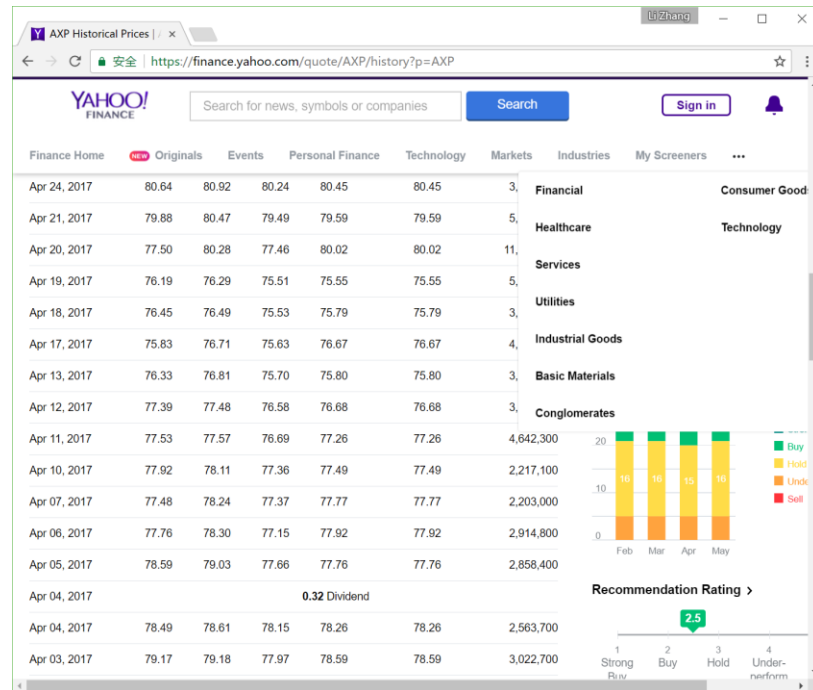
- 抓取
 - urllib内建模块
 - urllib.request
 - Requests第三方库
 - Scrapy框架
- 解析
 - BeautifulSoup库
 - re模块

道指成分股数据

6



dji



quotes

数据形式

7

0	1	2
0 MMM	3M	195.8
1 AXP	American Express	76.8
2 AAPL	Apple	153.06
3 BA	Boeing	180.76
4 CAT	Caterpillar	102.43
5 CVX	Chevron	106.52
6 CSCO	Cisco	31.21
7 KO	Coca-Cola	43.9
8 DIS	Disney	107.52
9 DD	E I du Pont de Nemours	77.82
10 XOM	Exxon Mobil	81.93
11 GE	General Electric	28.05
12 GS	Goldman Sachs	215.39
13 HD	Home Depot	156.3
14 IBM	IBM	151.98
15 INTC	Intel	35.4
16 JNJ	Johnson & Johnson	127
17 JPM	JPMorgan Chase	84.78
18 MCD	McDonald's	148.15
19 MRK	Merck	63.78
20 MSFT	Microsoft	67.69
21 NKE	Nike	51.77
22 PFE	Pfizer	32.46
23 PG	Procter & Gamble	86.24
24 TRV	Travelers Companies Inc	120.79
25 UTX	United Technologies	121.16
26 UNH	UnitedHealth	172.59
27 VZ	Verizon	45.42
28 V	Visa	92.48
29 WMT	Wal-Mart	78.77

djidf

	close	date	high	low	open	volume
0	76.8	1495200600	77.35	76.3	76.55	3278200
1	76.38	1495114200	76.85	75.97	76.27	3545700
2	76.37	1495027800	78.13	76.24	78.13	4441600
3	78.13	1494941400	78.64	77.84	78.6	2457500
4	78.33	1494855000	78.62	77.48	77.48	3327000
5	77.49	1494595800	77.81	77.22	77.7	2865800
6	77.92	1494509400	78.45	77.25	78.2	3780600
7	78.65	1494423000	78.66	78.14	78.28	2396900
8	78.44	1494336600	78.74	78.09	78.16	2570600
9	78.16	1494250200	78.74	77.95	78.5	2608600
10	78.32	1493991000	78.73	77.88	78.61	2936700
11	78.33	1493904600	79.42	77.99	79.23	3902200
12	78.83	1493818200	79.51	78.69	79.23	3800600
13	79.54	1493731800	79.66	79.15	79.15	3334900
14	79.23	1493645400	79.49	78.88	79.22	3458100
15	79.25	1493386200	80.17	79.05	79.94	5313200
16	80.33	1493299800	80.87	80.08	80.77	2922700
17	80.52	1493213400	80.92	80.15	80.62	3661600
18	80.63	1493127000	81.4	80.63	81.06	5061300
19	80.45	1493040600	80.92	80.24	80.64	3563200
20	79.59	1492781400	80.47	79.49	79.88	5837800

quotesdf

便捷网络数据获取

8



是否能够简单方便并且快速的方式获得财经网站上公司股票的历史数据？

Time Period: May 20, 2016 - May 20, 2017 ▾

Show: Historical Prices ▾

Frequency: Daily ▾

Apply

Currency in USD

[Download Data](#)

