Data Acquisition & Wrangling

Susan B. Davidson and Zachary G. Ives

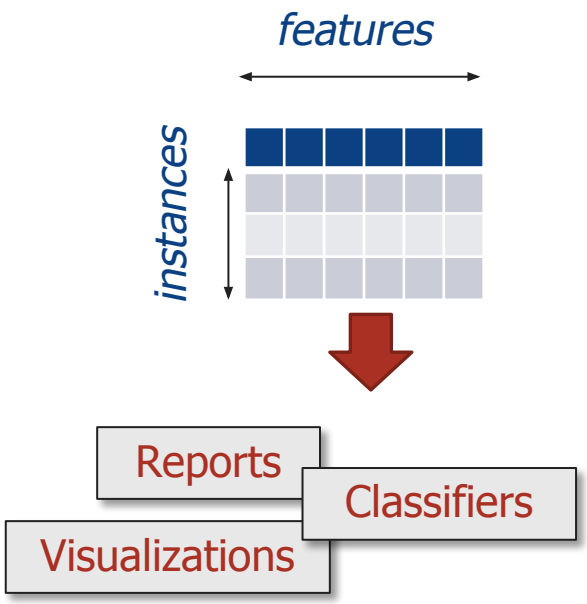
University of Pennsylvania

CIS 545 – Big Data Analytics

Portions of this lecture have been contributed to the OpenDS4All project, piloted by Penn, IBM, and the Linux Foundation

https://tinyurl.com/cis545-lecture-01-19-22

What We Need for Data Analytics: Well-Structured Data



The data "instances" are *observations* or *samples* from a population

Each has many *features* – measurable properties or characteristics

Requires a *transformation* from (the right) "raw data" to a single tabular representation!

https://tinyurl.com/cis545-lecture-01-19-22

Three Basic Encodings of Structured Data

Tables

Arrays

Nested

https://tinyurl.com/cis545-lecture-01-19-22

Three Basic Encodings of Structured Data

Tables

id	last	first	hair	height	handed	label
1	Li	C	Brn	5'6"	Left	T
2	Prakash	A	Blk	5'8"	Right	T
3	Jones	L	Blk	6'3"	Ambi	F

Comprised of **tuples** that may have different **field** types

Also called **dataframes** (Pandas, R, PySpark) and **relations** (SQL)

https://tinyurl.com/cis545-lecture-01-19-22

Three Basic Encodings of Structured Data

Tables

id	last	first	hair	height	handed	label
1	Li	C	Brn	5'6"	Left	T
2	Prakash	A	Blk	5'8"	Right	T
3	Jones	L	Blk	6'3"	Ambi	F

Comprised of **tuples** that may have different **field** types

Also called **dataframes** (Pandas, R, PySpark) and **relations** (SQL)

Arrays

1	3	2
3	1	0
2	0	1

Cells have same type, in a multi-dimensional coordinate space

Matrices, tensors

https://tinyurl.com/cis545-lecture-01-19-22

Three Basic Encodings of Structured Data

Tables

id	last	first	hair	height	handed	label
1	Li	C	Brn	5'6"	Left	T
2	Prakash	A	Blk	5'8"	Right	T
3	Jones	L	Blk	6'3"	Ambi	F

Comprised of **tuples** that may have different **field** types

Also called **dataframes** (Pandas, R, PySpark) and **relations** (SQL)

Arrays

1	3	2
3	1	0
2	0	1

Cells have same type, in a multi-dimensional coordinate space

Matrices, tensors

Nested


```
{
  "type":"FeatureCollection",
  "features":[
    {
      "type":"Feature",
      "properties":{
        "OBJECTID":1,
        "INSPECTION_UNIT":"Code Enforcement",
        "DISTRICT":"EAST",
        "STREET_ADDRESS":"7522 Castor Ave",
        "UNIT":null,
        "ZIP_CODE":"19152",
        "PHONE_NUMBER":"215-685-0535",
        "EMAIL":"CodeEnforcement.East@phila.gov"
      },
      "geometry":{
        "type":"Point",
        "coordinates":[
          -75.06289028615926,
          40.05512873142789
        ]
      }
    },
    {
```


opendataphilly.org

Compositions of **dictionaries (maps)** and **lists**
also: JSON, XML

https://tinyurl.com/cis545-lecture-01-19-22

Outline for Today

- The process of structuring our data
- A central example
- Techniques for acquiring and structuring data

https://tinyurl.com/cis545-lecture-01-19-22

https://canvas.upenn.edu/courses/1636888/quizzes/2771598

Quiz 02A

- Which of the following do we consider to be the three basic encodings of structured data?
  - graphs, heaps, strings
  - tables, arrays, nested
  - tables, nested, dataframes
  - streams, static, multimedia
- Which of the following are valid types of nested data?
  - dictionaries nested in lists
  - lists nested in dictionaries
  - all of the above

https://tinyurl.com/cis545-lecture-01-19-22

Structuring Data

Susan B. Davidson and Zachary G. Ives

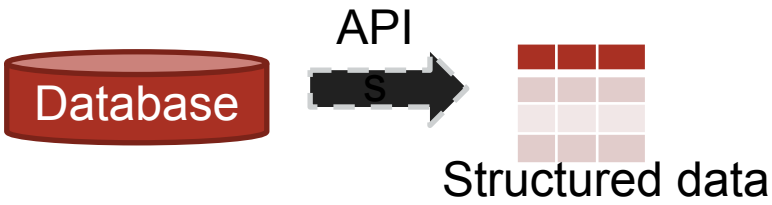
University of Pennsylvania

CIS 545 – Big Data Analytics

Portions of this lecture have been contributed to the OpenDS4All project, piloted by Penn, IBM, and the Linux Foundation

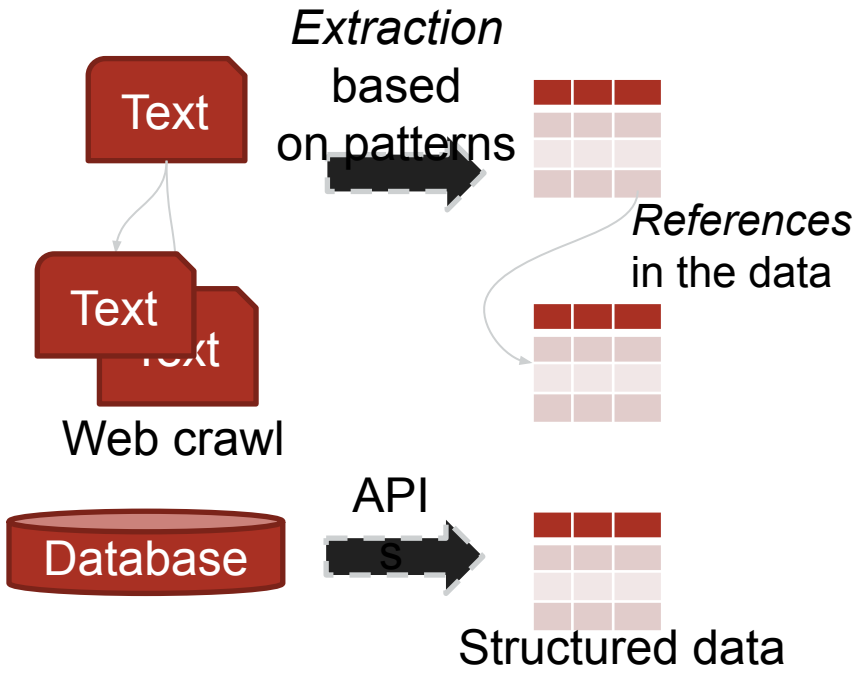
https://tinyurl.com/cis545-lecture-01-19-22

