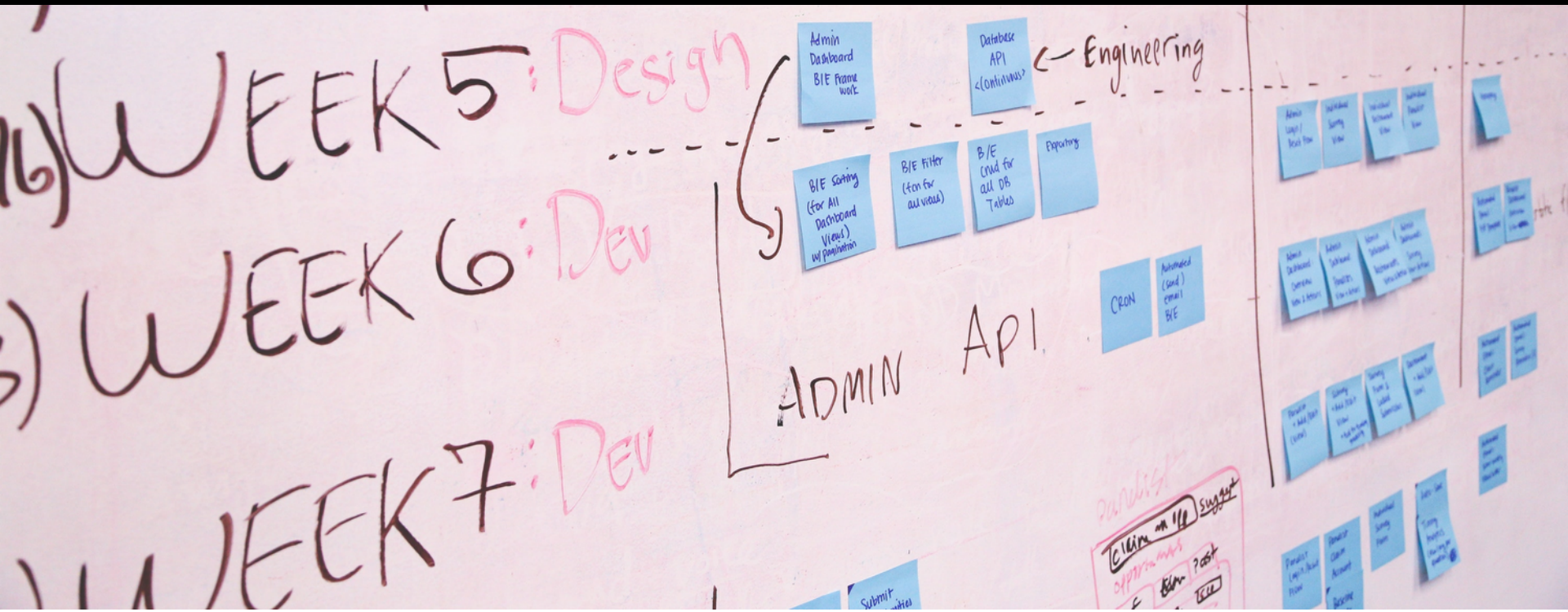


LEARNING GARDEN MAKER WORKSHOP:

WEB SCRAPING IN PYTHON BEAUTIFULSOUP

Bernard Suen
Center for Entrepreneurship
Chinese University of Hong Kong

Today's agenda.

1. What is web scraping ?
2. How does it work ?
3. The CRISP-DM model for handling data.
4. Demo of scraping in ParseHub.
5. Scraping in BeautifulSoup.
6. Beyond scraping: Legal & ethical considerations.

What is web scraping?

Web scraping is a process of fetching web pages from a website and extracting specific data from it.

- ✓ Scrap stock prices
- ✓ Scrap product information
- ✓ Scrap race scores
- ✓ Scrap market trend data
- ✓ Scrap demographics and census data
- ✓ Social media and web articles, etc.

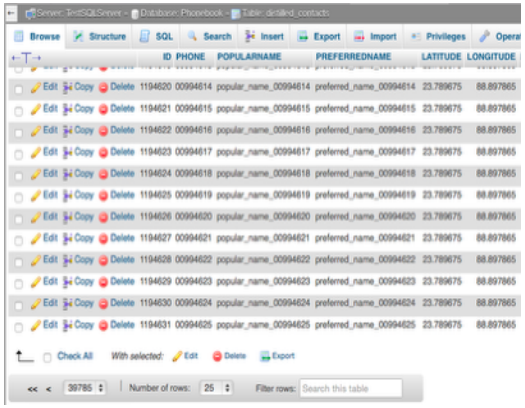
How does it work?

Data

Web Publishing

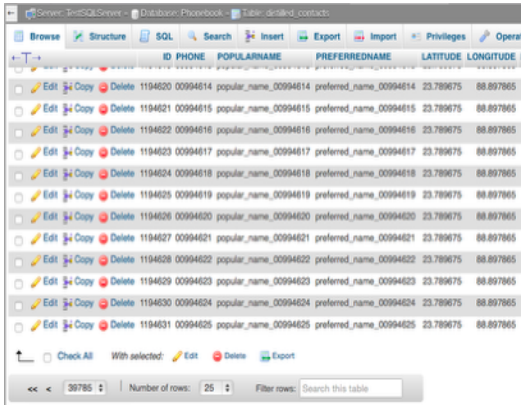
Web

Dynamic Web Building

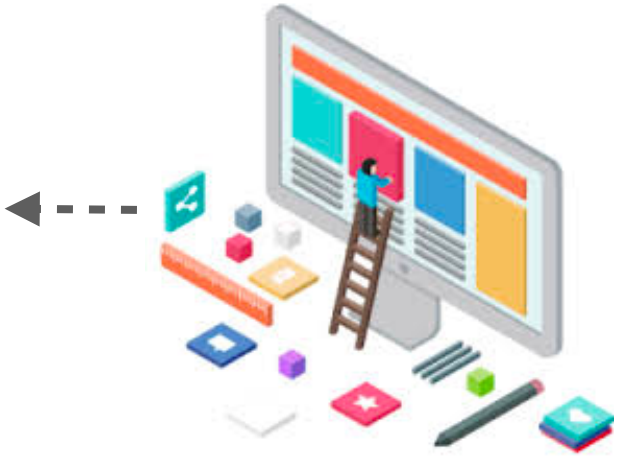


Static Web Building

Data



Web Scraping



Web

Browser View

Welcome to My Page!

My Favourite Fruits

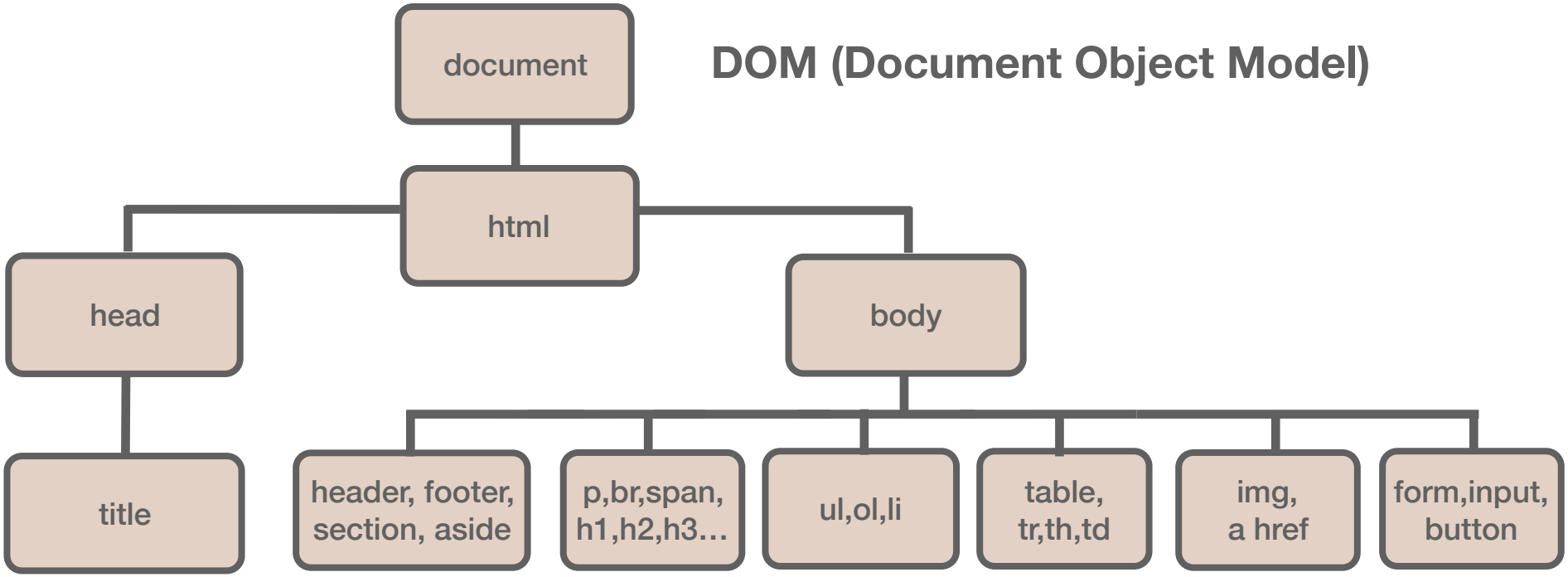
- apples
- oranges
- pineapples

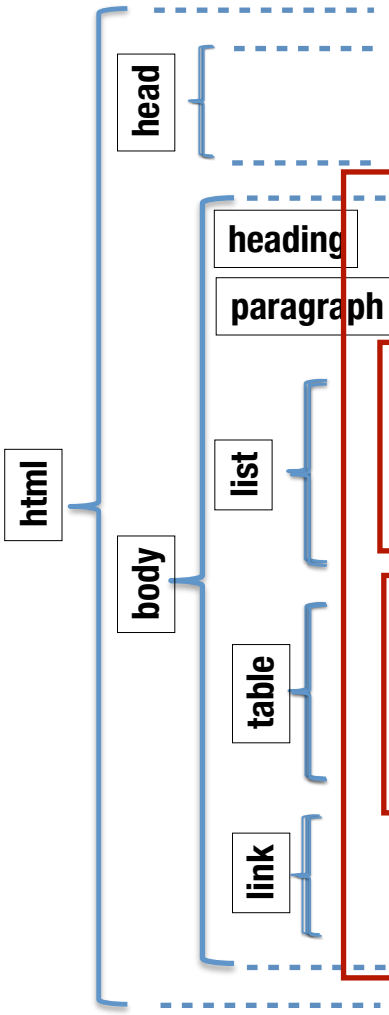
Name	Major
John Chan	English
David Wong	Computer Science

香 港 中 文 大 學
The Chinese University of Hong Kong

This is the link to CUHK's website.

DOM (Document Object Model)



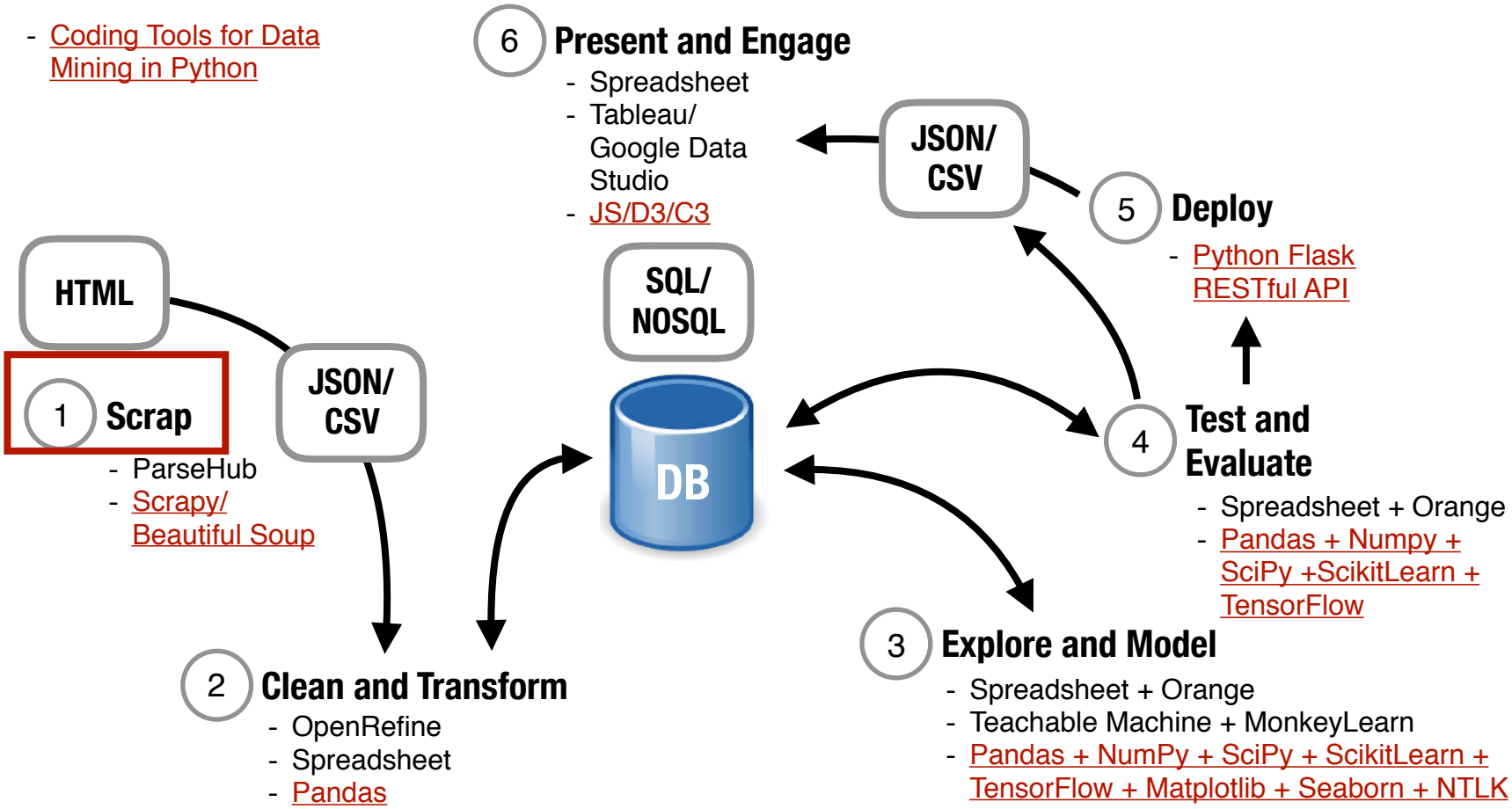



```
<!DOCTYPE html>
<html>
<head>
<title>My Web Page</title>
</head>
<body>
<h1>Welcome to My Page!</h1>
<p><font color="blue">My Favourite Fruits</font></p>

<ul>
  <li>apples</li>
  <li>oranges</li>
  <li>pineapples</li>
</ul>

<table cellspacing="0" cellpadding="0" border="1">
  <tr><th width="20%">Name</th><th width="20%">Major</th></tr>
  <tr><td height="50">John Chan</td><td height="50">English</td></tr>
  <tr><td>David Wong</td><td>Computer Science</td></tr>
</table>

<img src="http://www.cuhk.edu.hk/english/images/cuhk_logo_2x.png"></
img>
<a href="www.cuhk.edu.hk">This is the link to CUHK's website</a>.
</body>
</html>
```

How to scrap list and table data from the highlighted areas?

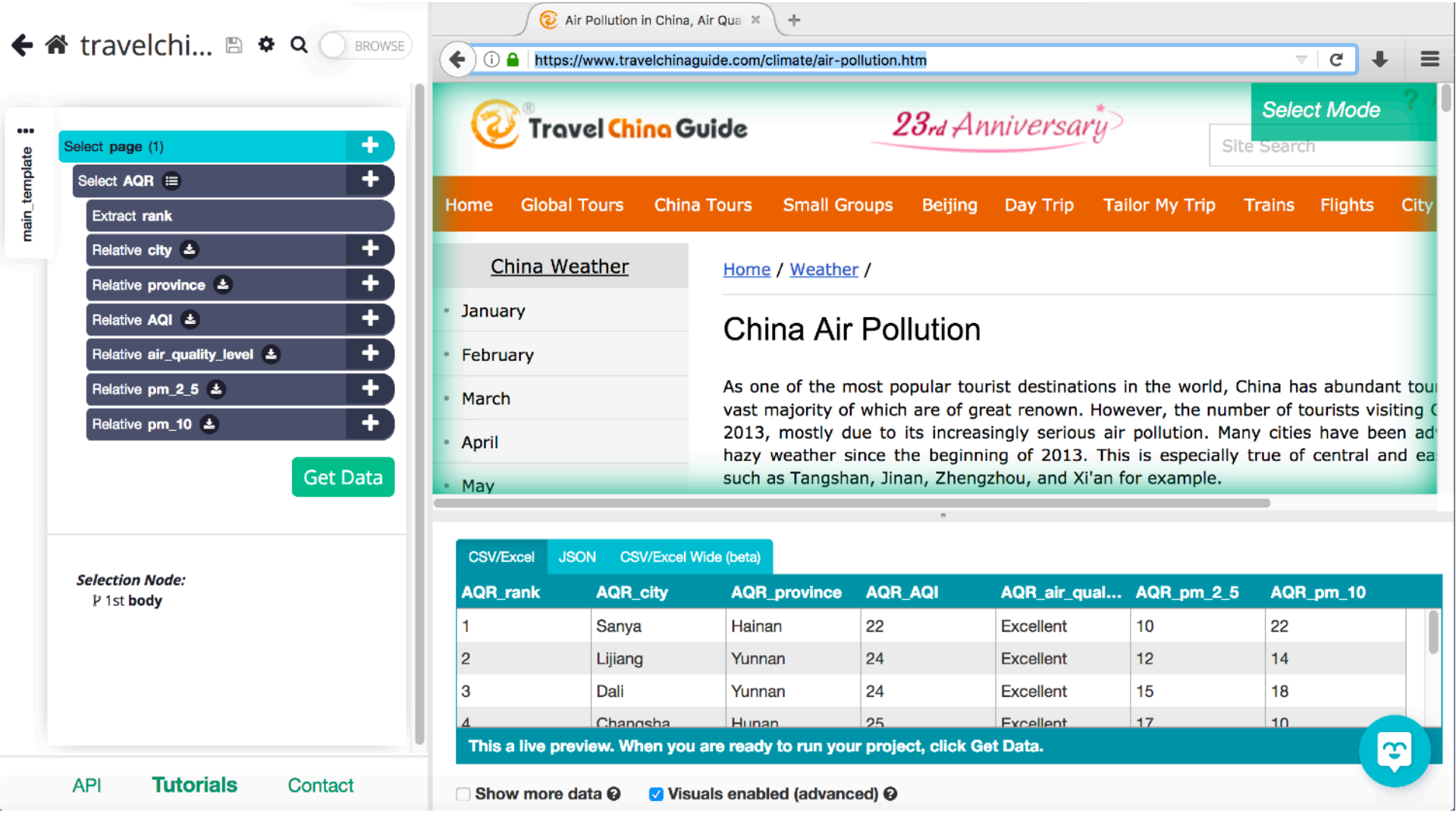
Scraping is part of a bigger picture.

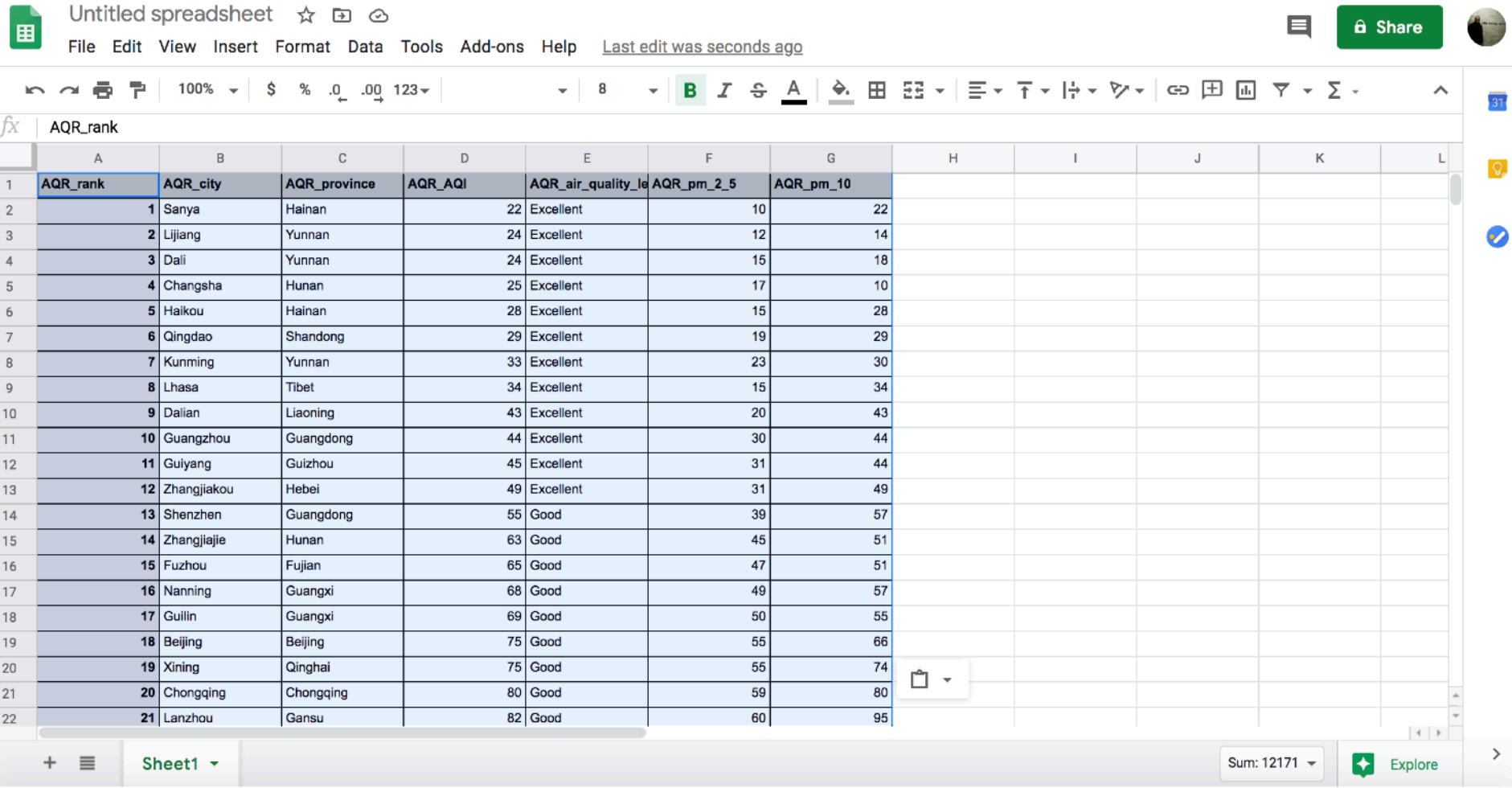
- Coding Tools for Data Mining in Python



Scraping Demo Using No-code Tools and Codes

Scraping with ParseHub


travelchi...
BROWSE
main_template
Select page (1)
Select AQR
Extract rank
Relative city
Relative province
Relative AQI
Relative air_quality_level
Relative pm_2_5
Relative pm_10
Get Data
Selection Node:
1st body
API
Tutorials
Contact
Air Pollution in China, Air Qua
https://www.travelchinaguide.com/climate/air-pollution.htm
Travel China Guide
23rd Anniversary
Select Mode
Site Search
Home
Global Tours
China Tours
Small Groups
Beijing
Day Trip
Tailor My Trip
Trains
Flights
City
China Weather
January
February
March
April
May
Home / Weather /
China Air Pollution
As one of the most popular tourist destinations in the world, China has abundant tou
vast majority of which are of great renown. However, the number of tourists visiting
2013, mostly due to its increasingly serious air pollution. Many cities have been ad
hazy weather since the beginning of 2013. This is especially true of central and ea
such as Tangshan, Jinan, Zhengzhou, and Xi'an for example.
CSV/Excel
JSON
CSV/Excel Wide (beta)
AQR_rank
AQR_city
AQR_province
AQR_AQI
AQR_air_qual...
AQR_pm_2_5
AQR_pm_10
1
Sanya
Hainan
22
Excellent
10
22
2
Lijiang
Yunnan
24
Excellent
12
14
3
Dali
Yunnan
24
Excellent
15
18
This a live preview. When you are ready to run your project, click Get Data.
Show more data
Visuals enabled (advanced)



AQR_rank	AQR_city	AQR_province	AQR_AQI	AQR_air_quality_le	AQR_pm_2_5	AQR_pm_10
1	Sanya	Hainan	22	Excellent	10	22
2	Lijiang	Yunnan	24	Excellent	12	14
3	Dali	Yunnan	24	Excellent	15	18
4	Changsha	Hunan	25	Excellent	17	10
5	Haikou	Hainan	28	Excellent	15	28
6	Qingdao	Shandong	29	Excellent	19	29
7	Kunming	Yunnan	33	Excellent	23	30
8	Lhasa	Tibet	34	Excellent	15	34
9	Dalian	Liaoning	43	Excellent	20	43
10	Guangzhou	Guangdong	44	Excellent	30	44
11	Guiyang	Guizhou	45	Excellent	31	44
12	Zhangjiakou	Hebei	49	Excellent	31	49
13	Shenzhen	Guangdong	55	Good	39	57
14	Zhangjiajie	Hunan	63	Good	45	51
15	Fuzhou	Fujian	65	Good	47	51
16	Nanning	Guangxi	68	Good	49	57
17	Guilin	Guangxi	69	Good	50	55
18	Beijing	Beijing	75	Good	55	66
19	Xining	Qinghai	75	Good	55	74
20	Chongqing	Chongqing	80	Good	59	80
21	Lanzhou	Gansu	82	Good	60	95

Untitled spreadsheet

File Edit View Insert Format Data Tools Add-ons Help Last edit was seconds ago

Share

100% $ % .0 .00 123 Default (Ari... 10

	A	B	D
1	AQR_rank	AQR_city	AQR_AQI
2	1	Sanya	22
3	2	Lijiang	24
4	3	Dali	24
5	4	Changsha	25
6	5	Haikou	28
7	6	Qingdao	29
8	7	Kunming	33
9	8	Lhasa	34
10	9	Dalian	43
11	10	Guangzhou	44
12	11	Guiyang	45
13	12	Zhangjiakou	49
14	13	Shenzhen	55
15	14	Zhangjiajie	63
16	15	Fuzhou	65
17	16	Nanning	68
18	17	Guilin	69
19	18	Beijing	75
20	19	Xining	75
21	20	Chongqing	80
22	21	Lanzhou	82

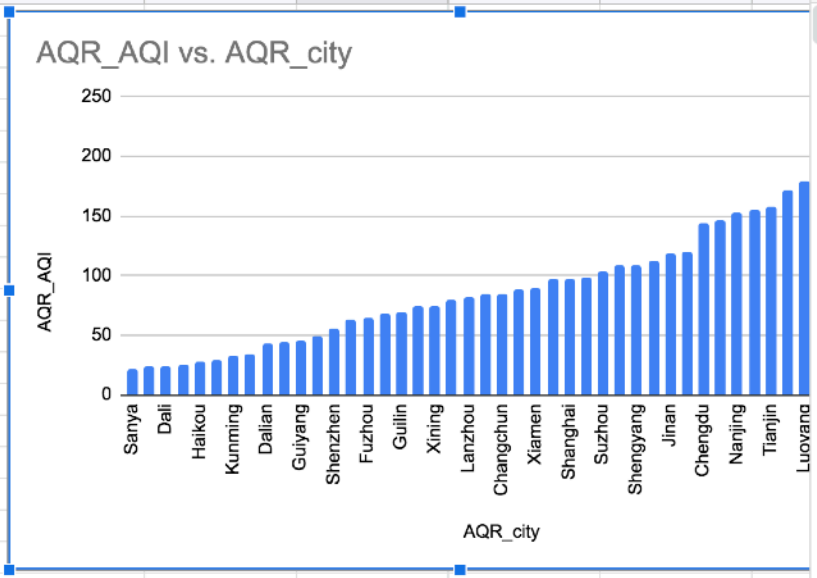


Chart editor

Setup Customize

Chart type

Column chart

Stacking

None

Data range

B1:D46

X-axis

AQR_city

Aggregate

Series

123 AQR_AQI

Sheet1

Explore

Untitled spreadsheet

Saved to Drive

File Edit View Insert Format Data Tools Add-ons Help Last edit was seconds ago

Share

100% $ % .0 .00 123 Default (Ari... 10

	B	D	F
1	AQR_city	AQR_AQI	AQR_pm_2_5
2	Sanya	22	10
3	Lijiang	24	12
4	Dali	24	15
5	Changsha	25	17
6	Haikou	28	15
7	Qingdao	29	19
8	Kunming	33	23
9	Lhasa	34	15
10	Dalian	43	20
11	Guangzhou	44	30
12	Guiyang	45	31
13	Zhangjiakou	49	31
14	Shenzhen	55	39
15	Zhangjiajie	63	45
16	Fuzhou	65	47
17	Nanning	68	49
18	Guilin	69	50
19	Beijing	75	55
20	Xining	75	55
21	Chongqing	80	59
22	Lanzhou	82	60

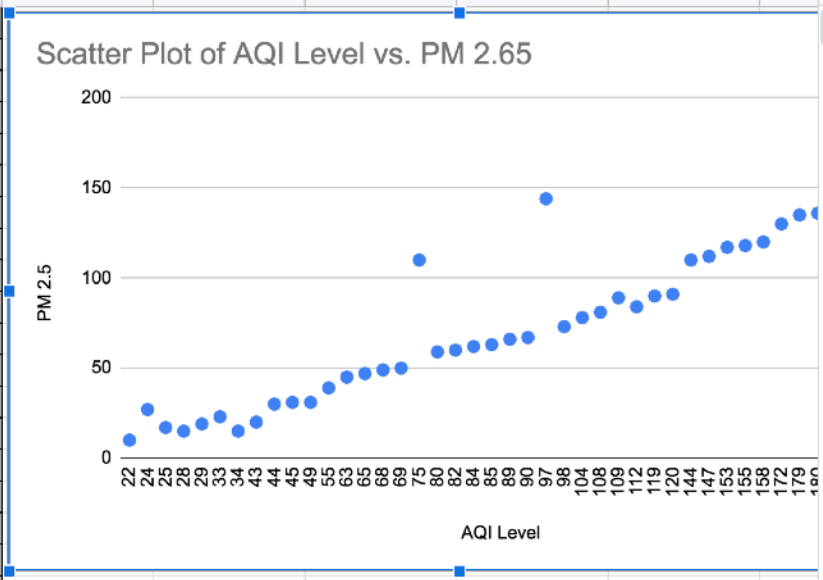


Chart editor

Setup Customize

Chart type

Scatter chart

Data range

D1:F46

X-axis

D1:D46

Aggregate

Series Sum

F1:F46 Sum

Add Series

Include hidden / filtered data

Sheet1

Explore

Untitled spreadsheet

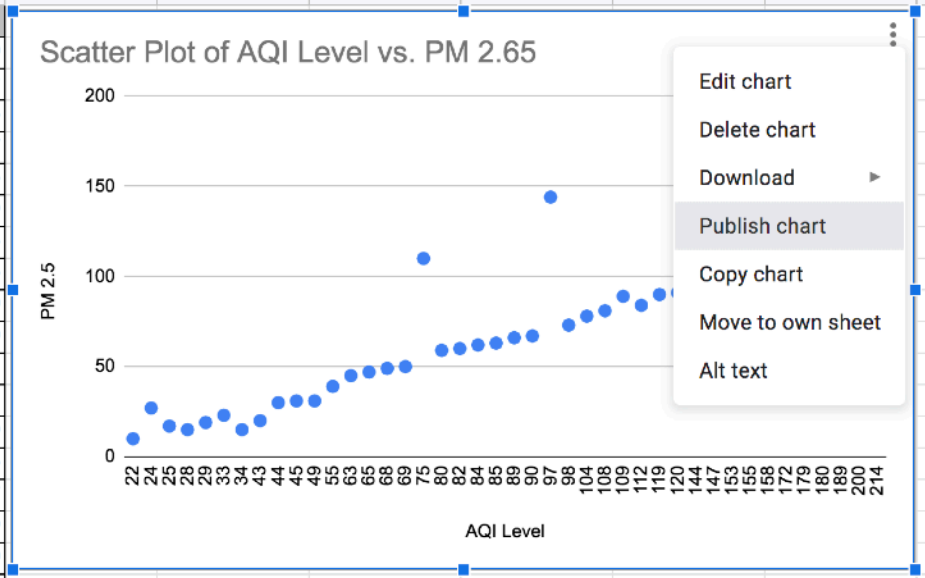
File Edit View Insert Format Data Tools Add-ons Help Last edit was 3 hours ago

Share

100% $ % .0 .00 123 Helvetica ... 8

fx AQR_rank

	A	B	D	F
1	AQR_rank	AQR_city	AQR_AQI	AQR_pm_2_5
2	1	Sanya	22	10
3	2	Lijiang	24	12
4	3	Dali	24	15
5	4	Changsha	25	17
6	5	Haikou	28	15
7	6	Qingdao	29	19
8	7	Kunming	33	23
9	8	Lhasa	34	15
10	9	Dalian	43	20
11	10	Guangzhou	44	30
12	11	Guiyang	45	31
13	12	Zhangjiakou	49	31
14	13	Shenzhen	55	39
15	14	Zhangjiajie	63	45
16	15	Fuzhou	65	47
17	16	Nanning	68	49
18	17	Guilin	69	50
19	18	Beijing	75	55
20	19	Xining	75	55
21	20	Chongqing	80	59
22	21	Lanzhou	82	60



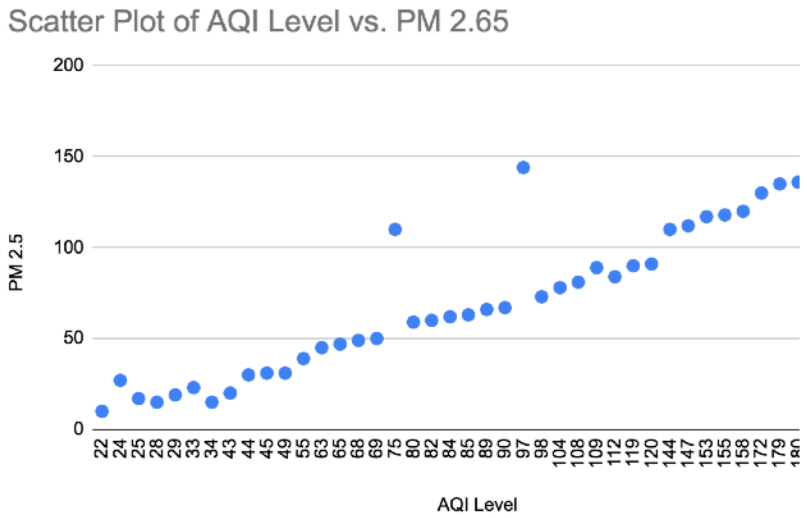
Sheet1

Explore

A wonderful serenity

Has taken possession of my entire soul, like these sweet mornings of spring which I enjoy with my whole heart.

Choose an option



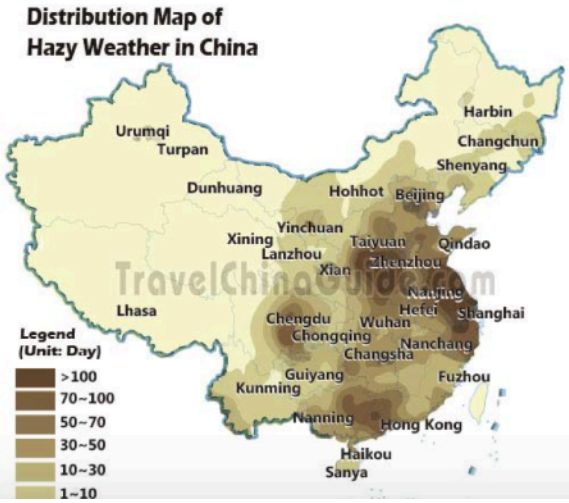
Scraping with Python Beautiful Soup

China Weather
- January
- February
- March
- April
- May
- June
- July
- August
- September
- October
- November
- December
- Climate Change
- Air Pollution

China Air Pollution

As one of the most popular tourist destinations in the world, China has abundant tourism resources, the vast majority of which are of great renown. However, the number of tourists visiting China declined from 2013, mostly due to its increasingly serious air pollution. Many cities have been adversely affected by hazy weather since the beginning of 2013. This is especially true of central and eastern China's cities such as Tangshan, Jinan, Zhengzhou, and Xi'an for example.