Date	Open	High	Low	Close	Adj Close	Volume
2016/5/20	63.16	64.14	62.95	63.92	63.92	5278200
2016/5/23	63.86	64.1	63.56	63.59	63.59	3074100
2016/5/24	63.79	65.1	63.79	64.87	64.87	3946100
2016/5/25	65.04	65.76	65.01	65.31	65.31	5755900
2016/5/26	65.29	65.37	64.95	65.23	65.23	3593500
2016/5/27	65.39	65.7	65.33	65.52	65.52	3925700
2016/5/31	65.7	65.92	65.4	65.76	65.76	5256000



```
# Filename: quotes_fromcsv.py
import pandas as pd
quotesdf = pd.read_csv('axp.csv')
print(quotesdf)
```


便捷网络数据获取



注意：目前此Api的部分功能已不对个人用户免费开放（网站和部分数据对个人关闭），重点理解Api的含义和使用方式

图书Api V2

[回Api V2 首页](#)

注意：1. 下文中提到的图书并不包括杂志。2. count最大为100，大于100的

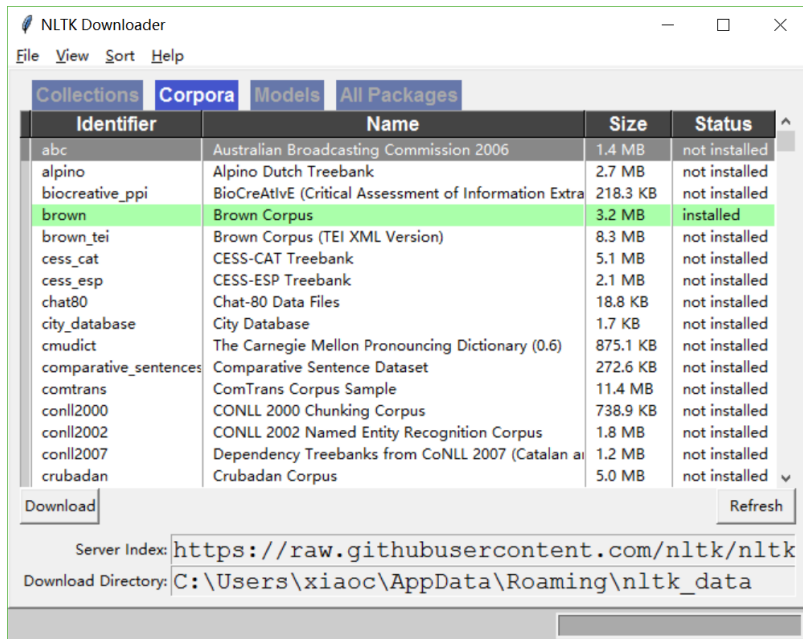
scope: book_basic_r

获取图书信息	GET	/v2/book/:id
根据isbn获取图书信息	GET	/v2/book/isbn/:name
搜索图书	GET	/v2/book/search



```
>>> r = requests.get('https://api.douban.com/v2/book/1084336')
>>> r.text
{"rating":{"max":10,"numRaters":218148,"average":"9.0","min":0},
"subtitle":"","author":["[法] 圣埃克苏佩里"],"pubdate":"2003-8",
"tags":[{"count":52078,"name":"小王子","title":"小王子"},
{"count":43966,"name":"童话", ... , "price":"22.00元"}]
```

NLTK语料库





```
>>> from nltk.corpus import gutenbergl brown
>>> import nltk
>>> print(gutenberg.fileids())
['austen-emma.txt', 'austen-persuasion.txt', 'austen-sense.txt', 'bible-kjv.txt', 'blake-
poems.txt', 'bryant-stories.txt', 'burgess-busterbrown.txt', 'carroll-alice.txt',
'chesterton-ball.txt', 'chesterton-brown.txt', 'chesterton-thursday.txt', 'edgeworth-
parents.txt', 'melville-moby_dick.txt', 'milton-paradise.txt', 'shakespeare-caesar.txt',
'shakespeare-hamlet.txt', 'shakespeare-macbeth.txt', 'whitman-leaves.txt']
>>> texts = gutenberg.words('shakespeare-hamlet.txt')
>>> print(texts)
['[', 'The', 'Tragedie', 'of', 'Hamlet', 'by', ...]
```

用Python玩转数据

2

数据准备

30支道指成分股
(dji) 股票数据的
逻辑结构

公司代码	公司名	最近一次成交价

美国运通公司
(quotes) 股票历
史数据的逻辑结构

收盘价	日期	最高价	最低价	开盘价	成交量

djidf加列索引 (columns)



```
# Filename: stock.py
```

```
import requests
```

```
import re
```

```
import pandas as pd
```

```
def retrieve_dji_list():
```

```
    ...
```

```
    return dji_list
```

```
dji_list = retrieve_dji_list()
```

```
djidf = pd.DataFrame(dji_list)
```

```
cols = ['code', 'name', 'lasttrade']
```

```
djidf.columns = cols
```

```
print(quotesdf)
```