Structuring Data: Acquisition, Wrangling, Integration, Cleaning



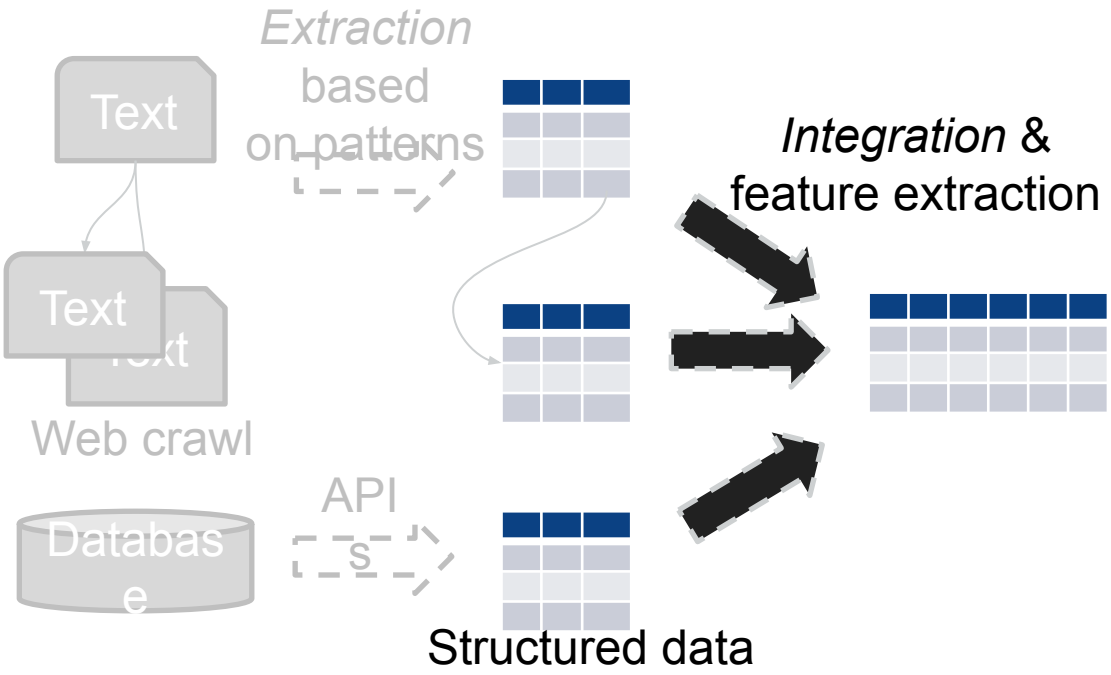
https://tinyurl.com/cis545-lecture-01-19-22

Structuring Data: Acquisition, Wrangling, Integration, Cleaning



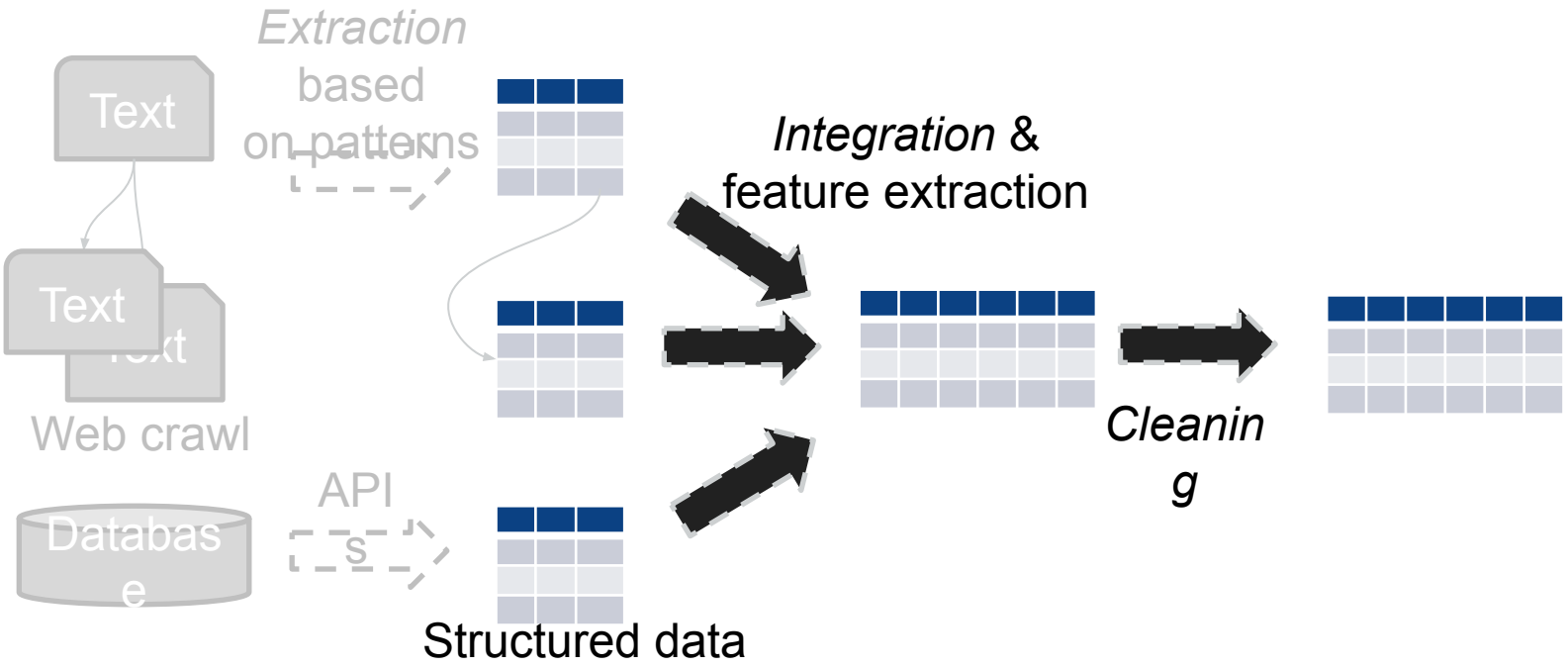
https://tinyurl.com/cis545-lecture-01-19-22

Structuring Data: Acquisition, Wrangling, Integration, Cleaning



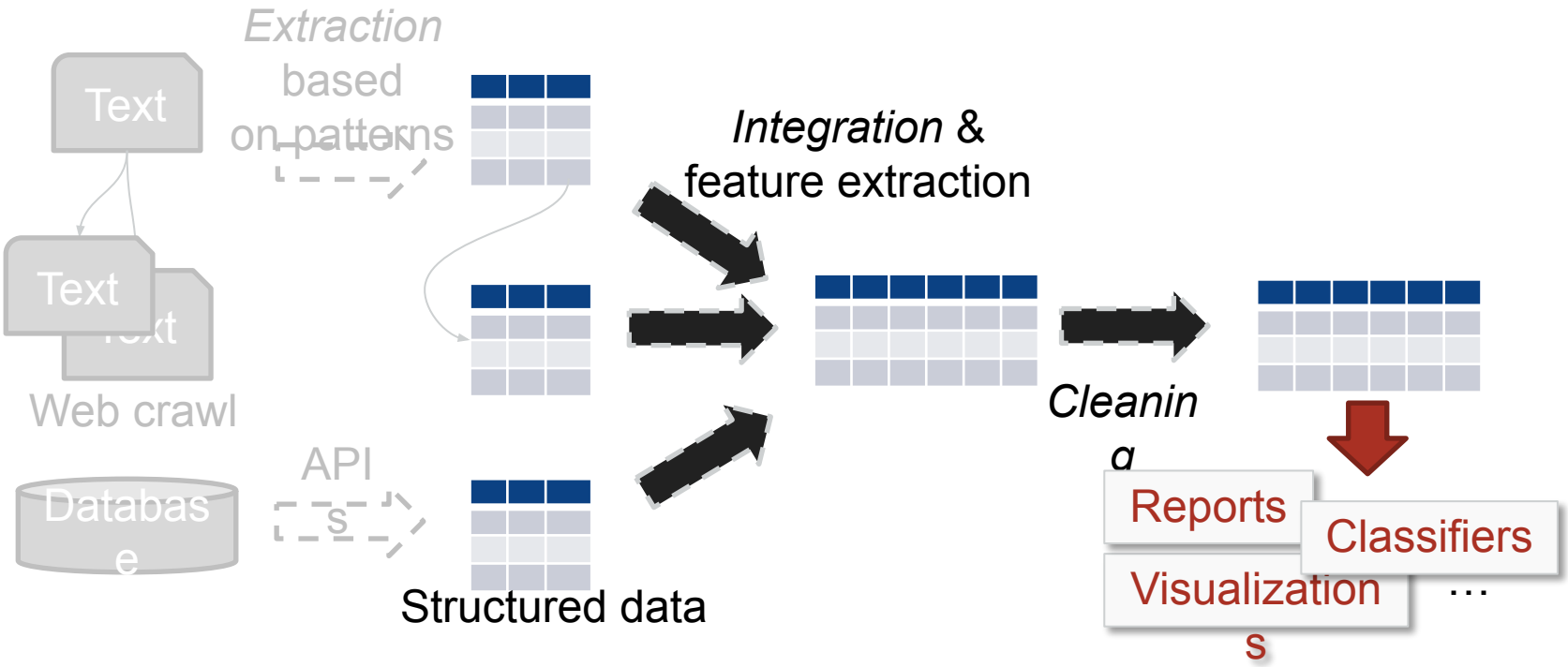
https://tinyurl.com/cis545-lecture-01-19-22