It is known to all that the rapid development of China's economy in the past 30 years resulted in environmental degradation. Nevertheless, the hazy weather records of China in 2013 attracted global attention. It is a pity that as a consequence some foreign residents choose to give up their jobs and life in China.



https://www.travelchinaguide.com/climate/air-pollution.htm

File Edit View Insert Cell Kernel Widgets Help Trusted Python 3

Run Code

Presenting China Pollution Figures Using Beautiful Soup, Pandas and Matplotlib

In [2]:

```
from bs4 import BeautifulSoup
import requests
import csv
import pandas as pd
import matplotlib.pyplot as plt

# Fetch URL
html_page = requests.get('https://www.travelchinaguide.com/climate/air-pollution.htm')
# Obtain the entire HTML page
soup = BeautifulSoup(html_page.content,'html.parser')
# Find all the HTML tables
tables = soup.find_all(class_="c_tableX")
# Access the second HTML table (i.e. tables[1] instrad of tables[0]) that contains Air Quality information
# with both table header and table data
table = tables[1] # assign 2nd table in the table list to variable called "table"
table_header = table.find_all('th') # extract table header based on the 'th' tag
# print table header tags and texts
# print(table_header)
header = []
data = []
for th in table.find_all('th'):
    # print header text
    # print(th.text)
    header.append(th.text)
# print list of headers
# print(header)
all_rows = table.find_all("tr") # extract all the table rows
# Enumerate all the rows to extract needed data
# Obtain both index and values using the enumerate function starting at 2nd row
```

File Edit View Insert Cell Kernel Widgets Help Trusted Python 3

Run Code

```
98 plt.show()
```

```
Air Quality Ranking

    Rank          City Province belongs to  AQI    Air Quality Level PM 2.5  \
0      1         Sanya               Hainan   22            Excellent     10
1      2       Lijiang              Yunnan   24            Excellent     12
2      3          Dali              Yunnan   24            Excellent     15
3      4      Changsha               Hunan   25            Excellent     17
4      5        Haikou              Hainan   28            Excellent     15
5      6       Qingdao            Shandong   29            Excellent     19
6      7       Kunming              Yunnan   33            Excellent     23
7      8         Lhasa               Tibet   34            Excellent     15
8      9        Dalian            Liaoning   43            Excellent     20
9     10     Guangzhou           Guangdong   44            Excellent     30
10    11       Guiyang             Guizhou   45            Excellent     31
11    12   Zhangjiakou               Hebei   49            Excellent     31
12    13      Shenzhen           Guangdong   55                 Good     39
13    14   Zhangjiajie               Hunan   63                 Good     45
14    15        Fuzhou              Fujian   65                 Good     47
15    16       Nanning             Guangxi   68                 Good     49
```

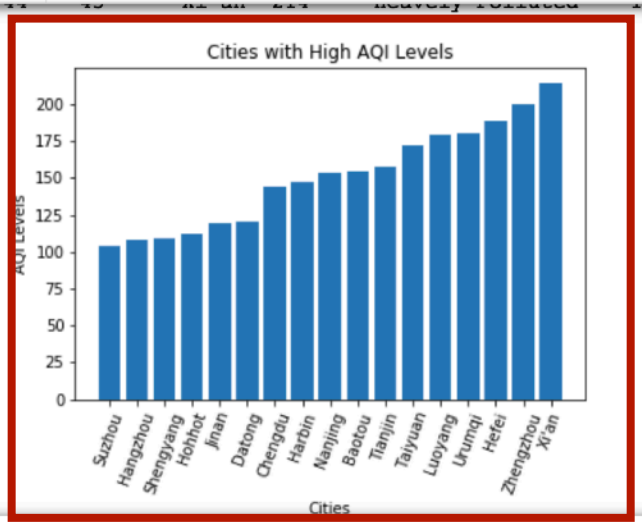
In [3]:

```
1 from matplotlib import pyplot as plt
2 # aqi_lvl = pc['AQI'].tolist()
3 # pm_2_5 = pc['PM 2.5'].tolist()
4 ax = plt.subplot()
5 plt.plot(aqi_lvl, pm_2_5, 'o')
6 plt.ylabel('PM 2.5')
7 plt.xlabel('AQI Levels')
8 plt.show()
```

File Edit View Insert Cell Kernel Widgets Help Trusted Python 3

Run Code

```
98 plt.show()
```



In [3]:

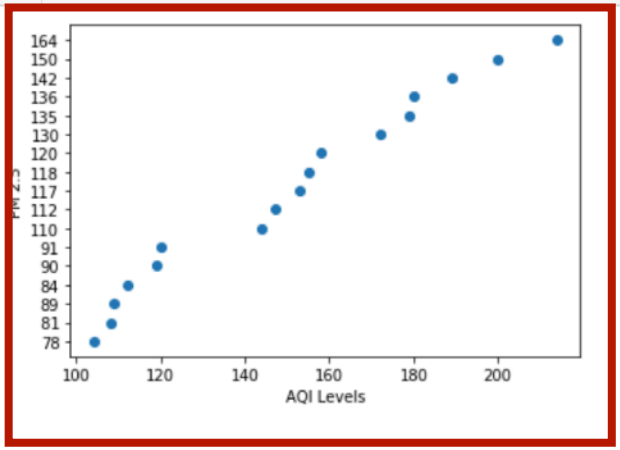
```
1 from matplotlib import pyplot as plt
2 # aqi_lvl = pc['AQI'].tolist()
3 # pm_2_5 = pc['PM 2.5'].tolist()
4 ax = plt.subplot()
5 plt.plot(aqi_lvl, pm_2_5, 'o')
6 plt.ylabel('PM 2.5')
7 plt.xlabel('AQI Levels')
8 plt.show()
```

File Edit View Insert Cell Kernel Widgets Help Trusted Python 3

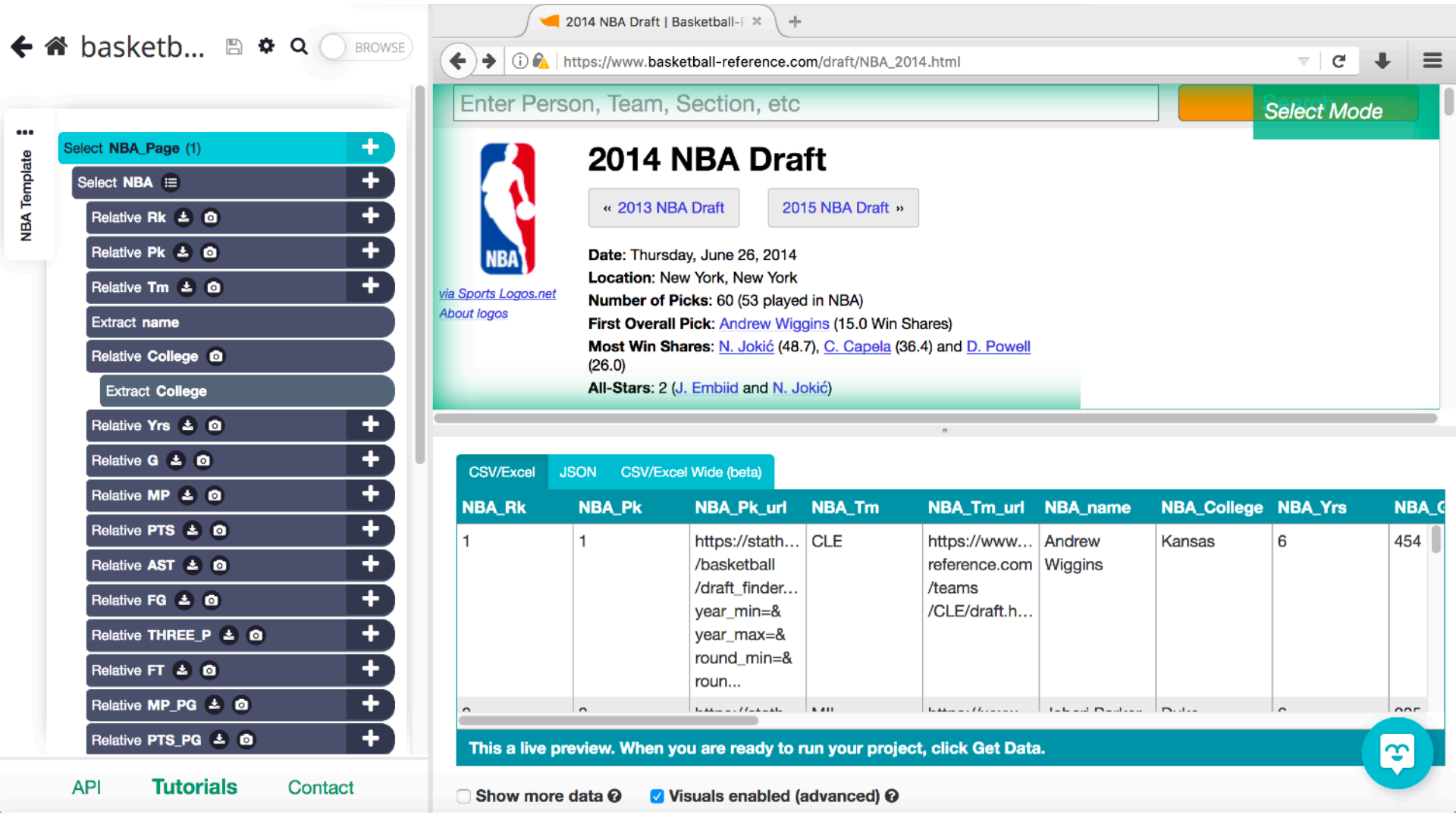
Run Code

In [3]:

```
from matplotlib import pyplot as plt
# aqi_lvl = pc['AQI'].tolist()
# pm_2_5 = pc['PM 2.5'].tolist()
ax = plt.subplot()
plt.plot(aqi_lvl, pm_2_5, 'o')
plt.ylabel('PM 2.5')
plt.xlabel('AQI Levels')
plt.show()
```



Clean up data in OpenRefine.


basketb...
BROWSE
NBA Template
Select NBA_Page (1)
Select NBA
Relative Rk
Relative Pk
Relative Tm
Extract name
Relative College
Extract College
Relative Yrs
Relative G
Relative MP
Relative PTS
Relative AST
Relative FG
Relative THREE_P
Relative FT
Relative MP_PG
Relative PTS_PG
API
Tutorials
Contact
2014 NBA Draft | Basketball-
https://www.basketball-reference.com/draft/NBA_2014.html
Enter Person, Team, Section, etc
Select Mode
2014 NBA Draft
« 2013 NBA Draft
2015 NBA Draft »
via Sports Logos.net
About logos
Date: Thursday, June 26, 2014
Location: New York, New York
Number of Picks: 60 (53 played in NBA)
First Overall Pick: Andrew Wiggins (15.0 Win Shares)
Most Win Shares: N. Jokić (48.7), C. Capela (36.4) and D. Powell (26.0)
All-Stars: 2 (J. Embiid and N. Jokić)
CSV/Excel
JSON
CSV/Excel Wide (beta)
NBA_Rk
NBA_Pk
NBA_Pk_url
NBA_Tm
NBA_Tm_url
NBA_name
NBA_College
NBA_Yrs
1
1
https://stath... /basketball /draft_finder... year_min=& year_max=& round_min=& roun...
CLE
https://www... reference.com /teams /CLE/draft.h...
Andrew Wiggins
Kansas
6
454
This a live preview. When you are ready to run your project, click Get Data.
Show more data
Visuals enabled (advanced)

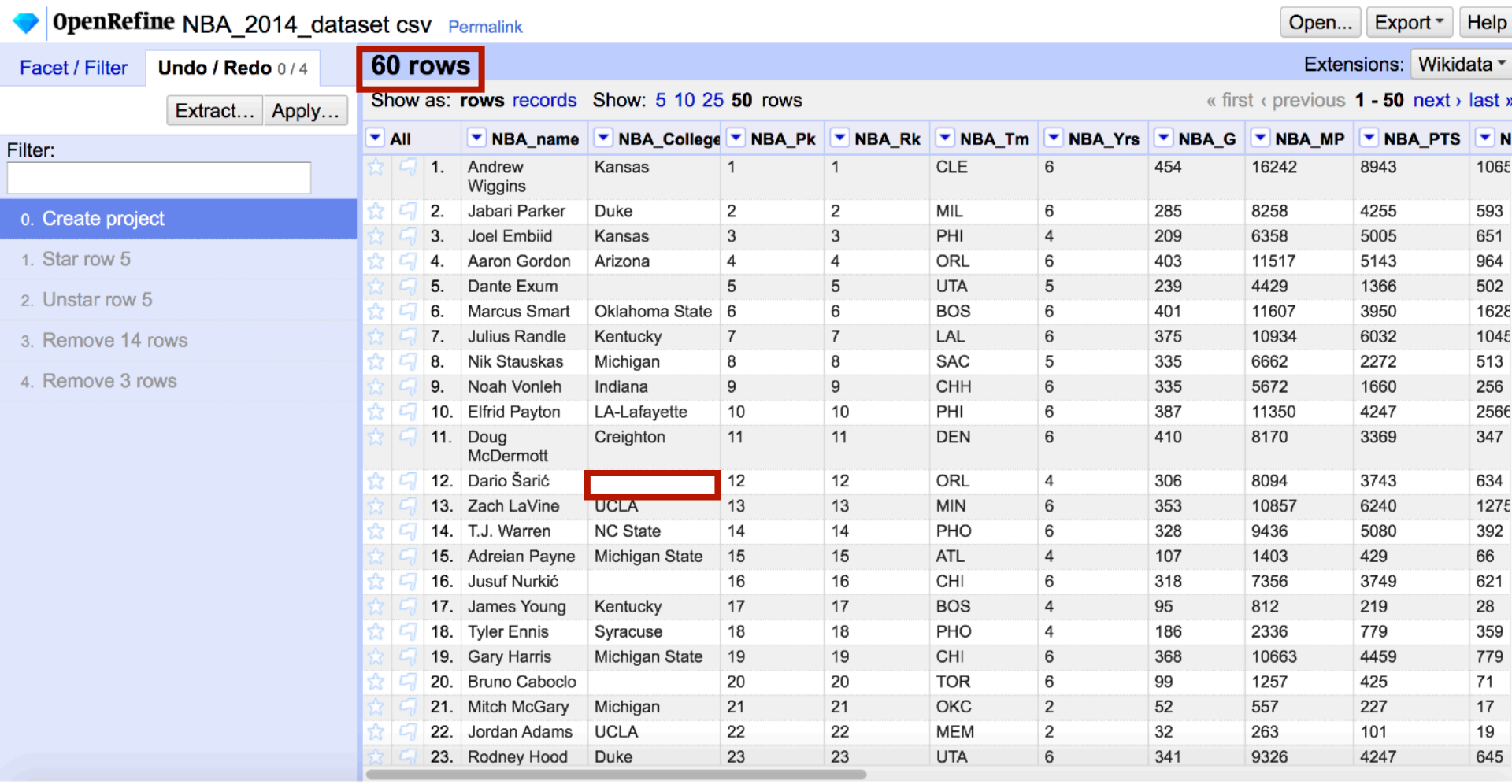
Most Basic Data Cleaning Tasks

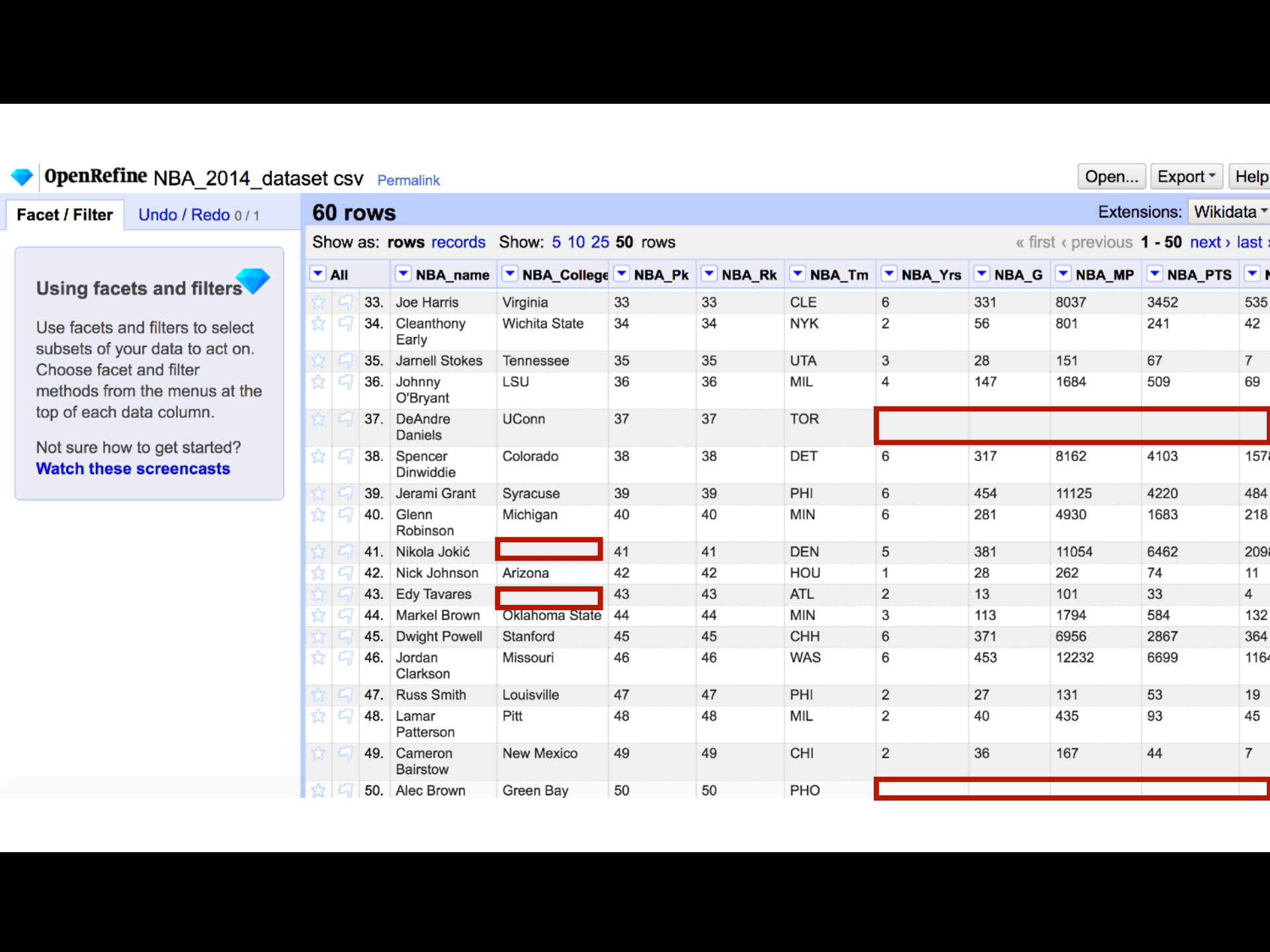
1. Remove blanks
2. Impute missing entries (need expertise to judge usage instead of relying on statistics alone)
3. Remove duplicates
4. Resolve inconsistent entries
5. Transform incorrect data (e.g. wrong data types such as text instead of numeric and calculations)
6. Rename, combine and split columns for making further data presentation and processing easier

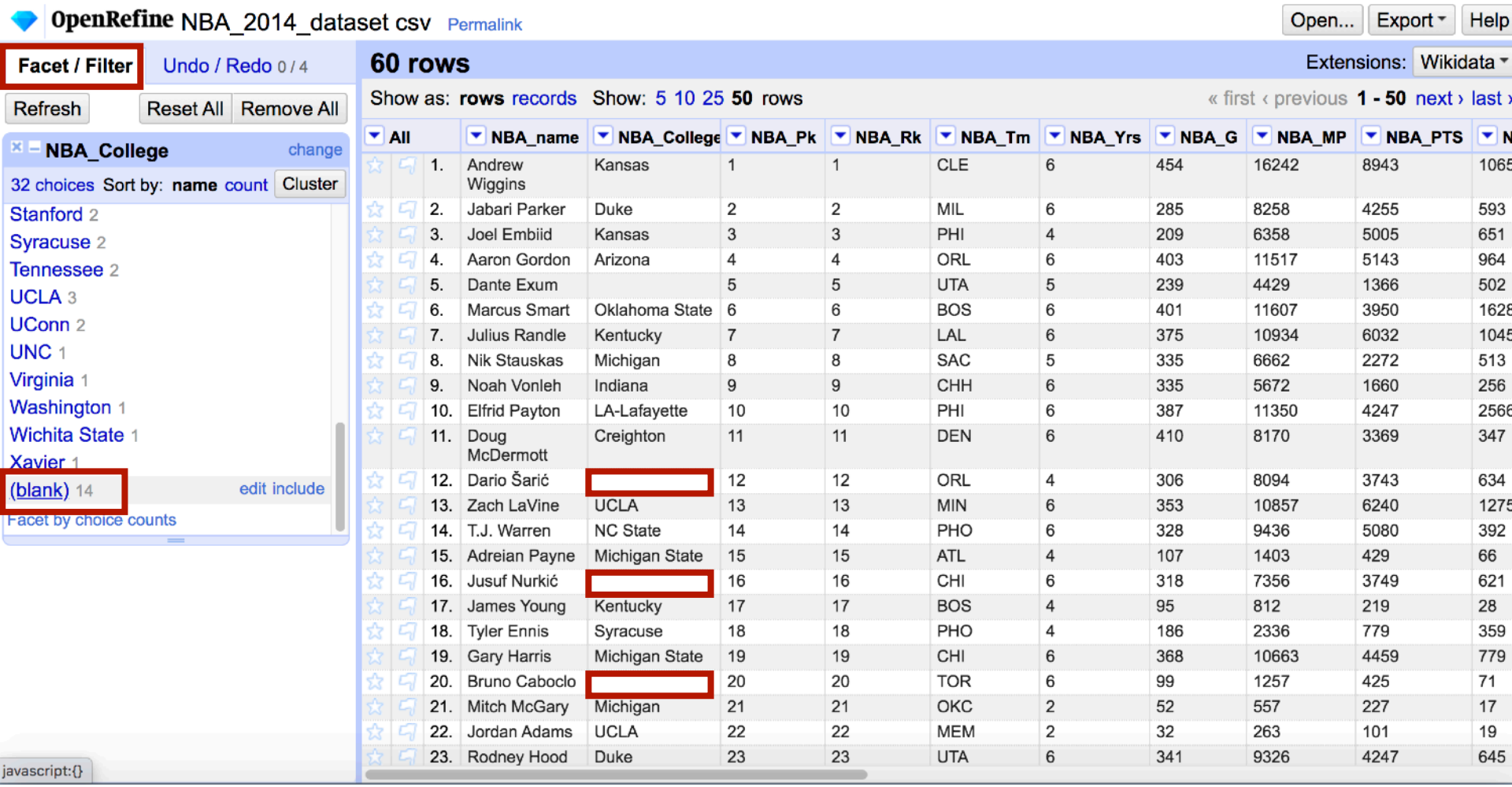
NBA_name	NBA_College	NBA_Pk	NBA_Rk	NBA_Tm	NBA_Yrs	NBA_G	NBA_MP	NBA_PTS	NBA_AST	NBA_FG	NBA_THREE_P	NBA_
Andrew Wiggins	Kansas	1	1	CLE	6	454	16242	8943	1065	0.441	0.332	0.7
Jabari Parker	Duke	2	2	MIL	6	285	8258	4255	593	0.493	0.324	0.7
Joel Embiid	Kansas	3	3	PHI	4	209	6358	5005	651	0.480	0.319	0.7
Aaron Gordon	Arizona	4	4	ORL	6	403	11517	5143	964	0.448	0.319	0.7
Dante Exum		5	5	UTA	5	239	4429	1366	502	0.407	0.308	0.7
Marcus Smart	Oklahoma State	6	6	BOS	6	401	11607	3950	1628	0.373	0.318	0.7
Julius Randle	Kentucky	7	7	LAL	6	375	10934	6032	1045	0.493	0.295	0.7
Nik Stauskas	Michigan	8	8	SAC	5	335	6662	2272	513	0.389	0.353	0.8
Noah Vonleh	Indiana	9	9	CHH	6	335	5672	1660	256	0.460	0.310	0.6
Elfrid Payton	LA-Lafayette	10	10	PHI	6	387	11350	4247	2566	0.452	0.289	0.6
Doug McDermott	Creighton	11	11	DEN	6	410	8170	3369	347	0.465	0.412	0.8
Dario Šarić		12	12	ORL	4	306	8094	3743	634	0.441	0.358	0.8
Zach LaVine	UCLA	13	13	MIN	6	353	10857	6240	1275	0.447	0.375	0.8
T.J. Warren	NC State	14	14	PHO	6	328	9436	5080	392	0.507	0.361	0.7
Adreian Payne	Michigan State	15	15	ATL	4	107	1403	429	66	0.406	0.254	0.6
Jusuf Nurkić		16	16	CHI	6	318	7356	3749	621	0.492	0.096	0.6
James Young	Kentucky	17	17	BOS	4	95	812	219	28	0.367	0.277	0.5
Tyler Ennis	Syracuse	18	18	PHO	4	186	2336	779	359	0.419	0.317	0.7
Gary Harris	Michigan State	19	19	CHI	6	368	10663	4459	779	0.454	0.360	0.8
Bruno Caboclo		20	20	TOR	6	99	1257	425	71	0.400	0.316	0.8
Mitch McGary	Michigan	21	21	OKC	2	52	557	227	17	0.527	0.000	0.5
Jordan Adams	UCLA	22	22	MEM	2	32	263	101	19	0.402	0.385	0.6
Rodney Hood	Duke	23	23	UTA	6	341	9326	4247	645	0.426	0.372	0.8

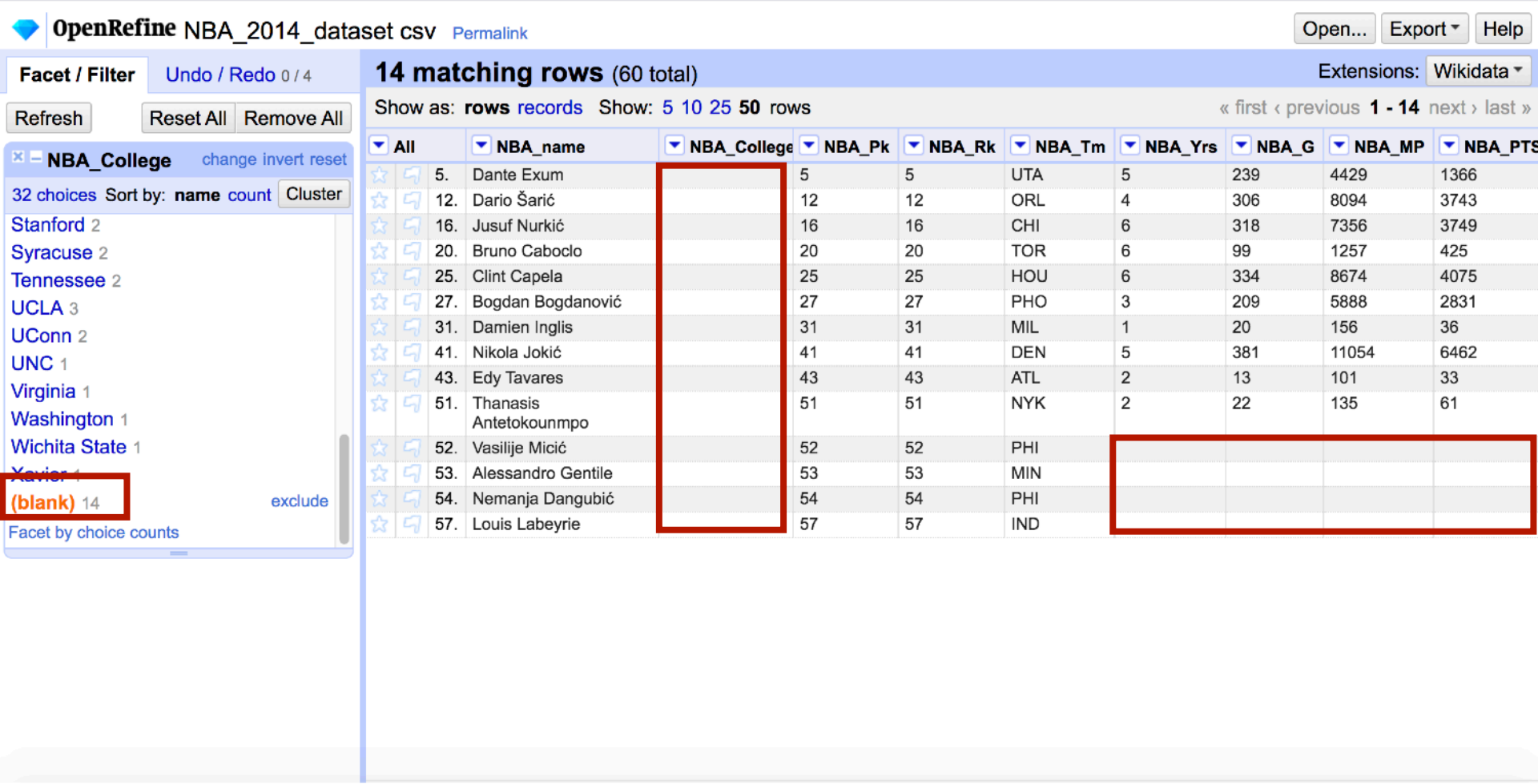
Sheet 1

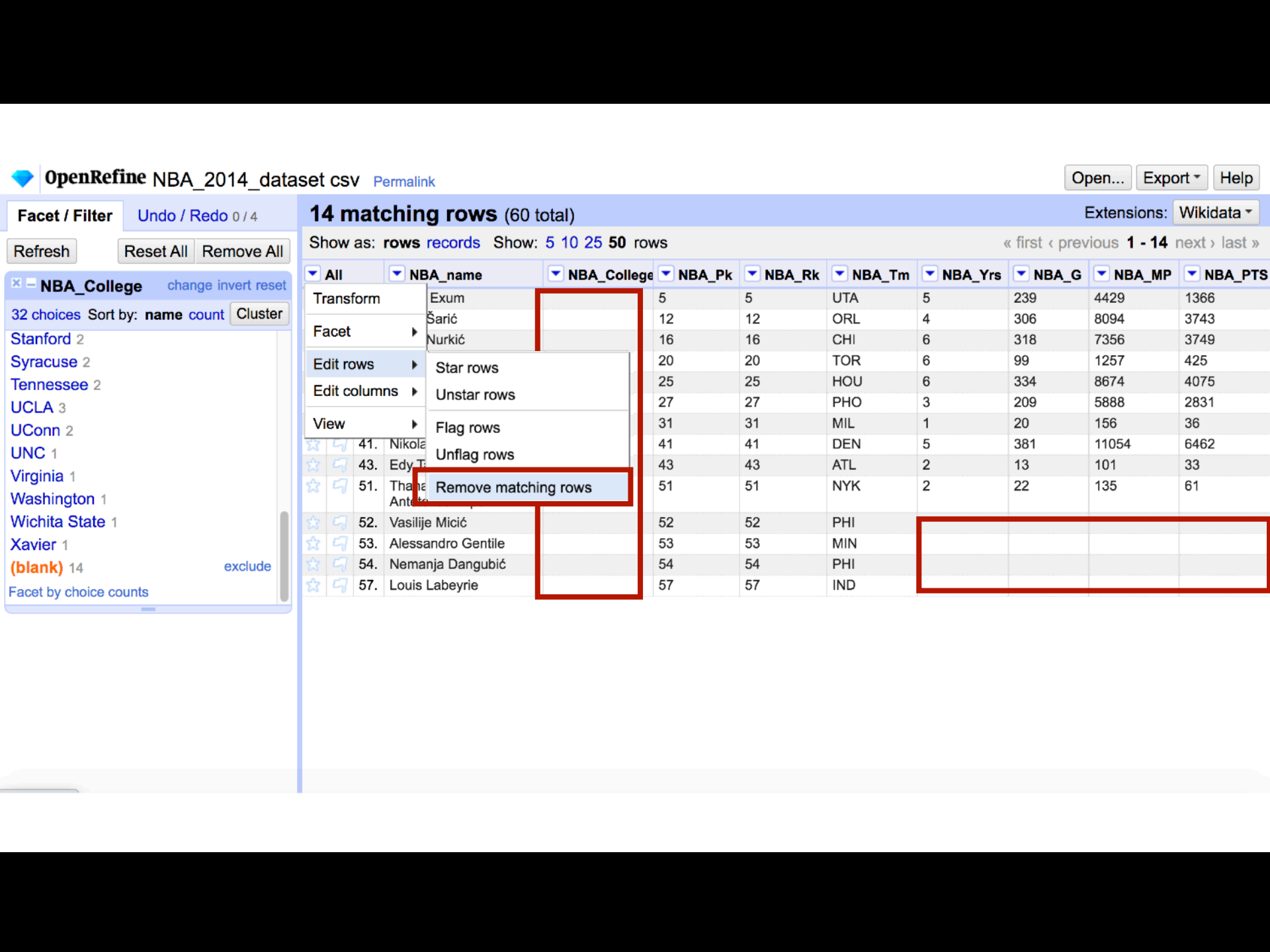
DeAndre Daniels	UConn	37	37	TOR								
Spencer Dinwiddie	Colorado	38	38	DET	6	317	8162	4103	1578	0.410	0.318	0.7
Jerami Grant	Syracuse	39	39	PHI	6	454	11125	4220	484	0.465	0.347	0.6
Glenn Robinson	Michigan	40	40	MIN	6	281	4930	1683	218	0.459	0.373	0.7
Nikola Jokić		41	41	DEN	5	381	11054	6462	2098	0.524	0.338	0.8
Nick Johnson	Arizona	42	42	HOU	1	28	262	74	11	0.347	0.238	0.6
Edy Tavares		43	43	ATL	2	13	101	33	4	0.625		0.2
Markel Brown	Oklahoma State	44	44	MIN	3	113	1794	584	132	0.380	0.295	0.7
Dwight Powell	Stanford	45	45	CHH	6	371	6956	2867	364	0.563	0.293	0.7
Jordan Clarkson	Missouri	46	46	WAS	6	453	12232	6699	1164	0.446	0.342	0.8
Russ Smith	Louisville	47	47	PHI	2	27	131	53	19	0.319	0.188	0.7
Lamar Patterson	Pitt	48	48	MIL	2	40	435	93	45	0.326	0.236	0.7
Cameron Bairstow	New Mexico	49	49	CHI	2	36	167	44	7	0.296	0.200	0.8
Alec Brown	Green Bay	50	50	PHO								
Thanasis Antetokounmpo		51	51	NYK	2	22	135	61	15	0.519	0.000	0.4
Vasilije Micić		52	52	PHI								
Alessandro Gentile		53	53	MIN								
Nemanja Dangubić		54	54	PHI								
Semaj Christon	Xavier	55	55	MIA	1	64	973	183	130	0.345	0.190	0.5
Devyn Marble	Iowa	56	56	DEN	2	44	457	97	29	0.304	0.222	0.3
Louis Labeyrie		57	57	IND								
Jordan McRae	Tennessee	58	58	SAS	4	123	1696	846	167	0.417	0.355	0.7
Xavier Thames	San Diego State	59	59	TOR								
Cory Jefferson	Baylor	60	60	SAS	2	58	581	205	16	0.444	0.125	0.5

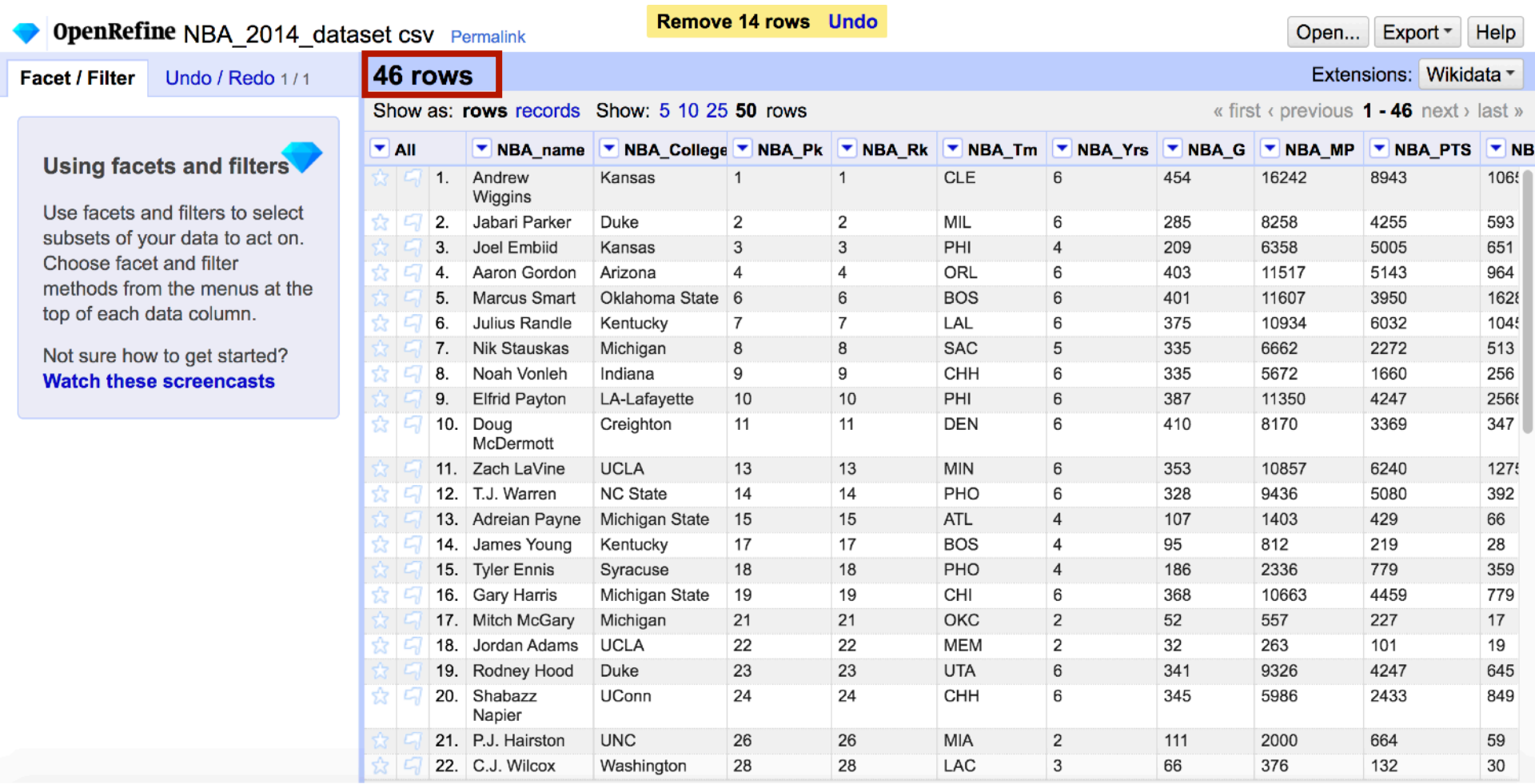

OpenRefine NBA_2014_dataset csv Permalink
Open... Export Help
Facet / Filter
Undo / Redo 0 / 4
Extract... Apply...
Filter:
0. Create project
1. Star row 5
2. Unstar row 5
3. Remove 14 rows
4. Remove 3 rows
60 rows
Extensions: Wikidata
Show as: rows records Show: 5 10 25 50 rows
« first ‹ previous 1 - 50 next › last »
All
NBA_name
NBA_College
NBA_Pk
NBA_Rk
NBA_Tm
NBA_Yrs
NBA_G
NBA_MP
NBA_PTS
1. Andrew Wiggins Kansas 1 1 CLE 6 454 16242 8943 1065
2. Jabari Parker Duke 2 2 MIL 6 285 8258 4255 593
3. Joel Embiid Kansas 3 3 PHI 4 209 6358 5005 651
4. Aaron Gordon Arizona 4 4 ORL 6 403 11517 5143 964
5. Dante Exum 5 5 UTA 5 239 4429 1366 502
6. Marcus Smart Oklahoma State 6 6 BOS 6 401 11607 3950 1628
7. Julius Randle Kentucky 7 7 LAL 6 375 10934 6032 1045
8. Nik Stauskas Michigan 8 8 SAC 5 335 6662 2272 513
9. Noah Vonleh Indiana 9 9 CHH 6 335 5672 1660 256
10. Elfrid Payton LA-Lafayette 10 10 PHI 6 387 11350 4247 2566
11. Doug McDermott Creighton 11 11 DEN 6 410 8170 3369 347
12. Dario Šarić 12 12 ORL 4 306 8094 3743 634
13. Zach LaVine UCLA 13 13 MIN 6 353 10857 6240 1275
14. T.J. Warren NC State 14 14 PHO 6 328 9436 5080 392
15. Adreian Payne Michigan State 15 15 ATL 4 107 1403 429 66
16. Jusuf Nurkić 16 16 CHI 6 318 7356 3749 621
17. James Young Kentucky 17 17 BOS 4 95 812 219 28
18. Tyler Ennis Syracuse 18 18 PHO 4 186 2336 779 359
19. Gary Harris Michigan State 19 19 CHI 6 368 10663 4459 779
20. Bruno Caboclo 20 20 TOR 6 99 1257 425 71
21. Mitch McGary Michigan 21 21 OKC 2 52 557 227 17
22. Jordan Adams UCLA 22 22 MEM 2 32 263 101 19
23. Rodney Hood Duke 23 23 UTA 6 341 9326 4247 645


OpenRefine NBA_2014_dataset csv Permalink
Open... Export Help
Facet / Filter Undo / Redo 0 / 1
60 rows
Extensions: Wikidata
Show as: rows records Show: 5 10 25 50 rows
« first ‹ previous 1 - 50 next › last
Using facets and filters
Use facets and filters to select subsets of your data to act on. Choose facet and filter methods from the menus at the top of each data column.
Not sure how to get started?
Watch these screencasts
All
NBA_name
NBA_College
NBA_Pk
NBA_Rk
NBA_Tm
NBA_Yrs
NBA_G
NBA_MP
NBA_PTS
33. Joe Harris Virginia 33 33 CLE 6 331 8037 3452 535
34. Cleanthony Early Wichita State 34 34 NYK 2 56 801 241 42
35. Jarnell Stokes Tennessee 35 35 UTA 3 28 151 67 7
36. Johnny O'Bryant LSU 36 36 MIL 4 147 1684 509 69
37. DeAndre Daniels UConn 37 37 TOR
38. Spencer Dinwiddie Colorado 38 38 DET 6 317 8162 4103 157
39. Jerami Grant Syracuse 39 39 PHI 6 454 11125 4220 484
40. Glenn Robinson Michigan 40 40 MIN 6 281 4930 1683 218
41. Nikola Jokić 41 41 DEN 5 381 11054 6462 209
42. Nick Johnson Arizona 42 42 HOU 1 28 262 74 11
43. Edy Tavares 43 43 ATL 2 13 101 33 4
44. Markel Brown Oklahoma State 44 44 MIN 3 113 1794 584 132
45. Dwight Powell Stanford 45 45 CHH 6 371 6956 2867 364
46. Jordan Clarkson Missouri 46 46 WAS 6 453 12232 6699 116
47. Russ Smith Louisville 47 47 PHI 2 27 131 53 19
48. Lamar Patterson Pitt 48 48 MIL 2 40 435 93 45
49. Cameron Bairstow New Mexico 49 49 CHI 2 36 167 44 7
50. Alec Brown Green Bay 50 50 PHO