0		1	2
0	MMM	3M	195.8
1	AXP	American Express	76.8
2	AAPL	Apple	153.06
3	BA	Boeing	180.76
4	CAT	Caterpillar	102.43
5	CVX	Chevron	106.52
6	CSCO	Cisco	31.21
7	KO	Coca-Cola	43.9
8	DIS	Disney	107.52
9	DD	E I du Pont de Nemours	77.82
10	XOM	Exxon Mobil	81.93
11	GE	General Electric	28.05
12	GS	Goldman Sachs	215.39
13	HD	Home Depot	156.3
14	IBM	IBM	151.98
15	INTC	Intel	35.4
16	JNJ	Johnson & Johnson	127
17	JPM	JPMorgan Chase	84.78
18	MCD	McDonald's	148.15
19	MRK	Merck	63.78
20	MSFT	Microsoft	67.69
21	NKE	Nike	51.77
22	PFE	Pfizer	32.46
23	PG	Procter & Gamble	86.24
24	TRV	Travelers Companies Inc	120.79
25	UTX	United Technologies	121.16
26	UNH	UnitedHealth	172.59
27	VZ	Verizon	45.42
28	V	Visa	92.48
29	WMT	Wal-Mart	78.77

djidf数据：加完
columns的形式

code	name	lasttrade
MMM		
AXP		
AAPL		
...		
WMT		


quotesdf数据：
原始数据中已有
columns

close	date	high	low	open	volume
	1464010200				
	1464096600				
	1464183000				
	...				
	1495200600				


用1,2,...作为index (行索引)

```
quotesdf = pd.DataFrame(quotes)
```

```
quotesdf.index = range(1,len(quotes)+1)
```



	close	date	high	low	open	volume
0	63.590000	1464010200	64.099998	63.560001	63.860001	3074100
1	64.870003	1464096600	65.099998	63.790001	63.790001	3946100
2	65.309998	1464183000	65.760002	65.010002	65.040001	5755900
3	65.230003	1464269400	65.370003	64.949997	65.290001	3593500
4	65.519997	1464355800	65.699997	65.330002	65.389999	3925700



	close	date	high	low	open	volume
1	63.590000	1464010200	64.099998	63.560001	63.860001	3074100
2	64.870003	1464096600	65.099998	63.790001	63.790001	3946100
3	65.309998	1464183000	65.760002	65.010002	65.040001	5755900
4	65.230003	1464269400	65.370003	64.949997	65.290001	3593500
5	65.519997	1464355800	65.699997	65.330002	65.389999	3925700



如果可以直接用date作为索引，quotes的时间能否转换成普通日期形式（如下图中的效果）？

1464010200

	close	high	low	open	volume
2016-05-23	63.590000	64.099998	63.560001	63.860001	3074100
2016-05-24	64.870003	65.099998	63.790001	63.790001	3946100
2016-05-25	65.309998	65.760002	65.010002	65.040001	5755900
2016-05-26	65.230003	65.370003	64.949997	65.290001	3593500
2016-05-27	65.519997	65.699997	65.330002	65.389999	3925700
2016-05-31	65.760002	65.919998	65.400002	65.699997	5256000
2016-06-01	65.910004	65.959999	65.180000	65.760002	3816000
2016-06-02	66.410004	66.410004	65.599998	65.860001	3052200
2016-06-03	65.489998	65.820000	64.769997	65.529999	4336100
2016-06-06	65.940002	66.199997	65.500000	65.550003	3915200
2016-06-07	65.889999	66.599998	65.879997	66.150002	3779500
2016-06-08	66.260002	66.580002	65.940002	65.940002	2601100



```
>>> from datetime import date
>>> firstday = date.fromtimestamp(1464010200)
>>> lastday = date.fromtimestamp(1495200600)
>>> firstday
datetime.date(2016, 5, 23)
>>> lastday
datetime.date(2017, 5, 19)
```

F
ile

```
# Filename: quotes_history_v2.py
```

```
def retrieve_quotes_historical(stock_code):
```

```
...
```

```
    return [item for item in quotes if not 'type' in item]
```

```
quotes = retrieve_quotes_historical('AXP')
```

```
list1 = []
```

```
for i in range(len(quotes)):
```

```
    x = date.fromtimestamp(quotes[i]['date'])
```

转换成常规时间

```
    y = date.strftime(x, '%Y-%m-%d')
```

转换成固定格式

```
    list1.append(y)
```

```
quotesdf_ori = pd.DataFrame(quotes, index = list1)
```

```
quotesdf_m = quotesdf_ori.drop(['adjclose'], axis = 1)
```

删除adjclose列，原
为unadjclose属性

```
quotesdf = quotesdf_m.drop(['date'], axis = 1)
```

删除原date列

```
print(quotesdf)
```

创建时间序列



```
>>> import pandas as pd
>>> dates = pd.date_range('20170520', periods=7)
>>> dates
<class 'pandas.tseries.index.DatetimeIndex'>
[2017-05-20, ..., 2017-05-26]
Length: 7, Freq: D, Timezone: None
>>> import numpy as np
>>> datesdf = pd.DataFrame(np.random.randn(7,3), index=dates, columns = list('ABC'))
>>> datesdf
```

	A	B	C
2017-05-20	1.302600	-1.214708	1.411628
2017-05-21	-0.512343	2.277474	0.403811
2017-05-22	-0.788498	-0.217161	0.173284
2017-05-23	1.042167	-0.453329	-2.107163
2017-05-24	-1.628075	1.663377	0.943582
2017-05-25	-0.091034	0.335884	2.455431
2017-05-26	-0.679055	-0.865973	0.246970