Structuring Data: Acquisition, Wrangling, Integration, Cleaning



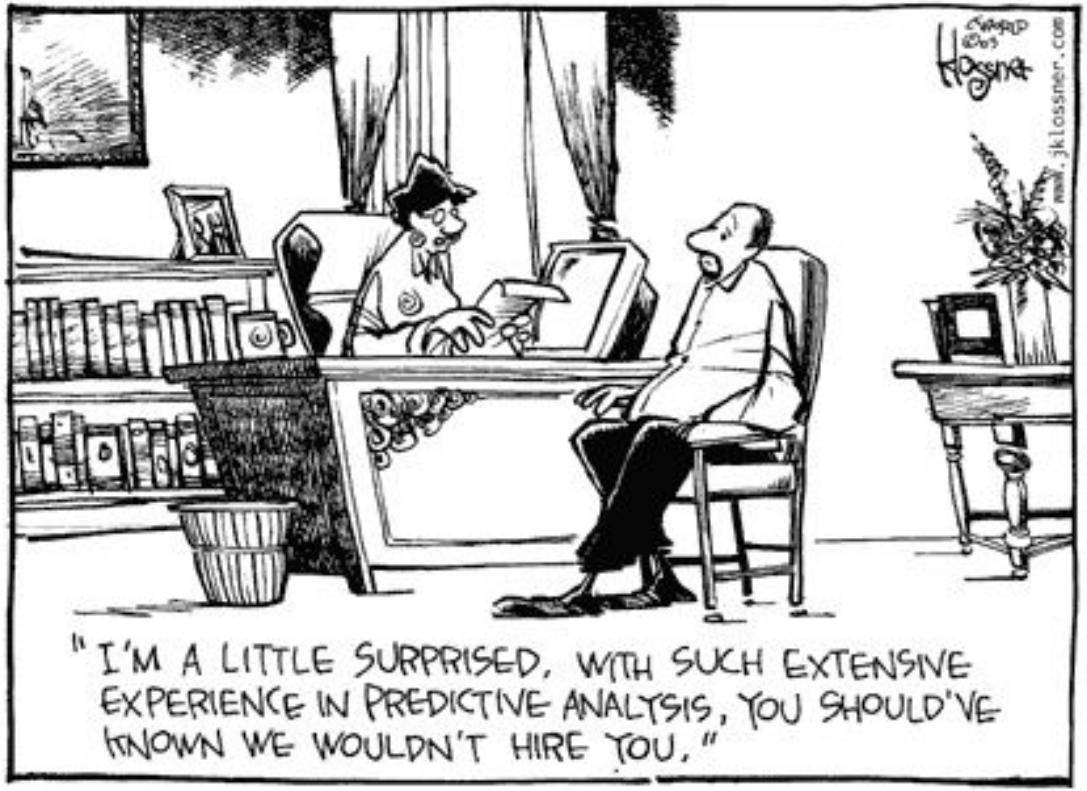
https://tinyurl.com/cis545-lecture-01-19-22

Structuring Data: Acquisition, Wrangling, Integration, Cleaning



https://tinyurl.com/cis545-lecture-01-19-22

https://www.kdnuggets.com/websites/cartoons.html



https://tinyurl.com/cis545-lecture-01-19-22

Looking in More Detail…

1. Processing data from structured sources
2. *Extracting* data from the web

Setting the stage for the next modules, which will be on *linking and cleaning*

https://tinyurl.com/cis545-lecture-01-19-22

https://canvas.upenn.edu/courses/1636888/quizzes/2771548

Quiz 02B

- What kinds of data might need pattern-based *extraction*?
  - web or text data
  - database data
  - Python dataframes

- Which of the following is *not* a part of the standard *data structuring* process?
  - training
  - acquisition
  - wrangling
  - integration

https://tinyurl.com/cis545-lecture-01-19-22

Data Wrangling, by Example

Susan B. Davidson and Zachary G. Ives

University of Pennsylvania

CIS 545 – Big Data Analytics

Portions of this lecture have been contributed to the OpenDS4All project, piloted by Penn, IBM, and the Linux Foundation

https://tinyurl.com/cis545-lecture-01-19-22

Virtually All Slides in this Course Will Be Backed by Real Code

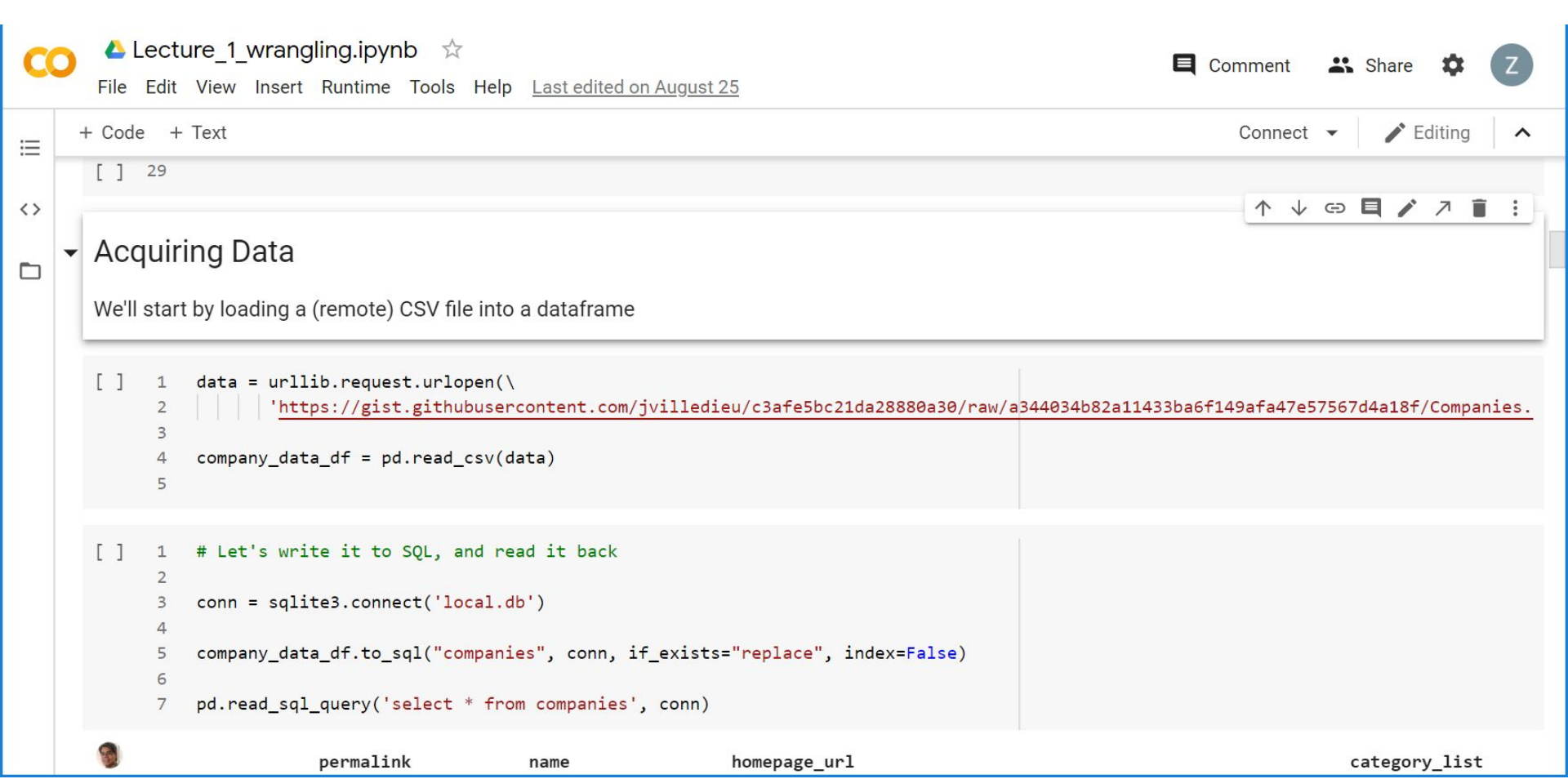
Hands-on examples wherever possible!