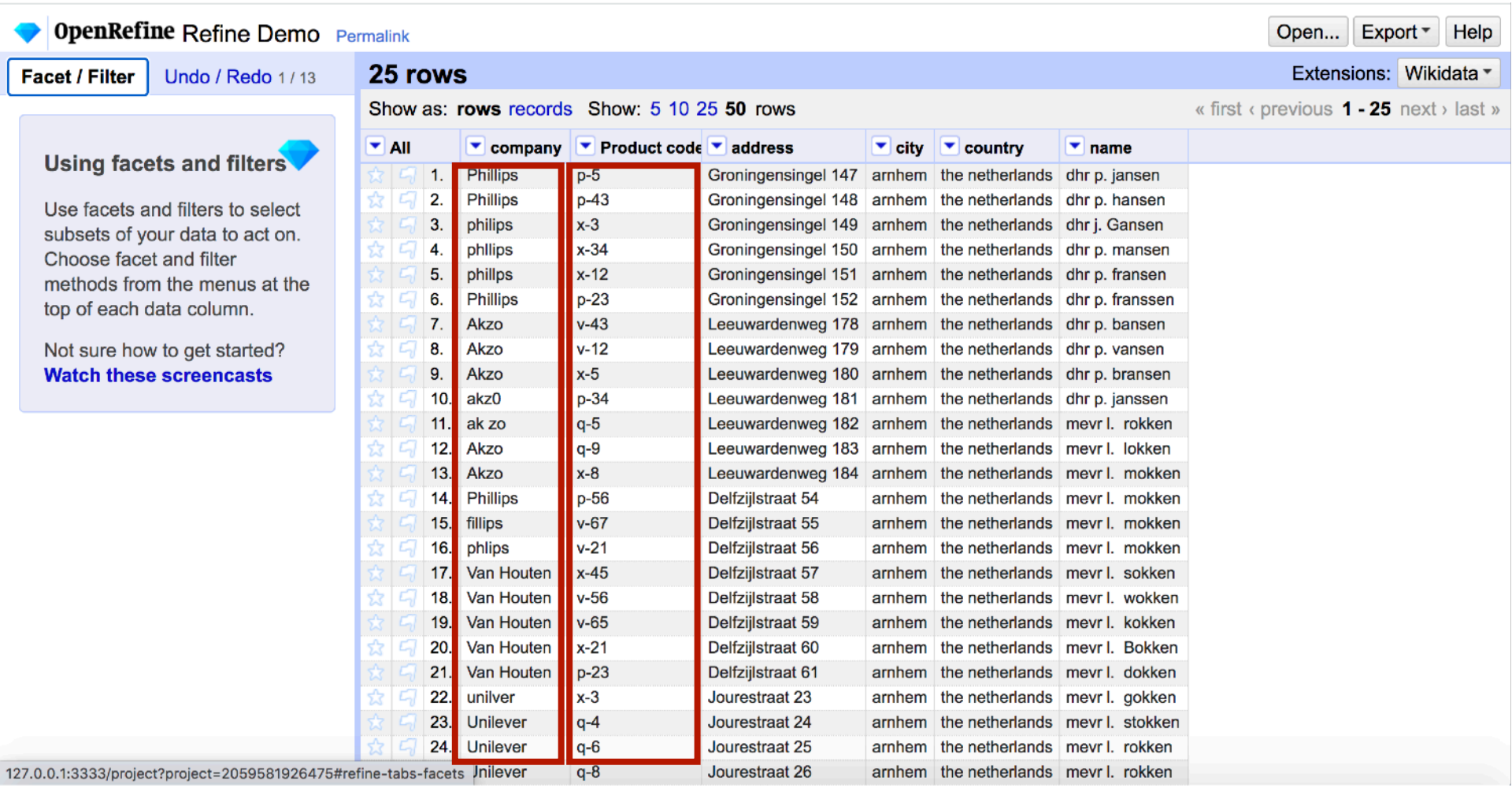

OpenRefine NBA_2014_dataset csv
Permalink
Open...
Export
Help
Facet / Filter
Undo / Redo 0 / 4
Refresh
Reset All
Remove All
NBA_College
change
32 choices Sort by: name count
Cluster
Stanford 2
Syracuse 2
Tennessee 2
UCLA 3
UConn 2
UNC 1
Virginia 1
Washington 1
Wichita State 1
Xavier 1
(blank) 14
edit include
Facet by choice counts
60 rows
Extensions: Wikidata
Show as: rows records
Show: 5 10 25 50 rows
« first ‹ previous 1 - 50 next › last »
All
NBA_name
NBA_College
NBA_Pk
NBA_Rk
NBA_Tm
NBA_Yrs
NBA_G
NBA_MP
NBA_PTS
1. Andrew Wiggins Kansas 1 1 CLE 6 454 16242 8943 1065
2. Jabari Parker Duke 2 2 MIL 6 285 8258 4255 593
3. Joel Embiid Kansas 3 3 PHI 4 209 6358 5005 651
4. Aaron Gordon Arizona 4 4 ORL 6 403 11517 5143 964
5. Dante Exum 5 5 UTA 5 239 4429 1366 502
6. Marcus Smart Oklahoma State 6 6 BOS 6 401 11607 3950 1628
7. Julius Randle Kentucky 7 7 LAL 6 375 10934 6032 1045
8. Nik Stauskas Michigan 8 8 SAC 5 335 6662 2272 513
9. Noah Vonleh Indiana 9 9 CHH 6 335 5672 1660 256
10. Elfrid Payton LA-Lafayette 10 10 PHI 6 387 11350 4247 2566
11. Doug McDermott Creighton 11 11 DEN 6 410 8170 3369 347
12. Dario Šarić 12 12 ORL 4 306 8094 3743 634
13. Zach LaVine UCLA 13 13 MIN 6 353 10857 6240 1275
14. T.J. Warren NC State 14 14 PHO 6 328 9436 5080 392
15. Adreian Payne Michigan State 15 15 ATL 4 107 1403 429 66
16. Jusuf Nurkić 16 16 CHI 6 318 7356 3749 621
17. James Young Kentucky 17 17 BOS 4 95 812 219 28
18. Tyler Ennis Syracuse 18 18 PHO 4 186 2336 779 359
19. Gary Harris Michigan State 19 19 CHI 6 368 10663 4459 779
20. Bruno Caboclo 20 20 TOR 6 99 1257 425 71
21. Mitch McGary Michigan 21 21 OKC 2 52 557 227 17
22. Jordan Adams UCLA 22 22 MEM 2 32 263 101 19
23. Rodney Hood Duke 23 23 UTA 6 341 9326 4247 645
javascript:{}

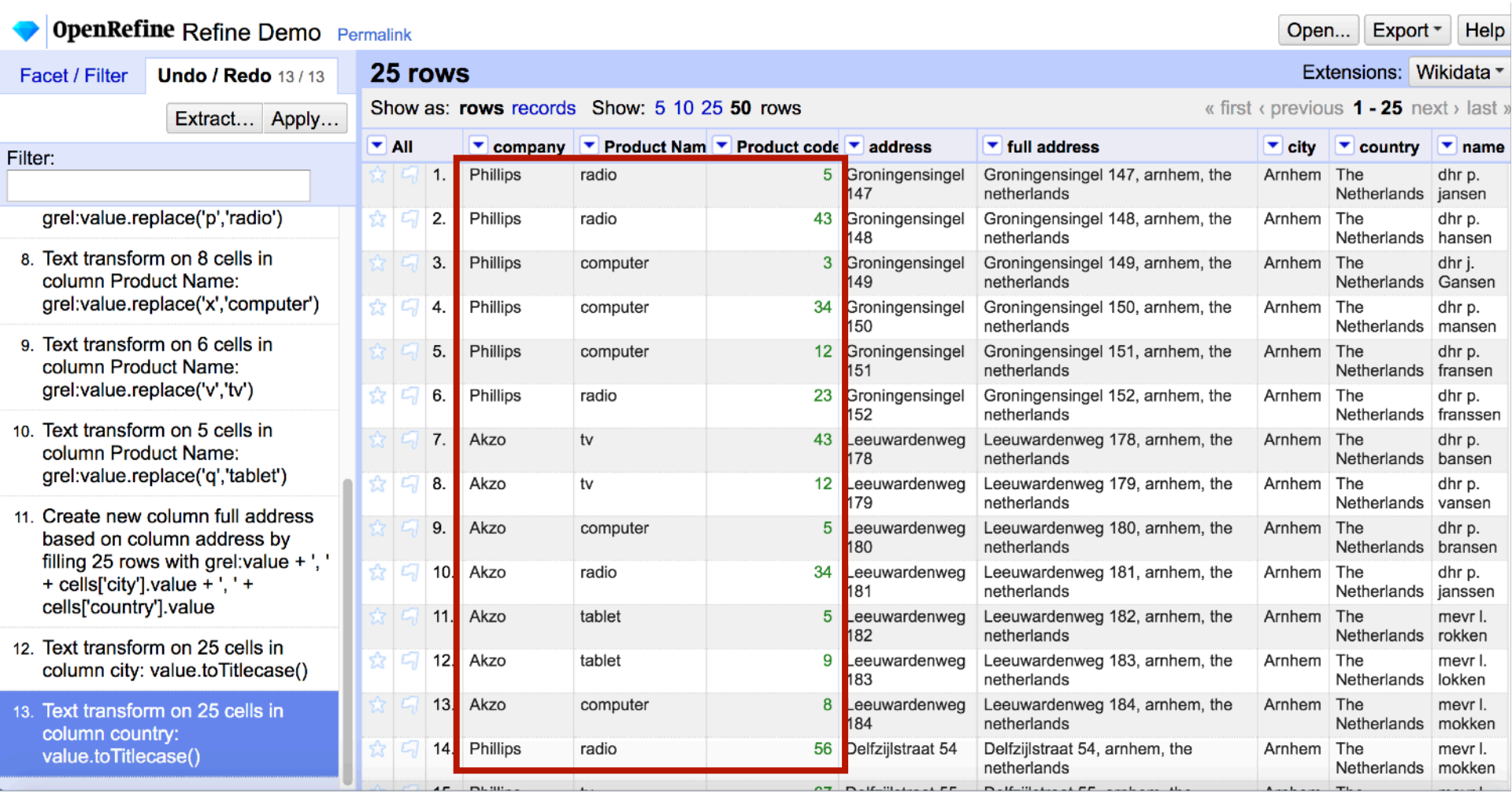

OpenRefine NBA_2014_dataset csv Permalink
Open... Export Help
Facet / Filter Undo / Redo 0 / 4
Refresh Reset All Remove All
NBA_College change invert reset
32 choices Sort by: name count Cluster
Stanford 2
Syracuse 2
Tennessee 2
UCLA 3
UConn 2
UNC 1
Virginia 1
Washington 1
Wichita State 1
(blank) 14 exclude
Facet by choice counts
14 matching rows (60 total)
Extensions: Wikidata
Show as: rows records Show: 5 10 25 50 rows
« first ‹ previous 1 - 14 next › last »
All NBA_name NBA_College NBA_Pk NBA_Rk NBA_Tm NBA_Yrs NBA_G NBA_MP NBA_PTS
5. Dante Exum 5 5 UTA 5 239 4429 1366
12. Dario Šarić 12 12 ORL 4 306 8094 3743
16. Jusuf Nurkić 16 16 CHI 6 318 7356 3749
20. Bruno Caboclo 20 20 TOR 6 99 1257 425
25. Clint Capela 25 25 HOU 6 334 8674 4075
27. Bogdan Bogdanović 27 27 PHO 3 209 5888 2831
31. Damien Inglis 31 31 MIL 1 20 156 36
41. Nikola Jokić 41 41 DEN 5 381 11054 6462
43. Edy Tavares 43 43 ATL 2 13 101 33
51. Thanasis Antetokounmpo 51 51 NYK 2 22 135 61
52. Vasilije Micić 52 52 PHI
53. Alessandro Gentile 53 53 MIN
54. Nemanja Dangubić 54 54 PHI
57. Louis Labeyrie 57 57 IND

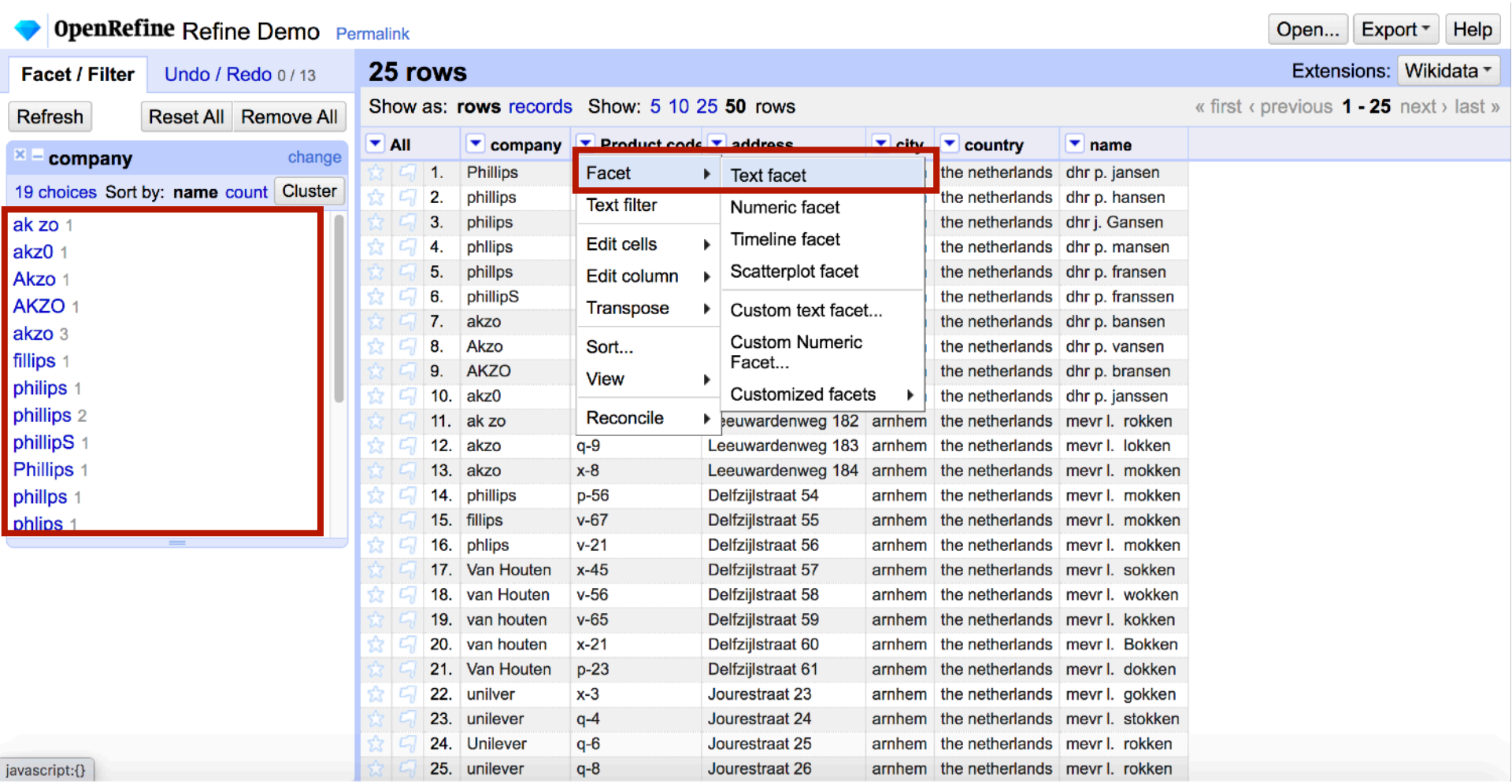

OpenRefine NBA_2014_dataset csv
Permalink
Open...
Export
Help
Facet / Filter
Undo / Redo 0 / 4
Refresh
Reset All
Remove All
NBA_College
change invert reset
32 choices Sort by: name count
Cluster
Stanford 2
Syracuse 2
Tennessee 2
UCLA 3
UConn 2
UNC 1
Virginia 1
Washington 1
Wichita State 1
Xavier 1
(blank) 14
exclude
Facet by choice counts
14 matching rows (60 total)
Extensions: Wikidata
Show as: rows records
Show: 5 10 25 50 rows
« first ‹ previous 1 - 14 next › last »
All
NBA_name
NBA_College
NBA_Pk
NBA_Rk
NBA_Tm
NBA_Yrs
NBA_G
NBA_MP
NBA_PTS
Transform
Facet
Edit rows
Edit columns
View
Star rows
Unstar rows
Flag rows
Unflag rows
Remove matching rows
Exum 5 5 UTA 5 239 4429 1366
Šarić 12 12 ORL 4 306 8094 3743
Nurkić 16 16 CHI 6 318 7356 3749
20 20 TOR 6 99 1257 425
25 25 HOU 6 334 8674 4075
27 27 PHO 3 209 5888 2831
31 31 MIL 1 20 156 36
41. Nikola 41 41 DEN 5 381 11054 6462
43. Edy T 43 43 ATL 2 13 101 33
51. Tha Ant 51 51 NYK 2 22 135 61
52. Vasilije Micić 52 52 PHI
53. Alessandro Gentile 53 53 MIN
54. Nemanja Dangubić 54 54 PHI
57. Louis Labeyrie 57 57 IND

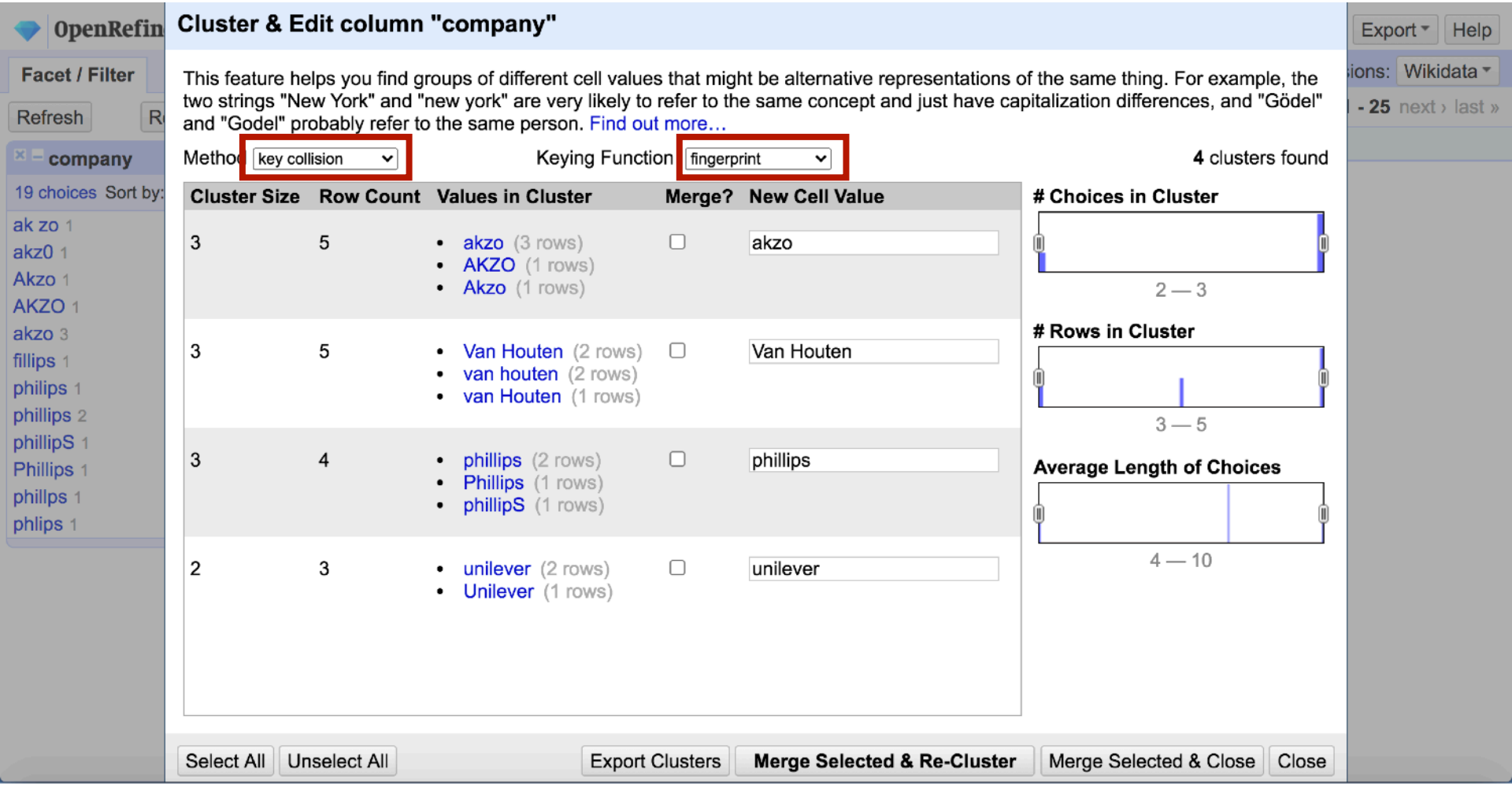

OpenRefine NBA_2014_dataset csv Permalink
Remove 14 rows Undo
Open... Export Help
Facet / Filter Undo / Redo 1 / 1
46 rows
Extensions: Wikidata
Show as: rows records Show: 5 10 25 50 rows
« first ‹ previous 1 - 46 next › last »
Using facets and filters
Use facets and filters to select subsets of your data to act on. Choose facet and filter methods from the menus at the top of each data column.
Not sure how to get started?
Watch these screencasts
All
NBA_name
NBA_College
NBA_Pk
NBA_Rk
NBA_Tm
NBA_Yrs
NBA_G
NBA_MP
NBA_PTS
1. Andrew Wiggins Kansas 1 1 CLE 6 454 16242 8943
2. Jabari Parker Duke 2 2 MIL 6 285 8258 4255 593
3. Joel Embiid Kansas 3 3 PHI 4 209 6358 5005 651
4. Aaron Gordon Arizona 4 4 ORL 6 403 11517 5143 964
5. Marcus Smart Oklahoma State 6 6 BOS 6 401 11607 3950
6. Julius Randle Kentucky 7 7 LAL 6 375 10934 6032
7. Nik Stauskas Michigan 8 8 SAC 5 335 6662 2272 513
8. Noah Vonleh Indiana 9 9 CHH 6 335 5672 1660 256
9. Elfrid Payton LA-Lafayette 10 10 PHI 6 387 11350 4247
10. Doug McDermott Creighton 11 11 DEN 6 410 8170 3369 347
11. Zach LaVine UCLA 13 13 MIN 6 353 10857 6240
12. T.J. Warren NC State 14 14 PHO 6 328 9436 5080 392
13. Adreian Payne Michigan State 15 15 ATL 4 107 1403 429 66
14. James Young Kentucky 17 17 BOS 4 95 812 219 28
15. Tyler Ennis Syracuse 18 18 PHO 4 186 2336 779 359
16. Gary Harris Michigan State 19 19 CHI 6 368 10663 4459 779
17. Mitch McGary Michigan 21 21 OKC 2 52 557 227 17
18. Jordan Adams UCLA 22 22 MEM 2 32 263 101 19
19. Rodney Hood Duke 23 23 UTA 6 341 9326 4247 645
20. Shabazz Napier UConn 24 24 CHH 6 345 5986 2433 849
21. P.J. Hairston UNC 26 26 MIA 2 111 2000 664 59
22. C.J. Wilcox Washington 28 28 LAC 3 66 376 132 30

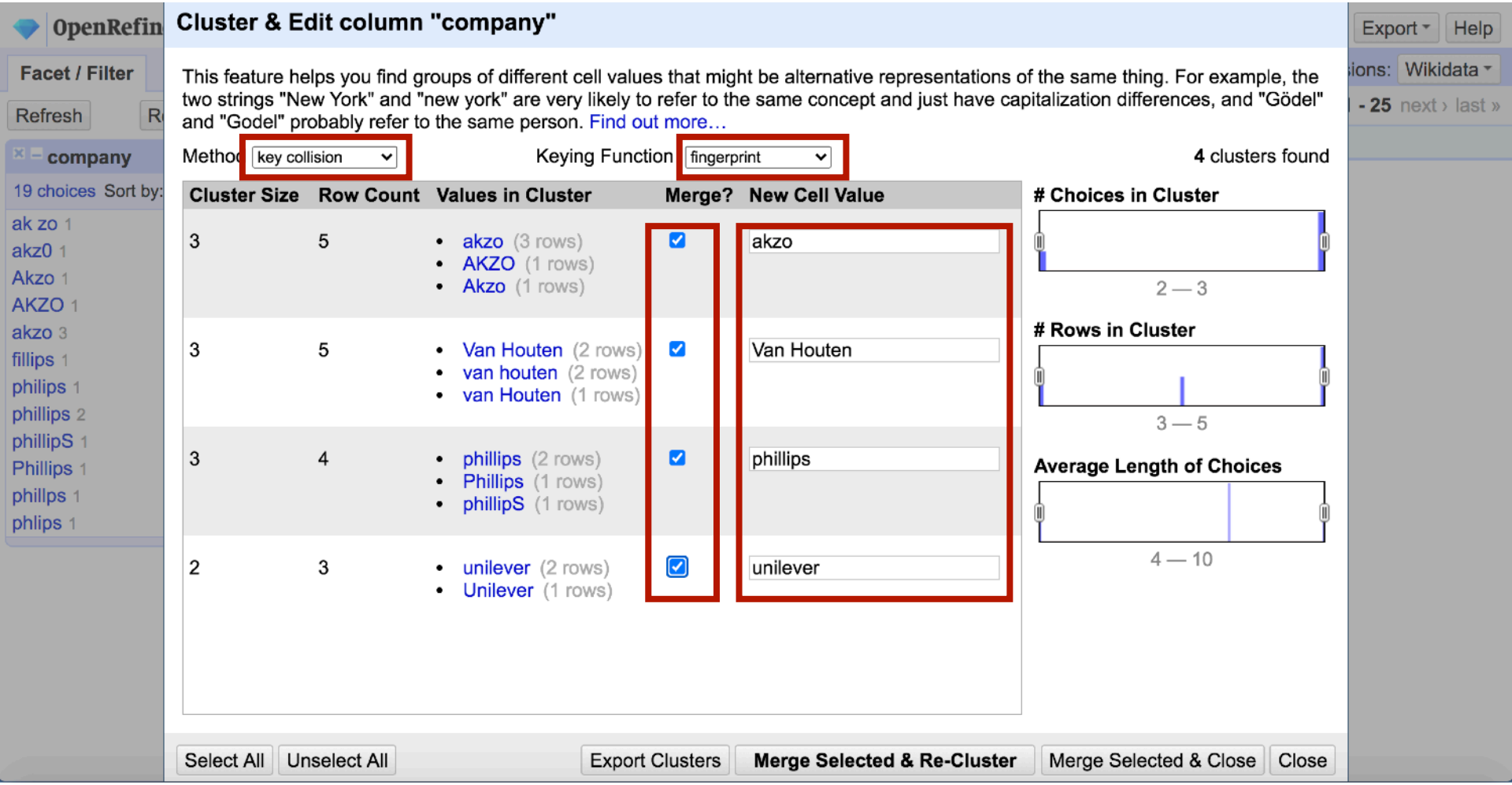
Another example.

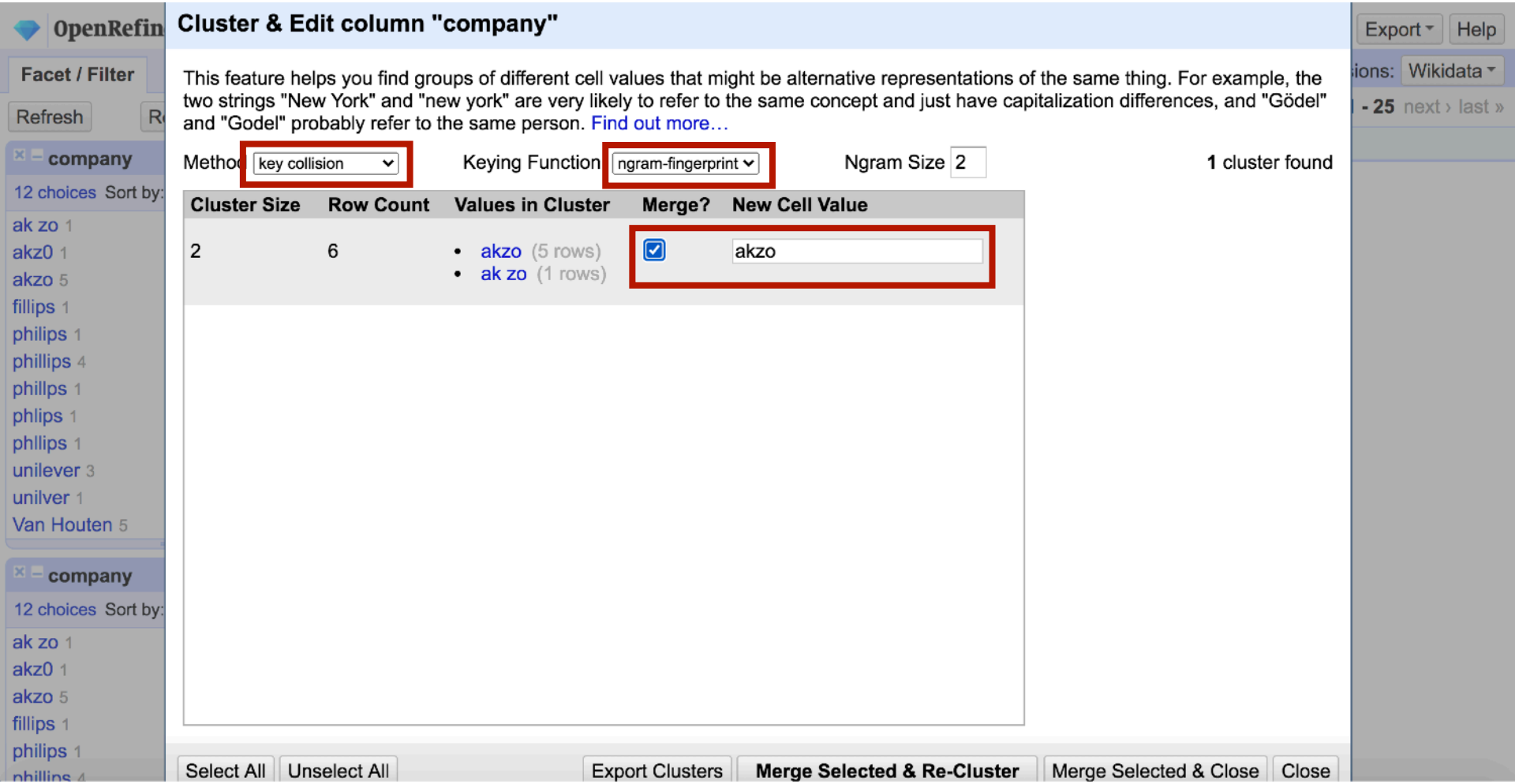

OpenRefine Refine Demo
Permalink
Open...
Export
Help
Facet / Filter
Undo / Redo 1 / 13
25 rows
Extensions: Wikidata
Show as: rows records Show: 5 10 25 50 rows
« first ‹ previous 1 - 25 next › last »
Using facets and filters
Use facets and filters to select subsets of your data to act on. Choose facet and filter methods from the menus at the top of each data column.
Not sure how to get started?
Watch these screencasts
All
company
Product code
address
city
country
name
1. Phillips p-5 Groningensingel 147 arnhem the netherlands dhr p. jansen
2. Phillips p-43 Groningensingel 148 arnhem the netherlands dhr p. hansen
3. philips x-3 Groningensingel 149 arnhem the netherlands dhr j. Gansen
4. phllips x-34 Groningensingel 150 arnhem the netherlands dhr p. mansen
5. phillps x-12 Groningensingel 151 arnhem the netherlands dhr p. fransen
6. Phillips p-23 Groningensingel 152 arnhem the netherlands dhr p. franssen
7. Akzo v-43 Leeuwardenweg 178 arnhem the netherlands dhr p. bansen
8. Akzo v-12 Leeuwardenweg 179 arnhem the netherlands dhr p. vansen
9. Akzo x-5 Leeuwardenweg 180 arnhem the netherlands dhr p. bransen
10. akz0 p-34 Leeuwardenweg 181 arnhem the netherlands dhr p. janssen
11. ak zo q-5 Leeuwardenweg 182 arnhem the netherlands mevr l. rokken
12. Akzo q-9 Leeuwardenweg 183 arnhem the netherlands mevr l. lokken
13. Akzo x-8 Leeuwardenweg 184 arnhem the netherlands mevr l. mokken
14. Phillips p-56 Delfzijlstraat 54 arnhem the netherlands mevr l. mokken
15. fillips v-67 Delfzijlstraat 55 arnhem the netherlands mevr l. mokken
16. phlips v-21 Delfzijlstraat 56 arnhem the netherlands mevr l. mokken
17. Van Houten x-45 Delfzijlstraat 57 arnhem the netherlands mevr l. sokken
18. Van Houten v-56 Delfzijlstraat 58 arnhem the netherlands mevr l. wokken
19. Van Houten v-65 Delfzijlstraat 59 arnhem the netherlands mevr l. kokken
20. Van Houten x-21 Delfzijlstraat 60 arnhem the netherlands mevr l. Bokken
21. Van Houten p-23 Delfzijlstraat 61 arnhem the netherlands mevr l. dokken
22. unilver x-3 Jourestraat 23 arnhem the netherlands mevr l. gokken
23. Unilever q-4 Jourestraat 24 arnhem the netherlands mevr l. stokken
24. Unilever q-6 Jourestraat 25 arnhem the netherlands mevr l. rokken
Unilever q-8 Jourestraat 26 arnhem the netherlands mevr l. rokken
127.0.0.1:3333/project?project=2059581926475#refine-tabs-facets

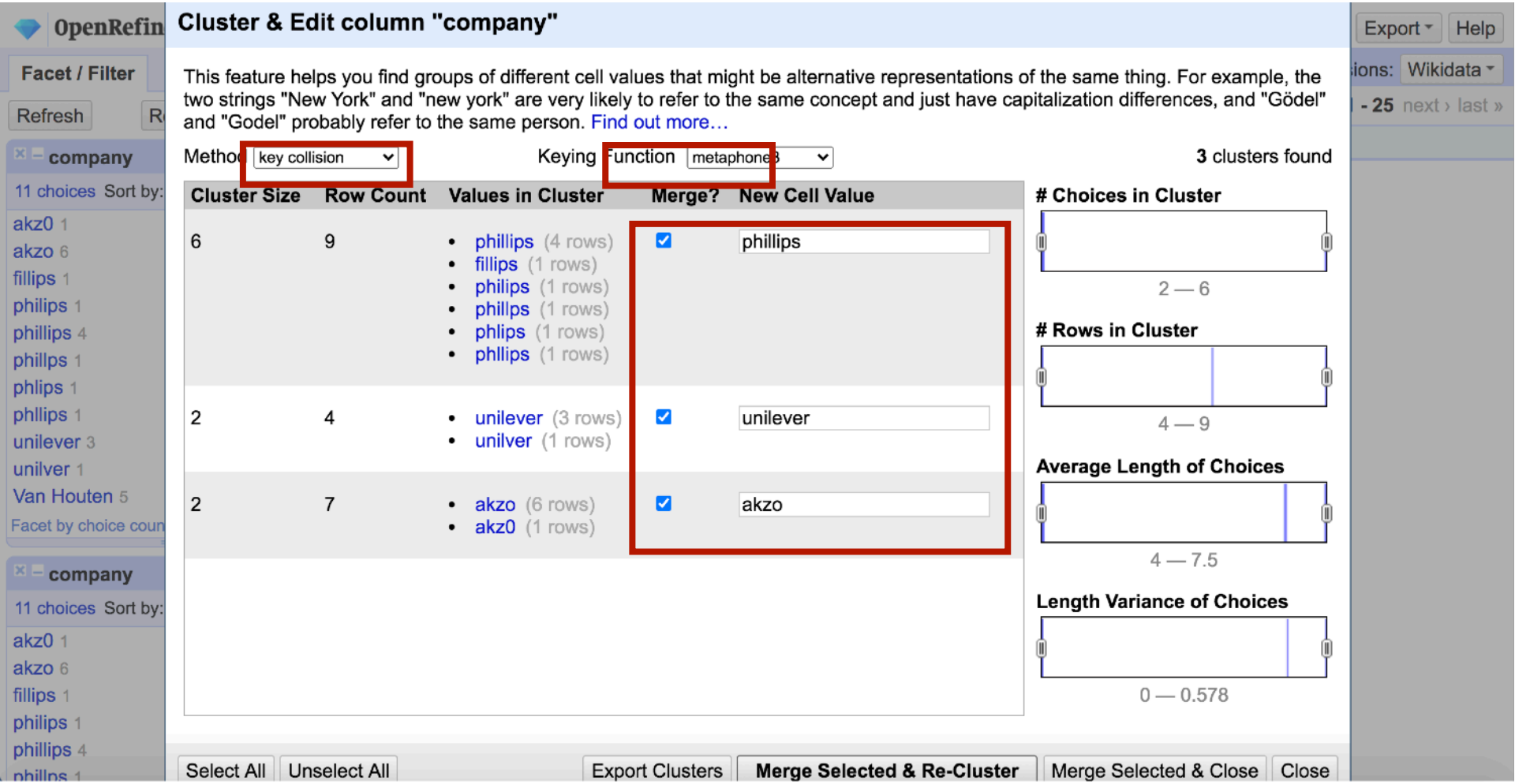

OpenRefine Refine Demo
Permalink
Open...
Export
Help
Facet / Filter
Undo / Redo 13 / 13
Extract...
Apply...
Filter:
grel:value.replace('p','radio')
8. Text transform on 8 cells in column Product Name: grel:value.replace('x','computer')
9. Text transform on 6 cells in column Product Name: grel:value.replace('v','tv')
10. Text transform on 5 cells in column Product Name: grel:value.replace('q','tablet')
11. Create new column full address based on column address by filling 25 rows with grel:value + ', ' + cells['city'].value + ', ' + cells['country'].value
12. Text transform on 25 cells in column city: value.toTitlecase()
13. Text transform on 25 cells in column country: value.toTitlecase()
25 rows
Extensions: Wikidata
Show as: rows records Show: 5 10 25 50 rows
« first ‹ previous 1 - 25 next › last »
All
company
Product Nam
Product code
address
full address
city
country
name
1. Phillips radio 5 Groningensingel 147 Groningensingel 147, arnhem, the netherlands Arnhem The Netherlands dhr p. jansen
2. Phillips radio 43 Groningensingel 148 Groningensingel 148, arnhem, the netherlands Arnhem The Netherlands dhr p. hansen
3. Phillips computer 3 Groningensingel 149 Groningensingel 149, arnhem, the netherlands Arnhem The Netherlands dhr j. Gansen
4. Phillips computer 34 Groningensingel 150 Groningensingel 150, arnhem, the netherlands Arnhem The Netherlands dhr p. mansen
5. Phillips computer 12 Groningensingel 151 Groningensingel 151, arnhem, the netherlands Arnhem The Netherlands dhr p. fransen
6. Phillips radio 23 Groningensingel 152 Groningensingel 152, arnhem, the netherlands Arnhem The Netherlands dhr p. franssen
7. Akzo tv 43 Leeuwardenweg 178 Leeuwardenweg 178, arnhem, the netherlands Arnhem The Netherlands dhr p. bansen
8. Akzo tv 12 Leeuwardenweg 179 Leeuwardenweg 179, arnhem, the netherlands Arnhem The Netherlands dhr p. vansen
9. Akzo computer 5 Leeuwardenweg 180 Leeuwardenweg 180, arnhem, the netherlands Arnhem The Netherlands dhr p. bransen
10. Akzo radio 34 Leeuwardenweg 181 Leeuwardenweg 181, arnhem, the netherlands Arnhem The Netherlands dhr p. janssen
11. Akzo tablet 5 Leeuwardenweg 182 Leeuwardenweg 182, arnhem, the netherlands Arnhem The Netherlands mevr l. rokken
12. Akzo tablet 9 Leeuwardenweg 183 Leeuwardenweg 183, arnhem, the netherlands Arnhem The Netherlands mevr l. lokken
13. Akzo computer 8 Leeuwardenweg 184 Leeuwardenweg 184, arnhem, the netherlands Arnhem The Netherlands mevr l. mokken
14. Phillips radio 56 Delfzijlstraat 54 Delfzijlstraat 54, arnhem, the netherlands Arnhem The Netherlands mevr l. mokken