3

用Python玩转数据

数据显示

数据显示

21

code	name	lasttrade
0	MMM	3M 195.80
1	AXP	American Express 76.80
2	AAPL	Apple 153.06
3	BA	Boeing 180.76
4	CAT	Caterpillar 102.43
5	CVX	Chevron 106.52
6	CSCO	Cisco 31.21
7	KO	Coca-Cola 43.90
8	DIS	Disney 107.52
9	DD	E I du Pont de Nemours and Co 77.82
10	XOM	Exxon Mobil 81.93
11	GE	General Electric 28.05
12	GS	Goldman Sachs 215.39
13	HD	Home Depot 156.30
14	IBM	IBM 151.98
15	INTC	Intel 35.40
16	JNJ	Johnson & Johnson 127.00
17	JPM	JPMorgan Chase 84.78
18	MCD	McDonald's 148.15
19	MRK	Merck 63.78
20	MSFT	Microsoft 67.69
21	NKE	Nike 51.77
22	PFE	Pfizer 32.46
23	PG	Procter & Gamble 86.24
24	TRV	Travelers Companies Inc 120.79
25	UTX	United Technologies 121.16
26	UNH	UnitedHealth 172.59
27	VZ	Verizon 45.42
28	V	Visa 92.48
29	WMT	Wal-Mart 78.77

djidf

	close	high	low	open	volume
2016-05-23	63.590000	64.099998	63.560001	63.860001	3074100
2016-05-24	64.870003	65.099998	63.790001	63.790001	3946100
2016-05-25	65.309998	65.760002	65.010002	65.040001	5755900
2016-05-26	65.230003	65.370003	64.949997	65.290001	3593500
2016-05-27	65.519997	65.699997	65.330002	65.389999	3925700
2016-05-31	65.760002	65.919998	65.400002	65.699997	5256000
2016-06-01	65.910004	65.959999	65.180000	65.760002	3816000
2016-06-02	66.410004	66.410004	65.599998	65.860001	3052200
2016-06-03	65.489998	65.820000	64.769997	65.529999	4336100
2016-06-06	65.940002	66.199997	65.500000	65.550003	3915200
2016-06-07	65.889999	66.599998	65.879997	66.150002	3779500
2016-06-08	66.260002	66.580002	65.940002	65.940002	2601100
2016-06-09	65.709999	65.779999	64.900002	65.720001	3883800
2016-06-10	64.970001	65.480003	64.709999	65.260002	3939100
2016-06-13	63.669998	64.889999	63.630001	64.800003	5883400
2016-06-14	61.070000	63.660000	60.380001	63.590000	12323200
2016-06-15	61.419998	62.160000	60.860001	61.470001	5979900

quotesdf

显示方式:

- 显示行索引
- 显示列索引
- 显示数据的值
- 显示数据描述



```
>>> list(djidf.index)
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22,
23, 24, 25, 26, 27, 28, 29]
>>> list(djidf.columns)
['code', 'name', 'lasttrade']
>>> djidf.values
array([[ 'MMM', '3M', 195.8],
       ...,
       ['WMT', 'Wal-Mart', 78.77]], dtype=object)
>>> djidf.describe
<bound method NDFrame.describe of
0          code      name  lasttrade
0          MM       3M    195.80
...
29         WMT    Wal-Mart    78.77>
```

数据的格式



```
>>> djidf.lasttrade
```

```
1    199.54
```

```
2     77.44
```

```
3    153.87
```

```
...
```

```
30    78.31
```

```
Name: lasttrade, dtype: float64
```

```
dji_list = []
```

```
for item in dji_list_in_text:
```

```
    dji_list.append([item[0], item[1], float(item[2])])
```


数据显示



查看道指成分股
中前5只和后5
只的股票基本信
息?

显示方式:

- 显示行
 - 专用方式
 - 切片
- 显示列

 `>>> djidf.head(5)`

	code	name	lasttrade
0	MMM	3M	195.80
1	AXP	American Express	76.80
2	AAPL	Apple	153.06
3	BA	Boeing	180.76
4	CAT	Caterpillar	102.43

`>>> djidf.tail(5)`

	code	name	lasttrade
25	UTX	United Technologies	121.16
26	UNH	UnitedHealth	172.59
27	VZ	Verizon	45.42
28	V	Visa	92.48
29	WMT	Wal-Mart	78.77

用Python玩转数据

4

数据选择

数据选择

	code	name	lasttrade
0	MMM	3M	195.80
1	AXP	American Express	76.80
2	AAPL	Apple	153.06
3	BA	Boeing	180.76
4	CAT	Caterpillar	102.43
5	CVX	Chevron	106.52
6	CSCO	Cisco	31.21
7	KO	Coca-Cola	43.90
8	DIS	Disney	107.52
9	DD	E I du Pont de Nemours and Co	77.82
10	XOM	Exxon Mobil	81.93
11	GE	General Electric	28.05
12	GS	Goldman Sachs	215.39
13	HD	Home Depot	156.30
14	IBM	IBM	151.98
15	INTC	Intel	35.40
16	JNJ	Johnson & Johnson	127.00
17	JPM	JPMorgan Chase	84.78
18	MCD	McDonald's	148.15
19	MRK	Merck	63.78
20	MSFT	Microsoft	67.69
21	NKE	Nike	51.77
22	PFE	Pfizer	32.46
23	PG	Procter & Gamble	86.24
24	TRV	Travelers Companies Inc	120.79
25	UTX	United Technologies	121.16
26	UNH	UnitedHealth	172.59
27	VZ	Verizon	45.42
28	V	Visa	92.48
29	WMT	Wal-Mart	78.77