After lecture: experiment with the notebooks!

Notebooks on **Google Colab**, cloud-hosted Jupyter!

For this module: **https://tinyurl.com/cis545-001**

https://tinyurl.com/cis545-lecture-01-19-22



https://tinyurl.com/cis545-lecture-01-19-22

Hypothesis: CEOs of major companies are typically in their 40s+

Our Running Example for this Segment

Identify top companies, CEOs, and their *distribution of ages*

Challenges:

- Where do we find authoritative data?
- How do we load it in?
- How do we link it together?

Start with three simple data sources:

1. A file with a big list of companies and categories – in comma-separated values
2. A table of companies and CEOs in a Wikipedia page
3. Content from HTML pages on people (from Wikipedia)

https://tinyurl.com/cis545-lecture-01-19-22

Structured Data Sources

Susan B. Davidson and Zachary G. Ives

University of Pennsylvania

CIS 545 – Big Data Analytics

Portions of this lecture have been contributed to the OpenDS4All project, piloted by Penn, IBM, and the Linux Foundation

https://tinyurl.com/cis545-lecture-01-19-22

Let's Start by Loading Tabular Structured Data

We'll use Pandas dataframes as our tables (aka a relations)

We'll look at how to do this from:

- Structured files (aka "CSV" files)
- HTML files on the web with tables
- Database management systems (DBMSs)

Then we'll look at staging the data in the DBMS

https://tinyurl.com/cis545-lecture-01-19-22

Acquiring Data (1/3): Reading structured files [Company categories]

Companies and categories from a comma-separated value file, https://gist.githubusercontent.com/jvilledieu/c3afe5bc21da28880a30/raw/a344034b82a11433ba6f149afa47e57567d4a18f/Companies.csv

```
permalink,name,homepage_url,category_list,market,funding_total_usd,status,country_code,state_code,region,city,funding_rounds,founded_at,founded_month,founded_quarter,
founded_year,first_funding_at,last_funding_at
/organization/waywire,#waywire,http://www.waywire.com,|Entertainment|Politics|Social Media|News|, News , 1 750 000 ,acquired,USA,NY,New York City,New
York,1,01/06/2012,2012-06,2012-Q2,2012,30/06/2012,30/06/2012
/organization/tv-communications,&TV Communications,http://enjoyandtv.com,|Games|,Games, 4 000 000 ,operating,USA,CA,Los Angeles,Los
Angeles,2,,,,,04/06/2010,23/09/2010
/organization/rock-your-paper,'Rock' Your Paper,http://www.rockyourpaper.org,|Publishing|Education|,Publishing, 40 000
,operating,EST,,Tallinn,Tallinn,1,26/10/2012,2012-10,2012-Q4,2012,09/08/2012,09/08/2012
/organization/in-touch-network,(In)Touch Network,http://www.InTouchNetwork.com,|Electronics|Guides|Coffee|Restaurants|Music|iPhone|Apps|Mobile|iOS|E-
Commerce|,Electronics, 1 500 000 ,operating,GBR,,London,London,1,01/04/2011,2011-04,2011-Q2,2011,01/04/2011,01/04/2011
/organization/n-plusn,+n (PlusN),http://plusn.com,|Software|, Software , 1 200 000 ,operating,USA,NY,New York City,New York,2,01/01/2012,2012-01,2012-
Q1,2012,29/08/2012,04/09/2014
/organization/r-ranch-and-mine,-R- Ranch and Mine,,Entertainment|Games|,Games, 10 000 ,operating,USA,TX,Dallas,Fort Worth,1,08/07/2014,2014-07,2014-
Q3,2014,17/08/2014,17/08/2014
/organization/club-domains,.Club Domains,http://nic.club/,|Software|, Software , 7 000 000 ,,USA,FL,Ft. Lauderdale,Oakland Park,1,10/10/2011,2011-10,2011-
Q4,2011,31/05/2013,31/05/2013
/organization/fox-networks,.Fox Networks,http://www.dotfox.com,|Advertising|,Advertising, 4 912 393 ,closed,ARG,,Buenos Aires,Buenos Aires,1,,,,,16/01/2007,16/01/2007
/organization/0-6-com,0-6.com,http://www.0-6.com,|Curated Web|,Curated Web, 2 000 000 ,operating,,,,,1,01/01/2007,2007-01,2007-Q1,2007,19/03/2008,19/03/2008
/organization/004-technologies,004 Technologies,http://004gmbh.de/en/004-interact,|Software|, Software , -   ,operating,USA,IL,"Springfield,
Illinois",Champaign,1,01/01/2010,2010-01,2010-Q1,2010,24/07/2014,24/07/2014
/organization/01games-technology,01Games Technology,http://www.01games.hk/,|Games|,Games, 41 250 ,operating,HKG,,Hong Kong,,1,,,,,01/07/2014,01/07/2014
```

https://tinyurl.com/cis545-lecture-01-19-22

Acquiring Data (1/3): Reading structured files [Company categories]

Fetch using **urllib**, read using Pandas' **read_csv, (or sometimes read_json)**:

```
import urllib
import pandas as pd

data = urllib.request.urlopen('https://gist.githubusercontent.com/../Companies.csv')
company_data_df = pd.read_csv(data)
```

	permalink	name	homepage_url	category_list
0	/organization/waywire	#waywire	http://www.waywire.com	|Entertainment|Politics|Social Media|News|
1	/organization/tv-communications	&TV Communications	http://enjoyandtv.com	|Games|
2	/organization/rock-your-paper	'Rock' Your Paper	http://www.rockyourpaper.org	|Publishing|Education|
3	/organization/in-touch-network	(In)Touch Network	http://www.InTouchNetwork.com	|Electronics|Guides|Coffee|Restaurants|Music|i...

https://tinyurl.com/cis545-lecture-01-19-22

Acquiring Data (2/3): Reading web tables [Company CEOs]

WIKIPEDIA
The Free Encyclopedia

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

Contribute
Help
Community portal
Recent changes
Upload file

Tools
What links here
Related changes
Special pages
Permanent link
Page information
Cite this page
Wikidata item

Print/export

List of chief executive officers

From Wikipedia, the free encyclopedia
(Redirected from CEOs of major corporations)

The following is a **list of chief executive officers** of notable companies. The list also includes lead executives with a position corresponding to chief executive officer (CEO), such as managing director (MD), and any concurrent positions held.

Contents [hide]
1 List of CEOs
2 See also
3 References
4 External links

List of CEOs [edit]

Company	Executive	Title	Since	Notes	Updated
Accenture	Julie Sweet	CEO[1]	2019	Succeeded Pierre Nanterme, died	2019-01-31
Aditya Birla Group	Kumar Birla	Chairman[2]	1995[2]	Part of the Birla family business house in India	2018-10-01
Adobe Systems	Shantanu Narayen	Chairman, president and CEO[3]	2007	Formerly with Apple Inc.	2018-10-01
Agenus	Garo H. Armen	Founder, chairman, CEO[4]	1994	Founder of the Children of Armenia Fund (COAF)	2018-10-01

https://tinyurl.com/cis545-lecture-01-19-22

Acquiring Data (2/3): Reading web tables [Company CEOs]

Import from **HTML tables** using Pandas' **read_html**:

```
company_ceos_df = pd.read_html(
'https://en.wikipedia.org/wiki/List_of_chief_executive_officers#List_of_CEOs')[0]
```

Takes the first (0^{th}) HTML table and turns it into a DataFrame!

```
<table class="wikitable sortable">

<tbody><tr>
<th>Company</th>
<th>Executive</th>
<th>Title</th>
<th>Since</th>
<th class="unsortable">Notes</th>
<th>Updated
</th></tr>
<tr>
<td><a href="/wiki/Accenture" title="Accenture">Accenture</a>
</td>
```

...

https://tinyurl.com/cis545-lecture-01-19-22

Acquiring Data (2/3):
Reading web tables [Company CEOs]

Import from **HTML tables** using Pandas' **read_html**:

```
company_ceos_df = pd.read_html(
'https://en.wikipedia.org/wiki/List_of_chief_executive_officers#List_of_CEOs')[0]
```

Takes the first (0^{th}) HTML table and turns it into a DataFrame!