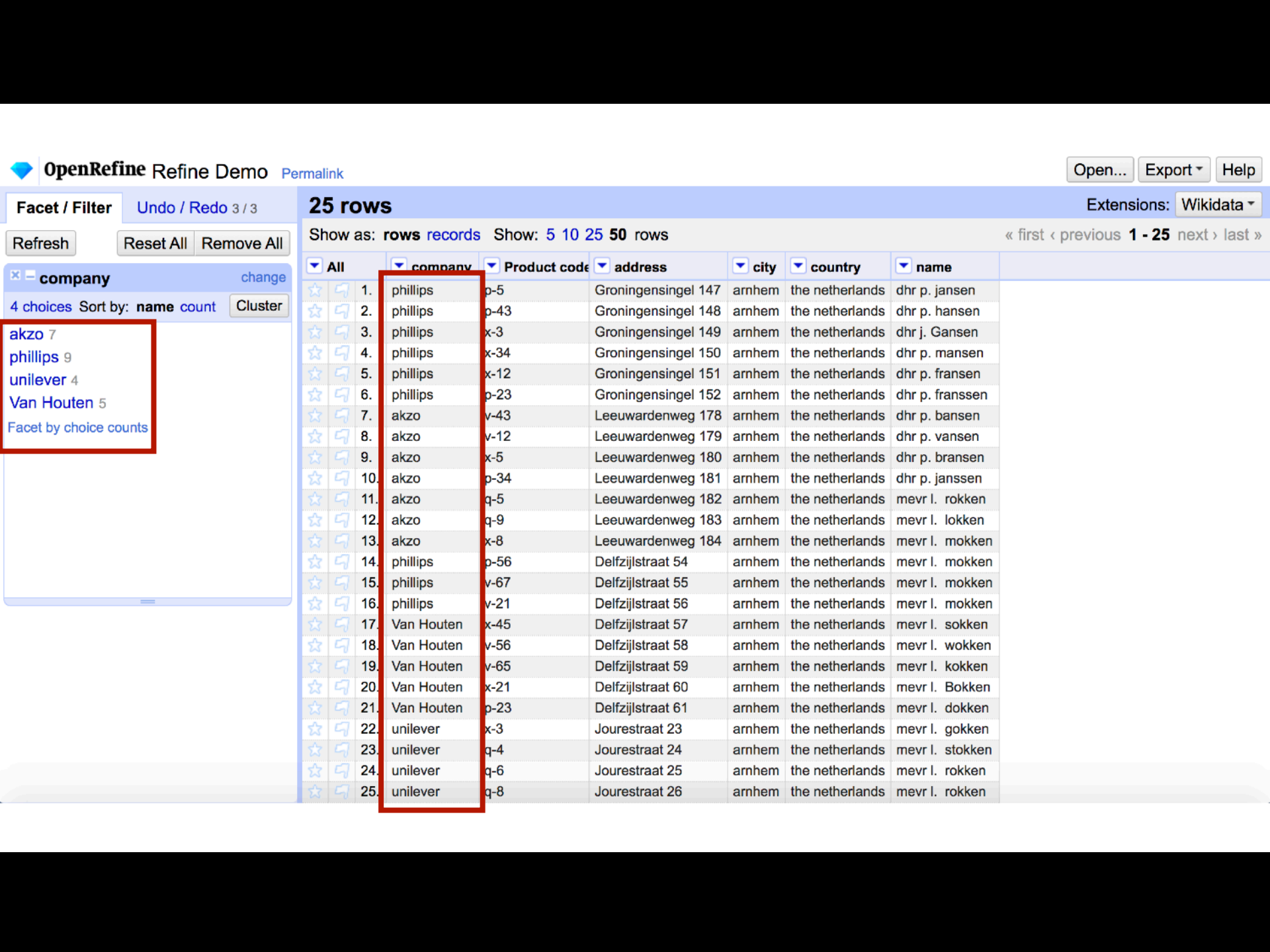

OpenRefine Refine Demo
Permalink
Open...
Export
Help
Facet / Filter
Undo / Redo 0 / 13
25 rows
Extensions:
Wikidata
Refresh
Reset All
Remove All
Show as: rows records
Show: 5 10 25 50 rows
« first ‹ previous 1 - 25 next › last »
company
change
19 choices Sort by: name count
Cluster
ak zo 1
akz0 1
Akzo 1
AKZO 1
akzo 3
fillips 1
philips 1
phillips 2
phillipS 1
Phillips 1
phillps 1
phlips 1
All
company
Product code
address
city
country
name
Facet
Text filter
Edit cells
Edit column
Transpose
Sort...
View
Reconcile
Text facet
Numeric facet
Timeline facet
Scatterplot facet
Custom text facet...
Custom Numeric Facet...
Customized facets
1. Phillips
2. phillips
3. philips
4. phllips
5. phillps
6. phillipS
7. akzo
8. Akzo
9. AKZO
10. akz0
11. ak zo
12. akzo q-9 Leeuwardenweg 183 arnhem the netherlands mevr l. lokken
13. akzo x-8 Leeuwardenweg 184 arnhem the netherlands mevr l. mokken
14. phillips p-56 Delfzijlstraat 54 arnhem the netherlands mevr l. mokken
15. fillips v-67 Delfzijlstraat 55 arnhem the netherlands mevr l. mokken
16. phlips v-21 Delfzijlstraat 56 arnhem the netherlands mevr l. mokken
17. Van Houten x-45 Delfzijlstraat 57 arnhem the netherlands mevr l. sokken
18. van Houten v-56 Delfzijlstraat 58 arnhem the netherlands mevr l. wokken
19. van houten v-65 Delfzijlstraat 59 arnhem the netherlands mevr l. kokken
20. van houten x-21 Delfzijlstraat 60 arnhem the netherlands mevr l. Bokken
21. Van Houten p-23 Delfzijlstraat 61 arnhem the netherlands mevr l. dokken
22. unilver x-3 Jourestraat 23 arnhem the netherlands mevr l. gokken
23. unilever q-4 Jourestraat 24 arnhem the netherlands mevr l. stokken
24. Unilever q-6 Jourestraat 25 arnhem the netherlands mevr l. rokken
25. unilever q-8 Jourestraat 26 arnhem the netherlands mevr l. rokken
javascript:{}


OpenRefin
Facet / Filter
Refresh
company
19 choices Sort by:
ak zo 1
akz0 1
Akzo 1
AKZO 1
akzo 3
fillips 1
philips 1
phillips 2
phillipS 1
Phillips 1
phillps 1
phlips 1
Export
Help
Wikidata
- 25 next › last »
Cluster & Edit column "company"
This feature helps you find groups of different cell values that might be alternative representations of the same thing. For example, the two strings "New York" and "new york" are very likely to refer to the same concept and just have capitalization differences, and "Gödel" and "Godel" probably refer to the same person. Find out more…
Method key collision
Keying Function fingerprint
4 clusters found
Cluster Size
Row Count
Values in Cluster
Merge?
New Cell Value
3 5 akzo (3 rows) AKZO (1 rows) Akzo (1 rows) akzo
3 5 Van Houten (2 rows) van houten (2 rows) van Houten (1 rows) Van Houten
3 4 phillips (2 rows) Phillips (1 rows) phillipS (1 rows) phillips
2 3 unilever (2 rows) Unilever (1 rows) unilever
Choices in Cluster
2 — 3
Rows in Cluster
3 — 5
Average Length of Choices
4 — 10
Select All
Unselect All
Export Clusters
Merge Selected & Re-Cluster
Merge Selected & Close
Close

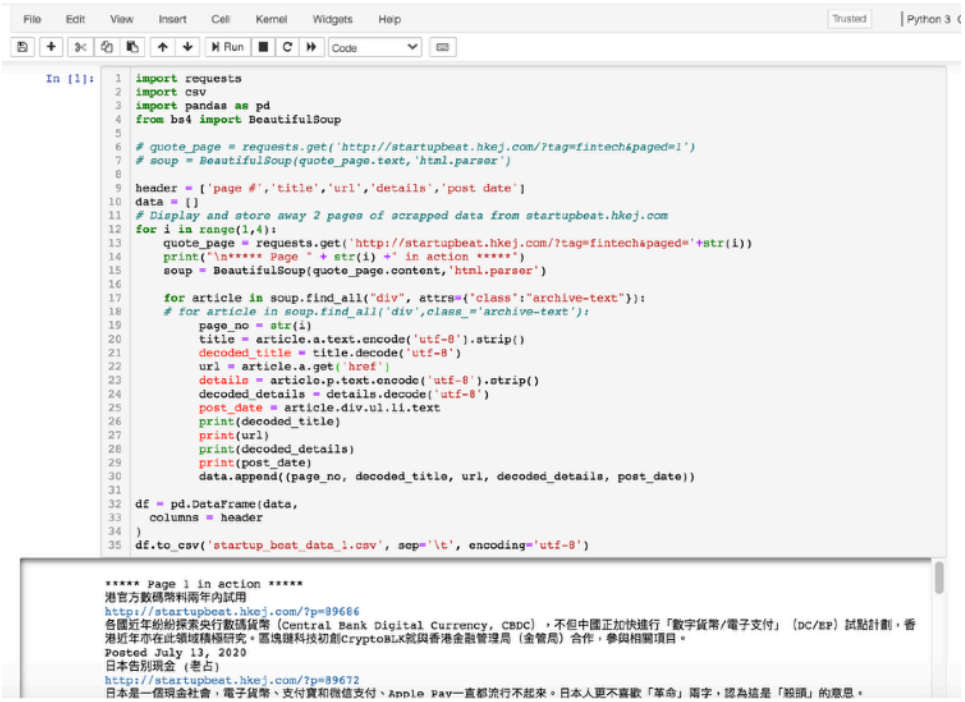

OpenRefin
Facet / Filter
Refresh
company
19 choices Sort by:
ak zo 1
akz0 1
Akzo 1
AKZO 1
akzo 3
fillips 1
philips 1
phillips 2
phillipS 1
Phillips 1
phillps 1
phlips 1
Cluster & Edit column "company"
Export
Help
Wikidata
- 25 next › last »
This feature helps you find groups of different cell values that might be alternative representations of the same thing. For example, the two strings "New York" and "new york" are very likely to refer to the same concept and just have capitalization differences, and "Gödel" and "Godel" probably refer to the same person. Find out more…
Method key collision
Keying Function fingerprint
4 clusters found
Cluster Size
Row Count
Values in Cluster
Merge?
New Cell Value
3
5
akzo (3 rows)
AKZO (1 rows)
Akzo (1 rows)
akzo
3
5
Van Houten (2 rows)
van houten (2 rows)
van Houten (1 rows)
Van Houten
3
4
phillips (2 rows)
Phillips (1 rows)
phillipS (1 rows)
phillips
2
3
unilever (2 rows)
Unilever (1 rows)
unilever
Choices in Cluster
2 — 3
Rows in Cluster
3 — 5
Average Length of Choices
4 — 10
Select All
Unselect All
Export Clusters
Merge Selected & Re-Cluster
Merge Selected & Close
Close


OpenRefin
Facet / Filter
Refresh
company
12 choices Sort by:
ak zo 1
akz0 1
akzo 5
fillips 1
philips 1
phillips 4
phillps 1
phlips 1
phllips 1
unilever 3
unilver 1
Van Houten 5
company
12 choices Sort by:
ak zo 1
akz0 1
akzo 5
fillips 1
philips 1
Export
Help
Wikidata
- 25 next › last »
Cluster & Edit column "company"
This feature helps you find groups of different cell values that might be alternative representations of the same thing. For example, the two strings "New York" and "new york" are very likely to refer to the same concept and just have capitalization differences, and "Gödel" and "Godel" probably refer to the same person. Find out more…
Metho
key collision
Keying Function
ngram-fingerprint
Ngram Size 2
1 cluster found
Cluster Size
Row Count
Values in Cluster
Merge?
New Cell Value
2
6
akzo (5 rows)
ak zo (1 rows)
akzo
Select All
Unselect All
Export Clusters
Merge Selected & Re-Cluster
Merge Selected & Close
Close


OpenRefin
Facet / Filter
Refresh
company
11 choices Sort by:
akz0 1
akzo 6
fillips 1
philips 1
phillips 4
phillps 1
phlips 1
phllips 1
unilever 3
unilver 1
Van Houten 5
Facet by choice coun
company
11 choices Sort by:
akz0 1
akzo 6
fillips 1
philips 1
phillips 4
Export
Help
ions:
Wikidata
- 25 next › last »
Cluster & Edit column "company"
This feature helps you find groups of different cell values that might be alternative representations of the same thing. For example, the two strings "New York" and "new york" are very likely to refer to the same concept and just have capitalization differences, and "Gödel" and "Godel" probably refer to the same person. Find out more…
Method key collision
Keying Function metaphone3
3 clusters found
Cluster Size
Row Count
Values in Cluster
Merge?
New Cell Value
6
9
phillips (4 rows)
fillips (1 rows)
philips (1 rows)
phillps (1 rows)
phlips (1 rows)
phllips (1 rows)
phillips
2
4
unilever (3 rows)
unilver (1 rows)
unilever
2
7
akzo (6 rows)
akz0 (1 rows)
akzo
Choices in Cluster
2 — 6
Rows in Cluster
4 — 9
Average Length of Choices
4 — 7.5
Length Variance of Choices
0 — 0.578
Select All
Unselect All
Export Clusters
Merge Selected & Re-Cluster
Merge Selected & Close
Close

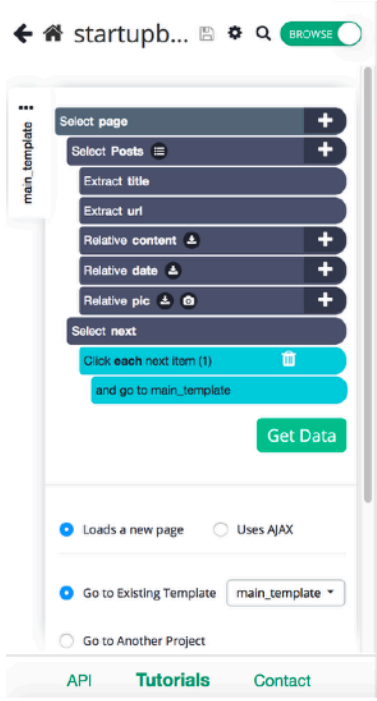

OpenRefine Refine Demo
Permalink
Open...
Export
Help
Facet / Filter
Undo / Redo 3 / 3
Refresh
Reset All
Remove All
company
change
4 choices Sort by: name count
Cluster
akzo 7
phillips 9
unilever 4
Van Houten 5
Facet by choice counts
25 rows
Extensions: Wikidata
Show as: rows records Show: 5 10 25 50 rows
« first ‹ previous 1 - 25 next › last »
All
company
Product code
address
city
country
name
1. phillips p-5 Groningensingel 147 arnhem the netherlands dhr p. jansen
2. phillips p-43 Groningensingel 148 arnhem the netherlands dhr p. hansen
3. phillips x-3 Groningensingel 149 arnhem the netherlands dhr j. Gansen
4. phillips x-34 Groningensingel 150 arnhem the netherlands dhr p. mansen
5. phillips x-12 Groningensingel 151 arnhem the netherlands dhr p. fransen
6. phillips p-23 Groningensingel 152 arnhem the netherlands dhr p. franssen
7. akzo v-43 Leeuwardenweg 178 arnhem the netherlands dhr p. bansen
8. akzo v-12 Leeuwardenweg 179 arnhem the netherlands dhr p. vansen
9. akzo x-5 Leeuwardenweg 180 arnhem the netherlands dhr p. bransen
10. akzo p-34 Leeuwardenweg 181 arnhem the netherlands dhr p. janssen
11. akzo q-5 Leeuwardenweg 182 arnhem the netherlands mevr l. rokken
12. akzo q-9 Leeuwardenweg 183 arnhem the netherlands mevr l. lokken
13. akzo x-8 Leeuwardenweg 184 arnhem the netherlands mevr l. mokken
14. phillips p-56 Delfzijlstraat 54 arnhem the netherlands mevr l. mokken
15. phillips v-67 Delfzijlstraat 55 arnhem the netherlands mevr l. mokken
16. phillips v-21 Delfzijlstraat 56 arnhem the netherlands mevr l. mokken
17. Van Houten x-45 Delfzijlstraat 57 arnhem the netherlands mevr l. sokken
18. Van Houten v-56 Delfzijlstraat 58 arnhem the netherlands mevr l. wokken
19. Van Houten v-65 Delfzijlstraat 59 arnhem the netherlands mevr l. kokken
20. Van Houten x-21 Delfzijlstraat 60 arnhem the netherlands mevr l. Bokken
21. Van Houten p-23 Delfzijlstraat 61 arnhem the netherlands mevr l. dokken
22. unilever x-3 Jourestraat 23 arnhem the netherlands mevr l. gokken
23. unilever q-4 Jourestraat 24 arnhem the netherlands mevr l. stokken
24. unilever q-6 Jourestraat 25 arnhem the netherlands mevr l. rokken
25. unilever q-8 Jourestraat 26 arnhem the netherlands mevr l. rokken

Web Scrapping



```
In [1]:
import requests
import csv
import pandas as pd
from bs4 import BeautifulSoup

# quote_page = requests.get('http://startupbeat.hkej.com/?tag=fintech&paged=1')
# soup = BeautifulSoup(quote_page.text,'html.parser')

header = ['page #','title','url','details','post date']
data = []
# Display and store away 2 pages of scrapped data from startupbeat.hkej.com
for i in range(1,4):
    quote_page = requests.get('http://startupbeat.hkej.com/?tag=fintech&paged='+str(i))
    print("\n***** Page " + str(i) +" in action *****")
    soup = BeautifulSoup(quote_page.content,'html.parser')

    for article in soup.find_all("div", attrs={"class":"archive-text"}):
    # for article in soup.find_all('div',class_='archive-text'):
        page_no = str(i)
        title = article.a.text.encode('utf-8').strip()
        decoded_title = title.decode('utf-8')
        url = article.a.get('href')
        details = article.p.text.encode('utf-8').strip()
        decoded_details = details.decode('utf-8')
        post_date = article.div.ul.li.text
        print(decoded_title)
        print(url)
        print(decoded_details)
        print(post_date)
        data.append((page_no, decoded_title, url, decoded_details, post_date))

df = pd.DataFrame(data,
  columns = header
)
df.to_csv('startup_beat_data_1.csv', sep='\t', encoding='utf-8')
```

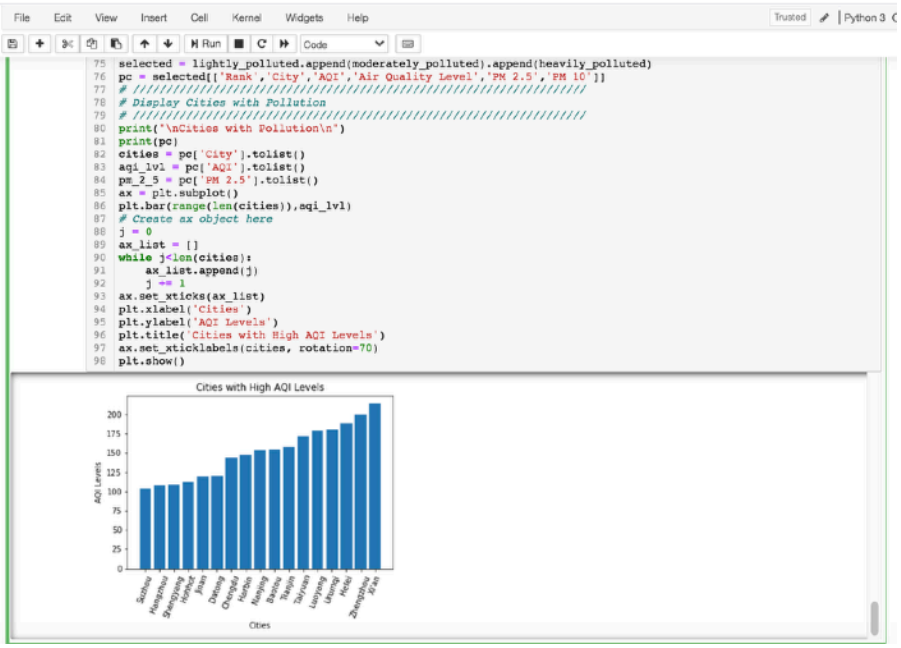
```
***** Page 1 in action *****
港官方數碼幣料兩年內試用
http://startupbeat.hkej.com/?p=89686
各國近年紛紛探索央行數碼貨幣（Central Bank Digital Currency，CBDC），不但中國正加快進行「數字貨幣/電子支付」（DC/EP）試點計劃，香港近年亦在此領域積極研究。區塊鏈科技初創CryptoBLK就與香港金融管理局（金管局）合作，參與相關項目。
Posted July 13, 2020
日本告別現金（老古）
http://startupbeat.hkej.com/?p=89672
日本是一個現金社會，電子貨幣、支付寶和微信支付、Apple Pay一直都流行不起來。日本人更不喜歡「革命」兩字，認為這是「殺頭」的意思。
```

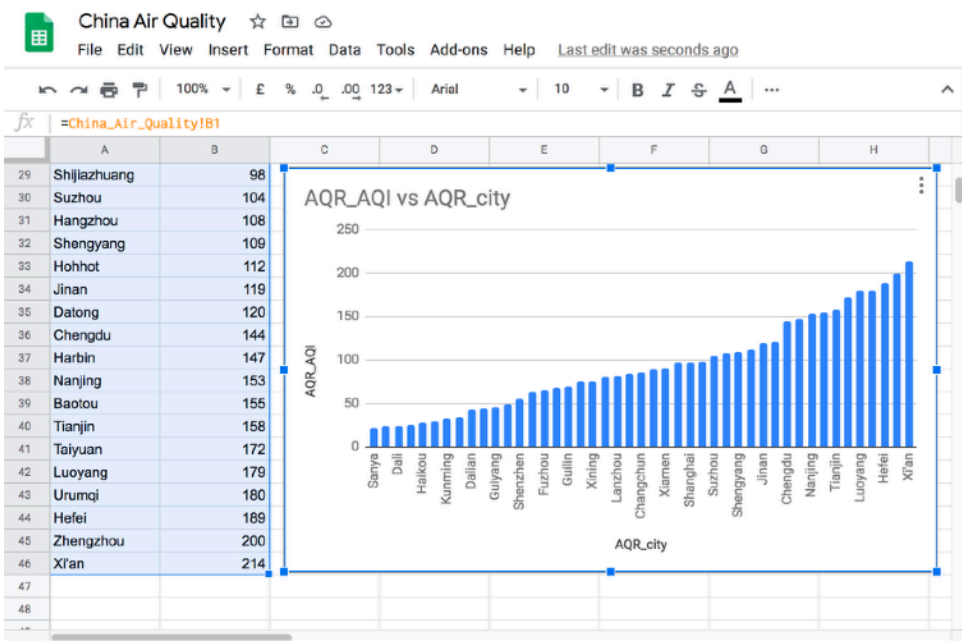


Programmer Track

Non-Programmer Track

Making Charts





	A	B
29	Shijiazhuang	98
30	Suzhou	104
31	Hangzhou	108
32	Shengyang	109
33	Hohhot	112
34	Jinan	119
35	Datong	120
36	Chengdu	144
37	Harbin	147
38	Nanjing	153
39	Baotou	155
40	Tianjin	158
41	Taiyuan	172
42	Luoyang	179
43	Urumqi	180
44	Hefei	189
45	Zhengzhou	200
46	Xi'an	214

Programmer Track

Non-Programmer Track

Tutorials on OpenRefine

Tutorial 1
https://programminghistorian.org/en/lessons/cleaning-data-with-openrefine#getting-started-installing-openrefine-and-importing-data

Practice Dataset 1
https://programminghistorian.org/assets/phm-collection.tsv

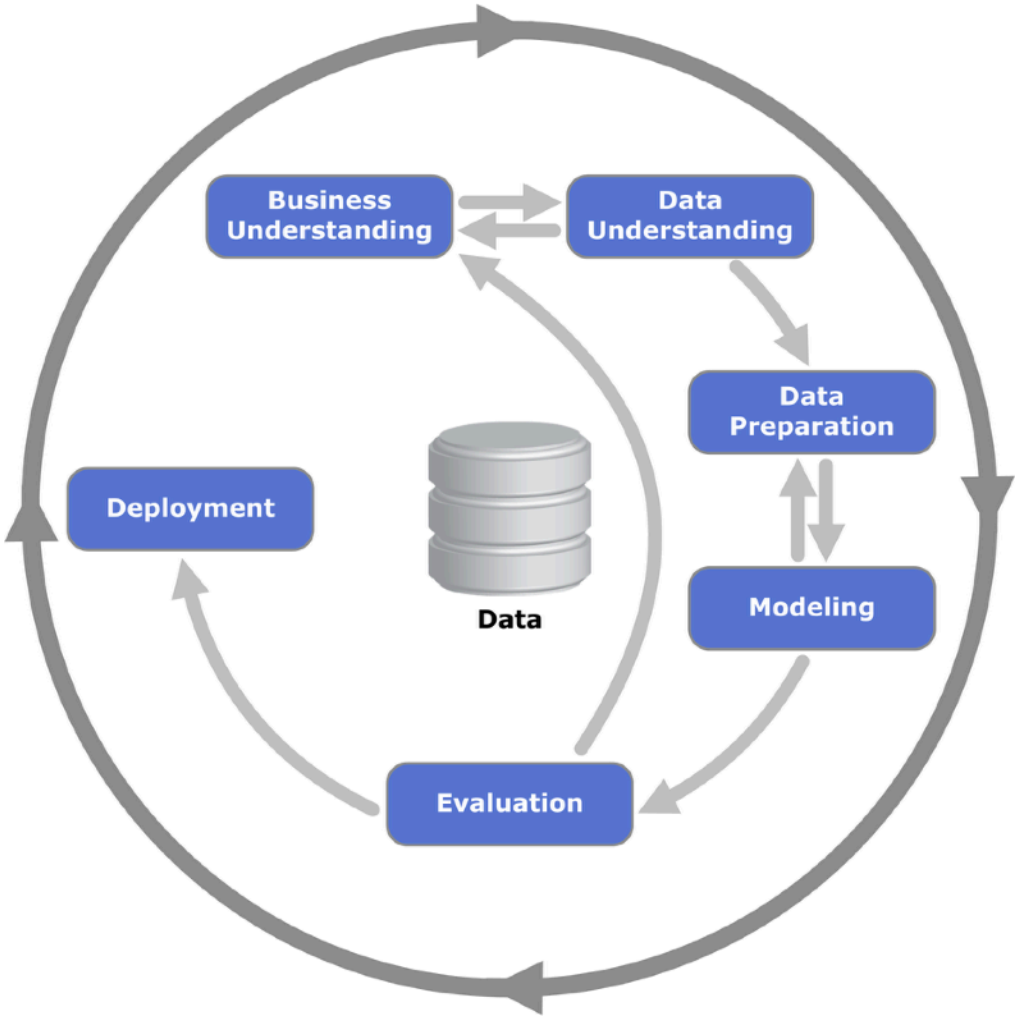
Tutorial 2

http://d3-media.blogspot.hk/2013/11/how-to-refine-your-data.html

Practice Dataset 2

https://docs.google.com/spreadsheets/d/171GoZt1fF3E1e79gb_HAh2uImq111DscD2_dCtjkNVw/edit?copiedFromTrash#gid=0

The CRISP-DM Model



CRoss Industry Standard Process for Data Mining

Harvard Business Review

DATA

Data Scientist: The Sexiest Job of the 21st Century

by Thomas H. Davenport and D.J. Patil

From the October 2012 Issue

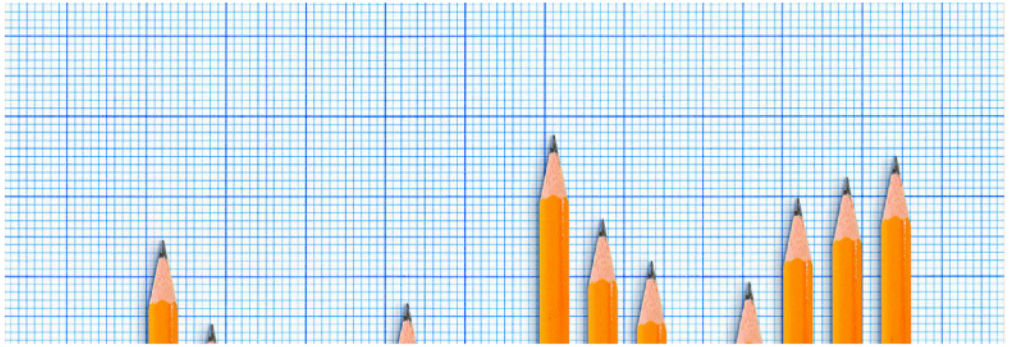
https://hbr.org/2012/10/data-scientist-the-sexiest-job-of-the-21st-century

ANALYTICS

What Data Scientists Really Do, According to 35 Data Scientists

by Hugo Bowne-Anderson

August 15, 2018



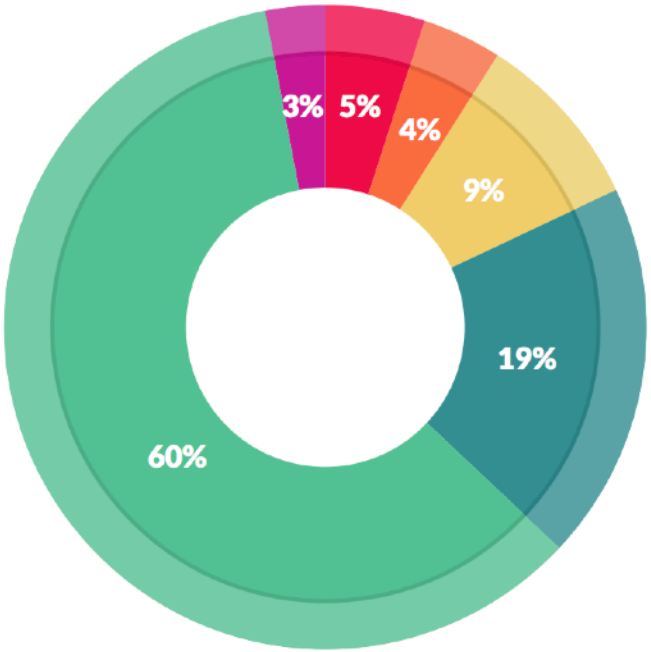
https://hbr.org/2018/08/what-data-scientists-really-do-according-to-35-data-scientists?referral=03758&cm_vc=rr_item_page.top_right

Source: By Hugo Bowen-Anderson August 15, 2018 Harvard Business Review

What data scientists do. We now know how data science works, at least in the tech industry. First, data scientists lay a solid data foundation in order to perform robust analytics. Then they use online experiments, among other methods, to achieve sustainable growth. Finally, they build machine learning pipelines and personalized data products to better understand their business and customers and to make better decisions. In other words, in tech, data science is about infrastructure, testing, machine learning for decision making, and data products.

How a Data Scientist Spends Their Day

Here's where the popular view of data scientists diverges pretty significantly from reality. Generally, we think of data scientists building algorithms, exploring data, and doing predictive analysis. That's actually not what they spend most of their time doing, however.

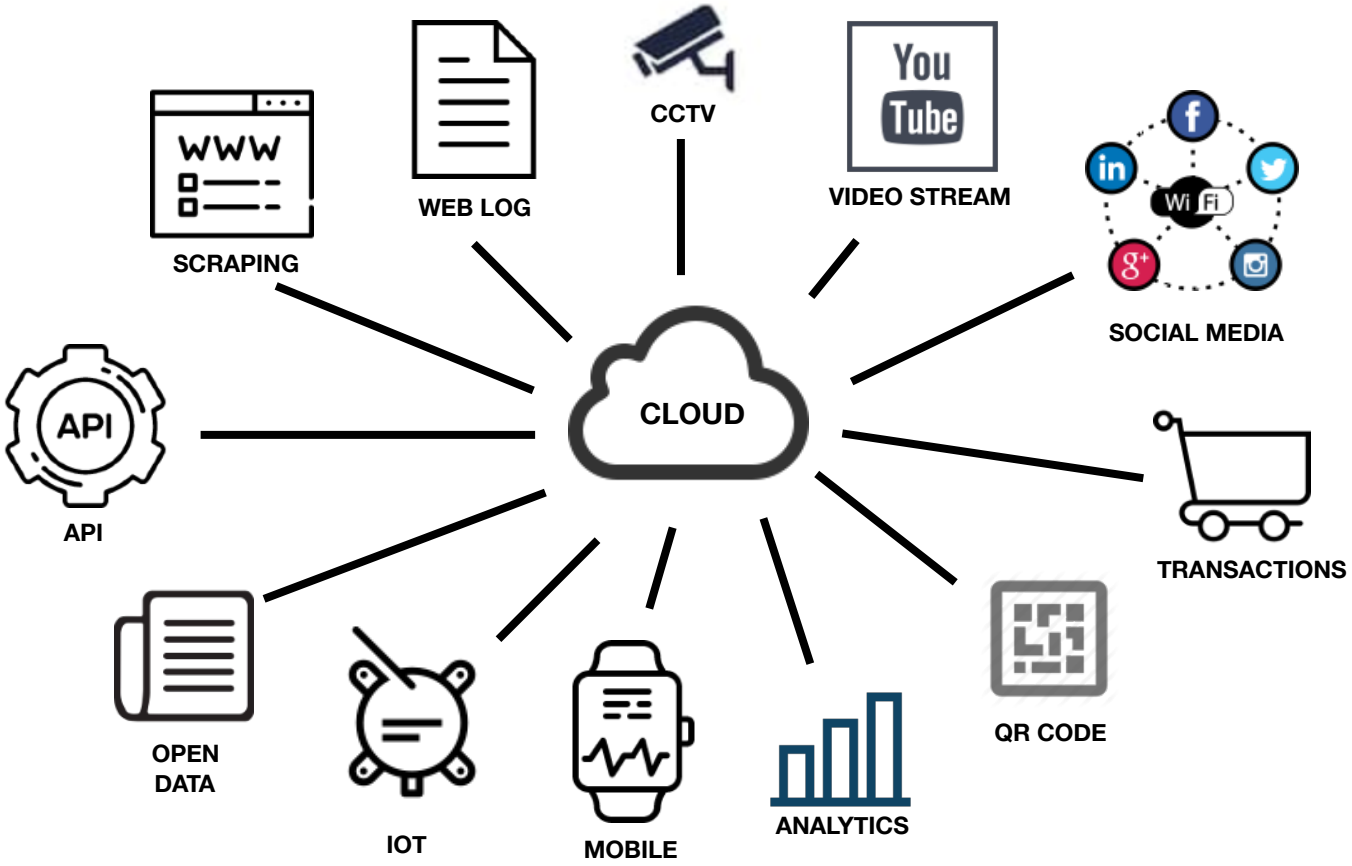


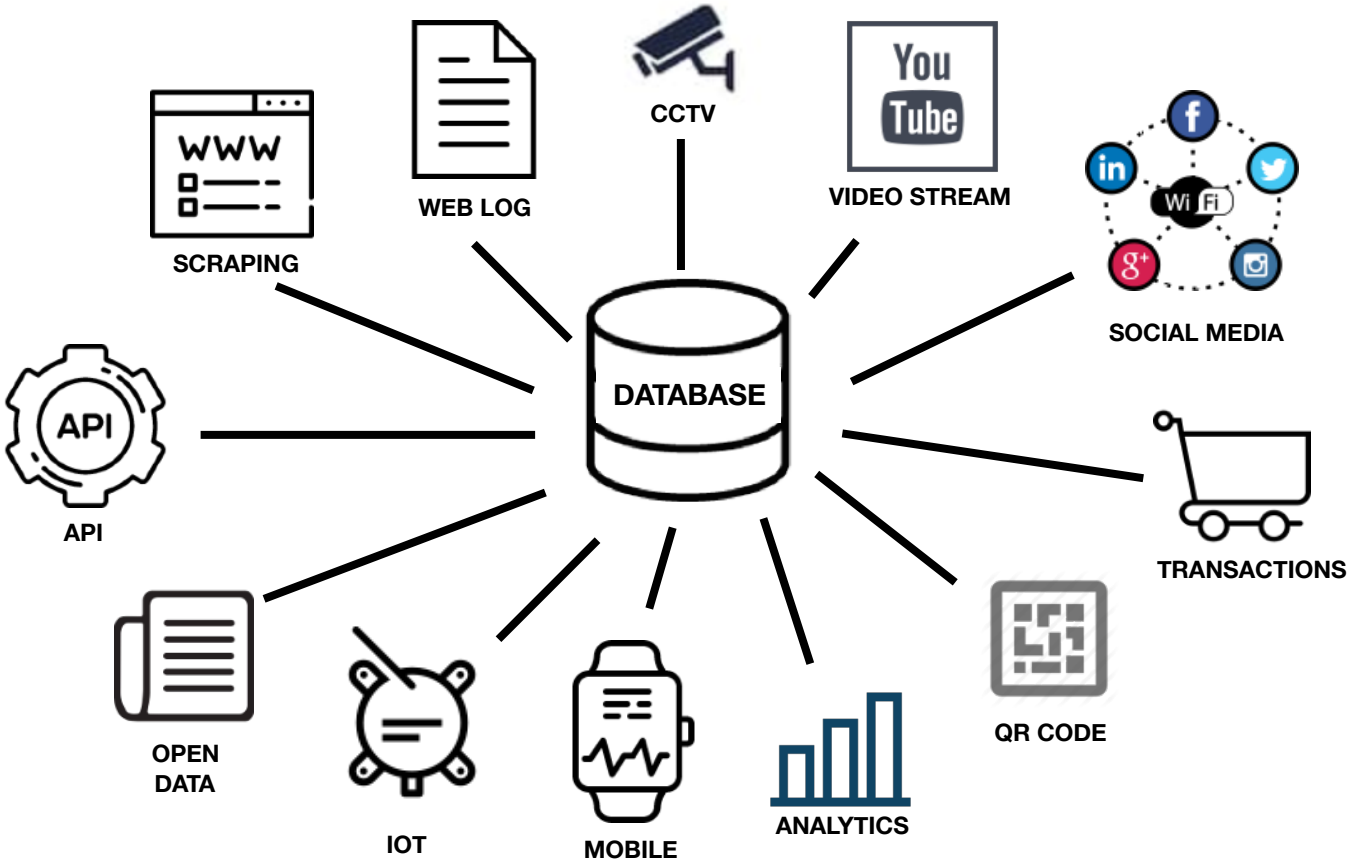
What data scientists spend the most time doing

- *Building training sets: 3%*
- *Cleaning and organizing data: 60%*
- *Collecting data sets; 19%*
- *Mining data for patterns: 9%*
- *Refining algorithms: 4%*
- *Other: 5%*

Source: Data Science 2016 Report by CrowdFlower

Other sources of data besides scraping.


WWW
SCRAPING
WEB LOG
CCTV
You
Tube
VIDEO STREAM
Wi Fi
SOCIAL MEDIA
API
API
CLOUD
TRANSACTIONS
OPEN
DATA
IOT
MOBILE
ANALYTICS
QR CODE


WWW
SCRAPING
WEB LOG
CCTV
You
Tube
VIDEO STREAM
Wi Fi
SOCIAL MEDIA
API
API
DATABASE
TRANSACTIONS
OPEN
DATA
IOT
MOBILE
ANALYTICS
QR CODE

NYC | OpenData 311 | Search all NYC.gov websites

NYC OpenData

Home Data About Learn Alerts Contact Us Blog

Open Data for All New Yorkers

Where can you find public Wi-Fi in your neighborhood? What kind of tree is in front of your office? Learn about where you live, work, eat, shop and play using NYC Open Data.

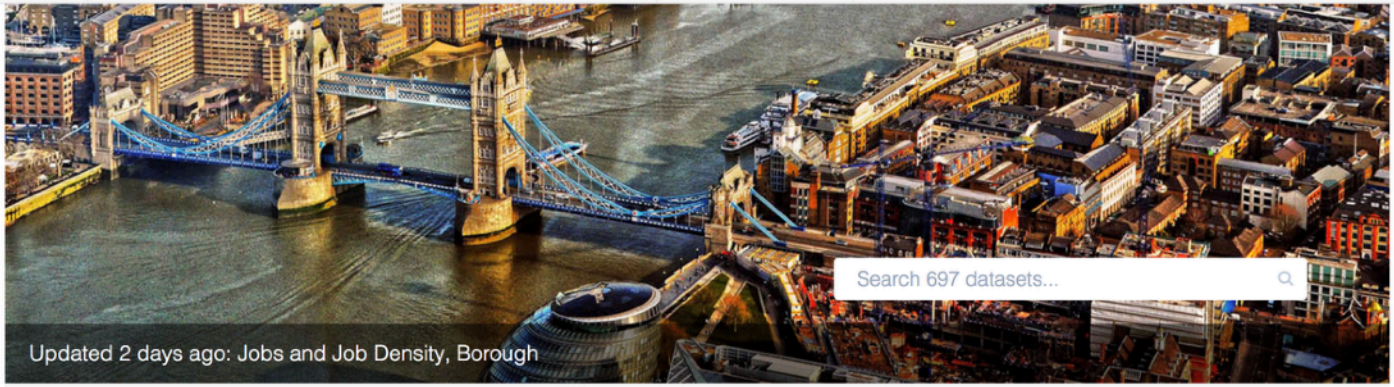
Search Open Data for things like 311, Buildings, Crime

MAYOR OF LONDON LONDONASSEMBLY

LONDON DATASTORE

Register an Account Login

Blog Data Apps & Analysis Developers Boroughs City Data Strategy More



JOBS AND ECONOMY TRANSPORT ENVIRONMENT COMMUNITY SAFETY HOUSING COMMUNITIES HEALTH LONDON AS A WORLD CITY GLA PERFORMANCE

Click on a circle to see more...