选择方式:

- 选择行
- 选择列
- 选择区域
- 筛选 (条件选择)

	close	high	low	open	volume
2016-05-23	63.590000	64.099998	63.560001	63.860001	3074100
2016-05-24	64.870003	65.099998	63.790001	63.790001	3946100
2016-05-25	65.309998	65.760002	65.010002	65.040001	5755900
2016-05-26	65.230003	65.370003	64.949997	65.290001	3593500
2016-05-27	65.519997	65.699997	65.330002	65.389999	3925700
2016-05-31	65.760002	65.919998	65.400002	65.699997	5256000
2016-06-01	65.910004	65.959999	65.180000	65.760002	3816000
2016-06-02	66.410004	66.410004	65.599998	65.860001	3052200
2016-06-03	65.489998	65.820000	64.769997	65.529999	4336100
2016-06-06	65.940002	66.199997	65.500000	65.550003	3915200
2016-06-07	65.889999	66.599998	65.879997	66.150002	3779500
2016-06-08	66.260002	66.580002	65.940002	65.940002	2601100
2016-06-09	65.709999	65.779999	64.900002	65.720001	3883800
2016-06-10	64.970001	65.480003	64.709999	65.260002	3939100
2016-06-13	63.669998	64.889999	63.630001	64.800003	5883400
2016-06-14	61.070000	63.660000	60.380001	63.590000	12323200
2016-06-15	61.419998	62.160000	60.860001	61.470001	5979900



美国运通公司
2017年5月1日至
2017年5月5日间的
股票交易信息?

选择方式:

- 选择行
 - 切片
 - 索引



```
>>> quotesdf['2017-05-01':'2017-05-05']
```

	close	high	low	open	volume
2017-05-01	79.230003	79.489998	78.879997	79.220001	3458100
2017-05-02	79.540001	79.660004	79.150002	79.150002	3334900
2017-05-03	78.830002	79.510002	78.690002	79.230003	3800600
2017-05-04	78.330002	79.419998	77.989998	79.230003	3902200
2017-05-05	78.320000	78.730003	77.879997	78.610001	2936700

数据选择



道指成分股公司
代码?

选择方式:

- 选择列
 - 列名



```
>>> djidf['code']
0    MMM
1    AXP
2    AAPL
...
29   WMT
Name: code, dtype: object
>>> djidf.code
0    MMM
1    AXP
2    AAPL
...
29   WMT
Name: code, dtype: object
```

不支持

`djidf['code', 'lasttrade']`

`djidf['code':'lasttrade']`

支持

`djidf[['code', 'lasttrade']]`

`djidf[['code':'lasttrade']]`

数据选择



道指成分股中索引是1至5的股票信息以及所有股票的代码和最近一次交易价？

选择方式：

- 行、列
 - 标签label (loc)



```
>>> djidf.loc[1:5,]
```

```
code
```

```
1 AXP
```

```
2 AAPL
```

```
3 BA
```

```
4 CAT
```

```
5 CVX
```

```
>>> djidf.loc[:, ['code', 'lasttrade']]
```

```
code lasttrade
```

```
0 MMM 195.80
```

```
1 AXP 76.80
```

```
2 AAPL 153.06
```

```
...
```

```
29 WMT 78.77
```

	name	lasttrade
	American Express	76.80
	Apple	153.06
	Boeing	180.76
	Caterpillar	102.43
	Chevron	106.52

数据选择



道指成分股中行索引是1至5的股票代码和最近一次交易价?行索引是1的股票的最近一次交易价?

选择方式:

- 行和列的区域
 - 标签label (loc)
- 单个值
 - at



```
>>> djidf.loc[1:5, ['code', 'lasttrade']]
```

	code	lasttrade
1	AXP	76.80
2	AAPL	153.06
3	BA	180.76
4	CAT	102.43
5	CVX	106.52

```
>>> djidf.loc[1, 'lasttrade']
```

```
76.799999999999997
```

```
>>> djidf.at[1, 'lasttrade']
```

```
76.799999999999997
```

选择方式:

- 行、列和区域
 - 用iloc (位置)
- 取某个值
 - iat

Source

```
>>> djidf.loc[1:5,['code','lasttrade']]
```

	code	lasttrade
1	AXP	76.80
2	AAPL	153.06
3	BA	180.76
4	CAT	102.43
5	CVX	106.52

如果直接写成
[1:6, 0:2]则表示
列索引即第0
和第1列

Source

```
>>> djidf.iloc[1:6,[0,2]]
```

	code	lasttrade
1	AXP	76.80
2	AAPL	153.06
3	BA	180.76
4	CAT	102.43
5	CVX	106.52

Source

```
>>> djidf.loc[1,'lasttrade']
```

```
76.799999999999997
```

```
>>> djidf.at[1,'lasttrade']
```

```
76.799999999999997
```

Source

```
>>> djidf.iloc[1,2]
```

```
76.799999999999997
```

```
>>> djidf.iat[1,2]
```

```
76.799999999999997
```



美国运通公司
2017年度3月份
的股票信息？进
一步寻找美国运
通公司2017年度
一季度收盘价大
于等于80的记录？

选择方式：

- 条件筛选



```
>>> quotesdf[(quotesdf.index >= '2017-03-01') & (quotesdf.index <= '2017-03-31')]
```

	close	high	low	open	volume
2017-03-01	81.919998	82.000000	81.019997	81.050003	4746400
2017-03-02	80.099998	81.660004	80.059998	81.660004	4409800

...