	Company	Executive	Title	Since	Notes	Updated
0	Accenture	Julie Sweet	CEO[1]	2019	Succeeded Pierre Nanterme, Passed Away	2019-01-31
1	Aditya Birla Group	Kumar Birla	Chairman[2]	1995[2]	Part of the Birla family business house in India	2018-10-01
2	Being Short	Meghan	Chairman, president and CEO[3]	2007	Formerly with Apple Inc.	2018-10-01
3	Agenus	Garo H. Armen	Founder, chairman, CEO[4]	1994	Founder of the Children of Armenia Fund (COAF)	2018-10-01

https://tinyurl.com/cis545-lecture-01-19-22

Acquiring Data (3/3): DBMSs

Database management system runs in the background (often in a server on the cloud)

Handles:

- Storage of (typically tabular) data
- Updates to the data
- Queries / transformations over the data

https://tinyurl.com/cis545-lecture-01-19-22

Acquiring Data (3/3): DBMSs

Two components:

- An active *connection* to the remote machine – with appropriate user ID, password, etc.
- A request to fetch data from a table or via a query

We'll start off with a simplified DBMS called **sqlite** – it doesn't need special setup

(If you want to share data with others you would probably use something like PostgreSQL)

https://tinyurl.com/cis545-lecture-01-19-22

Acquiring Data (3/3): Reading from a DBMS

```
import sqlite3

conn = sqlite3.connect('local.db')
pd.read_sql_query('select * from companies', conn)
```

Connection (sqlite doesn't need a user ID and password)

Followed by a query to fetch all columns from **companies** into a result (to be output to the console)

https://tinyurl.com/cis545-lecture-01-19-22

Saving Dataframes to & Reading Them from a DBMS

```
import sqlite3

conn = sqlite3.connect('local.db')

company_data.to_sql("companies", conn, if_exists="replace", index=False)

pd.read_sql_query('select * from companies', conn)
```

Like the previous example, but *we* save to **companies** this time

We'll overwrite (replace) and we won't store the Pandas index

Useful to capture a snapshot, or to avoid repeating work!

https://tinyurl.com/cis545-lecture-01-19-22

The Story So Far

Acquiring data is pretty easy IF:

- It's in CSV, with labels
- It's in HTML tables, with regular rows and columns
- It's in relational database tables

But: Importing from CSV or HTML can be slow, and links can break

So, using a DBMS to store the data locally is a good idea!

(Beware that the DBMS is NOT part of your Jupyter Notebook, so you may need to host the DBMS on a **server** on the cloud)

Now, let's consider what happens if we don't have the data in a table…

https://tinyurl.com/cis545-lecture-01-19-22

https://canvas.upenn.edu/courses/1636888/quizzes/2771561

Quiz 02D

- When we read a structured file into a dataframe:
  - column names may be assigned based on the file header
  - row names may be assigned based on the file header
  - column names are assigned based on the inferred meaning of the data
  - rows are reordered

- When we store a dataframe in a SQL database, what do we have an option of saving or dropping?
  - the dataframe index
  - the data
  - the first 5 rows

https://tinyurl.com/cis545-lecture-01-19-22

Web (HTML and XML) Data

Susan B. Davidson and Zachary G. Ives

University of Pennsylvania

CIS 545 – Big Data Analytics

Portions of this lecture have been contributed to the OpenDS4All project, piloted by Penn, IBM, and the Linux Foundation

https://tinyurl.com/cis545-lecture-01-19-22

So Far

Our example has been bringing in data to test the hypothesis –

Most company CEOs are at least 40 years old

- Have a CSV file on companies
- And an HTML table on companies' CEOs
- But – we're missing the *ages* of the CEOs!

https://tinyurl.com/cis545-lecture-01-19-22

Finding Complementary Web Data

From `company_ceos_df`, find additional information on the **webpages of CEOs**

	Company	Executive	Title	Since	Notes	Updated
0	Accenture	Julie Sweet	CEO[1]	2019	Succeeded Pierre Nanterme, Passed Away	2019-01-31
1	Aditya Birla Group	Kumar Birla	Chairman[2]	1995[2]	Part of the Birla family business house in India	2018-10-01
2	Being Short	Meghan	Chairman, president and CEO[3]	2007	Formerly with Apple Inc.	2018-10-01
3	Agenus	Garo H. Armen	Founder, chairman, CEO[4]	1994	Founder of the Children of Armenia Fund (COAF)	2018-10-01

https://tinyurl.com/cis545-lecture-01-19-22

Finding Complementary Web Data

From `company_ceos_df`, find additional information on the **webpages of CEOs**

Executive names to URLs: "Julie Sweet" □

"https://en.wikipedia.org/wiki/Julie_Sweet"

	Company	Executive	Title	Since	Notes	Updated
0	Accenture	Julie Sweet	CEO[1]	2019	Succeeded Pierre Nanterme, Passed Away	2019-01-31
1	Aditya Birla Group	Kumar Birla	Chairman[2]	1995[2]	Part of the Birla family business house in India	2018-10-01
2	Being Short	Meghan	Chairman, president and CEO[3]	2007	Formerly with Apple Inc.	2018-10-01
3	Agenus	Garo H. Armen	Founder, chairman, CEO[4]	1994	Founder of the Children of Armenia Fund (COAF)	2018-10-01

https://tinyurl.com/cis545-lecture-01-19-22

Finding Complementary Web Data by *Projection*

From `company_ceos_df`, find additional information on the **webpages of CEOs**

Executive names to URLs: "Julie Sweet"

"https://en.wikipedia.org/wiki/Julie_Sweet"

```
company_ceos_df['Executive']
```

	Company	Executive	Title	Since	Notes	Updated
0	Accenture	Julie Sweet	CEO[1]	2019	Succeeded Pierre Nanterme, Passed Away	2019-01-31
1	Aditya Birla Group	Kumar Birla	Chairman[2]	1995[2]	Part of the Birla family business house in India	2018-10-01
2	Being Short	Meghan	Chairman, president and CEO[3]	2007	Formerly with Apple Inc.	2018-10-01
3	Agenus	Garo H. Armen	Founder, chairman, CEO[4]	1994	Founder of the Children of Armenia Fund (COAF)	2018-10-01

https://tinyurl.com/cis545-lecture-01-19-22

Finding Complementary Web Data by *Projection* and Iteration

From `company_ceos_df`, find additional information on the **webpages of CEOs**

Executive names to URLs: "Julie Sweet"

"https://en.wikipedia.org/wiki/Julie_Sweet"

	Company	Executive	Title	Since	Notes	Updated
0	Accenture	Julie			Succeeded Pierre	
1	Aditya Birla Group	Kuma				
2	Being Short	M				
3	Agenus	G Armen	Founder, chairman, CEO[4]	1994	of Armenia Fund (COAF)	2018-10-01

```
company_ceos_df['Executive']
```

```
crawl_list = []

for executive in company_ceos_df['Executive']:
  crawl_list.append('https://en.wikipedia.org/wiki/' + executive.replace(' ', '_'))

crawl_list
```

https://tinyurl.com/cis545-lecture-01-19-22

Finding Complementary Web Data by *Projection* and Iteration

From `company_ceos_df`, find additional information on the **webpages of CEOs**

Executive names to URLs: “Julie Sweet” □

```
['https://en.wikipedia.org/wiki/Julie_Sweet',
'https://en.wikipedia.org/wiki/Kumar_Birla',
'https://en.wikipedia.org/wiki/Meghan',
'https://en.wikipedia.org/wiki/Garo_H._Armen',
'https://en.wikipedia.org/wiki/Guillaume_Faury',
'https://en.wikipedia.org/wiki/Daniel_Zhang',
'https://en.wikipedia.org/wiki/Jeff_Bezos',
'https://en.wikipedia.org/wiki/Lisa_Su',
'https://en.wikipedia.org/wiki/Stephen_Squeri',
'https://en.wikipedia.org/wiki/Doug_Parker',
'https://en.wikipedia.org/wiki/Joseph_R._Swedish',
```

Notes	Updated
Succeeded Pierre	
of Armenia Fund (COAF)	2018-10-01

```
['Executive']:
wikipedia.org/wiki
```

https://tinyurl.com/cis545-lecture-01-19-22

Fetching Documents

```
['https://en.wikipedia.org/wiki/Julie_Sweet',
'https://en.wikipedia.org/wiki/Kumar_Birla',
```

```
for url in crawl_list:
    ...
    response = urllib.request.urlopen(url)
```

https://tinyurl.com/cis545-lecture-01-19-22

What Does a CEO Webpage Look Like?