Welcome to the Datastore

The London Datastore is a free and open data-sharing portal where anyone can access data relating to the capital. Whether you're a citizen, business owner, researcher or developer, the site provides over 700 datasets to help you understand the city and develop solutions to London's problems Please do have a look around, and let us

Office of the Government Chief Information Officer
The Government of the Hong Kong Special Administrative Region

[Hong Kong
Smart City Blueprint]

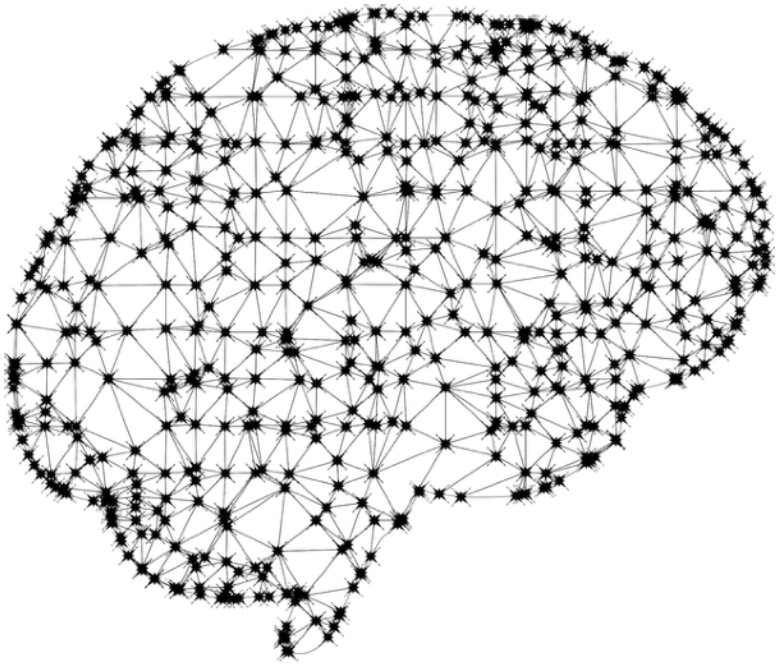
Home | Vision & Mission | Development Plans | Open e-Blueprint | Download Blueprint | 繁 簡 A A A

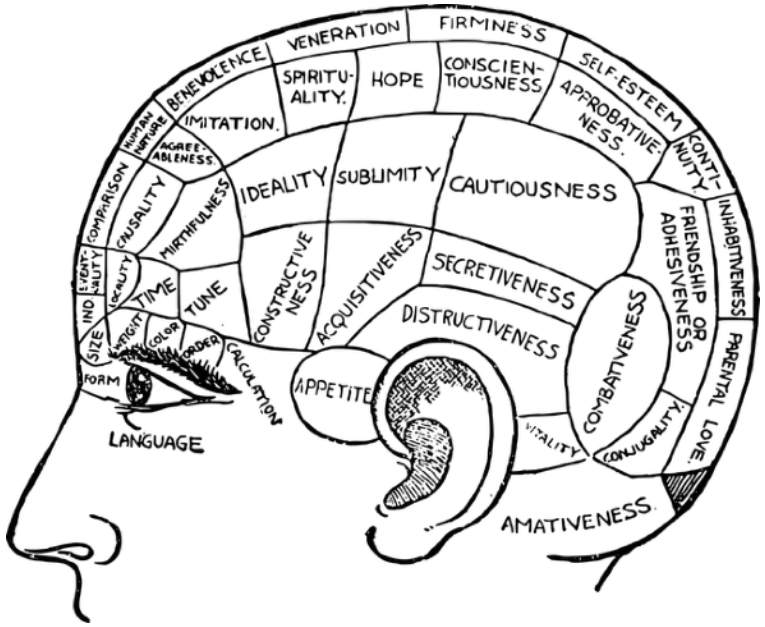
Open Data

Home > Open Data

Smart Mobility | Smart Living | Smart Environment | Smart People | Smart Government | Smart Economy | Open Data

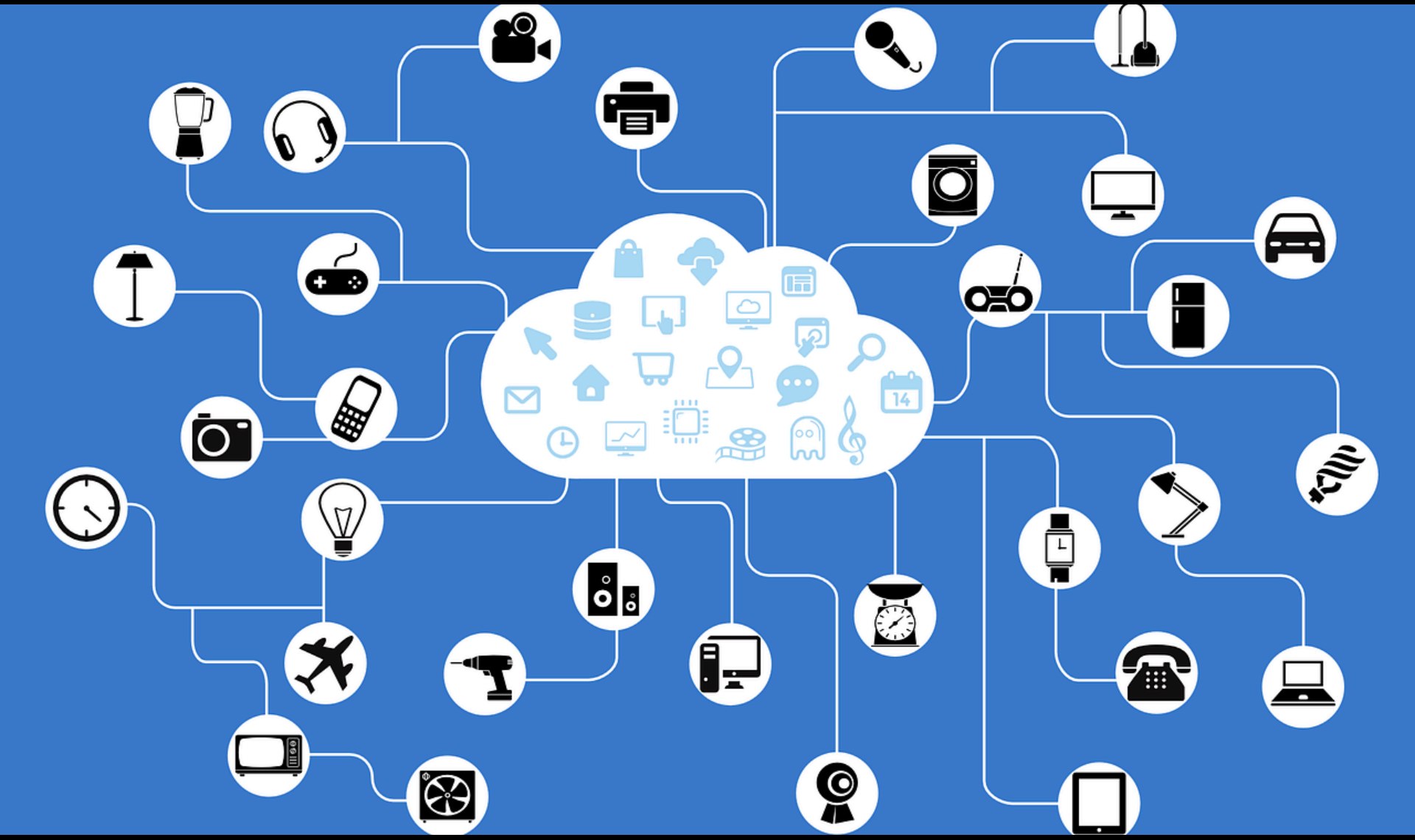
Digital infrastructure for 21st century living.

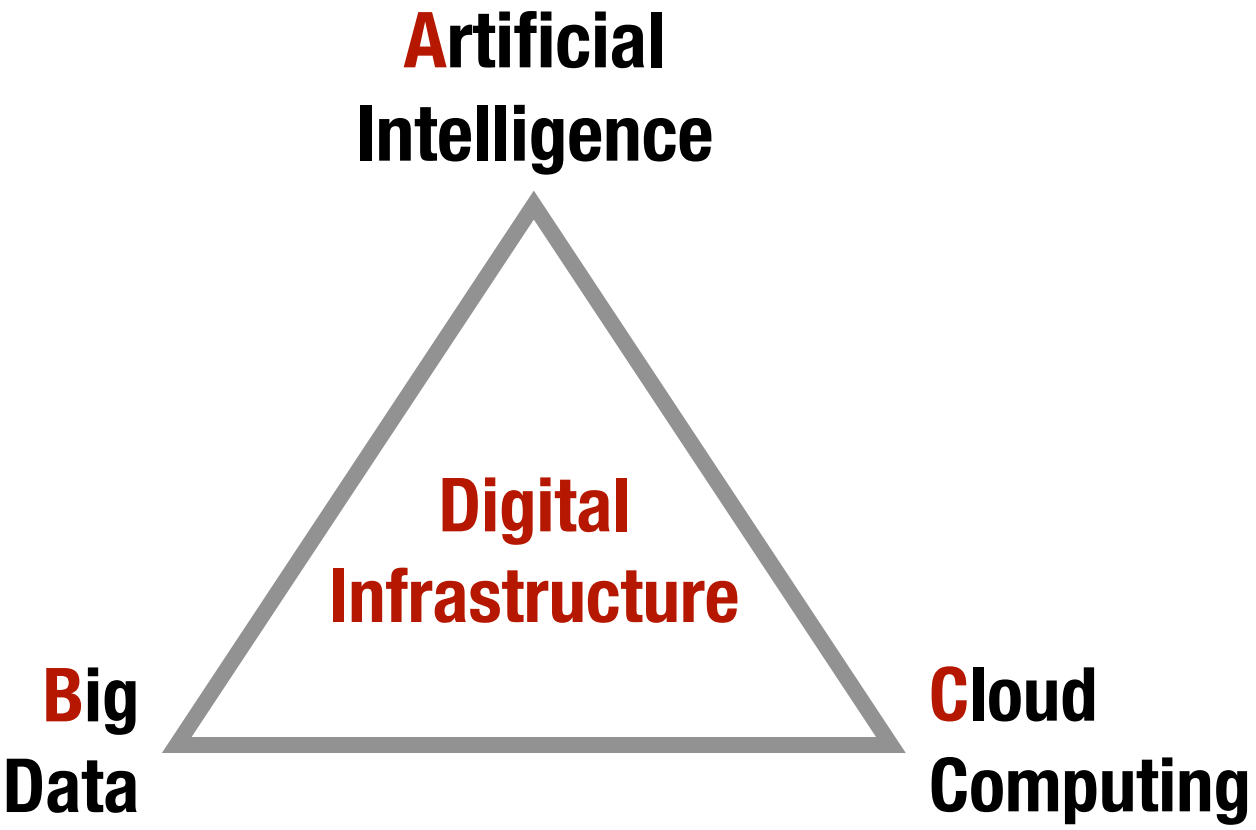




ARTIFICIAL INTELLIGENCE

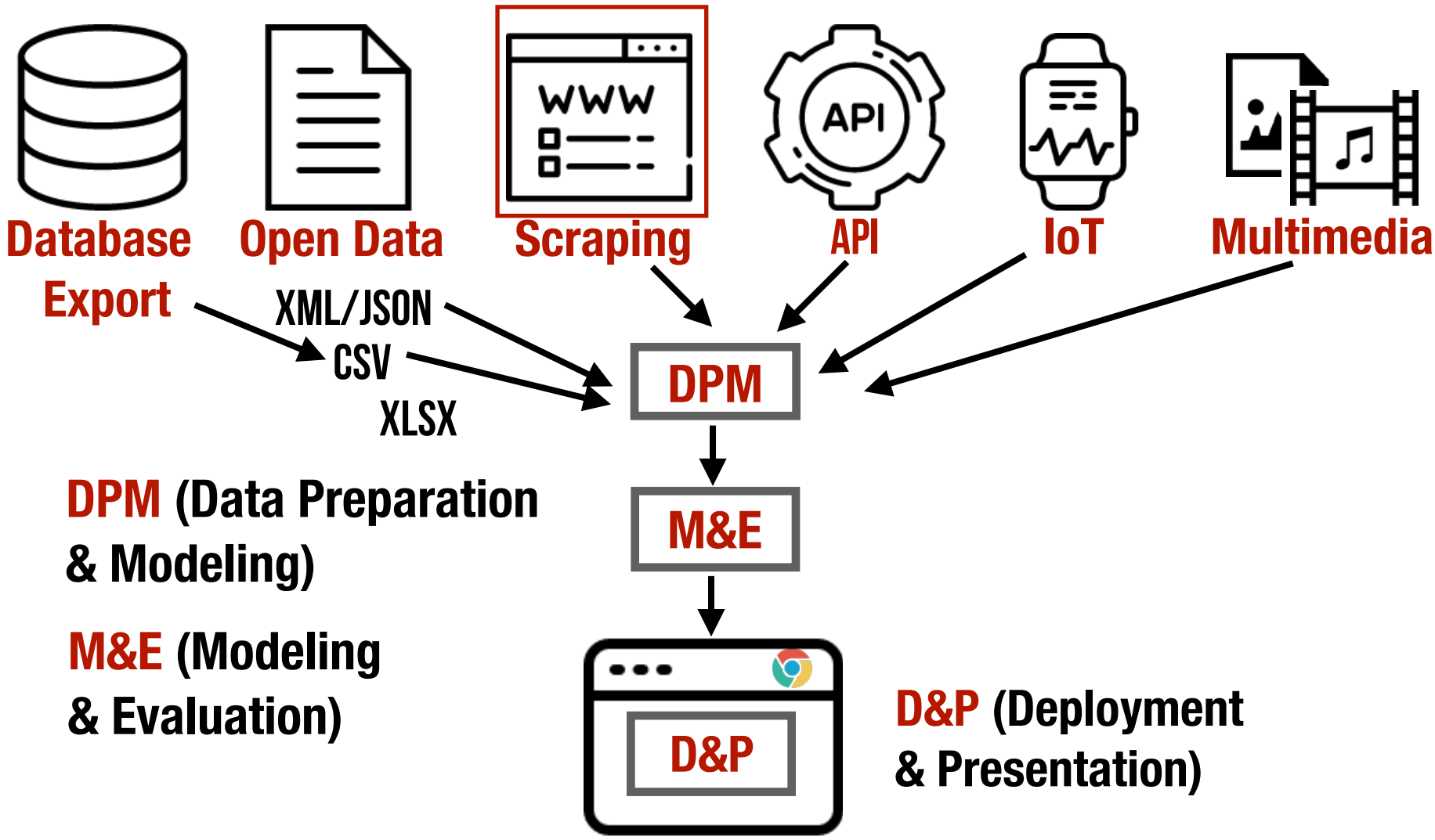

BIG DATA




Artificial
Intelligence
Digital
Infrastructure
Big
Data
Cloud
Computing

香港智慧城市

Hong Kong Smart City 藍圖 Blueprint


Database Export
Open Data
Scraping
API
IoT
Multimedia
XML/JSON
CSV
XLSX
DPM
M&E
D&P
DPM (Data Preparation & Modeling)
M&E (Modeling & Evaluation)
D&P (Deployment & Presentation)

Scraping single page table.

Using Jupyter Notebook for Scraping

File Edit View Insert Cell Kernel Widgets Help Not Trusted | Python 3

Run Markdown

Quick Revision of Python List and Dictionary

Process list elements inside a loop.

In [18]:

```
name_list = ['John',"Mike","Mary","Jane"]
hrs_list = [30.0,40.0,50.0,60.0]
rate_list = [65.0,75.0,65.0,75.0]
index  = 0
for number in name_list:
    print(number)
    hrs = hrs_list[index]
    rate = rate_list[index]
    fee = fee = hrs * rate
    money_made = name_list[index] + " makes $" + str(fee) + "."
    print(hrs,rate,fee, money_made)
    index = index + 1
```

```
John
30.0 65.0 1950.0 John makes $1950.0.
Mike
40.0 75.0 3000.0 Mike makes $3000.0.
Mary
50.0 65.0 3250.0 Mary makes $3250.0.
Jane
60.0 75.0 4500.0 Jane makes $4500.0.
```

Input and process list elements inside a loop.

In [24]:
```
# Input first name and last name and print out full name
print('Enter your first name:')
fname = input()
print('Enter your last name:')
lname = input()
print('Hello, ' + fname + ' ' + lname + '.')
```

```
Enter your first name:
Bernard
Enter your last name:
Suen
Hello, Bernard Suen.
```

```
Formula for converting Celcius into Fahrenheit: (23°C × 9/5) + 32 = 73.4°F
Create a list of temperature in Fahrenheit for the week based on the temperature of Celcius as follows:

    Sun  Mon  Tue  Wed  Thu  Fri  Sat
    23°C 24°C  28°C 19°C 17°C  21°C 25°C
```

```
For more flexible handling: Entering the temperature in Celcius through an input box inside a loop and print out
each day's temperature and calculate the weekly average in Fahrenheit.
```

In [25]:
```
weekly_temp = []
total_temp = 0
week_of_day =['Sun','Mon','Tue','Wed','Thu','Fri','Sat']
for i in week_of_day:
    temp = input("Enter " + i + "'s temperature in celcius:")
    temp = (float(temp) * 9/5) + 32
    weekly_temp.append(int(temp))
```

Process dictionary elements inside a loop.

In [29]:
```
# dictinarly as value pairs
dict = {'Peter':80,"David":90,"Mary":100}
print(dict)
print(dict.keys())
print(dict.values())
# accessing individualy element through its key
print(dict["David"])
```

```
{'Peter': 80, 'David': 90, 'Mary': 100}
dict_keys(['Peter', 'David', 'Mary'])
dict_values([80, 90, 100])
90
```

In [30]:
```
dict = {'Peter':80,"David":90,"Mary":100}
# print out key and value referenced by the key
for key in dict:
  print(key, dict[key])
```

```
Peter 80
David 90
Mary 100
```

Creating Pandas Data Frames from Lists and Dictionaries

Data Frames from list of Dictionaries and list of lists

Creating Pandas Data Frames from Lists and Dictionaries

Data Frames from list of Dictionaries and list of lists

In [14]:
```
import pandas as pd
# Define a list of dictionaries
scores = [{'assignment scores': 'David Chan', 'Jan': 90, 'Feb': 85, 'Mar': 88},
          {'assignment scores': 'Peter Lee',  'Jan': 72, 'Feb': 75, 'Mar': 68},
          {'assignment scores': 'John Lui',   'Jan': 60,  'Feb': 80,  'Mar': 100 }]
df = pd.DataFrame(scores) # Assign the list of dictionaries to a dataframe
df # display dataframe
```

Out[14]:

	assignment scores	Jan	Feb	Mar
0	David Chan	90	85	88
1	Peter Lee	72	75	68
2	John Lui	60	80	100

In [15]:
```
import pandas as pd
# define dataframe as a list of lists
# with a header list called columns which contains the column names
df = pd.DataFrame([
  [1, 'David Chan', 'dchan@cuhk.edu.hk'],
  [2, 'Peter Lee', 'plee@cuhk.edu.hk'],
  [3,'John Lui','jlui@cuhk.edu.hk'],
  [4,'Mary Lu','mlu@cuhk.edu.hk']
],
```

Data Frames from Dictionary of Lists

In [5]:
```
# Define a dictionary of list
scores = {'assignment scores': ['David Chan', 'Peter Lee', 'John Lui'],
          'Jan': [90, 72, 60],
          'Feb': [85, 75, 80],
          'Mar': [88, 68, 100]}
df = pd.DataFrame.from_dict(scores) # Assign the dictionary of lists to a dataframe
df # display dataframe
```

Out[5]:

	assignment scores	Jan	Feb	Mar
0	David Chan	90	85	88
1	Peter Lee	72	75	68
2	John Lui	60	80	100

In [12]:
```
# Define a dictionary of list directily inside the data frame
import pandas as pd
df = pd.DataFrame({ # Define dataframe as a dictionary object
  'Student ID': [1111, 1112, 1113, 1114],
'Student Name': ['David Chan','Peter Lee','John Lui','Mary Lu'],
'Email Address':['dchan@cuhk.edu.hk','plee@cuhk.edu.hk','jlui@cuhk.edu.hk','mlu@cuhk.edu.hk']
})
df
```

Out[12]:

	Student ID	Student Name	Email Address
0	1111	David Chan	dchan@cuhk.edu.hk
1	1112	Peter Lee	plee@cuhk.edu.hk
2	1113	John Lui	jlui@cuhk.edu.hk

Saving To and Retrieving From CSV Files

Saving Pandas Data Frame To a CSV File

In [ ]:

```
import pandas as pd
# Define a list of dictionaries
scores = [{'assignment scores': 'David Chan', 'Jan': 90, 'Feb': 85, 'Mar': 88},
          {'assignment scores': 'Peter Lee',  'Jan': 72, 'Feb': 75, 'Mar': 68},
          {'assignment scores': 'John Lui',   'Jan': 60,  'Feb': 80,  'Mar': 100 }]
df = pd.DataFrame(scores) # Assign the list of dictionaries to a dataframe
df # display dataframe
df.to_csv('students.csv',mode='w',index=False)
df
```

Reading Data From a CSV File Into a Pandas Data Frame

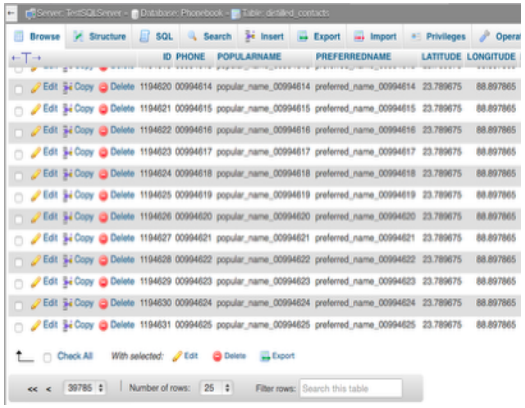
In [65]:

```
import pandas as pd
df = pd.read_csv("students.csv") # import csv file by using the read_csv functon in Pandas
df # list the dataframe
```

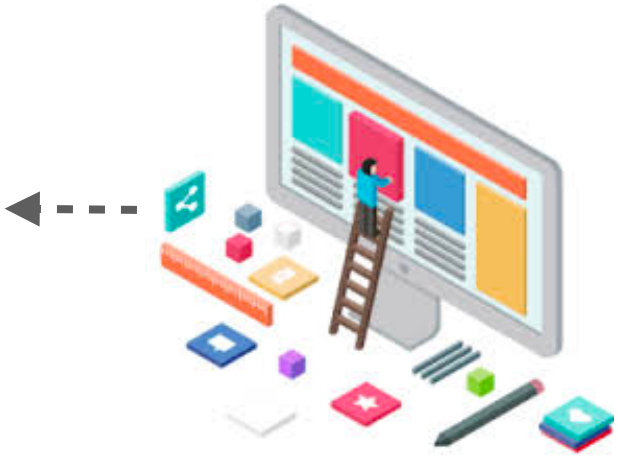
Out[65]:

	assignment scores	Jan	Feb	Mar
0	David Chan	90	85	88
1	Peter Lee	72	75	68
2	John Lui	60	80	100

Data



Web Scraping



Web

Browser View

Welcome to My Page!

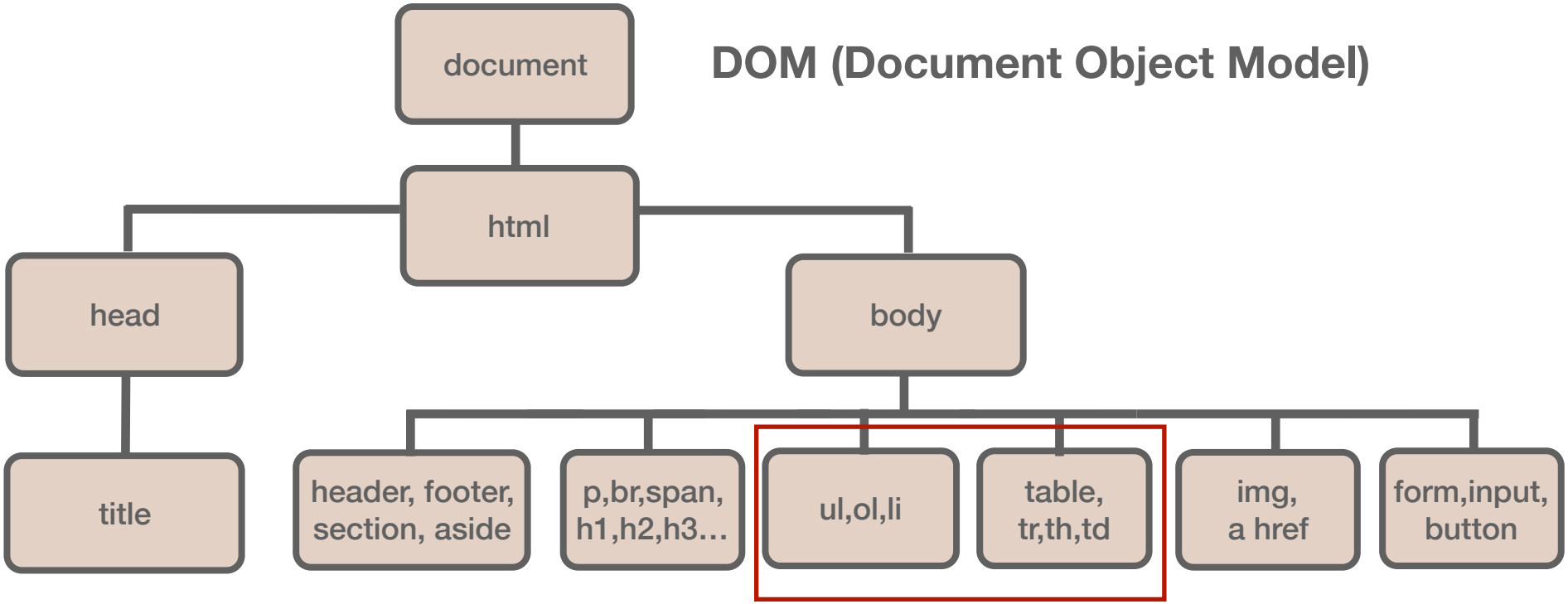
My Favourite Fruits

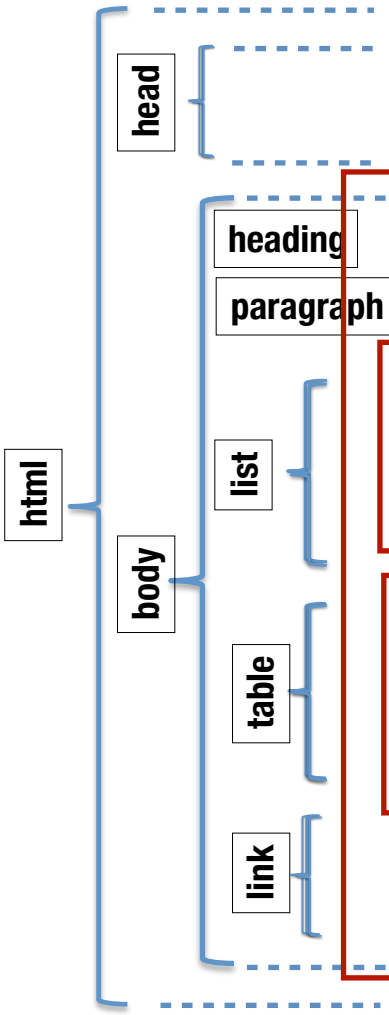
- apples
- oranges
- pineapples

Name	Major
John Chan	English
David Wong	Computer Science

香 港 中 文 大 學
The Chinese University of Hong Kong

This is the link to CUHK's website.


DOM (Document Object Model)
document
html
head
body
title
header, footer, section, aside
p,br,span, h1,h2,h3…
ul,ol,li
table, tr,th,td
img, a href
form,input, button




```
<!DOCTYPE html>
<html>
<head>
<title>My Web Page</title>
</head>
<body>
<h1>Welcome to My Page!</h1>
<p><font color="blue">My Favourite Fruits</font></p>

<ul>
  <li>apples</li>
  <li>oranges</li>
  <li>pineapples</li>
</ul>

<table cellspacing="0" cellpadding="0" border="1">
  <tr><th width="20%">Name</th><th width="20%">Major</th></tr>
  <tr><td height="50">John Chan</td><td height="50">English</td></tr>
  <tr><td>David Wong</td><td>Computer Science</td></tr>
</table>

<img src="http://www.cuhk.edu.hk/english/images/cuhk_logo_2x.png"></
img>
<a href="www.cuhk.edu.hk">This is the link to CUHK's website</a>.
</body>
</html>
```

How to scrap list and table data from the highlighted areas?

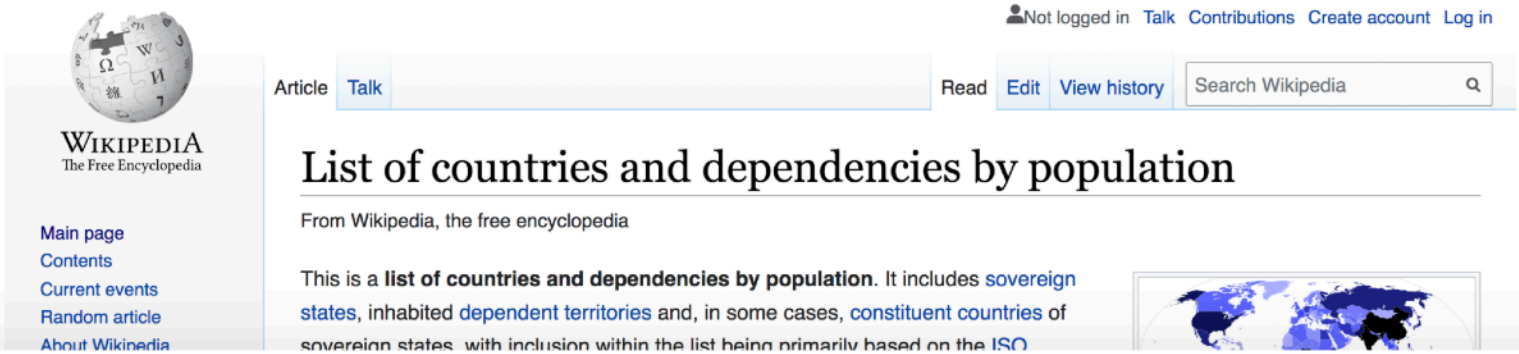
Reading Data From a CSV File Into a Pandas Data Frame

In [65]:
```
1 import pandas as pd
2 df = pd.read_csv("students.csv") # import csv file by using the read_csv functon in Pandas
3 df # list the dataframe
```

Out[65]:

	assignment scores	Jan	Feb	Mar
0	David Chan	90	85	88
1	Peter Lee	72	75	68
2	John Lui	60	80	100

Scrap Single Page Table Using BeautifulSoup



Main page
Contents
Current events
Random article
About Wikipedia
Contact us
Donate

Contribute
Help
Learn to edit
Community portal
Recent changes
Upload file

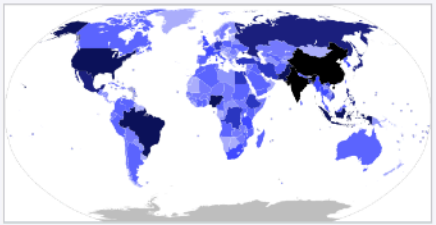
Tools
What links here
Related changes
Special pages
Permanent link

List of countries and dependencies by population

From Wikipedia, the free encyclopedia

This is a **list of countries and dependencies by population**. It includes sovereign states, inhabited dependent territories and, in some cases, constituent countries of sovereign states, with inclusion within the list being primarily based on the ISO standard ISO 3166-1. For instance, the United Kingdom is considered as a single entity, while the constituent countries of the Kingdom of the Netherlands are considered separately. In addition, this list includes certain states with limited recognition not found in ISO 3166-1.

Also given in percent is each country's population compared with the world population, which the United Nations estimates at 7.82 billion as of today.



Map of countries and territories by population in 2019

A cartogram of the world population in 2018

Contents [hide]

https://en.wikipedia.org/wiki/List_of_countries_and_dependencies_by_population

thead

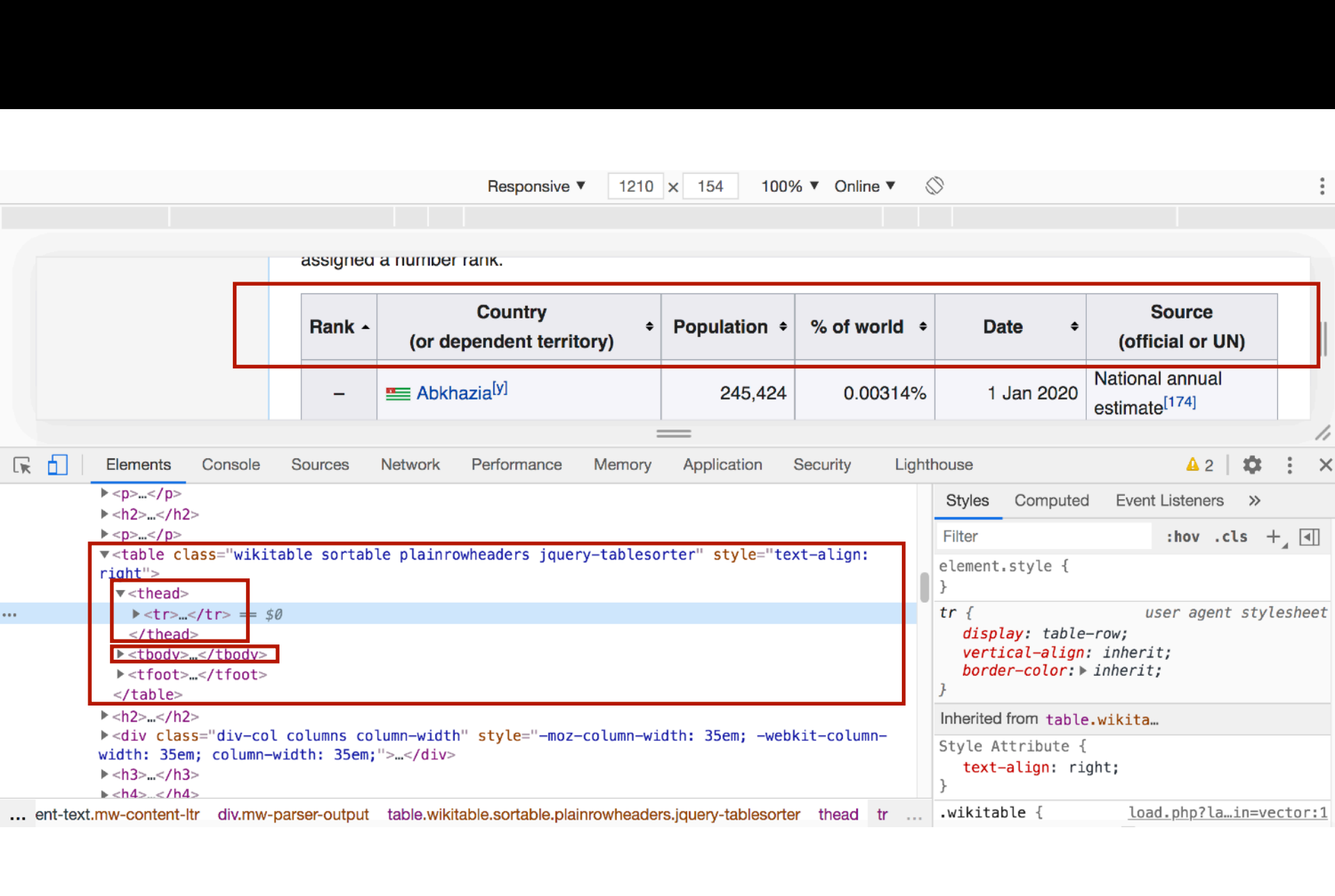
tbody

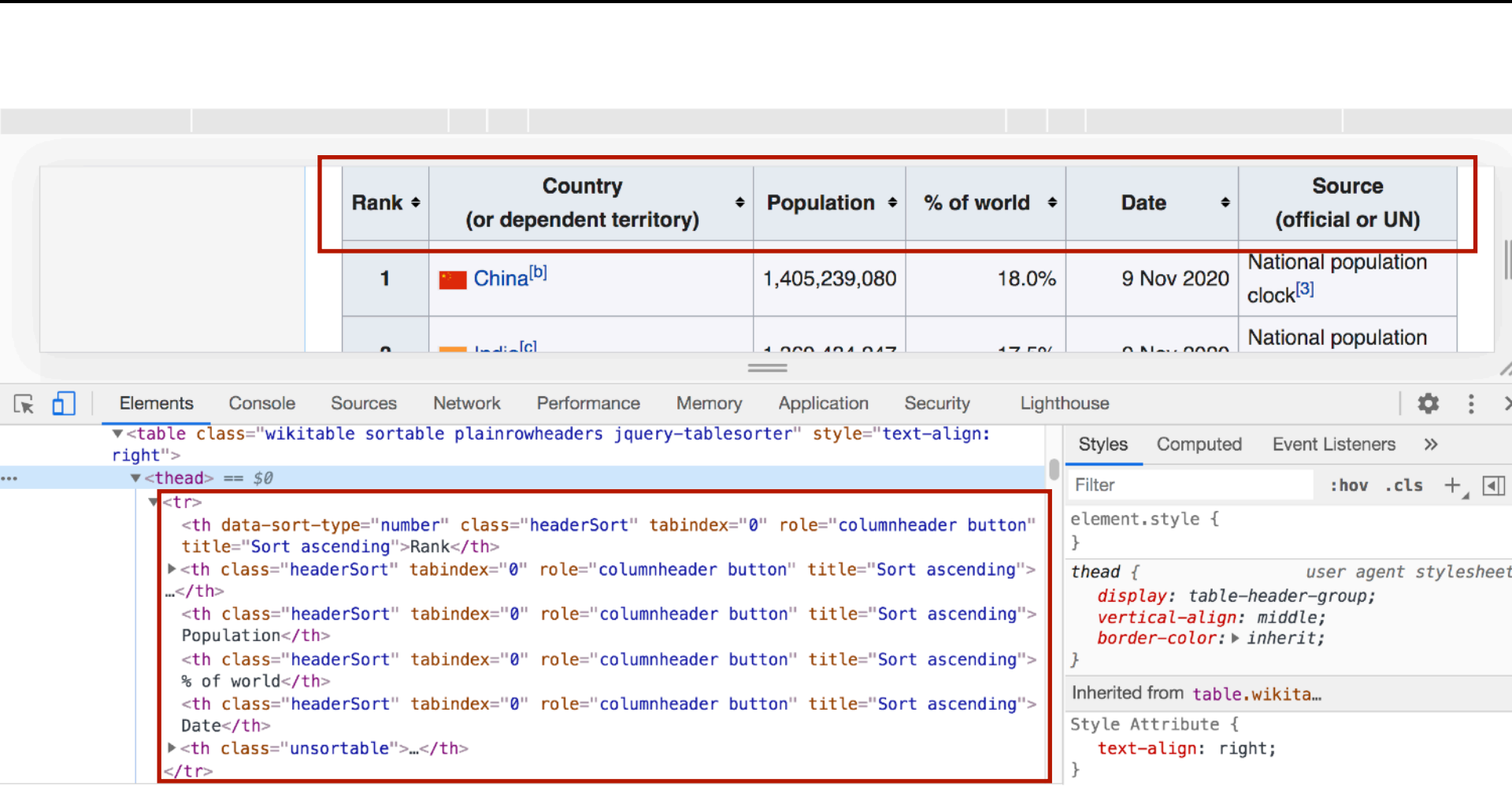
Sovereign states and dependencies by population [edit]

Note: A number rank is assigned to the 193 member states of the United Nations and the two observer states to the United Nations General Assembly. Dependent territories and constituent countries that are parts of sovereign states are shown in *italics* and not assigned a numbered rank. In addition, sovereign states with limited recognition are not assigned a number rank.