2017-03-31	79.110001	79.430000	78.800003	78.930000	5228400
------------	-----------	-----------	-----------	-----------	---------

```
>>> quotesdf[(quotesdf.index >= '2017-01-01') & (quotesdf.index <= '2017-03-31') & (quotesdf.close >= 80)]
```

	open	close	high	low	volume
2017-02-23	80.050003	80.449997	79.769997	79.870003	3339500
2017-02-27	80.169998	80.309998	79.589996	79.750000	2619400
2017-02-28	80.059998	80.489998	79.769997	80.120003	4415300
2017-03-01	81.919998	82.000000	81.019997	81.050003	4746400
2017-03-02	80.099998	81.660004	80.059998	81.660004	4409800

用Python玩转数据

5 简单统计与处理

简单统计与筛选



求道指成分股中
30只股票最近一
次成交价的平均值?
股票最近一次成交
价大于等于180的
公司名?



```
>>> djidf.lasttrade.mean()
101.26500000000001
>>> djidf[djidf.lasttrade >= 180].name
0          3M
3          Boeing
12  Goldman Sachs
Name: name, dtype: object
```

简单统计与筛选



统计美国运通公司近一年股票涨和跌分别的天数?



```
>>> len(quotesdf[quotesdf.close > quotesdf.open])  
123  
>>> len(quotesdf)-123  
128
```



统计美国运通公司近一年相邻两天收盘价的涨跌情况?



```
>>> status = np.sign(np.diff(quotesdf.close))  
>>> status  
array([ 1.,  1., -1., ..., -1.,  1.,  1.])  
>>> status[np.where( status == 1.)].size  
132  
>>> status[np.where( status == -1.)].size  
118
```



按最近一次成交价对30只道指成分股股票进行排序。根据排序结果列出前三家公司名。



```
>>> tempdf = djidf.sort_values(by = 'lasttrade', ascending = False)
```

	code	name	lasttrade
12	GS	Goldman Sachs	215.39
0	MMM	3M	195.80
3	BA	Boeing	180.76
26	UNH	UnitedHealth	172.59

...

```
>>> tempdf[:3].name
```

```
12  Goldman Sachs
```

```
0      3M
```

```
3      Boeing
```

```
Name: name, dtype: object
```



统计2017年度1月份的股票开盘天数?



```
>>> t = quotesdf[(quotesdf.index >= '2017-01-01') & (quotesdf.index < '2017-02-01')]  
>>> len(t)  
20
```



统计近一年每个月的股票开盘天数?



Filename: quotes_month.py

import time

...

listtemp = []

for i in range(len(quotesdf)):

temp = time.strptime(quotesdf.index[i], "%Y-%m-%d")

listtemp.append(temp.tm_mon)

tempdf = quotesdf.copy()

tempdf['month'] = listtemp

print(tempdf['month'].value_counts())

	close	high	low	open	volume	month
2016-05-24	64.870003	65.099998	63.790001	63.790001	3946100	5
2016-05-25	65.309998	65.760002	65.010002	65.040001	5755900	5
2016-05-26	65.230003	65.370003	64.949997	65.290001	3593500	5
2016-05-27	65.519997	65.699997	65.330002	65.389999	3925700	5
2016-05-31	65.760002	65.919998	65.400002	65.699997	5256000	5
2016-06-01	65.910004	65.959999	65.180000	65.760002	3816000	6
2016-06-02	66.410004	66.410004	65.599998	65.860001	3052200	6

Output:

8 23

3 23

6 22

12 21

11 21

10 21

9 21

5 21

7 20

1 20

4 19

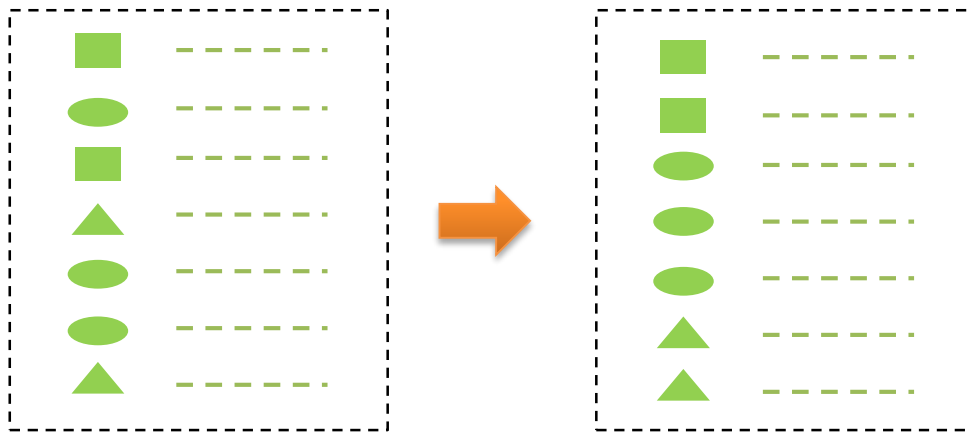
2 19

Name: month,
dtype: int64

用Python玩转数据

6

GROUPING



Grouping的顺序

- ① Splitting
- ② Applying
- ③ Combining



统计近一年每个月的股票开盘天数？



	close	high	low	open	volume	month
2016-05-24	64.870003	65.099998	63.790001	63.790001	3946100	5
2016-05-25	65.309998	65.760002	65.010002	65.040001	5755900	5
2016-05-26	65.230003	65.370003	64.949997	65.290001	3593500	5
2016-05-27	65.519997	65.699997	65.330002	65.389999	3925700	5
2016-05-31	65.760002	65.919998	65.400002	65.699997	5256000	5
2016-06-01	65.910004	65.959999	65.180000	65.760002	3816000	6
2016-06-02	66.410004	66.410004	65.599998	65.860001	3052200	6

```
>>> x = tempdf.groupby('month').count()
```

```
      close  high  low  open  volume
```

```
month
```

```
1      20      20      20      20      20
```

```
2      19      19      19      19      19
```

```
3      23      23      23      23      23
```

```
...
```

```
11     21      21      21      21      21
```

```
12     21      21      21      21      21
```

```
>>> x.close
```

Output:

month

1 20

2 19

3 23

4 19

5 21

6 22

7 20

8 23

9 21

10 21

11 21

12 21

Name: month, dtype: int64



统计近一年每个月的总成交量？

Source

```
>>> tempdf.groupby('month').sum().volume
```

```
month
```

```
1    103887100
```

```
2     65816600
```

```
3     98700800
```

```
4     77893800
```

```
...
```

```
10   116243400
```

```
11    99527200
```

```
12    75948200
```

```
Name: volume, dtype: float64
```

mean()

min()

max()

...



如果更高效统计近一年每个月的总成交量？



```
tempdf.groupby('month').sum().volume
```



```
>>> tempdf.groupby('month').volume.sum()
```

```
month
```

```
1    103887100
```

```
2     65816600
```

```
3     98700800
```

```
4     77893800
```

```
...
```

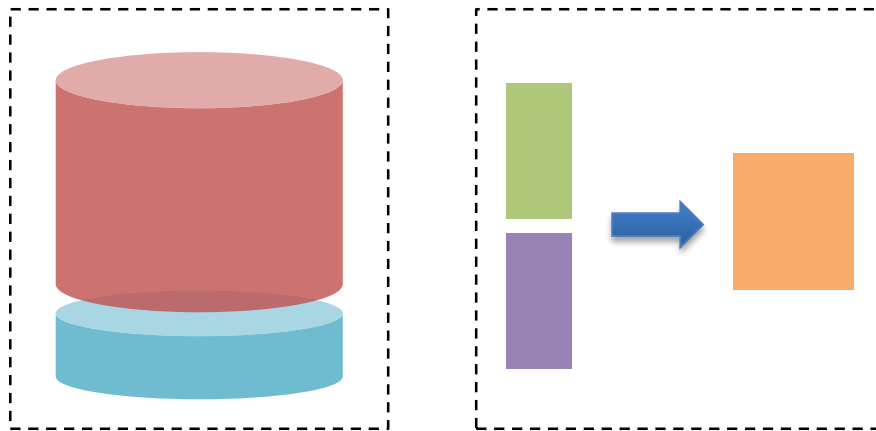
```
12    75948200
```

```
Name: volume, dtype: float64
```

用Python玩转数据

7

MERGE



Merge的形式

- Append
 - 加行到DataFrame
- Concat
 - 连接pandas对象
- Join
 - SQL类型的连接

Append



Source

把美国运通公司
本年度1月1日
至1月5日间的
股票交易信息合
并到近一年中前
两天的股票信息
中?

```
>>> p = quotesdf[:2]
>>> p
```

	open	close	high	low	volume
2016-05-23	63.590000	64.099998	63.560001	63.860001	3074100
2016-05-24	64.870003	65.099998	63.790001	63.790001	3946100

```
>>> q = quotesdf['2017-01-01':'2017-01-05']
>>> q
```

	open	close	high	low	volume
2017-01-03	75.349998	75.750000	74.739998	74.889999	5853900
2017-01-04	76.260002	76.550003	75.059998	75.260002	4635800
2017-01-05	75.320000	76.180000	74.820000	76.000000	3383000

```
>>> p.append(q)
```

	open	close	high	low	volume
2016-05-23	63.590000	64.099998	63.560001	63.860001	3074100
2016-05-24	64.870003	65.099998	63.790001	63.790001	3946100
2017-01-03	75.349998	75.750000	74.739998	74.889999	5853900
2017-01-04	76.260002	76.550003	75.059998	75.260002	4635800
2017-01-05	75.320000	76.180000	74.820000	76.000000	3383000

Concat

47



将美国运通
公司近一年
股票数据中
的前5个和
后5个合并。



或tempdf[-5:]

```
>>> pieces = [tempdf[:5], tempdf[len(tempdf)-5:]]  
>>> pd.concat(pieces)
```

	open	close	high	low	volume	month
2016-05-23	63.590000	64.099998	63.560001	63.860001	3074100	5
2016-05-24	64.870003	65.099998	63.790001	63.790001	3946100	5
2016-05-25	65.309998	65.760002	65.010002	65.040001	5755900	5
2016-05-26	65.230003	65.370003	64.949997	65.290001	3593500	5
2016-05-27	65.519997	65.699997	65.330002	65.389999	3925700	5
2017-05-15	78.330002	78.620003	77.480003	77.480003	3327000	5
2017-05-16	78.129997	78.639999	77.839996	78.599998	2457500	5
2017-05-17	76.370003	78.129997	76.239998	78.129997	4441600	5
2017-05-18	76.379997	76.849998	75.970001	76.269997	3545700	5
2017-05-19	76.800003	77.349998	76.300003	76.550003	3278200	5