Timothy Donald Cook (born November 1, 1960)[3] is an American business executive and industrial engineer. Cook is the chief executive officer of Apple Inc., and previously served as the company's chief operating officer under its cofounder Steve Jobs.[4]

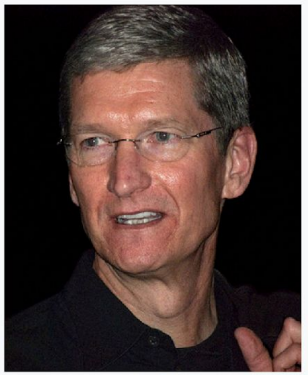
Cook joined Apple in March 1998 as a senior vice president for worldwide operations, and then served as the executive vice president for worldwide sales and operations.[5] He was made the chief executive on August 24, 2011, prior to Jobs' death in October of that year.[6] During his tenure as the chief executive, he has advocated for the political reformation of international and domestic surveillance, cybersecurity, corporate taxation, American manufacturing, and environmental preservation.

In 2014, Cook became the first chief executive of a Fortune 500 company to publicly come out as gay.[7] Cook also serves on the boards of directors of Nike, Inc.,[6] the National Football Foundation,[8] and is a trustee of Duke University.[9] In March 2015, he said he planned to donate his entire stock fortune to charity.[10]

Contents [hide]

1 Early life and education
2 Career
  2.1 Pre-Apple era
  2.2 Apple era
    2.2.1 Early career

Tim Cook



Cook in 2009

Born	Timothy Donald Cook November 1, 1960 (age 58) Mobile, Alabama, U.S.

- **Task:** extract information about date of birth!
- We can pull the HTML page via **urllib** and inspect it…

https://tinyurl.com/cis545-lecture-01-19-22

What Does the HTML Look Like?

… (lots of stuff before the portion of interest)

```
<table class="infobox biography vcard" style="width:22em"><tbody><tr><th colspan="2" style="text-align:center;font-size:125%;font-
weight:bold"><div class="fn" style="display:inline">Tim Cook</div></th></tr><tr><td colspan="2" style="text-align:center"><a
href="/wiki/File:Tim Cook (2017, cropped).jpg" class="image"><img alt="Tim Cook (2017, cropped).jpg"
src="//upload.wikimedia.org/wikipedia/commons/thumb/9/9e/Tim Cook %282017%2C cropped%29.jpg/220px-Tim Cook %282017%2C cropped%29.jpg"
decoding="async" width="220" height="286"
srcset="//upload.wikimedia.org/wikipedia/commons/thumb/9/9e/Tim Cook %282017%2C cropped%29.jpg/330px-Tim Cook %282017%2C cropped%29.jpg
1.5x, //upload.wikimedia.org/wikipedia/commons/thumb/9/9e/Tim Cook %282017%2C cropped%29.jpg/440px-Tim Cook %282017%2C cropped%29.jpg 2x"
data-file-width="2302" data-file-height="2994" /></a></td></tr><tr><th scope="row">Born</th><td><div style="display:inline"
class="nickname">Timothy Donald Cook</div><br /><span style="display:none"> (<span class="bday">1960-11-01</span>) </span>November 1,
```


```
<th scope="row">Born</th><td><div style="display:inline"
y:none"> (<span class="bday">1960-11-01</span>) </span>November 1,
```


https://tinyurl.com/cis545-lecture-01-19-22

How Do We Handle this?

The story so far: to get our data, we need to:

1. look for patterns in the HTML (`<span class="bday">...</...>`)
2. extract the text from between the tags

Option 1: build a custom **parser** (a terrible idea!)

Option 2: use a Python HTML parser

Let's see how this works, next…

https://tinyurl.com/cis545-lecture-01-19-22

https://canvas.upenn.edu/courses/1636888/quizzes/2771562

Quiz 02E

- What does *projection* do?
  - returns a subset of the columns in a dataframe
  - returns a person from a dataframe
  - returns a subset of rows from the dataframe
  - allows us to do a computation over all rows in a dataframe

https://tinyurl.com/cis545-lecture-01-19-22

Web Data and the Document Object Model

Susan B. Davidson and Zachary G. Ives

University of Pennsylvania

CIS 545 – Big Data Analytics

Portions of this lecture have been contributed to the OpenDS4All project, piloted by Penn, IBM, and the Linux Foundation

https://tinyurl.com/cis545-lecture-01-19-22

Let's Take a Look at How We Handle Web (HTML/XML) Data…

```
<table class="infobox biography vcard" style="width:22em"><tbody><tr><th colspan="2" style="text-align:center;font-size:125%;font-weight:bold"><div class="fn"
style="display:inline">Tim Cook</div></th></tr><tr><td colspan="2" style="text-align:center"><a href="/wiki/File:Tim_Cook_2009_cropped.jpg" class="image"><img alt="Tim
Cook 2009 cropped.jpg" src="//upload.wikimedia.org/wikipedia/commons/thumb/d/d5/Tim_Cook_2009_cropped.jpg/220px-Tim_Cook_2009_cropped.jpg" decoding="async" width="220"
height="272" srcset="//upload.wikimedia.org/wikipedia/commons/thumb/d/d5/Tim_Cook_2009_cropped.jpg/330px-Tim_Cook_2009_cropped.jpg 1.5x,
//upload.wikimedia.org/wikipedia/commons/d/d5/Tim_Cook_2009_cropped.jpg 2x" data-file-width="346" data-file-height="428" /></a><div>Cook in 2009</div></td></tr><tr><th
scope="row">Born</th><td><div style="display:inline" class="nickname">Timothy Donald Cook</div><br /><span style="display:none"> (<span class="bday">1960-11-01</span>)
</span>November 1, 1960<span class="noprint ForceAgeToShow"> (age 58)</span><br /><div style="display:inline" class="birthplace"><a href="/wiki/Mobile,_Alabama"
title="Mobile, Alabama">Mobile</a>, <a href="/wiki/Alabama" title="Alabama">Alabama</a>, U.S.</div></td></tr><tr><th scope="row">Residence</th><td class="label"><a
href="/wiki/Palo_Alto,_California" title="Palo Alto, California">Palo Alto, California</a>, U.S.</td></tr><tr><th scope="row">Education</th><td><a
href="/wiki/Auburn_University" title="Auburn University">Auburn University</a> (<a href="/wiki/Bachelor_of_Science" title="Bachelor of Science">BS</a>)<br /><a
href="/wiki/Duke_University" title="Duke University">Duke University</a> (<a href="/wiki/Master_of_Business_Administration" title="Master of Business
Administration">MBA</a>)</td></tr><tr><th scope="row">Employer</th><td class="org"><div class="plainlist">
<ul><li><a href="/wiki/IBM" title="IBM">IBM</a> (1982–1994)</li>
<li>Intelligent Electronics (1994–1998)</li>
```

- HTML and XML are **hierarchical** data – tags and content nested within tags
- The **lxml.etree** parser gives us a hierarchical (tree) data structure, the **Document Object Model**!