Rank	Country (or dependent territory)	Population	% of world	Date	Source (official or UN)
1	China[b]	1,405,239,080	18.0%	9 Nov 2020	National population clock[3]
2	India[c]	1,369,434,247	17.5%	9 Nov 2020	National population clock[4]
3	United States[d]	330,617,844	4.22%	9 Nov 2020	National population clock[5]
4	Indonesia	269,603,400	3.45%	1 Jul 2020	National annual projection[6]
5	Pakistan[e]	220,892,331	2.82%	1 Jul 2020	UN Projection[2]
6	Brazil	212,311,986	2.71%	9 Nov 2020	National population clock[7]
7	Nigeria	206,139,587	2.63%	1 Jul 2020	UN Projection[2]
8	Bangladesh	160,615,266	2.17%	9 Nov 2020	National population

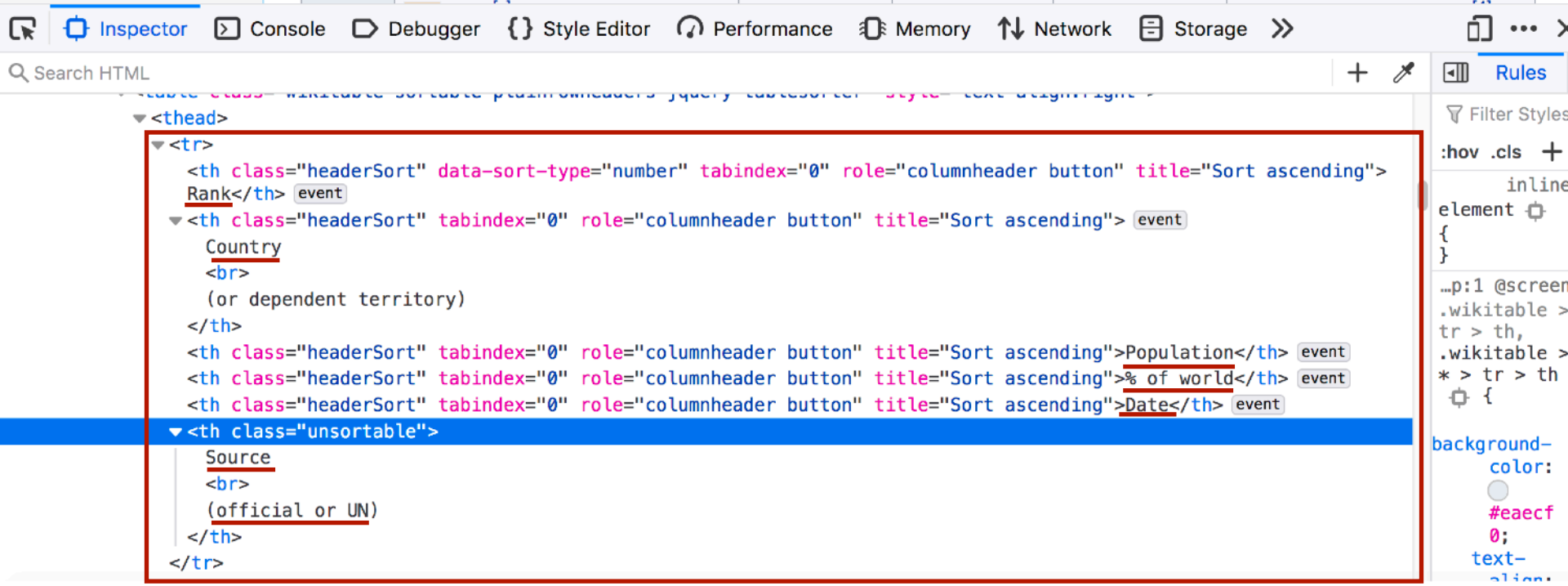
Fetching the thead items.

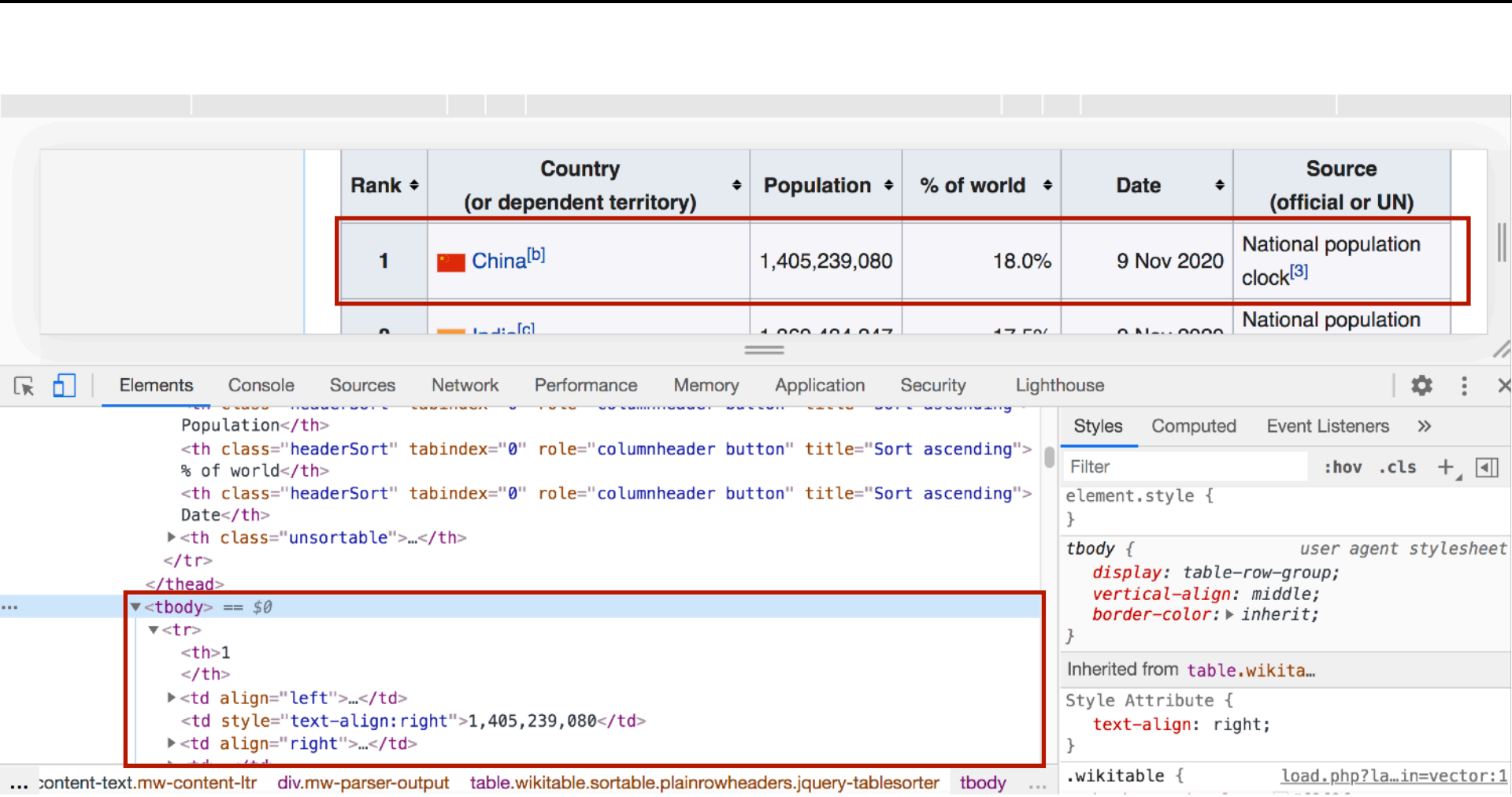

Responsive
1210 x 154
100%
Online
assigned a number rank.
Rank
Country (or dependent territory)
Population
% of world
Date
Source (official or UN)
–
Abkhazia[y]
245,424
0.00314%
1 Jan 2020
National annual estimate[174]
Elements
Console
Sources
Network
Performance
Memory
Application
Security
Lighthouse
…
<h2>…</h2>
…
<table class="wikitable sortable plainrowheaders jquery-tablesorter" style="text-align: right">
<thead>
<tr>…</tr> == $0
</thead>
<tbody>…</tbody>
<tfoot>…</tfoot>
</table>
<h2>…</h2>
…
<h3>…</h3>
Styles
Computed
Event Listeners
Filter
:hov .cls
element.style {
}
tr { user agent stylesheet
display: table-row;
vertical-align: inherit;
border-color: inherit;
}
Inherited from table.wikita…
Style Attribute {
text-align: right;
}
.wikitable { load.php?la…in=vector:1
ent-text.mw-content-ltr
div.mw-parser-output
table.wikitable.sortable.plainrowheaders.jquery-tablesorter
thead
tr

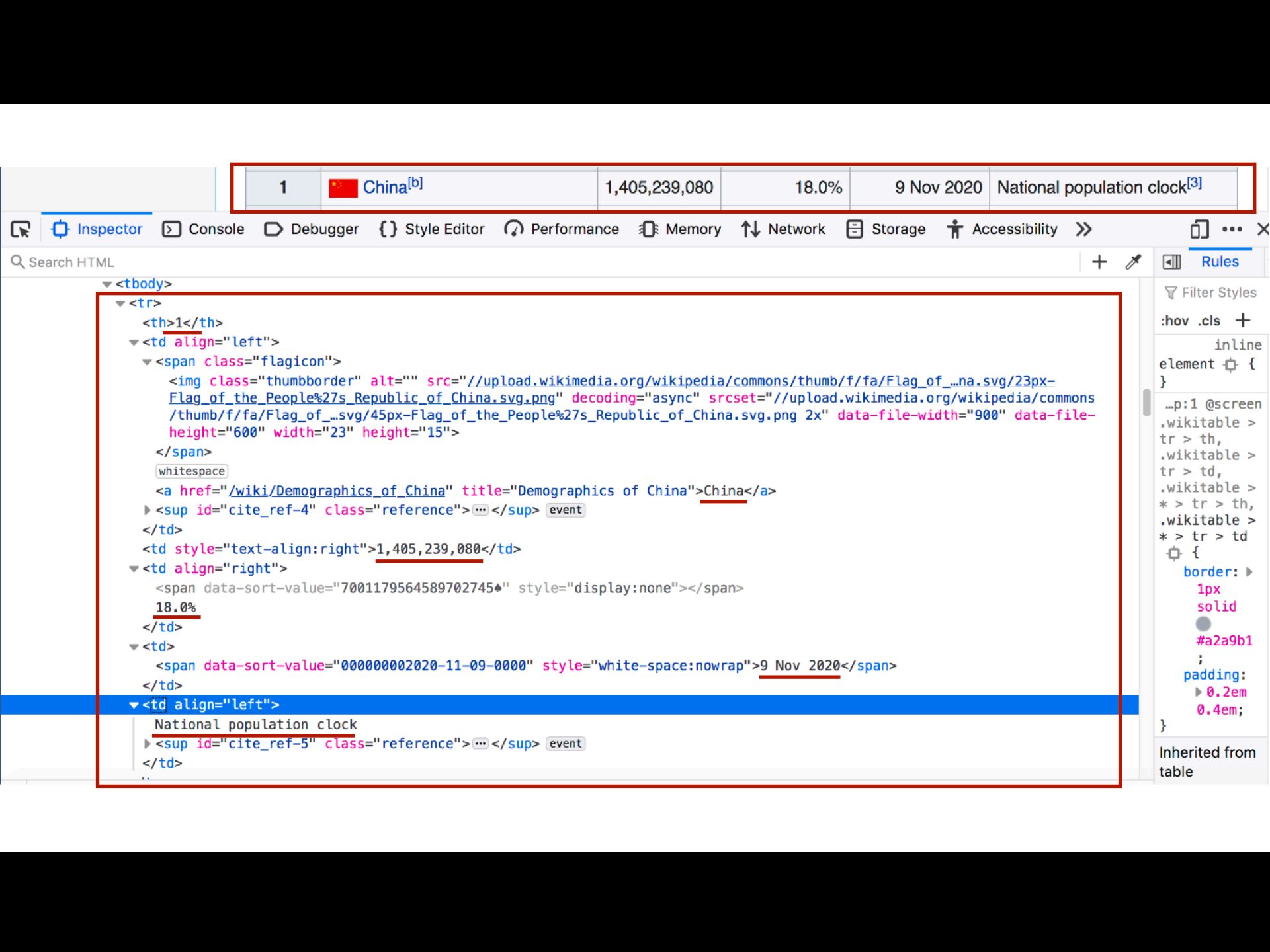

Rank
Country (or dependent territory)
Population
% of world
Date
Source (official or UN)
1
China[b]
1,405,239,080
18.0%
9 Nov 2020
National population clock[3]
National population
Elements
Console
Sources
Network
Performance
Memory
Application
Security
Lighthouse
<table class="wikitable sortable plainrowheaders jquery-tablesorter" style="text-align: right">
<thead> == $0
<tr>
<th data-sort-type="number" class="headerSort" tabindex="0" role="columnheader button" title="Sort ascending">Rank</th>
<th class="headerSort" tabindex="0" role="columnheader button" title="Sort ascending">…</th>
<th class="headerSort" tabindex="0" role="columnheader button" title="Sort ascending">Population</th>
<th class="headerSort" tabindex="0" role="columnheader button" title="Sort ascending">% of world</th>
<th class="headerSort" tabindex="0" role="columnheader button" title="Sort ascending">Date</th>
<th class="unsortable">…</th>
</tr>
Styles
Computed
Event Listeners
Filter
:hov .cls
element.style {
}
thead { user agent stylesheet
display: table-header-group;
vertical-align: middle;
border-color: inherit;
}
Inherited from table.wikita…
Style Attribute {
text-align: right;
}

assigned a numbered rank. In addition, sovereign states with limited recognition are not assigned a number rank.

Rank	Country (or dependent territory)	Population	% of world	Date	Source (official or UN)
1	China[b]	1,405,239,080	18.0%	9 Nov 2020	National population clock[3]




Rank
Country (or dependent territory)
Population
% of world
Date
Source (official or UN)
1
China[b]
1,405,239,080
18.0%
9 Nov 2020
National population clock[3]
National population
Elements
Console
Sources
Network
Performance
Memory
Application
Security
Lighthouse
Population</th>
<th class="headerSort" tabindex="0" role="columnheader button" title="Sort ascending">
% of world</th>
<th class="headerSort" tabindex="0" role="columnheader button" title="Sort ascending">
Date</th>
<th class="unsortable">…</th>
</tr>
</thead>
<tbody> == $0
<tr>
<th>1
</th>
<td align="left">…</td>
<td style="text-align:right">1,405,239,080</td>
<td align="right">…</td>
Styles
Computed
Event Listeners
Filter
:hov .cls
element.style {
}
tbody { user agent stylesheet
display: table-row-group;
vertical-align: middle;
border-color: inherit;
}
Inherited from table.wikita…
Style Attribute {
text-align: right;
}
.wikitable { load.php?la…in=vector:1
content-text.mw-content-ltr
div.mw-parser-output
table.wikitable.sortable.plainrowheaders.jquery-tablesorter
tbody


1
China[b]
1,405,239,080
18.0%
9 Nov 2020
National population clock[3]
Inspector
Console
Debugger
Style Editor
Performance
Memory
Network
Storage
Accessibility
Search HTML
Rules
Filter Styles
:hov .cls
<tbody>
<tr>
<th>1</th>
<td align="left">

<img class="thumbborder" alt="" src="//upload.wikimedia.org/wikipedia/commons/thumb/f/fa/Flag_of_…na.svg/23px-Flag_of_the_People%27s_Republic_of_China.svg.png" decoding="async" srcset="//upload.wikimedia.org/wikipedia/commons/thumb/f/fa/Flag_of_…svg/45px-Flag_of_the_People%27s_Republic_of_China.svg.png 2x" data-file-width="900" data-file-height="600" width="23" height="15">

whitespace
China
[…] event
</td>
<td style="text-align:right">1,405,239,080</td>
<td align="right">

18.0%
</td>
<td>
9 Nov 2020
</td>
<td align="left">
National population clock
[…] event
</td>
inline element { }
…p:1 @screen .wikitable > tr > th, .wikitable > tr > td, .wikitable > * > tr > th, .wikitable > * > tr > td {
border: 1px solid #a2a9b1;
padding: 0.2em 0.4em;
}
Inherited from table

Import external libraries.

In [95]:

```
from urllib.request import urlopen
from bs4 import BeautifulSoup
import pandas as pd
import csv

url = "https://en.wikipedia.org/wiki/List_of_countries_and_dependencies_by_population"
html = urlopen(url)
soup = BeautifulSoup(html,'html.parser')
table = soup.find("table", attrs={"class": "wikitable"})
table_data = table.tbody.find_all("tr")
# Fetch table headers
headings = []
rows = []
for th in table_data[0].find_all("th"):
    # print(th.getText())
    # headings.append(th.getText())
    headings.append(th.getText().strip())
# Fetch table body
h0 = headings[0]
h1 = headings[1]
h2 = headings[2]
h3 = headings[3]
h4 = headings[4]
h5 = headings[5]
row_index = 1 # row index for controlling the loop through all rows
for tbody in table_data[1:len(table_data)]:
    row = tbody.find_all("td")
    col_index = 0 # column index for controlling the loop through all columns
    dict = {}
    dict[h0] = tbody.find("th").getText().strip()
    val = dict[h0]
    # print(row_index,col_index,val)
    for td in row:
        if col_index == 0:
```

The soup class object.

In [95]:

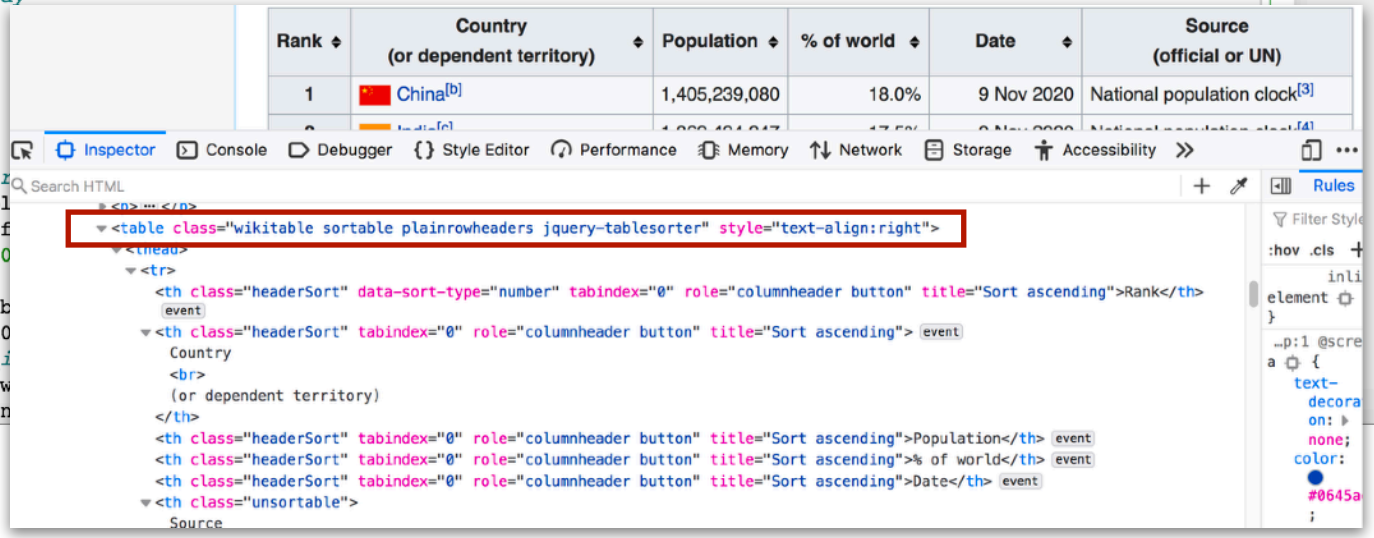
```
from urllib.request import urlopen
from bs4 import BeautifulSoup
import pandas as pd
import csv

url = "https://en.wikipedia.org/wiki/List_of_countries_and_dependencies_by_population"
html = urlopen(url)
soup = BeautifulSoup(html,'html.parser')
table = soup.find("table", attrs={"class": "wikitable"})
table_data = table.tbody.find_all("tr")
# Fetch table headers
headings = []
rows = []
for th in table_data[0].find_all("th"):
    # print(th.getText())
    # headings.append(th.getText())
    headings.append(th.getText().strip())
# Fetch table body
h0 = headings[0]
h1 = headings[1]
h2 = headings[2]
h3 = headings[3]
h4 = headings[4]
h5 = headings[5]
row_index = 1 # row index for controlling the loop through all rows
for tbody in table_data[1:len(table_data)]:
    row = tbody.find_all("td")
    col_index = 0 # column index for controlling the loop through all columns
    dict = {}
    dict[h0] = tbody.find("th").getText().strip()
    val = dict[h0]
    # print(row_index,col_index,val)
    for td in row:
        if col_index == 0:
```

The find_all and find methods.

In [95]:

```
from urllib.request import urlopen
from bs4 import BeautifulSoup
import pandas as pd
import csv

url = "https://en.wikipedia.org/wiki/List_of_countries_and_dependencies_by_population"
html = urlopen(url)
soup = BeautifulSoup(html,'html.parser')
table = soup.find("table", attrs={"class": "wikitable"})
table_data = table.tbody.find_all("tr")
# Fetch table headers
headings = []
rows = []
for th in table_data[0].find_all("th"):
    # print(th.getText())
    # headings.append(th.getText())
    headings.append(th.getText().strip())
# Fetch table body
h0 = headings[0]
h1 = headings[1]
h2 = headings[2]
h3 = headings[3]
h4 = headings[4]
h5 = headings[5]
row_index = 1 # r
for tbody in tabl
    row = tbody.f
    col_index = 0
    dict = {}
    dict[h0] = tb
    val = dict[h0
    # print(row_i
    for td in row
        if col_in
```

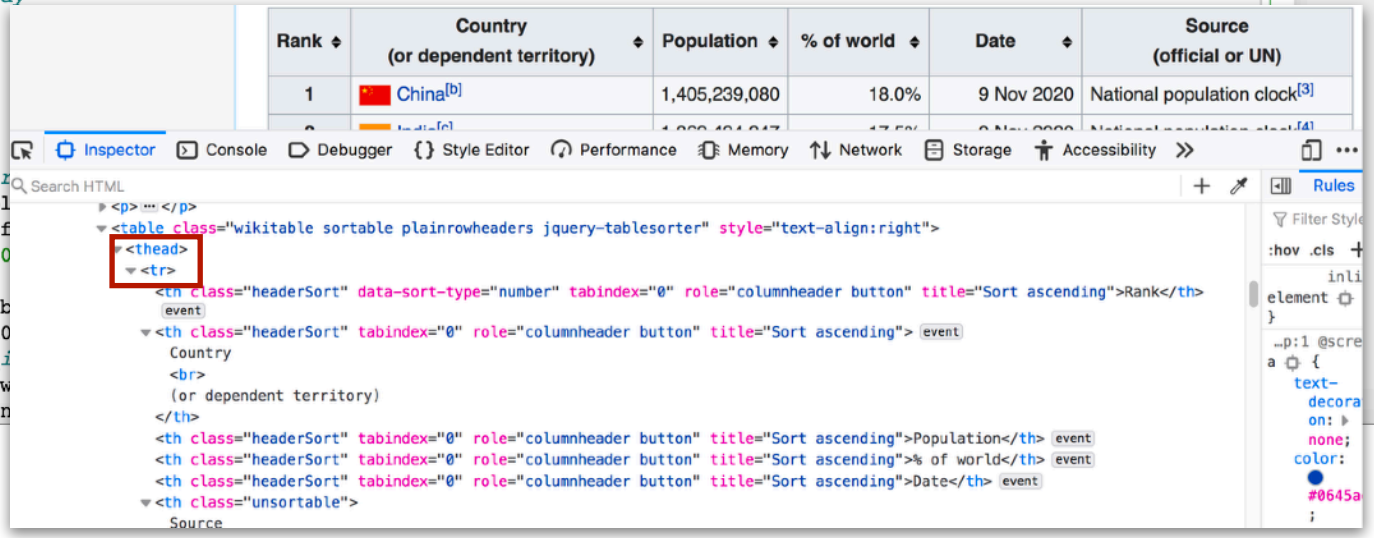
Rank	Country (or dependent territory)	Population	% of world	Date	Source (official or UN)
1	China[b]	1,405,239,080	18.0%	9 Nov 2020	National population clock[3]

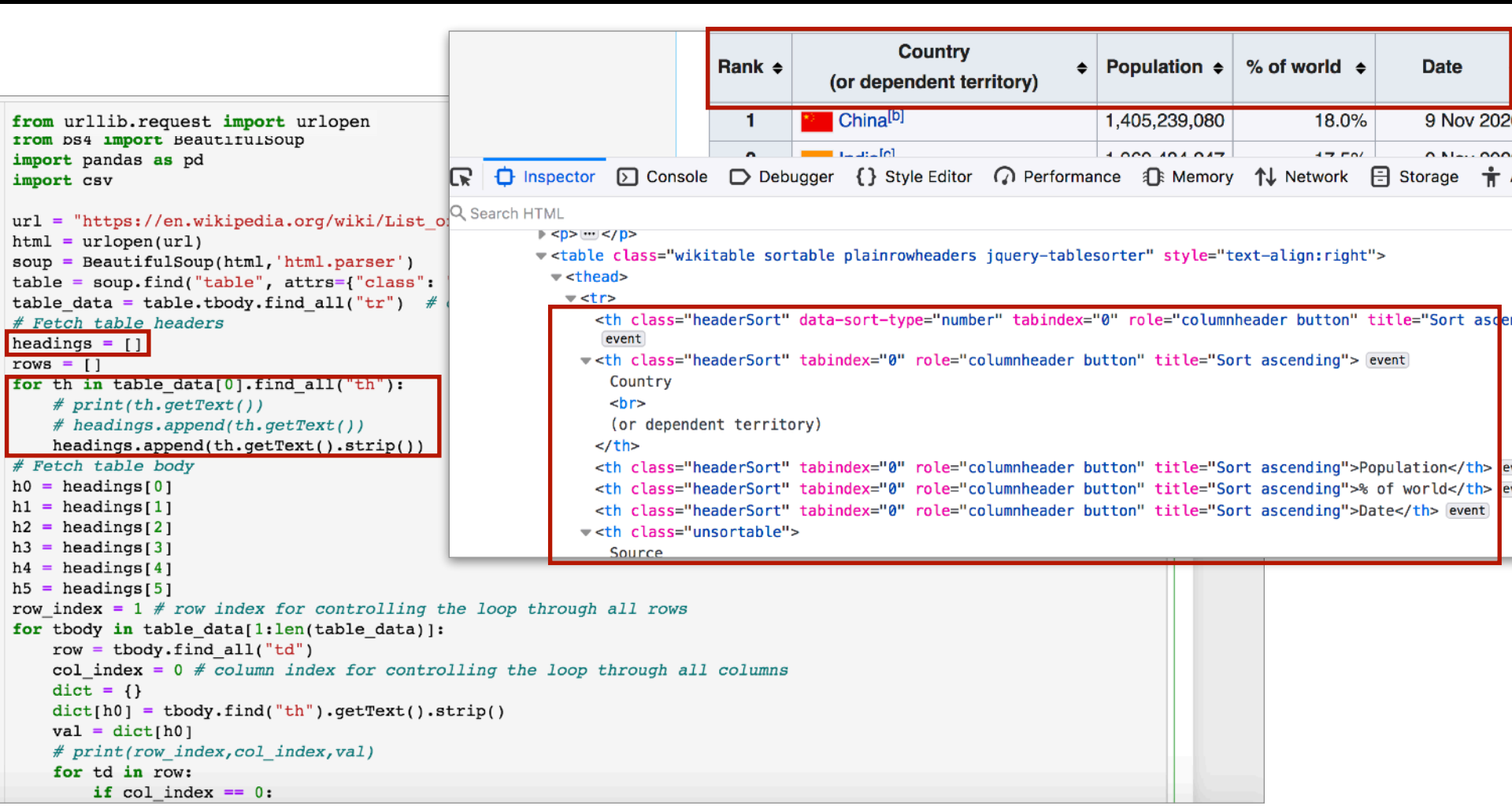


In [95]:

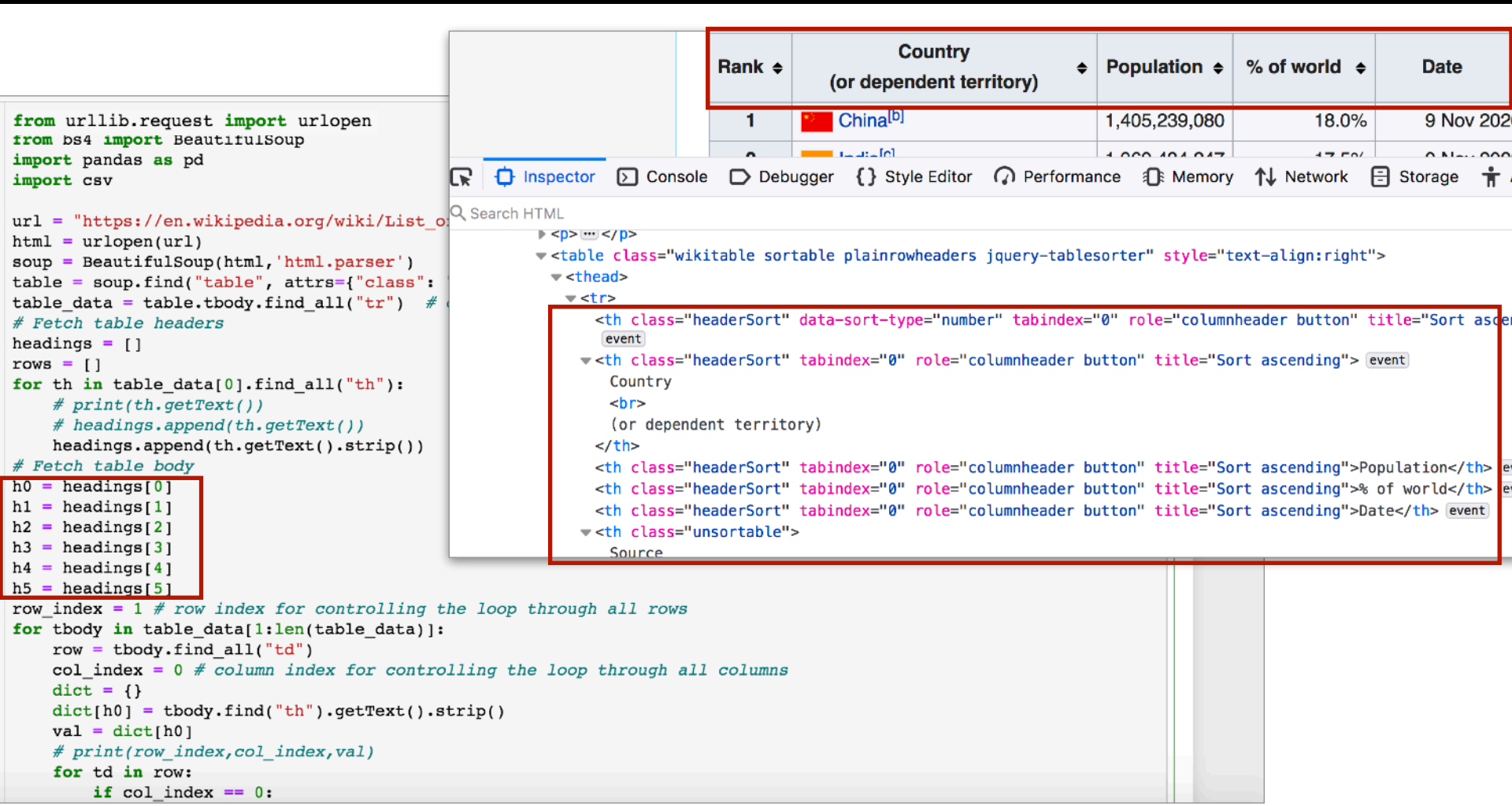
```
from urllib.request import urlopen
from bs4 import BeautifulSoup
import pandas as pd
import csv

url = "https://en.wikipedia.org/wiki/List_of_countries_and_dependencies_by_population"
html = urlopen(url)
soup = BeautifulSoup(html,'html.parser')
table = soup.find("table", attrs={"class": "wikitable"})
table_data = table.tbody.find_all("tr")
# Fetch table headers
headings = []
rows = []
for th in table_data[0].find_all("th"):
    # print(th.getText())
    # headings.append(th.getText())
    headings.append(th.getText().strip())
# Fetch table body
h0 = headings[0]
h1 = headings[1]
h2 = headings[2]
h3 = headings[3]
h4 = headings[4]
h5 = headings[5]
row_index = 1 # r
for tbody in tabl
    row = tbody.f
    col_index = 0
    dict = {}
    dict[h0] = tb
    val = dict[h0
    # print(row_i
    for td in row
        if col_in
```

Rank	Country (or dependent territory)	Population	% of world	Date	Source (official or UN)
1	China[b]	1,405,239,080	18.0%	9 Nov 2020	National population clock[3]



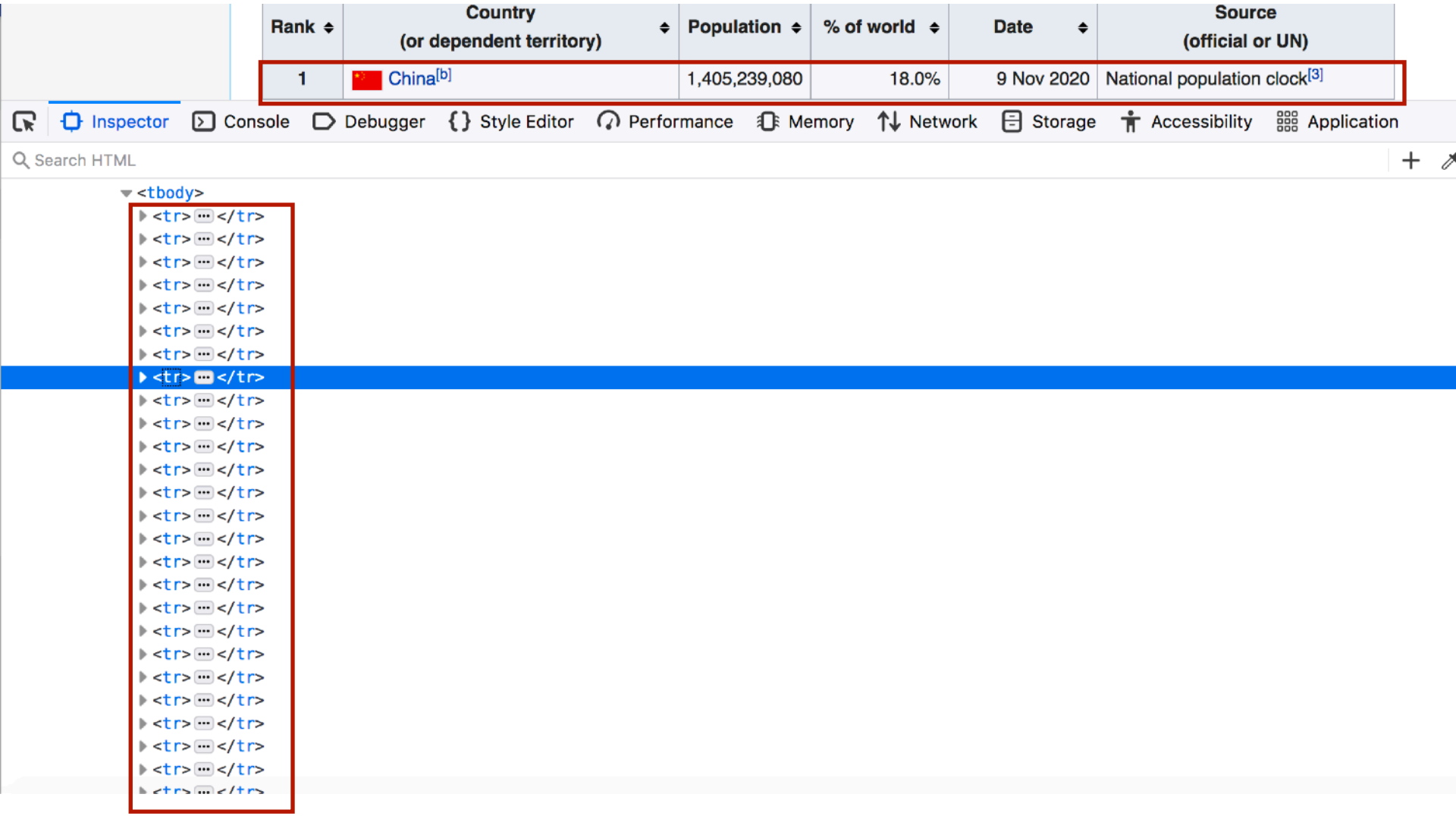

from urllib.request import urlopen
from bs4 import BeautifulSoup
import pandas as pd
import csv
url = "https://en.wikipedia.org/wiki/List_o
html = urlopen(url)
soup = BeautifulSoup(html,'html.parser')
table = soup.find("table", attrs={"class":
table_data = table.tbody.find_all("tr")
# Fetch table headers
headings = []
rows = []
for th in table_data[0].find_all("th"):
# print(th.getText())
# headings.append(th.getText())
headings.append(th.getText().strip())
# Fetch table body
h0 = headings[0]
h1 = headings[1]
h2 = headings[2]
h3 = headings[3]
h4 = headings[4]
h5 = headings[5]
row_index = 1 # row index for controlling the loop through all rows
for tbody in table_data[1:len(table_data)]:
row = tbody.find_all("td")
col_index = 0 # column index for controlling the loop through all columns
dict = {}
dict[h0] = tbody.find("th").getText().strip()
val = dict[h0]
# print(row_index,col_index,val)
for td in row:
if col_index == 0:
Rank
Country (or dependent territory)
Population
% of world
Date
1
China[b]
1,405,239,080
18.0%
9 Nov 202
Inspector
Console
Debugger
Style Editor
Performance
Memory
Network
Storage
Search HTML
 …
<table class="wikitable sortable plainrowheaders jquery-tablesorter" style="text-align:right">
<thead>
<tr>
<th class="headerSort" data-sort-type="number" tabindex="0" role="columnheader button" title="Sort asce
event
<th class="headerSort" tabindex="0" role="columnheader button" title="Sort ascending"> event
Country

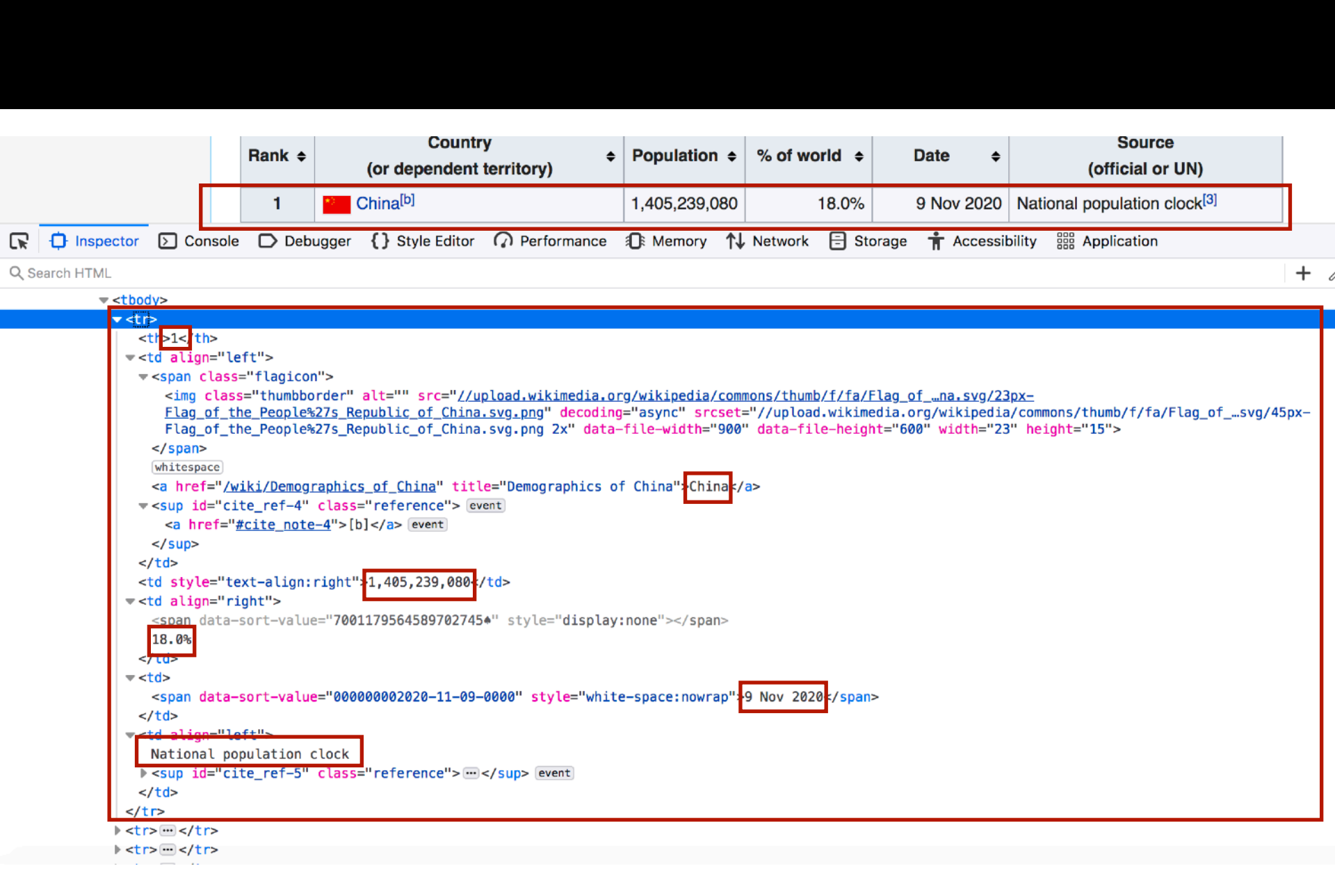
(or dependent territory)
</th>
<th class="headerSort" tabindex="0" role="columnheader button" title="Sort ascending">Population</th>
<th class="headerSort" tabindex="0" role="columnheader button" title="Sort ascending">% of world</th>
<th class="headerSort" tabindex="0" role="columnheader button" title="Sort ascending">Date</th> event
<th class="unsortable">
Source




```
from urllib.request import urlopen
from bs4 import BeautifulSoup
import pandas as pd
import csv

url = "https://en.wikipedia.org/wiki/List_o
html = urlopen(url)
soup = BeautifulSoup(html,'html.parser')
table = soup.find("table", attrs={"class":
table_data = table.tbody.find_all("tr")  #
# Fetch table headers
headings = []
rows = []
for th in table_data[0].find_all("th"):
    # print(th.getText())
    # headings.append(th.getText())
    headings.append(th.getText().strip())
# Fetch table body
h0 = headings[0]
h1 = headings[1]
h2 = headings[2]
h3 = headings[3]
h4 = headings[4]
h5 = headings[5]
row_index = 1 # row index for controlling the loop through all rows
for tbody in table_data[1:len(table_data)]:
    row = tbody.find_all("td")
    col_index = 0 # column index for controlling the loop through all columns
    dict = {}
    dict[h0] = tbody.find("th").getText().strip()
    val = dict[h0]
    # print(row_index,col_index,val)
    for td in row:
        if col_index == 0:
```

Fetching the tbody items.