两个不同逻辑结构
的对象能否连接?



```
>>> piece1 = quotesdf[:3]
>>> piece2 = tempdf[:3]
>>> pd.concat([piece1,piece2], ignore_index = True)
```

objs	axis
join	join_axes
keys	levels
names	verify_integrity
ignore_index	

	close	high	low	month	open	volume
0	63.590000	64.099998	63.560001	NaN	63.860001	3074100
1	64.870003	65.099998	63.790001	NaN	63.790001	3946100
2	65.309998	65.760002	65.010002	NaN	65.040001	5755900
3	63.590000	64.099998	63.560001	5.0	63.860001	3074100
4	64.870003	65.099998	63.790001	5.0	63.790001	3946100
5	65.309998	65.760002	65.010002	5.0	65.040001	5755900

Join

49

code	name
AXP	
KO	

volume	code	month
	AXP	
	AXP	
	KO	
	KO	



code	name	volume	month
AXP			
AXP			
KO			
KO			

Join

50



将美国运通公司
和可口可乐公司
近一年中每个月
的交易总量表
(包含公司代码)
与30只道琼斯
成分股股票信息
合并。

code | name | volume | month

	code	name	lasttrade
0	MMM	3M	195.80
1	AXP	American Express	76.80
2	AAPL	Apple	153.06
3	BA	Boeing	180.76
4	CAT	Caterpillar	102.43
5	CVX	Chevron	106.52
6	CSCO	Cisco	31.21
7	KO	Coca-Cola	43.90
8	DIS	Disney	107.52
9	DD	E I du Pont de Nemours and Co	77.82
10	XOM	Exxon Mobil	81.93
11	GE	General Electric	28.05
12	GS	Goldman Sachs	215.39
13	HD	Home Depot	156.30
14	IBM	IBM	151.98
15	INTC	Intel	35.40
16	JNJ	Johnson & Johnson	127.00
17	JPM	JPMorgan Chase	84.78
18	MCD	McDonald's	148.15
19	MRK	Merck	63.78
20	MSFT	Microsoft	67.69
21	NKE	Nike	51.77
22	PFE	Pfizer	32.46
23	PG	Procter & Gamble	86.24
24	TRV	Travelers Companies Inc	120.79
25	UTX	United Technologies	121.16
26	UNH	UnitedHealth	172.59
27	VZ	Verizon	45.42
28	V	Visa	92.48
29	WMT	Wal-Mart	78.77

	volume	code	month
month			
1	103887100	AXP	1
2	65816600	AXP	2
3	98700800	AXP	3
4	77893800	AXP	4
5	76209200	AXP	5
6	121788800	AXP	6
7	90064900	AXP	7
8	77514100	AXP	8
9	95572800	AXP	9
10	116243400	AXP	10
11	99527200	AXP	11
12	75948200	AXP	12
1	240321400	KO	1
2	333983800	KO	2
3	339185400	KO	3
4	232465400	KO	4
5	239687800	KO	5
6	265483400	KO	6
7	235959400	KO	7
8	235118300	KO	8
9	251007200	KO	9
10	264839100	KO	10
11	316557000	KO	11
12	283871000	KO	12

AKdf

djidf

Join

51

Source

```
>>> pd.merge(djidf.drop(['lasttrade'], axis = 1), AKdf, on = 'code')
```

	code	name	volume	month
0	AXP	American Express	103887100	1
1	AXP	American Express	65816600	2
2	AXP	American Express	98700800	3
3	AXP	American Express	77893800	4
4	AXP	American Express	76209200	5
...				
19	KO	Coca-Cola	235118300	8
20	KO	Coca-Cola	251007200	9
21	KO	Coca-Cola	264839100	10
22	KO	Coca-Cola	316557000	11
23	KO	Coca-Cola	283871000	12

	code	name	volume	month
0	AXP	American Express	103887100	1
1	AXP	American Express	65816600	2
2	AXP	American Express	98700800	3
3	AXP	American Express	77893800	4
4	AXP	American Express	76209200	5
5	AXP	American Express	121788800	6
6	AXP	American Express	90064900	7
7	AXP	American Express	77514100	8
8	AXP	American Express	95572800	9
9	AXP	American Express	116243400	10
10	AXP	American Express	99527200	11
11	AXP	American Express	75948200	12
12	KO	Coca-Cola	240321400	1
13	KO	Coca-Cola	333983800	2
14	KO	Coca-Cola	339185400	3
15	KO	Coca-Cola	232465400	4
16	KO	Coca-Cola	239687800	5
17	KO	Coca-Cola	265483400	6
18	KO	Coca-Cola	235959400	7
19	KO	Coca-Cola	235118300	8
20	KO	Coca-Cola	251007200	9
21	KO	Coca-Cola	264839100	10
22	KO	Coca-Cola	316557000	11
23	KO	Coca-Cola	283871000	12

merge函数的参数

52

left	right	how
on	left_on	right_on
left_index	right_index	sort
suffixes	copy	