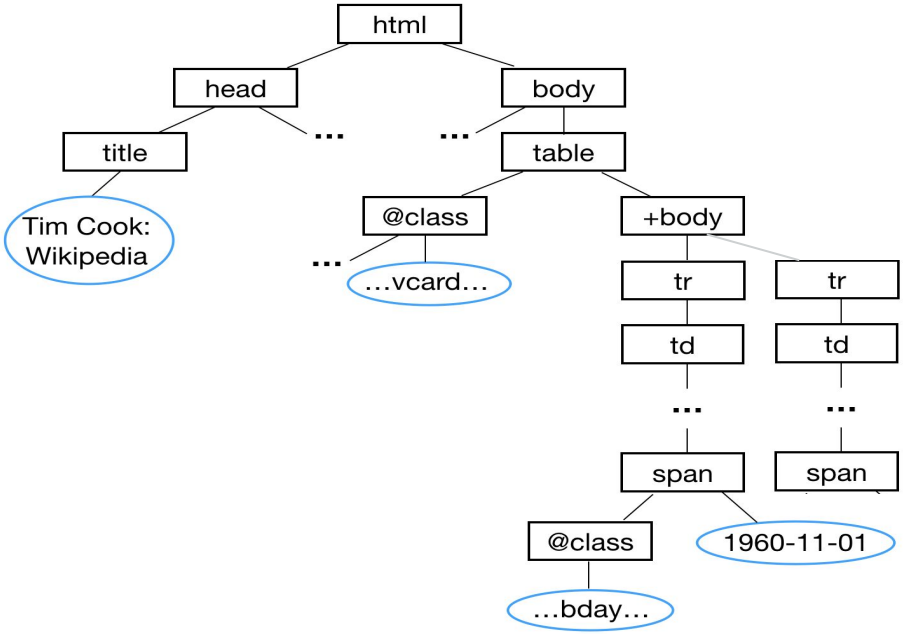
https://tinyurl.com/cis545-lecture-01-19-22

Tree Representation of the HTML Page ("Document Object Model")

We parse the page into a tree of **elements**

We can navigate this tree one node at a time…

… But it's much easier to use **XPath**, which is inspired by paths in UNIX!



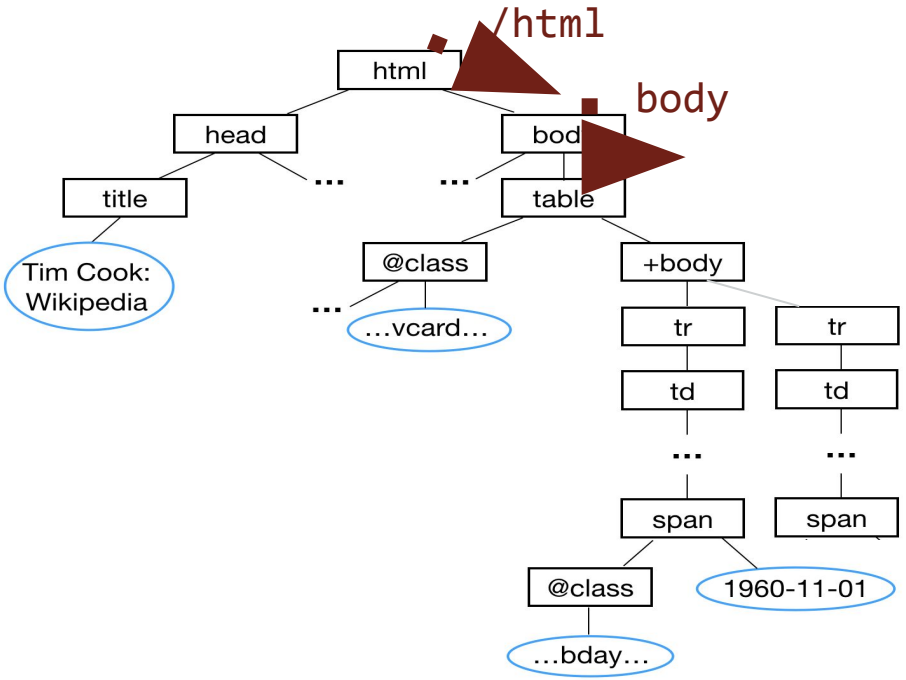
https://tinyurl.com/cis545-lecture-01-19-22

XPath in a Nutshell

XPath describes a series of *steps* to find matching nodes

- Each XPath returns a (possibly empty) **set of nodes in order**

```
/html/body
```



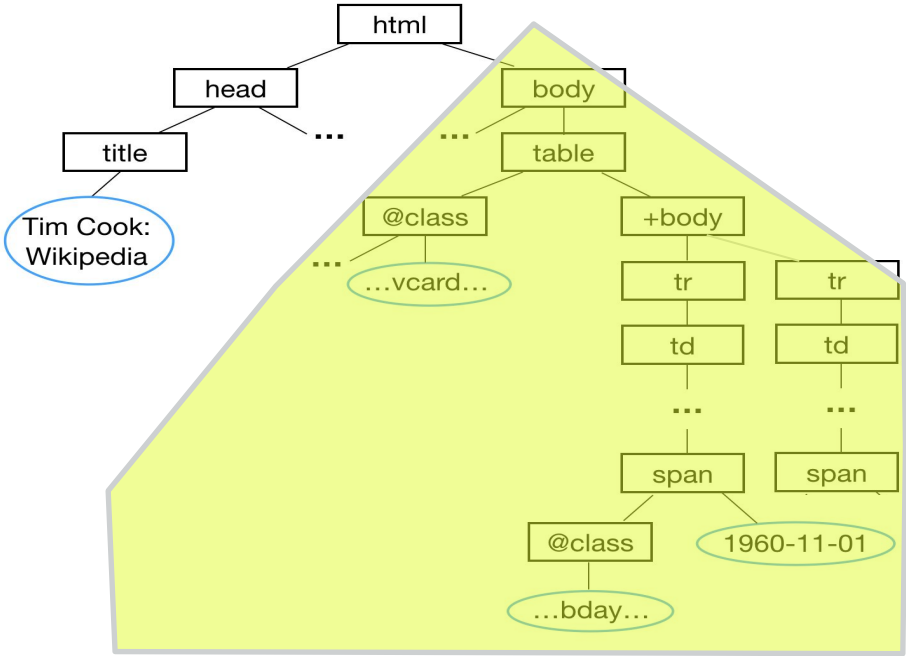
https://tinyurl.com/cis545-lecture-01-19-22

XPath in a Nutshell

XPath describes a series of *steps* to find matching nodes

- Each XPath returns a (possibly empty) **set of nodes in order**

```
/html/body
```



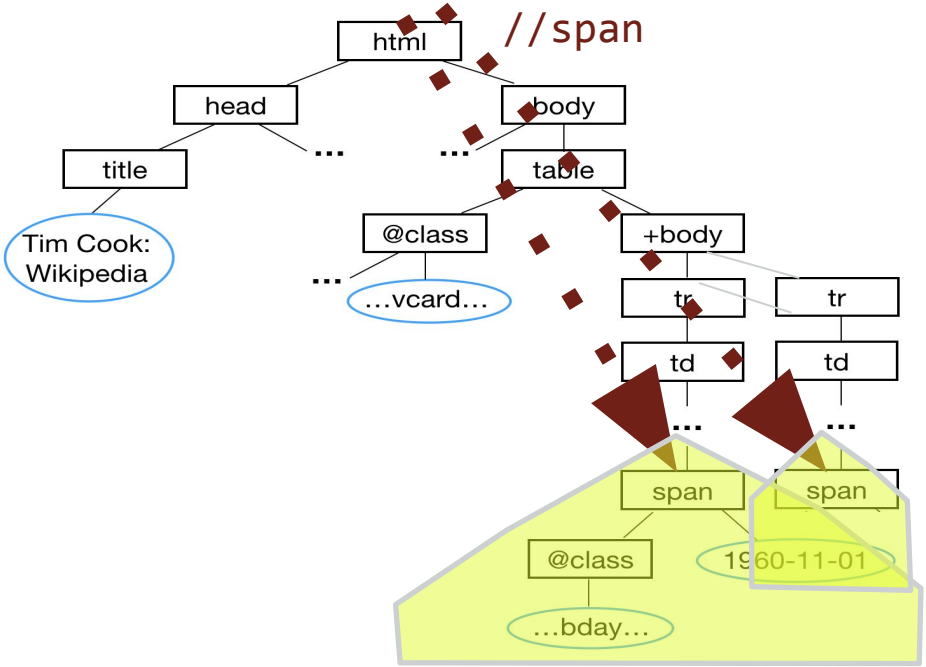
https://tinyurl.com/cis545-lecture-01-19-22