Rank
Country (or dependent territory)
Population
% of world
Date
Source (official or UN)
1
China[b]
1,405,239,080
18.0%
9 Nov 2020
National population clock[3]
Inspector
Console
Debugger
Style Editor
Performance
Memory
Network
Storage
Accessibility
Application
Search HTML
<tbody>
<tr> ··· </tr>


Rank
Country (or dependent territory)
Population
% of world
Date
Source (official or UN)
1
China[b]
1,405,239,080
18.0%
9 Nov 2020
National population clock[3]
Inspector
Console
Debugger
Style Editor
Performance
Memory
Network
Storage
Accessibility
Application
Search HTML
<tbody>
<tr>
<th>1</th>
<td align="left">

<img class="thumbborder" alt="" src="//upload.wikimedia.org/wikipedia/commons/thumb/f/fa/Flag_of_…na.svg/23px-Flag_of_the_People%27s_Republic_of_China.svg.png" decoding="async" srcset="//upload.wikimedia.org/wikipedia/commons/thumb/f/fa/Flag_of_…svg/45px-Flag_of_the_People%27s_Republic_of_China.svg.png 2x" data-file-width="900" data-file-height="600" width="23" height="15">

whitespace
China
<sup id="cite_ref-4" class="reference"> event
[b] event
</sup>
</td>
<td style="text-align:right">1,405,239,080</td>
<td align="right">

18.0%
</td>
<td>
9 Nov 2020
</td>
<td align="left">
National population clock
[…] event
</td>
</tr>
<tr>…</tr>
<tr>…</tr>
China
1,405,239,080
18.0%
9 Nov 2020
National population clock

```
row_index = 1 # row index for controlling the loop through all rows
for tbody in table_data[1:len(table_data)]:
    row = tbody.find_all("td")
    col_index = 0 # column index for controlling the loop through all columns
    dict = {}
    dict[h0] = tbody.find("th").getText().strip()
    val = dict[h0]
    # print(row_index,col_index,val)
    for td in row:
        if col_index == 0:
            dict[h1] = td.getText().strip()
            val = dict[h1]
        if col_index == 1:
            dict[h2] = td.getText().strip()
            val = dict[h2]
        if col_index == 2:
            dict[h3] = td.getText().strip()
            val = dict[h3]
        if col_index == 3:
            dict[h4] = td.getText().strip()
            val = dict[h4]
        if col_index == 4:
            dict[h5] = td.getText().strip()
            val = dict[h5]
        # print(row_index,col_index+1,val)
        col_index = col_index + 1
    rows.append(dict)
    row_index = row_index + 1
# Define a list of dictionaries
df = pd.DataFrame(rows) # Assign the list of dictionaries to a dataframe
df # display dataframe
df.to_csv('population.csv',mode='w',index=False)
df
```

Rank	Country(or dependent territory)	Population	% of world	Date	Source(official or UN)

```
row_index = 1 # row index for controlling the loop th
for tbody in table_data[1:len(table_data)]:
    row = tbody.find_all("td")
    col_index = 0 # column index for controlling the
    dict = {}
    dict[h0] = tbody.find("th").getText().strip()
    val = dict[h0]
    # print(row_index,col_index,val)
    for td in row:
        if col_index == 0:
            dict[h1] = td.getText().strip()
            val = dict[h1]
        if col_index == 1:
            dict[h2] = td.getText().strip()
            val = dict[h2]
        if col_index == 2:
            dict[h3] = td.getText().strip()
            val = dict[h3]
        if col_index == 3:
            dict[h4] = td.getText().strip()
            val = dict[h4]
        if col_index == 4:
            dict[h5] = td.getText().strip()
            val = dict[h5]
        # print(row_index,col_index+1,val)
        col_index = col_index + 1
    rows.append(dict)
    row_index = row_index + 1
# Define a list of dictionaries
df = pd.DataFrame(rows) # Assign the list of dictiona
df # display dataframe
df.to_csv('population.csv',mode='w',index=False)
df
```

Rank	Country (or dependent territory)	Population	% of world
1	China[b]	1,405,239,080	18.0

Inspector | Console | Debugger | Style Editor | Performance | Memory | Network

Search HTML

```
<tbody>
  <tr>
    <th>1</th>
    <td align="left">
      <span class="flagicon">
        <img class="thumbborder" alt="" src="//upload.wikimedia.org/wikipedia/commons/thumb/f/
        Flag_of_the_People%27s_Republic_of_China.svg.png" decoding="async" srcset="//upload.wi
        Flag_of_the_People%27s_Republic_of_China.svg.png 2x" data-file-width="900" data-file-h
      </span>
      whitespace
      <a href="/wiki/Demographics_of_China" title="Demographics of China">China</a>
      <sup id="cite_ref-4" class="reference"> event
        <a href="#cite_note-4">[b]</a> event
      </sup>
    </td>
    <td style="text-align:right">1,405,239,080</td>
    <td align="right">
      <span data-sort-value="7001179564589702745♠" style="display:none"></span>
      18.0%
    </td>
    <td>
      <span data-sort-value="000000002020-11-09-0000" style="white-space:nowrap">9 Nov 2020</s
    </td>
    <td align="left">
      National population clock
      <sup id="cite_ref-5" class="reference">…</sup> event
    </td>
  </tr>
  <tr>…</tr>
  <tr>…</tr>
```

```
row_index = 1 # row index for controlling the loop th
for tbody in table_data[1:len(table_data)]:
    row = tbody.find_all("td")
    col_index = 0 # column index for controlling the
    dict = {}
    dict[h0] = tbody.find("th").getText().strip()
    val = dict[h0]
    # print(row_index,col_index,val)
    for td in row:
        if col_index == 0:
            dict[h1] = td.getText().strip()
            val = dict[h1]
        if col_index == 1:
            dict[h2] = td.getText().strip()
            val = dict[h2]
        if col_index == 2:
            dict[h3] = td.getText().strip()
            val = dict[h3]
        if col_index == 3:
            dict[h4] = td.getText().strip()
            val = dict[h4]
        if col_index == 4:
            dict[h5] = td.getText().strip()
            val = dict[h5]
        # print(row_index,col_index+1,val)
        col_index = col_index + 1
    rows.append(dict)
    row_index = row_index + 1
# Define a list of dictionaries
df = pd.DataFrame(rows) # Assign the list of dictiona
df # display dataframe
df.to_csv('population.csv',mode='w',index=False)
df
```

Rank	Country (or dependent territory)	Population	% of world
1	China[b]	1,405,239,080	18.0

Inspector | Console | Debugger | Style Editor | Performance | Memory | Network

Search HTML

```
<tbody>
  <tr>
    <th>1</th>
    <td align="left">
      <span class="flagicon">
        <img class="thumbborder" alt="" src="//upload.wikimedia.org/wikipedia/commons/thumb/f/
        Flag_of_the_People%27s_Republic_of_China.svg.png" decoding="async" srcset="//upload.wi
        Flag_of_the_People%27s_Republic_of_China.svg.png 2x" data-file-width="900" data-file-
      </span>
      whitespace
      <a href="/wiki/Demographics_of_China" title="Demographics of China">China</a>
      <sup id="cite_ref-4" class="reference"> event
        <a href="#cite_note-4">[b]</a> event
      </sup>
    </td>
    <td style="text-align:right">1,405,239,080</td>
    <td align="right">
      <span data-sort-value="7001179564589702745♠" style="display:none"></span>
      18.0%
    </td>
    <td>
      <span data-sort-value="000000002020-11-09-0000" style="white-space:nowrap">9 Nov 2020</
    </td>
    <td align="left">
      National population clock
      <sup id="cite_ref-5" class="reference">…</sup> event
    </td>
  </tr>
  <tr>…</tr>
  <tr>…</tr>
```

```
row_index = 1 # row index for controlling the loop th
for tbody in table_data[1:len(table_data)]:
    row = tbody.find_all("td")
    col_index = 0 # column index for controlling the
    dict = {}
    dict[h0] = tbody.find("th").getText().strip()
    val = dict[h0]
    # print(row_index,col_index,val)
    for td in row:
        if col_index == 0:
            dict[h1] = td.getText().strip()
            val = dict[h1]
        if col_index == 1:
            dict[h2] = td.getText().strip()
            val = dict[h2]
        if col_index == 2:
            dict[h3] = td.getText().strip()
            val = dict[h3]
        if col_index == 3:
            dict[h4] = td.getText().strip()
            val = dict[h4]
        if col_index == 4:
            dict[h5] = td.getText().strip()
            val = dict[h5]
        # print(row_index,col_index+1,val)
        col_index = col_index + 1
    rows.append(dict)
    row_index = row_index + 1
# Define a list of dictionaries
df = pd.DataFrame(rows) # Assign the list of dictiona
df # display dataframe
df.to_csv('population.csv',mode='w',index=False)
df
```

Rank	Country (or dependent territory)	Population	% of world
1	China[b]	1,405,239,080	18.0

Inspector | Console | Debugger | Style Editor | Performance | Memory | Network

Search HTML

```
<tbody>
  <tr>
    <th>1</th>
    <td align="left">
      <span class="flagicon">
        <img class="thumbborder" alt="" src="//upload.wikimedia.org/wikipedia/commons/thumb/f/
        Flag_of_the_People%27s_Republic_of_China.svg.png" decoding="async" srcset="//upload.wi
        Flag_of_the_People%27s_Republic_of_China.svg.png 2x" data-file-width="900" data-file-h
      </span>
      whitespace
      <a href="/wiki/Demographics_of_China" title="Demographics of China">China</a>
      <sup id="cite_ref-4" class="reference"> event
        <a href="#cite_note-4">[b]</a> event
      </sup>
    </td>
    <td style="text-align:right">1,405,239,080</td>
    <td align="right">
      <span data-sort-value="7001179564589702745♠" style="display:none"></span>
      18.0%
    </td>
    <td>
      <span data-sort-value="000000002020-11-09-0000" style="white-space:nowrap">9 Nov 2020</s
    </td>
    <td align="left">
      National population clock
      <sup id="cite_ref-5" class="reference">…</sup> event
    </td>
  </tr>
  <tr>…</tr>
  <tr>…</tr>
```

```
row_index = 1 # row index for controlling the loop th
for tbody in table_data[1:len(table_data)]:
    row = tbody.find_all("td")
    col_index = 0 # column index for controlling the
    dict = {}
    dict[h0] = tbody.find("th").getText().strip()
    val = dict[h0]
    # print(row_index,col_index,val)
    for td in row:
        if col_index == 0:
            dict[h1] = td.getText().strip()
            val = dict[h1]
        if col_index == 1:
            dict[h2] = td.getText().strip()
            val = dict[h2]
        if col_index == 2:
            dict[h3] = td.getText().strip()
            val = dict[h3]
        if col_index == 3:
            dict[h4] = td.getText().strip()
            val = dict[h4]
        if col_index == 4:
            dict[h5] = td.getText().strip()
            val = dict[h5]
        # print(row_index,col_index+1,val)
        col_index = col_index + 1
    rows.append(dict)
    row_index = row_index + 1
# Define a list of dictionaries
df = pd.DataFrame(rows) # Assign the list of dictiona
df # display dataframe
df.to_csv('population.csv',mode='w',index=False)
df
```

Rank	Country (or dependent territory)	Population	% of world
1	China[b]	1,405,239,080	18.0

Inspector | Console | Debugger | Style Editor | Performance | Memory | Network

Search HTML

```
<tbody>
  <tr>
    <th>1</th>
    <td align="left">
      <span class="flagicon">
        <img class="thumbborder" alt="" src="//upload.wikimedia.org/wikipedia/commons/thumb/f/
        Flag_of_the_People%27s_Republic_of_China.svg.png" decoding="async" srcset="//upload.wi
        Flag_of_the_People%27s_Republic_of_China.svg.png 2x" data-file-width="900" data-file-
      </span>
      whitespace
      <a href="/wiki/Demographics_of_China" title="Demographics of China">China</a>
      <sup id="cite_ref-4" class="reference"> event
        <a href="#cite_note-4">[b]</a> event
      </sup>
    </td>
    <td style="text-align:right">1,405,239,080</td>
    <td align="right">
      <span data-sort-value="7001179564589702745♠" style="display:none"></span>
      18.0%
    </td>
    <td>
      <span data-sort-value="000000002020-11-09-0000" style="white-space:nowrap">9 Nov 2020</
    </td>
    <td align="left">
      National population clock
      <sup id="cite_ref-5" class="reference">…</sup> event
    </td>
  </tr>
  <tr>…</tr>
  <tr>…</tr>
```

```
row_index = 1 # row index for controlling the loop th
for tbody in table_data[1:len(table_data)]:
    row = tbody.find_all("td")
    col_index = 0 # column index for controlling the
    dict = {}
    dict[h0] = tbody.find("th").getText().strip()
    val = dict[h0]
    # print(row_index,col_index,val)
    for td in row:
        if col_index == 0:
            dict[h1] = td.getText().strip()
            val = dict[h1]
        if col_index == 1:
            dict[h2] = td.getText().strip()
            val = dict[h2]
        if col_index == 2:
            dict[h3] = td.getText().strip()
            val = dict[h3]
        if col_index == 3:
            dict[h4] = td.getText().strip()
            val = dict[h4]
        if col_index == 4:
            dict[h5] = td.getText().strip()
            val = dict[h5]
        # print(row_index,col_index+1,val)
        col_index = col_index + 1
    rows.append(dict)
    row_index = row_index + 1
# Define a list of dictionaries
df = pd.DataFrame(rows) # Assign the list of dictiona
df # display dataframe
df.to_csv('population.csv',mode='w',index=False)
df
```

Rank	Country (or dependent territory)	Population	% of world
1	China[b]	1,405,239,080	18.0

Inspector | Console | Debugger | Style Editor | Performance | Memory | Network

Search HTML

```
<tbody>
  <tr>
    <th>1</th>
    <td align="left">
      <span class="flagicon">
        <img class="thumbborder" alt="" src="//upload.wikimedia.org/wikipedia/commons/thumb/f/
        Flag_of_the_People%27s_Republic_of_China.svg.png" decoding="async" srcset="//upload.wi
        Flag_of_the_People%27s_Republic_of_China.svg.png 2x" data-file-width="900" data-file-h
      </span>
      whitespace
      <a href="/wiki/Demographics_of_China" title="Demographics of China">China</a>
      <sup id="cite_ref-4" class="reference"> event
        <a href="#cite_note-4">[b]</a> event
      </sup>
    </td>
    <td style="text-align:right">1,405,239,080</td>
    <td align="right">
      <span data-sort-value="7001179564589702745♠" style="display:none"></span>
      18.0%
    </td>
    <td>
      <span data-sort-value="000000002020-11-09-0000" style="white-space:nowrap">9 Nov 2020</s
    </td>
    <td align="left">
      National population clock
      <sup id="cite_ref-5" class="reference">…</sup> event
    </td>
  </tr>
  <tr>…</tr>
  <tr>…</tr>
```

jupyter population.csv Yesterday at 5:49 PM

Logout

File Edit View Language current mode

```
Rank,Country(or dependent territory),Population,% of world,Date,Source(official or UN)
1,China[b],"1,405,239,080",18.0%,9 Nov 2020,National population clock[3]
2,India[c],"1,369,434,247",17.5%,9 Nov 2020,National population clock[4]
3,United States[d],"330,617,844",4.22%,9 Nov 2020,National population clock[5]
4,Indonesia,"269,603,400",3.45%,1 Jul 2020,National annual projection[6]
5,Pakistan[e],"220,892,331",2.82%,1 Jul 2020,UN Projection[2]
6,Brazil,"212,311,986",2.71%,9 Nov 2020,National population clock[7]
7,Nigeria,"206,139,587",2.63%,1 Jul 2020,UN Projection[2]
8,Bangladesh,"169,615,266",2.17%,9 Nov 2020,National population clock[8]
9,Russia[f],"146,748,590",1.88%,1 Jan 2020,National annual estimate[9]
10,Mexico,"127,792,286",1.63%,1 Jul 2020,National annual projection[10]
11,Japan,"125,880,000",1.61%,1 Oct 2020,Monthly national estimate[11]
12,Philippines,"109,406,628",1.40%,9 Nov 2020,National population clock[12]
13,DR Congo,"101,935,800",1.30%,1 Jul 2020,National annual projection[13]
14,Egypt,"101,155,808",1.29%,9 Nov 2020,National population clock[14]
15,Ethiopia,"100,829,000",1.29%,1 Jul 2020,National annual projection[15]
16,Vietnam,"96,483,981",1.23%,1 Jul 2019,National annual estimate[16]
17,Iran,"83,933,996",1.07%,9 Nov 2020,National population clock[17]
18,Turkey,"83,154,997",1.06%,31 Dec 2019,National annual estimate[18]
19,Germany,"83,122,889",1.06%,30 Jun 2020,National quarterly estimate[19]
20,France[g],"67,146,000",0.858%,1 Oct 2020,Monthly national estimate[20]
21,United Kingdom[h],"66,796,807",0.854%,30 Jun 2019,National annual estimate[21]
22,Thailand,"66,574,242",0.851%,9 Nov 2020,National population clock[22]
23,Italy,"60,026,546",0.767%,30 Jun 2020,Monthly national estimate[23]
24,South Africa,"59,622,350",0.762%,1 Jul 2020,National annual estimate[24]
25,Tanzania[i],"57,637,628",0.737%,1 Jul 2020,National annual projection[25]
26,Myanmar,"54,817,919",0.700%,1 Jul 2020,National annual projection[26]
27,South Korea,"51,841,786",0.662%,1 Sep 2020,Monthly national estimate[27]
28,Colombia,"50,372,424",0.644%,30 Jun 2020,National annual projection[28]
29,Kenya,"47,564,296",0.608%,31 Aug 2019,2019 census result[29]
30,Spain,"47,329,981",0.605%,1 Jan 2020,National semi-annual estimate[30]
31,Argentina,"45,376,763",0.580%,1 Jul 2020,National annual projection[31]
```

Sheet 1

Table data was imported. Adjust Settings

population

Rank	Country(or dependent territory)	Population	% of world	Date	Source(official or UN)
1	China[b]	1,405,239,080	18.0%	9 Nov 2020	National population clock[3]
2	India[c]	1,369,434,247	17.5%	9 Nov 2020	National population clock[4]
3	United States[d]	330,617,844	4.22%	9 Nov 2020	National population clock[5]
4	Indonesia	269,603,400	3.45%	1 Jul 2020	National annual projection[6]
5	Pakistan[e]	220,892,331	2.82%	1 Jul 2020	UN Projection[2]
6	Brazil	212,311,986	2.71%	9 Nov 2020	National population clock[7]
7	Nigeria	206,139,587	2.63%	1 Jul 2020	UN Projection[2]
8	Bangladesh	169,615,266	2.17%	9 Nov 2020	National population clock[8]
9	Russia[f]	146,748,590	1.88%	1 Jan 2020	National annual estimate[9]
10	Mexico	127,792,286	1.63%	1 Jul 2020	National annual projection[10]
11	Japan	125,880,000	1.61%	1 Oct 2020	Monthly national estimate[11]
12	Philippines	109,406,628	1.40%	9 Nov 2020	National population clock[12]
13	DR Congo	101,935,800	1.30%	1 Jul 2020	National annual projection[13]
14	Egypt	101,155,808	1.29%	9 Nov 2020	National population clock[14]
15	Ethiopia	100,829,000	1.29%	1 Jul 2020	National annual projection[15]
16	Vietnam	96,483,981	1.23%	1 Jul 2019	National annual estimate[16]
17	Iran	83,933,996	1.07%	9 Nov 2020	National population clock[17]
18	Turkey	83,154,997	1.06%	31 Dec 2019	National annual estimate[18]
19	Germany	83,122,889	1.06%	30 Jun 2020	National quarterly estimate[19]
20	France[g]	67,146,000	0.858%	1 Oct 2020	Monthly national estimate[20]

Exercise time.

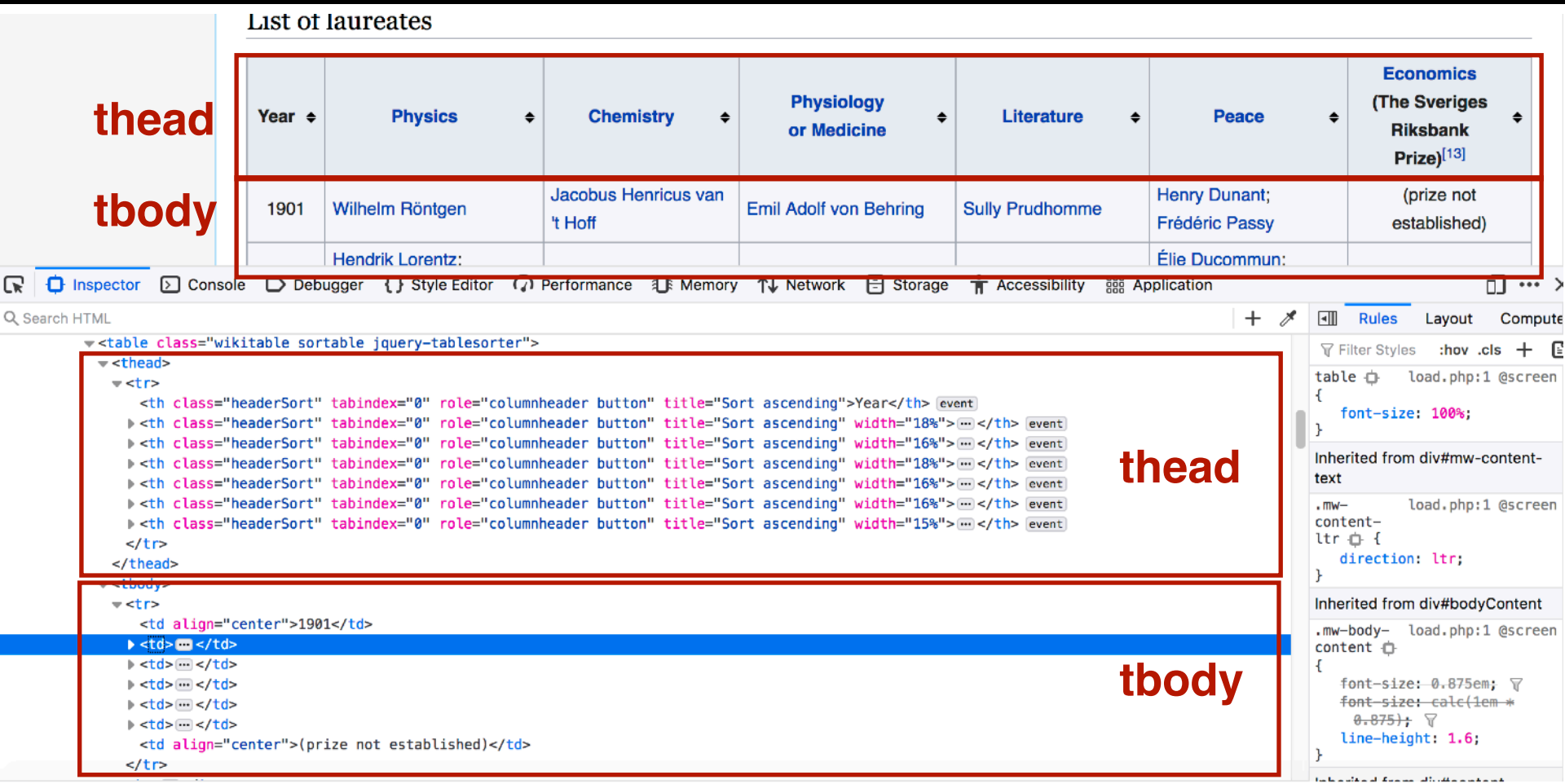
List of Nobel laureates

From Wikipedia, the free encyclopedia
(Redirected from List of nobel prize winners)

The Nobel Prizes (Swedish: *Nobelpriset*, Norwegian: *Nobelprisen*) are prizes awarded annually by the Royal Swedish Academy of Sciences, the Swedish Academy, the Karolinska Institutet, and the Norwegian Nobel Committee to individuals and organizations who make outstanding contributions in the fields of chemistry, physics, literature, peace, and physiology or medicine.[1] They were established by the 1895 will of Alfred Nobel, which dictates that the awards should be administered by the Nobel Foundation. The Nobel Memorial Prize in Economic Sciences was established in 1968 by the Sveriges Riksbank, the central bank of Sweden, for contributions to the field of economics. Each recipient, or "laureate", receives a gold medal, a diploma, and a sum of money, which is decided annually by the Nobel Foundation.[2]

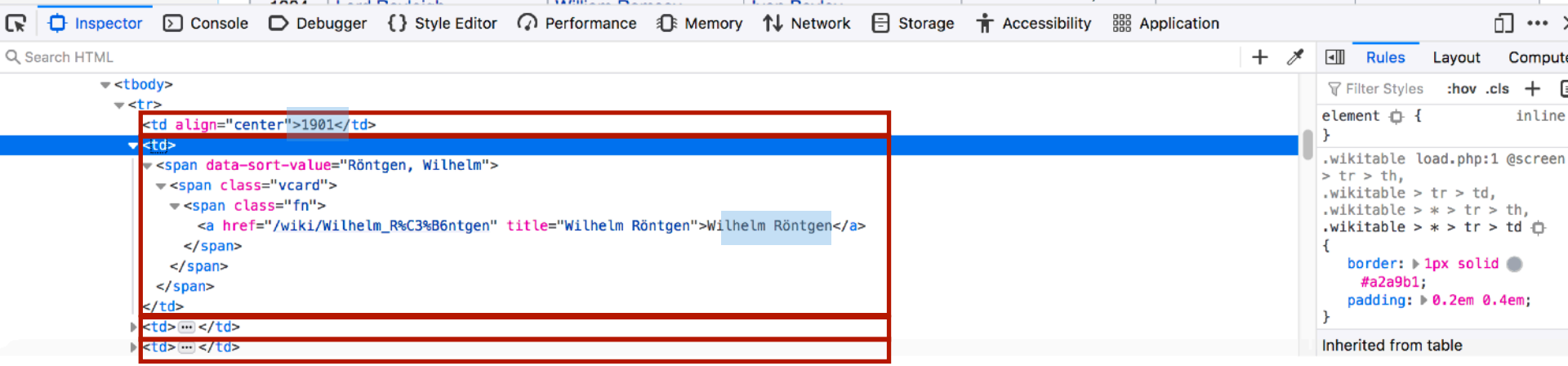
Nobel laureates receive a gold medal together with a diploma and (as of 2017) 9 million SEK (roughly US $1.0 million, €0.87 million).

https://en.wikipedia.org/wiki/List_of_Nobel_laureates


List of laureates
thead
tbody
Year
Physics
Chemistry
Physiology or Medicine
Literature
Peace
Economics (The Sveriges Riksbank Prize)[13]
1901
Wilhelm Röntgen
Jacobus Henricus van 't Hoff
Emil Adolf von Behring
Sully Prudhomme
Henry Dunant; Frédéric Passy
(prize not established)
Hendrik Lorentz:
Élie Ducommun:
Inspector
Console
Debugger
Style Editor
Performance
Memory
Network
Storage
Accessibility
Application
Search HTML
Rules
Layout
Compute
Filter Styles
:hov .cls
<table class="wikitable sortable jquery-tablesorter">
<thead>
<tr>
<th class="headerSort" tabindex="0" role="columnheader button" title="Sort ascending">Year</th> event
<th class="headerSort" tabindex="0" role="columnheader button" title="Sort ascending" width="18%">…</th> event
<th class="headerSort" tabindex="0" role="columnheader button" title="Sort ascending" width="16%">…</th> event
<th class="headerSort" tabindex="0" role="columnheader button" title="Sort ascending" width="18%">…</th> event
<th class="headerSort" tabindex="0" role="columnheader button" title="Sort ascending" width="16%">…</th> event
<th class="headerSort" tabindex="0" role="columnheader button" title="Sort ascending" width="16%">…</th> event
<th class="headerSort" tabindex="0" role="columnheader button" title="Sort ascending" width="15%">…</th> event
</tr>
</thead>
<tr>
<td align="center">1901</td>
<td>…</td>
<td>…</td>
<td>…</td>
<td>…</td>
<td>…</td>
<td align="center">(prize not established)</td>
</tr>
thead
tbody
table load.php:1 @screen
{
font-size: 100%;
}
Inherited from div#mw-content-text
.mw-content-ltr load.php:1 @screen
{
direction: ltr;
}
Inherited from div#bodyContent
.mw-body-content load.php:1 @screen
{
font-size: 0.875em;
font-size: calc(1em * 0.875);
line-height: 1.6;
}

List of laureates

Year	Physics	Chemistry	Physiology or Medicine	Literature	Peace	Economics (The Sveriges Riksbank Prize)[13]
1901	Wilhelm Röntgen	Jacobus Henricus van 't Hoff	Emil Adolf von Behring	Sully Prudhomme	Henry Dunant; Frédéric Passy	(prize not established)
1902	Hendrik Lorentz; Pieter Zeeman	Hermann Emil Fischer	Ronald Ross	Theodor Mommsen	Élie Ducommun; Charles Albert Gobat	—
1903	Henri Becquerel; Pierre Curie; Marie Curie	Svante Arrhenius	Niels Ryberg Finsen	Bjørnstjerne Bjørnson	Randal Cremer	—
				Frédéric Mistral;	Institut de Droit	



```
# from urllib.request import urlopen
from bs4 import BeautifulSoup
import pandas as pd
import csv

url = "https://en.wikipedia.org/wiki/List_of_Nobel_l
html = urlopen(url)
soup = BeautifulSoup(html,'html.parser')
table = soup.find("table", attrs={"class": "wikitabl
table_data = table.tbody.find_all("tr")  # contains
# Fetch table headers
headings = []
rows = []
for th in table_data[0].find_all("th"):
    # print(th.getText())
    # headings.append(th.getText())
    headings.append(th.getText().strip())
# Fetch table body
h0 = headings[0]
h1 = headings[1]
h2 = headings[2]
h3 = headings[3]
h4 = headings[4]
h5 = headings[5]
h6 = headings[6]
row_index = 1 # row index for controlling the loop t
for tbody in table_data[1:len(table_data)]:
    row = tbody.find_all("td")
    col_index = 0 # column index for controlling the
    dict = {}
    # dict[h0] = tbody.find("th").getText().strip()
    # val = dict[h0]
    # print(row_index,col_index,val)
    for td in row:
        if col_index == 0:
            dict[h0] = td.getText().strip()
            val = dict[h0]
        if col_index == 1:
            dict[h1] = td.getText().strip()
            val = dict[h1]
        if col_index == 2:
            dict[h2] = td.getText().strip()
            val = dict[h2]
        if col_index == 3:
            dict[h3] = td.getText().strip()
            val = dict[h3]
        if col_index == 4:
            dict[h4] = td.getText().strip()
            val = dict[h4]
        if col_index == 5:
            dict[h5] = td.getText().strip()
            val = dict[h5]
        if col_index == 6:
            dict[h6] = td.getText().strip()
            val = dict[h6]
        # print(row_index,col_index+1,val)
        col_index = col_index + 1
    rows.append(dict)
    row_index = row_index + 1
# Define a list of dictionaries
df = pd.DataFrame(rows) # Assign the list of diction
df # display dataframe
df.to_csv('nobel_laureates.csv',mode='w',index=False)
df
```

```
row_index = 1 # row index for controlling the loop through all rows
for tbody in table_data[1:len(table_data)]:
    row = tbody.find_all("td")
    col_index = 0 # column index for controlling the loop through all columns
    dict = {}
    # dict[h0] = tbody.find("th").getText().strip()
    # val = dict[h0]
    # print(row_index,col_index,val)
    for td in row:
        if col_index == 0:
            dict[h0] = td.getText().strip()
            val = dict[h0]
        if col_index == 1:
            dict[h1] = td.getText().strip()
            val = dict[h1]
        if col_index == 2:
            dict[h2] = td.getText().strip()
            val = dict[h2]
        if col_index == 3:
            dict[h3] = td.getText().strip()
            val = dict[h3]
        if col_index == 4:
            dict[h4] = td.getText().strip()
            val = dict[h4]
        if col_index == 5:
            dict[h5] = td.getText().strip()
            val = dict[h5]
        if col_index == 6:
            dict[h6] = td.getText().strip()
            val = dict[h6]
        # print(row_index,col_index+1,val)
        col_index = col_index + 1
    rows.append(dict)
    row_index = row_index + 1
```

Scraping multiple pages.

My Account | Login/ Register | English

Search entire store here...

CATALOG

My Cart HK$0.00

NEW IN | WOMEN | MEN | KIDS | FOOTWEAR | APPAREL | ACCESSORIES | BRANDS | SALE | KAKAO FRIENDS X CATALOG

Home ▸ Footwear

SHOP BY

CATEGORY

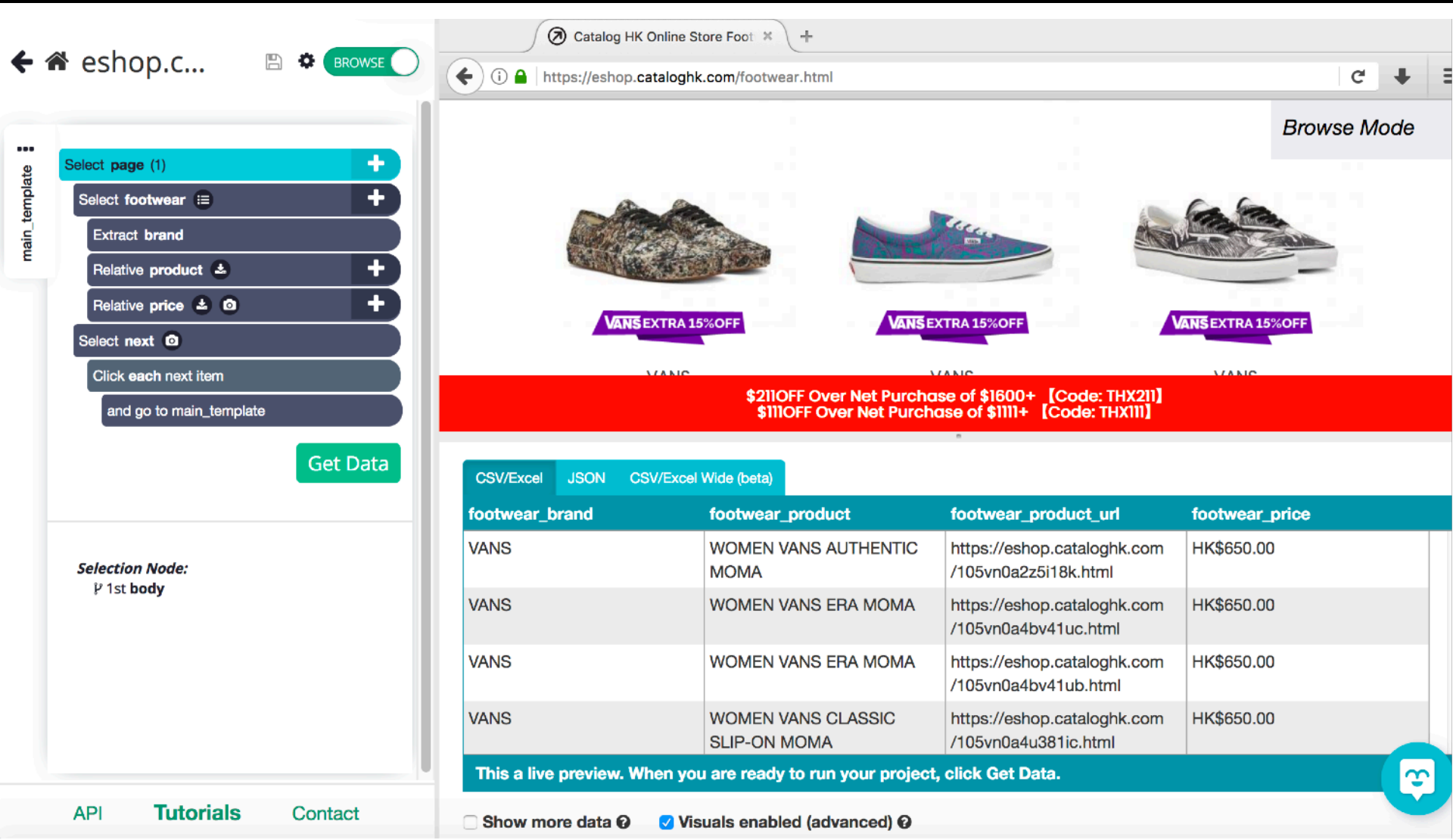
Sneakers	406
Casual Footwear	262
Sandals	133

BRANDS

adidas Age Asics Birkenstock BT21 MEETS CATALOG Converse Crocs Dr. Martens Havaianas Jason Markk Keen

Sort By: New ↓ Show: 60 Page: 1 2 3 4 5 ▸

https://eshop.cataloghk.com/footwear.html


eshop.c...
BROWSE
main_template
Select page (1)
Select footwear
Extract brand
Relative product
Relative price
Select next
Click each next item
and go to main_template
Get Data
Selection Node:
1st body
API
Tutorials
Contact
Catalog HK Online Store Foot
https://eshop.cataloghk.com/footwear.html
Browse Mode
VANS EXTRA 15%OFF
VANS EXTRA 15%OFF
VANS EXTRA 15%OFF
$211OFF Over Net Purchase of $1600+ 【Code: THX211】
$111OFF Over Net Purchase of $1111+ 【Code: THX111】
CSV/Excel
JSON
CSV/Excel Wide (beta)
footwear_brand
footwear_product
footwear_product_url
footwear_price
VANS
WOMEN VANS AUTHENTIC MOMA
https://eshop.cataloghk.com/105vn0a2z5i18k.html
HK$650.00
VANS
WOMEN VANS ERA MOMA
https://eshop.cataloghk.com/105vn0a4bv41uc.html
HK$650.00
VANS
WOMEN VANS ERA MOMA
https://eshop.cataloghk.com/105vn0a4bv41ub.html
HK$650.00
VANS
WOMEN VANS CLASSIC SLIP-ON MOMA
https://eshop.cataloghk.com/105vn0a4u381ic.html
HK$650.00
This a live preview. When you are ready to run your project, click Get Data.
Show more data
Visuals enabled (advanced)

Determine page counter.

https://eshop.cataloghk.com/footwear.html?p=2

Most Visited | Getting Started | Design Thinking & ... | 思潮 AVANT-GARDE... | Create Interactive c... | hkdc_dt_event phot...

My Account | Login/ Register | English

Search entire store here...

CATALOG

My Cart

NEW IN | WOMEN | MEN | KIDS | FOOTWEAR | APPAREL | ACCESSORIES | BRANDS | SALE | KAKAO FRIENDS X CATALOG

Home > Footwear

SHOP BY

CATEGORY

Sneakers	424
Casual Footwear	241
Sandals	133

BRANDS

adidas Age Asics Birkenstock
BT21 MEETS CATALOG Converse Crocs Havaianas

Sort By: New | Show: 60 | Page: 1 2 3 4 5



eshop.cataloghk.com(https://eshop.cataloghk.com/footwear.html)

In [214]:

```
import requests
import csv
import pandas as pd
from bs4 import BeautifulSoup

header = ['page #','brand','url','product','price']
data = []
# Display and store away 2 pages of scrapped data from startupbeat.hkej.com
for i in range(1,4):
    quote_page = requests.get('https://eshop.cataloghk.com/footwear.html?p='+str(i))
    # print("\n***** Page " + str(i) +" in action *****")
    soup = BeautifulSoup(quote_page.content,'html.parser')
    for item in soup.find_all("li", attrs={"class":"item product product-item"}):
        page_no = str(i)
        brand = item.find("div",attrs={"class":"product details product-item-details box-info"}).span.get_text()
        url = item.find("div",attrs={"class":"product details product-item-details box-info"}).a.get('href')
        product = item.find("a",attrs={"class":"product-item-link"}).get_text().strip()
        price = item.find("span",attrs={"class":"price"}).get_text().strip()
        # print(brand)
        # print(url)
        # print(product)
        # print(price)
        data.append((page_no, brand, url, product, price))

df = pd.DataFrame(data,
   columns = header
)
df.to_csv('eshop_catalog.csv', sep='\t', encoding='utf-8')
df
```

Define item header.

eshop.cataloghk.com(https://eshop.cataloghk.com/footwear.html)

In [214]:

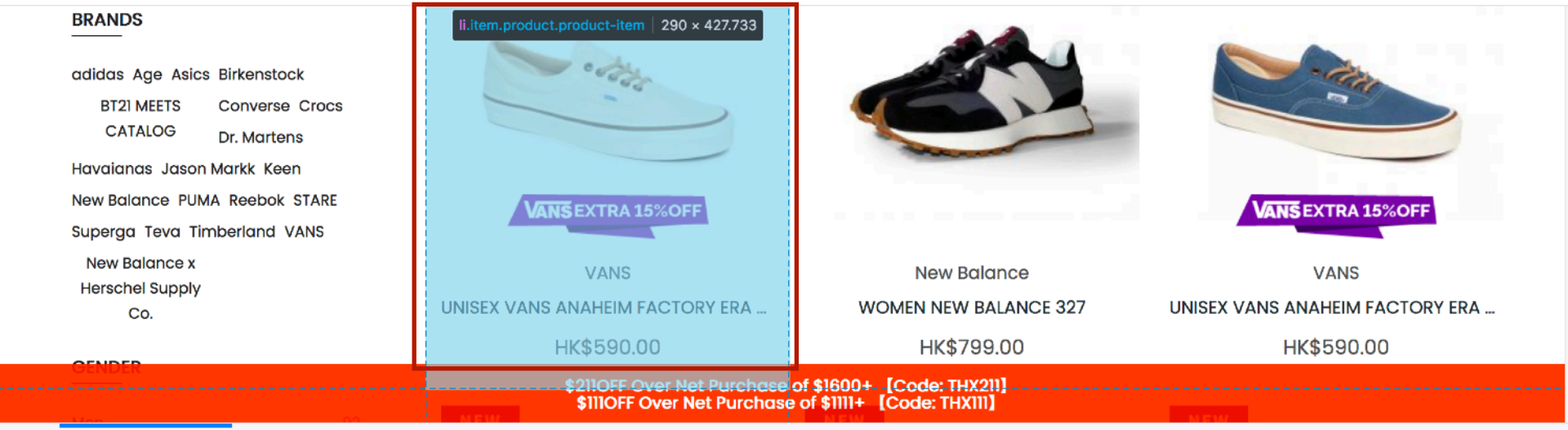
```
import requests
import csv
import pandas as pd
from bs4 import BeautifulSoup

header = ['page #','brand','url','product','price']
data = []
# Display and store away 2 pages of scrapped data from startupbeat.hkej.com
for i in range(1,4):
    quote_page = requests.get('https://eshop.cataloghk.com/footwear.html?p='+str(i))
    # print("\n***** Page " + str(i) +" in action *****")
    soup = BeautifulSoup(quote_page.content,'html.parser')
    for item in soup.find_all("li", attrs={"class":"item product product-item"}):
        page_no = str(i)
        brand = item.find("div",attrs={"class":"product details product-item-details box-info"}).span.get_text()
        url = item.find("div",attrs={"class":"product details product-item-details box-info"}).a.get('href')
        product = item.find("a",attrs={"class":"product-item-link"}).get_text().strip()
        price = item.find("span",attrs={"class":"price"}).get_text().strip()
        # print(brand)
        # print(url)
        # print(product)
        # print(price)
        data.append((page_no, brand, url, product, price))

df = pd.DataFrame(data,
   columns = header
)
df.to_csv('eshop_catalog.csv', sep='\t', encoding='utf-8')
df
```

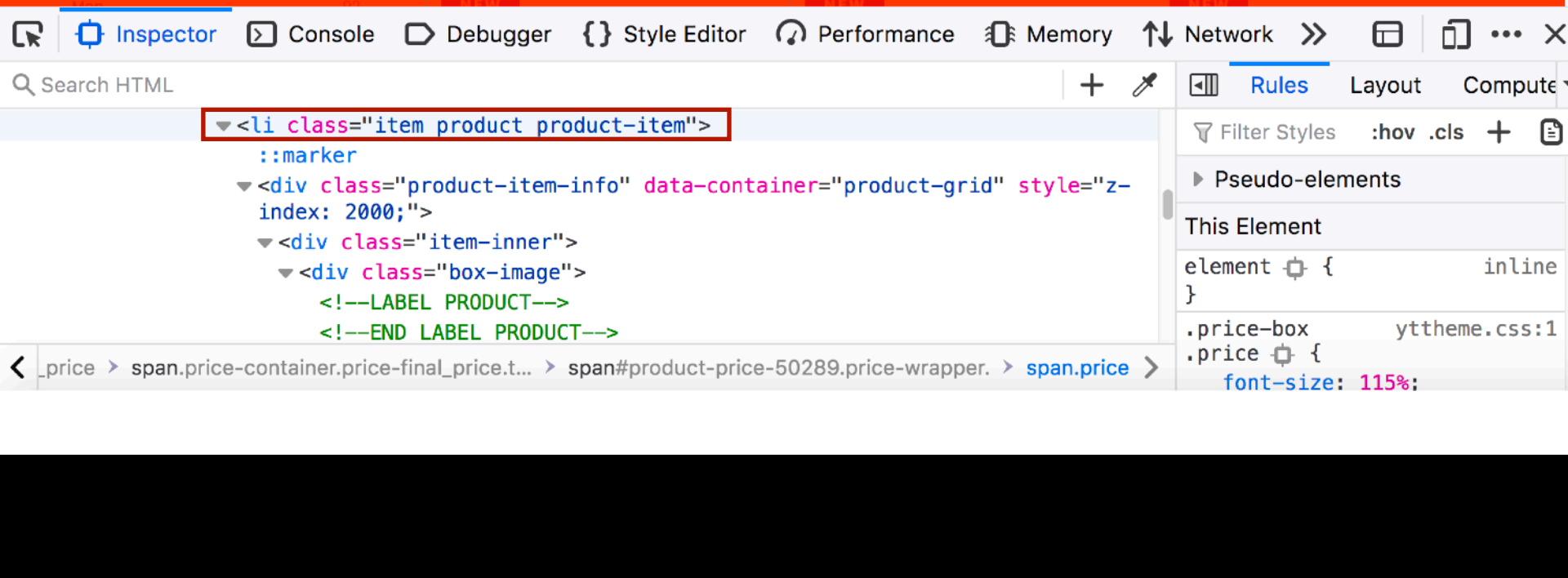
Loop through items to grab details.

eshop.cataloghk.com(https://eshop.cataloghk.com/footwear.html)

In [214]:

```
import requests
import csv
import pandas as pd
from bs4 import BeautifulSoup

header = ['page #','brand','url','product','price']
data = []
# Display and store away 2 pages of scrapped data from startupbeat.hkej.com
for i in range(1,4):
    quote_page = requests.get('https://eshop.cataloghk.com/footwear.html?p='+str(i))
    # print("\n***** Page " + str(i) +" in action *****")
    soup = BeautifulSoup(quote_page.content,'html.parser')
    for item in soup.find_all("li", attrs={"class":"item product product-item"}):
        page_no = str(i)
        brand = item.find("div",attrs={"class":"product details product-item-details box-info"}).span.get_text()
        url = item.find("div",attrs={"class":"product details product-item-details box-info"}).a.get('href')
        product = item.find("a",attrs={"class":"product-item-link"}).get_text().strip()
        price = item.find("span",attrs={"class":"price"}).get_text().strip()
        # print(brand)
        # print(url)
        # print(product)
        # print(price)
        data.append((page_no, brand, url, product, price))

df = pd.DataFrame(data,
   columns = header
)
df.to_csv('eshop_catalog.csv', sep='\t', encoding='utf-8')
df
```


BRANDS
adidas Age Asics Birkenstock
BT21 MEETS CATALOG
Converse Crocs
Dr. Martens
Havaianas Jason Markk Keen
New Balance PUMA Reebok STARE
Superga Teva Timberland VANS
New Balance x Herschel Supply Co.
GENDER
li.item.product.product-item | 290 × 427.733
VANS EXTRA 15%OFF
VANS
UNISEX VANS ANAHEIM FACTORY ERA ...
HK$590.00
New Balance
WOMEN NEW BALANCE 327
HK$799.00
VANS EXTRA 15%OFF
VANS
UNISEX VANS ANAHEIM FACTORY ERA ...
HK$590.00
$211OFF Over Net Purchase of $1600+ [Code: THX211]
$111OFF Over Net Purchase of $1111+ [Code: THX111]


Inspector
Console
Debugger
Style Editor
Performance
Memory
Network
Search HTML
<li class="item product product-item">
::marker

<!--LABEL PRODUCT-->
<!--END LABEL PRODUCT-->
Rules
Layout
Compute
Filter Styles
:hov .cls
Pseudo-elements
This Element
element { inline
}
.price-box ytheme.css:1
.price {
font-size: 115%;
_price
span.price-container.price-final_price.t...
span#product-price-50289.price-wrapper.
span.price

eshop.cataloghk.com(https://eshop.cataloghk.com/footwear.html)

In [214]:

```
import requests
import csv
import pandas as pd
from bs4 import BeautifulSoup

header = ['page #','brand','url','product','price']
data = []
# Display and store away 2 pages of scrapped data from startupbeat.hkej.com
for i in range(1,4):
    quote_page = requests.get('https://eshop.cataloghk.com/footwear.html?p='+str(i))
    # print("\n***** Page " + str(i) +" in action *****")
    soup = BeautifulSoup(quote_page.content,'html.parser')
    for item in soup.find_all("li", attrs={"class":"item product product-item"}):
        page_no = str(i)
        brand = item.find("div",attrs={"class":"product details product-item-details box-info"}).span.get_text()
        url = item.find("div",attrs={"class":"product details product-item-details box-info"}).a.get('href')
        product = item.find("a",attrs={"class":"product-item-link"}).get_text().strip()
        price = item.find("span",attrs={"class":"price"}).get_text().strip()
        # print(brand)
        # print(url)
        # print(product)
        # print(price)
        data.append((page_no, brand, url, product, price))

df = pd.DataFrame(data,
   columns = header
)
df.to_csv('eshop_catalog.csv', sep='\t', encoding='utf-8')
df
```

BT21 MEETS CATALOG
Converse Crocs
Havaianas
Jason Markk Keen New Balance
PUMA Reebok STARE Superga Teva
Timberland VANS
New Balance x Herschel Supply Co.



VANS
UNISEX VANS ANAHEIM FACTORY ERA ...
HK$590.00

New Balance
WOMEN NEW BALANCE 327
HK$799.00

VANS
UNISEX VANS ANAHEIM FACTORY ERA ...
HK$590.00

GENDER

Men 92

NEW NEW NEW

Women 281

$211OFF Over Net Purchase of $1600+ 【Code: THX211】
$111OFF Over Net Purchase of $1111+ 【Code: THX111】

Inspector | Console | Debugger | Style Editor | Performance | Memory | Network | Storage | Accessibility | Application

Search HTML

```
<div class="product details product-item-details box-info">
  <!--show the brand name of product-->
  <span>VANS</span>
<h2 class="product name product-item-name product-name">
  <a class="product-item-link" href="https://eshop.cataloghk.com/105vn0a2rr11ve.html">…</a>
</h2>
<div class="price-box price-final_price" data-role="priceBox" data-product-id="50289">
  <!--Updated by Tony Wong on 20180515 to add discount off label START-->
  <style>…</style>
  <!--Updated by Tony Wong on 20180515 to add discount off label END-->
  <span class="price-container price-final_price tax weee">
    <span id="product-price-50289" class="price-wrapper " data-price-amount="590" data-price-type="finalPrice">
      <span class="price">HK$590.00</span>
    </span>
```

y-product.products.wrapper.gr... › ol.products.list.items.product-items › li.item.product.product-item › div.product-item-info › div.item-inner › div.product.details.product-item-details...

Rules | Layout | Compute

Filter Styles :hov .cls +

Pseudo-elements

This Element

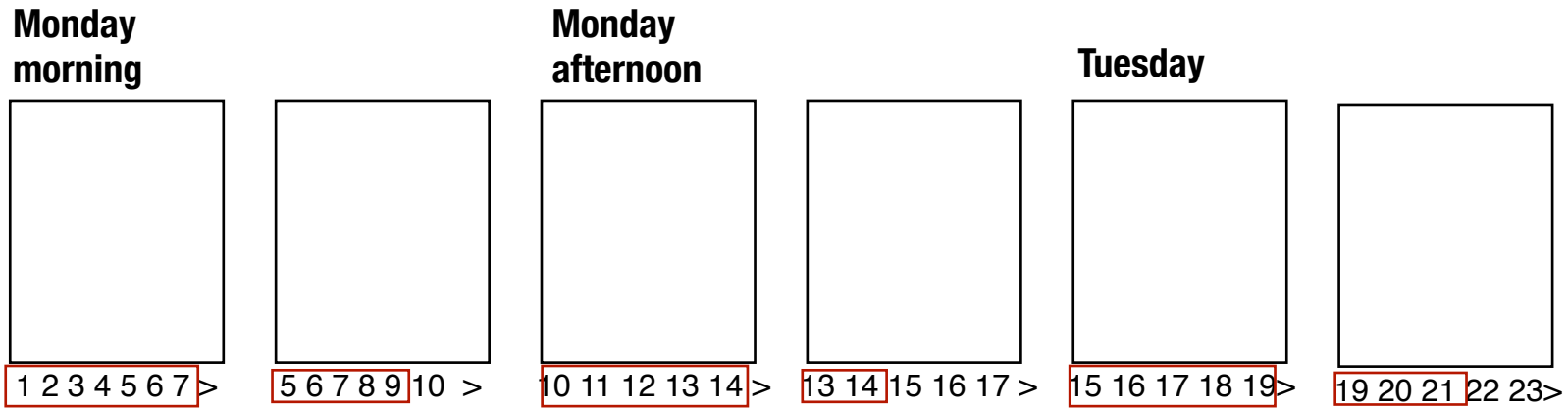
```
element { inline
}
.products-grid .item .item-inner .box-info { yttheme.css:1
  padding: 15px 0 5px;
  text-align: center;
}
* { yttheme.css:1
  -webkit-box-sizing:
  border-box;
```

	page #	brand	url	product	price
0	1	VANS	https://eshop.cataloghk.com/105vn0a2rr11ve.html	UNISEX VANS ANAHEIM FACTORY ERA 95 DX	HK$590.00
1	1	New Balance	https://eshop.cataloghk.com/112ws327kc.html	WOMEN NEW BALANCE 327	HK$799.00
2	1	VANS	https://eshop.cataloghk.com/105vn0a2rr11v2.html	UNISEX VANS ANAHEIM FACTORY ERA 95 DX	HK$590.00
3	1	Reebok	https://eshop.cataloghk.com/101fv2392.html	BIG KIDS REEBOK ROYAL PRIME 2.0 2V	HK$299.00
4	1	Converse	https://eshop.cataloghk.com/150569433c.html	WOMEN CONVERSE CHUCK 70 HIGH TOP	HK$639.00
...	...	...	...	...	...
175	3	New Balance	https://eshop.cataloghk.com/112ws327coc.html	WOMEN NEW BALANCE 327	HK$799.00
176	3	New Balance	https://eshop.cataloghk.com/112ms327sfa.html	UNISEX NEW BALANCE 327	HK$799.00
177	3	Reebok	https://eshop.cataloghk.com/101fv2395.html	LITTLE KIDS REEBOK ROYAL PRIME 2.0 2V	HK$259.00
178	3	Teva	https://eshop.cataloghk.com/1661116376brnbw.html	WOMEN TEVA FLATFORM UNIVERSAL STRIPE	HK$799.00
179	3	Teva	https://eshop.cataloghk.com/1661004006rwbc.html	MEN TEVA ORIGINAL UNIVERSAL	HK$499.00

180 rows × 5 columns

	page #	brand	url	product	price
0	1	VANS	https://eshop.cataloghk.com/105vn0a2rr11ve.html	UNISEX VANS ANAHEIM FACTORY ERA 95 DX	HK$590.00
1	1	New Balance	https://eshop.cataloghk.com/112ws327kc.html	WOMEN NEW BALANCE 327	HK$799.00
2	1	VANS	https://eshop.cataloghk.com/105vn0a2rr11v2.html	UNISEX VANS ANAHEIM FACTORY ERA 95 DX	HK$590.00
3	1	Reebok	https://eshop.cataloghk.com/101fv2392.html	BIG KIDS REEBOK ROYAL PRIME 2.0 2V	HK$299.00
4	1	Converse	https://eshop.cataloghk.com/150569433c.html	WOMEN CONVERSE CHUCK 70 HIGH TOP	HK$639.00
5	1	adidas	https://eshop.cataloghk.com/108eg3289.html	WOMEN ADIDAS ORIGINALS LITE RACER 2.0	HK$499.00
6	1	adidas	https://eshop.cataloghk.com/108g28109.html	BIG KIDS ADIDAS ORIGINALS U_PATH RUN	HK$669.00
7	1	adidas	https://eshop.cataloghk.com/108fw1722.html	MEN ADIDAS ORIGINALS LITE RACER 2.0	HK$499.00
8	1	adidas	https://eshop.cataloghk.com/108fy2418.html	UNISEX ADIDAS ORIGINALS U_PATH RUN	HK$699.00
9	1	Converse	https://eshop.cataloghk.com/150167331c.html	UNISEX CONVERSE JACK PURCELL LOW TOP	HK$151.00
10	1	PUMA	https://eshop.cataloghk.com/1pu37114903.html	UNISEX PUMA RIDER PLAY ON	HK$201.00
11	1	Reebok	https://eshop.cataloghk.com/101fw6024.html	UNISEX REEBOK FURYLITE 3.0	HK$201.00
12	1	Converse	https://eshop.cataloghk.com/150569539c.html	WOMEN CONVERSE BLACK ICE CHUCK 70 HI TOP	HK$639.00
13	1	Converse	https://eshop.cataloghk.com/150569540c.html	WOMEN CONVERSE BLACK ICE CHUCK 70 HI TOP	HK$639.00
14	1	Reebok	https://eshop.cataloghk.com/101q47272.html	UNISEX REEBOK AZTEC 79	HK$699.00
15	1	Reebok	https://eshop.cataloghk.com/101fv1577.html	UNISEX REEBOK INSTAPUMP FURY OG NM	HK$1,299.00
16	1	Reebok	https://eshop.cataloghk.com/101fw7699.html	UNISEX REEBOK INSTAPUMP FURY OG NM	HK$1,299.00
17	1	Reebok	https://eshop.cataloghk.com/101fy9093.html	LITTLE KIDS REEBOK MINIONS VERSA PUMP FURY	HK$399.00
18	1	Reebok	https://eshop.cataloghk.com/101fy3404.html	UNISEX REEBOK MINIONS INSTAPUMP FURY MU	HK$1,299.00
19	1	Reebok	https://eshop.cataloghk.com/101fy3405.html	LITTLE KIDS REEBOK VERSA PUMP FURY	HK$399.00
20	1	adidas	https://eshop.cataloghk.com/108fz1962.html	UNISEX ADIDAS ORIGINALS OZWEEGO	HK$1,099.00

Periodic scraping in smaller batches.


Monday
morning
Monday
afternoon
Tuesday
1 2 3 4 5 6 7 >
5 6 7 8 9 10 >
10 11 12 13 14 >
13 14 15 16 17 >
15 16 17 18 19 >
19 20 21 22 23 >

Exercise time.

登入 註冊 | 訂閱 / 續訂

信報財經新聞

熱門：港區國安法 新冠肺炎　搜尋：搶奪中環人 ESG報道系列

搜尋

齊心抗疫，優惠再延長！

信健康 StartUpBeat EJFQ 港股360 信報月刊 優雅生活 ejinsight 信報通識 信報出版 好·食·玩

主頁 即時新聞 今日信報 港股360 地產投資 財富管理 專題 信報視頻

最新 新聞專題 人物訪問 生活休閒 信報教育 活動專輯 全部

瘟疫大流行 >全球疫情發展

南韓增36宗新冠肺炎確診 逾半屬境外輸入

南韓中央防疫對策本部稱，當地新增36宗感染新型冠狀病毒肺炎的確診病例，累計近1.43萬宗。另外，當地新增1例死 ...

2020年7月31日

全球確診人數超過1745萬 美國達463萬

據Worldometer統計，截至香港時間周五早上8時，全球確診人數突破1745萬人，增加27.7

瘟疫大流行最新發展

Test Ru...

BROWSE

main_template

Select page

Select article (19)

Extract title

Extract url

Relative story

Relative post_date

Select next

Click each next item

and go to main_template

Selection Node: Edit

2nd a
> All element s

Wait up to 60 seconds for elements to appear.

API Tutorials Contact

全球疫情發展 - 信報網站 h | 全球疫情發展 - 信報網站 h | Test Run | 全球疫情發展 - 信報網站 h

https://www1.hkej.com/features/topic/tag/瘟疫大流行最新發展

2020年7月30日

Select Mode

更多內容

全球確診人數突破1700萬
美國逾456萬

據Worldometer統計，截至香港時間周四早上

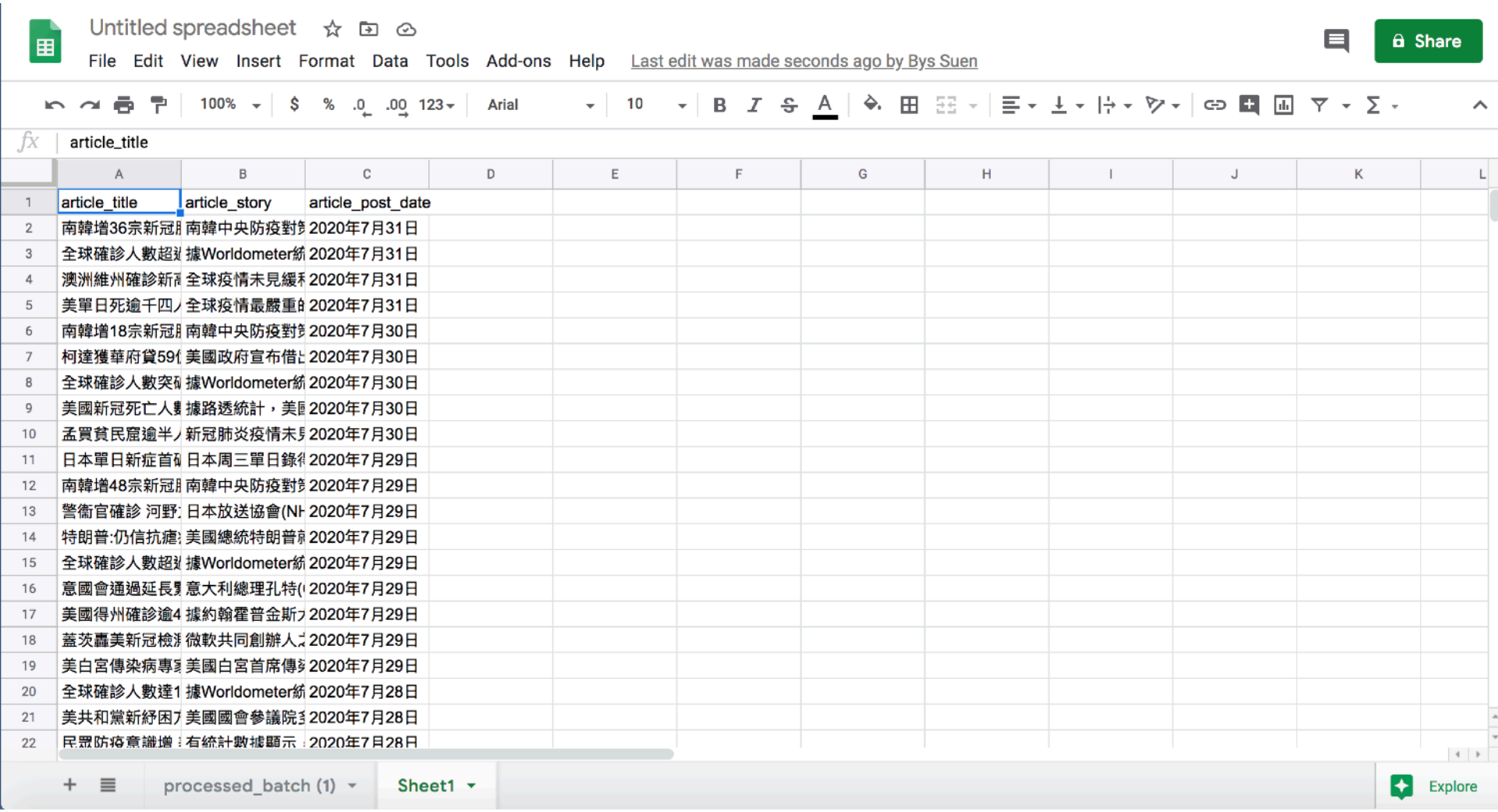
CSV/Excel JSON

article_title	article_url	article_story	article_story_url	article_post_date
南韓增36宗新冠肺炎確診 逾半屬境外輸入	https://www1.hkej.com /features/article?q= %23%E7%98%9F %E7%96%AB%E5 %A4%A7%E6%B5 %81%E8%A1%8C %E6%9C%80%E6 %96%B0%E7%99 %BC%E5%B1%95...	南韓中央防疫對策本部稱，當地新增36宗感染新型冠狀病毒肺炎的確診病例，累計近1.43萬宗。另外，當地新增1例死 ...	https://www1.hkej.com /features/article?q= %23%E7%98%9F %E7%96%AB%E5 %A4%A7%E6%B5 %81%E8%A1%8C %E6%9C%80%E6 %96%B0%E7%99 %BC%E5%B1%95...	2020年7月31日
全球確診人數超過1745萬 美國達463萬	https://www1.hkej.com /features/article?q= %23%E7%98%9F %E7%96%AB%E5	據Worldometer統計，截至香港時間周五早上8時，全球確診人數突破1745萬人，增加27.7萬人，死亡	https://www1.hkej.com /features/article?q= %23%E7%98%9F %E7%96%AB%E5	2020年7月31日

This a live preview. When you are ready to run your project, click Get Data.

Show more data Visuals enabled (advanced)

```
import requests
import csv
import pandas as pd
from bs4 import BeautifulSoup

header = ['page #','title','url','details','post date']
data = []
# Display and store away 2 pages of scrapped data from startupbeat.hkej.com
for i in range(1,4):
    quote_page = requests.get('https://www1.hkej.com/features/topic/tag/%E7%98%9F%E7%96%AB%E5%A4%A7%E6%B5%81%E8%A1%
    print("\n***** Page " + str(i) +" in action *****")
    soup = BeautifulSoup(quote_page.content,'html.parser')

    for article in soup.find_all("li", attrs={"class":"fea_list_n"}):
         page_no = str(i)
         title = article.find("div",attrs={"class":"fea_list_n_topic"}).get_text()
         url = article.a.get('href')
         details = article.find("div",attrs={"class":"fea_list_n_sub fea_list_n_recap"}).a.get_text()
         post_date = article.find("div",attrs={"class":"fea_list_n_sub"}).get_text()
         post_date = post_date[-13:].strip()
         print(title)
         print(url)
         print(details)
         print(post_date)
         data.append((page_no, title, url, details, post_date))

df = pd.DataFrame(data,
  columns = header
)
df.to_csv('startup_beat_data_2.csv', sep='\t', encoding='utf-8')
df
```


Untitled spreadsheet
File Edit View Insert Format Data Tools Add-ons Help
Last edit was made seconds ago by Bys Suen
Share
article_title
article_title article_story article_post_date
processed_batch (1)
Sheet1
Explore

Legal and Ethical Considerations.

scraping and how they play out.

Is Web Scraping Legal?

Your first thought might be to look at the legal side of things.

The truth is that the legality of web scraping is still relatively up in the air.

Meaning that there are currently no specific laws that refer to the legality of web scraping. So it is neither legal or illegal.

https://www.parsehub.com/blog/web-scraping-ethical/

Scraping Publicly Available Information

Another factor to keep in mind is the type of data you'd be scraping. In our case, we always refer to publicly available data.

This is data that has been made public by the owner of said data. Private and leaked information is not considered publicly available information.

Ethics in Web Scraping

James Densmore Jul 23, 2017 · 3 min read

We all scrape web data. Well, those of us who work with data do. Data scientists, marketers, data journalists, and the data curious alike. Lately, I've been thinking more about the ethics of the practice and have been dissatisfied by the lack of consensus on the topic.

Let me be clear that I'm talking **ethics** not the law. The law in regards to scraping web data is complex, fuzzy and ripe for reform, but that's another matter. It's not that no one is thinking, or writing, about the ethics in scraping but rather that both those scraping and those being scraped can't

https://towardsdatascience.com/ethics-in-web-scraping-b96b18136f01

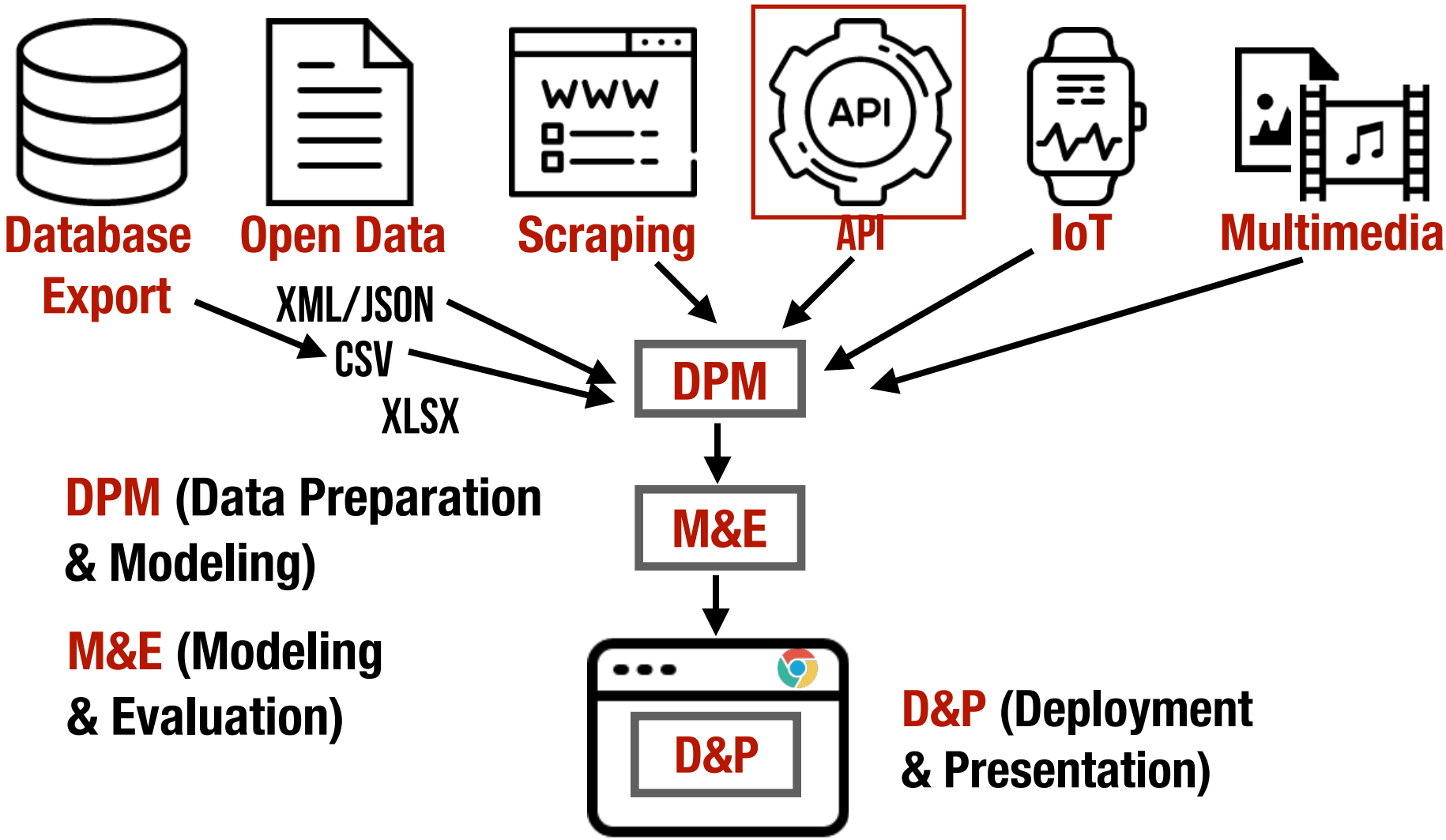
WRITTEN BY
James Densmore
Data Science and Data Engineering Consultant at Data Liftoff
https://www.dataliftoff.com

Follow

The Ethical Scraper

I, the web scraper will live by the following principles:

- If you have a public API that provides the data I'm looking for, I'll use it and avoid scraping all together.
- I will always provide a User Agent string that makes my intentions clear and provides a way for you to contact me with questions or concerns.
- I will request data at a reasonable rate. I will strive to never be confused for a DDoS attack.
- I will only save the data I absolutely need from your page. If all I need it OpenGraph meta-data, that's all I'll keep.
- I will respect any content I do keep. I'll never pass it off as my own.
- I will look for ways to return value to you. Maybe I can drive some (real) traffic to your site or credit you in an article or post.
- I will respond in a timely fashion to your outreach and work with you towards a resolution.
- I will scrape for the **purpose of creating new value from the data**, not to duplicate it.


Database Export
Open Data
Scraping
API
IoT
Multimedia
XML/JSON
CSV
XLSX
DPM
M&E
D&P
DPM (Data Preparation & Modeling)
M&E (Modeling & Evaluation)
D&P (Deployment & Presentation)

The Airbnb Story

API partner

Request access

Connect to our API.
Connect to millions of travelers on Airbnb.

Request access

Reach more guests

Reach millions of new guests by connecting with our global community.

Integrate and build

Set up your integration to give clients full control over how they want to list on Airbnb.

Access support

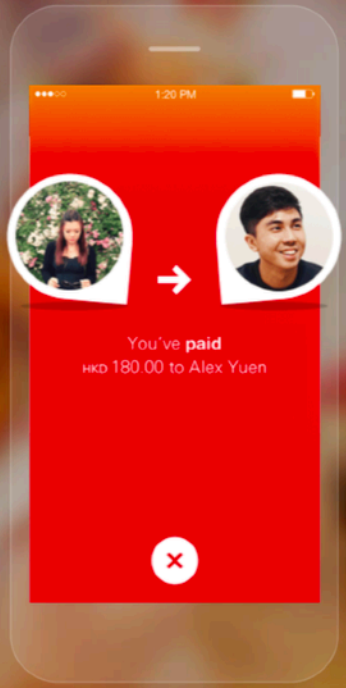
Explore in-depth technical documentation and get partner manager support.

AAABNB

PayMe | from HSBC

Personal | Business | What's new | Need help? | Download PayMe | English

Pay anyone with any bank



Pay your friends with just a few taps, regardless of which Hong Kong local bank they use and without ever asking for an account number.

How it works

Discover more

Google Cloud Platform

Prediction API

Natural Language API

Translation API

Vision API

PayMe Marketplace API
PayMe • paymeapi
Documentation Inspector

Download
API Blueprint

INTRODUCTION

API Capabilities and Structure

Getting Started and Connection Details

Service URLs

Credit Card Numbers for Testing Purposes

PayMe Marketplace API

INTRODUCTION

Our API is designed to allow platforms to offer a full payment solution as part of their product.

Using the API you can:

微信支付 WeChat Pay

Home

Documents

WeChat payment API development documents, Ch

Quick Pay

Buyers can present the pay code, vendor will scan the code to finish the transaction.

Alibaba Open Platform

Home Resources

Platform

Tmall.hk

API for Alibaba

Sourcing API： Click here

WholeSale API:Click here

Harvard Business Review

GROWTH STRATEGY

The Strategic Value of APIs

by Bala Iyer and Mohan Subramaniam

JANUARY 07, 2015

McKinsey&Company
Digital McKinsey

How We Help Clients | Our Insights | Careers | Our People | Contact Us

API經濟來了：從開放創新到Open API

by 《數位時代》整合行銷部 2014.11.04

McKinsey&Company
Digital McKinsey

How We Help Clients | Our Insights | Careers | Our People | Contact Us

Deloitte. Insights

Realizing the economic value of APIs

Download the report →

Get insights via email →

Organization and governance

Successful API development requires cross-functional decision-making and oversight. Traditionally, IT departments have developed APIs with little or no input from other functions within the organization. Proactive API-centric organizations generally engage individuals from the relevant lines of business in the API development cycle to provide needed guidance around the right level of desired functionality and customer experience.

These organizations also involve their procurement and legal departments in monitoring intellectual property, developing SLAs and contracting. Marketing professionals play an important role in addressing the branding and promotion of APIs, and focusing on the attraction and recruitment of partners and external developers. Risk managers evaluate potential impacts of security breaches as well as unintended use of the APIs by legitimate parties.

Skills and capabilities

The API economy has implications for both an organization's skills and cultural mindset. An API-centric business requires new capabilities, including product management, data science and intellectual property management, as well as a culture that stresses innovation and resource reuse. For example, the product management skills needed to oversee the entire lifecycle of API management are often different from the technical capabilities needed to architect and code the actual APIs themselves.

One way organizations have set out to address these issues is the use of innovation teams centered on creating an API-centric transformation. These businesses create physical and virtual spaces that bring together architects, developers, product managers and business leaders to rapidly exchange ideas and interact with one another. They provide education on API development, develop use-case opportunities, and leverage internal and external collaborative events to promote API value and usage.

APIs drive digital transformation[4]

International Airlines Group (IAG), one of the world's largest airline groups, needed to transform itself into an agile, digital organization. The company was facing competitive investments in innovation and digital technologies, as well as fear of disruption from a potential "uberization" of the airline industry. IAG set up a digital business team to oversee a digital transformation and connectivity program, with APIs as the underlying driver.

This resulted in closer collaboration between the business and IT functions, connecting operations from the front- to the back-office via APIs and reducing much of the complexity caused by disjointed silos. The airline began a "Connectivity Program" to extend the digital mindset across the company, with a focus on timesaving techniques and tools to help airline crews do their jobs.

Please visit dc.cuhkcfe.io
for more information.

Thank you for your time!