XPath in a Nutshell

XPath describes a series of *steps* to find matching nodes

- Each XPath returns a (possibly empty) **set of nodes in order**

```
//span
```



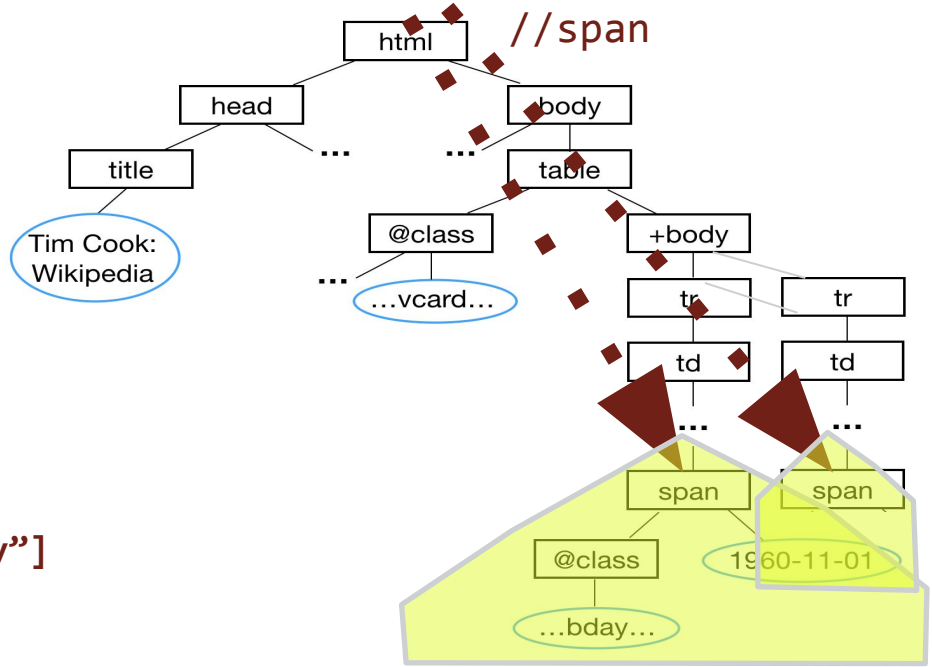
https://tinyurl.com/cis545-lecture-01-19-22

XPath in a Nutshell

XPath describes a series of *steps* to find matching nodes

- Each XPath returns a (possibly empty) **set of nodes in order**

```
//span[@class="bday"]
```



https://tinyurl.com/cis545-lecture-01-19-22

Back to Our Wikipedia CEO Birthdays: Finding the Vcard Table Subtree

```
(<span class="bday">1960-11-01</span>)
```

Perhaps we want to find any node (“//”, descendent traversal) with **label “table”** with an **attribute “class”** that contains the **substring “vcard”**.

```
//table[contains(@class,"vcard")]
```

Then extend that path to extract **bday** information:

```
//table[contains(@class,"vcard")]//span[@class="bday"]/text()
```

https://tinyurl.com/cis545-lecture-01-19-22

XPath and Its Use in Information Extraction (IE)

- XPath is a rich language – see https://www.w3schools.com/xml/xpath_intro.asp
- Main take-aways:
  - Simplifies matching within DOM,
  - Roughly corresponds to pathnames in Unix
  - Returns **ordered sets of nodes** (as Python lists)
  - Main parts are **path steps**
- Many tools are built using this, see https://blog.scrapinghub.com/
- *The simplest form of IE; later IE over text phrases!*

https://tinyurl.com/cis545-lecture-01-19-22

OK, Now We Have a Toolkit for Processing XML

Let's now finish out the data acquisition task for our "company CEO age" question!

https://tinyurl.com/cis545-lecture-01-19-22

Quiz 02F

- The document object model is what kind of data structure?
  - dictionary
  - array
  - tree
  - linked list

https://canvas.upenn.edu/courses/1636888/quizzes/2771567

- What does XPath return as its result?
  - multiset of nodes in arbitrary order
  - error if nothing matches
  - single node
  - set of matching nodes in order, as a list

The XPath //span matches

- all elements except span
- span as the root element
- the span element(s) at any level
- all leaf-level span elements

https://tinyurl.com/cis545-lecture-01-19-22

Completing Our Acquisition… And Setting up for Integration

Susan B. Davidson and Zachary G. Ives

University of Pennsylvania

CIS 545 – Big Data Analytics

Portions of this lecture have been contributed to the OpenDS4All project, piloted by Penn, IBM, and the Linux Foundation

https://tinyurl.com/cis545-lecture-01-19-22

Wrapping Up Our Discussion of Data Acquisition

XPath lets us pull content out of HTML (and XML)

Let's get our CEOs' birthdays now!

… We'll combine with our company data and company-CEO data (in the next module)

https://tinyurl.com/cis545-lecture-01-19-22

Let's Load Data

```
['https://en.wikipedia.org/wiki/Julie_Sweet',
'https://en.wikipedia.org/wiki/Kumar_Birla',
'https://en.wikipedia.org/wiki/Meghan',
```

```
for page in pages:
    tree = etree.HTML(page.read().decode( "utf-8"))
    url = page.geturl()
    bday = tree.xpath( '//table[contains(@class,"vcard")]//span[@class="bday
"]/text()')
    if len(bday) > 0:
        name = url[url.rfind( '/')+1:] # Last part of URL
        exec_df = exec_df.append({ 'name': name, 'page': url,
                    'born': datetime.datetime.strptime(bday[ 0], '%Y-%m-%d')}
, ignore_index=True)
```

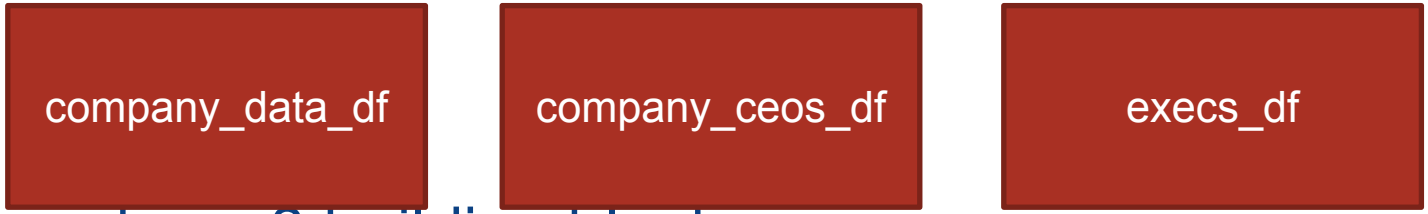
https://tinyurl.com/cis545-lecture-01-19-22

Acquiring Data, 3/3:Creating execs_df

	name	page	born
0	Julie_Sweet	https://en.wikipedia.org/wiki/Julie_Sweet	NaT
1	Kumar_Birla	https://en.wikipedia.org/wiki/Kumar_Birla	1967-06-14
2	Shantanu_Narayen	https://en.wikipedia.org/wiki/Shantanu_Narayen	1963-05-27
3	Garo_H._Armen	https://en.wikipedia.org/wiki/Garo_H._Armen	1953-01-31
4	Tom_Enders	https://en.wikipedia.org/wiki/Tom_Enders	NaT
5	Daniel_Zhang	https://en.wikipedia.org/wiki/Daniel_Zhang	1972-01-11
6	Jeff_Bezos	https://en.wikipedia.org/wiki/Jeff_Bezos	1964-01-12
7	Lisa_Su	https://en.wikipedia.org/wiki/Lisa_Su	1969-11-07
8	Stephen_Squeri	https://en.wikipedia.org/wiki/Stephen_Squeri	NaT

https://tinyurl.com/cis545-lecture-01-19-22

Recap



We now have 3 building blocks:

1. From CSV, company_data_df : company categories
2. From HTML tables, company_ceos_df: companies' CEOs
3. From HTML information extraction, execs_df: CEOs' birthdays

We're set for our next topic – *integrating* the data!

https://tinyurl.com/cis545-lecture-01-19-22