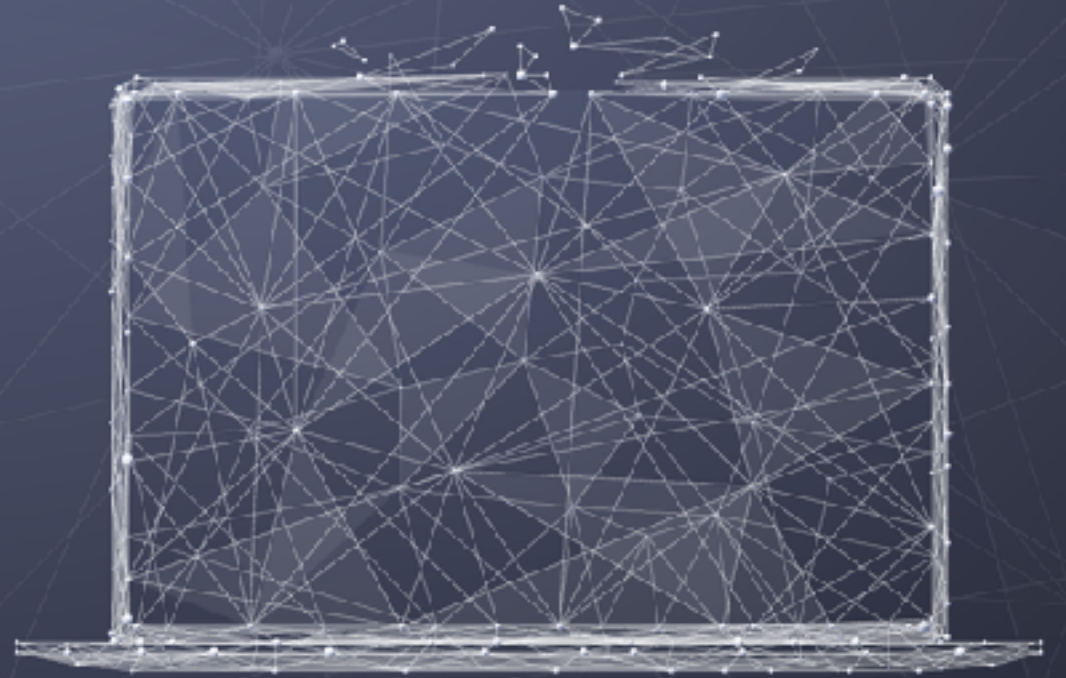


# **Data Science Data Engineering I**

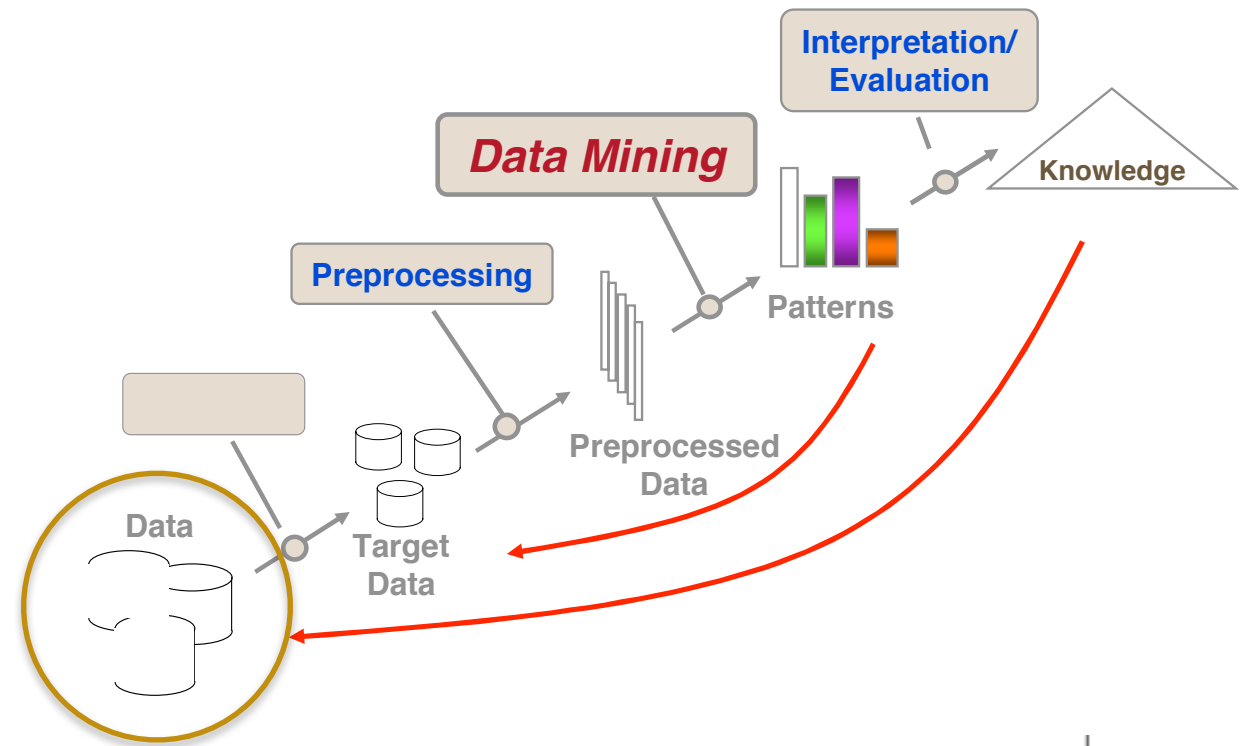
**What is data?**



**PURDUE**  
UNIVERSITY®

College of Science

# First step: obtain data





# What is data?

- Collection of entities and their attributes
- Attribute:** property or characteristic of an entity (e.g., eye color, temperature)
- Entity:** collection of attributes  
Aka: record, point, case, sample, object, or instance.

Entities

## Attributes

Name	Thread pitch (mm)	Minor diameter tolerance	Nominal diameter (mm)	Head shape	Price for 50 screws	Available at factory outlet?	Number in stock	Flat or Phillips head?
M4	0.7	4g	4	Pan	\$10.08	Yes	275	Flat
M5	0.8	4g	5	Round	\$13.89	Yes	183	Both
M6	1	5g	6	Button	\$10.42	Yes	1013	Flat
M8	1.25	5g	8	Pan	\$11.08	No	295	Phillips
M10	1.5	6g	10	Round	\$15.74	Yes	485	Phillips
M12	1.75	7g	12	Pan	\$18.20	No	995	Flat
M14	2	7g	14	Round	\$21.19	No	235	Phillips
M16	2	8g	16	Button	\$23.57	Yes	292	Both
M18	2.1	8g	18	Button	\$25.87	No	664	Both
M20	2.4	8g	20	Pan	\$29.09	Yes	486	Both
M24	2.55	9g	24	Round	\$33.01	Yes	982	Phillips
M28	2.7	10g	28	Button	\$35.86	No	1067	Phillips
M30	3.2	12g	30	Pan	\$41.32	No	434	Both
M50	4.5	15g	50	Pan	\$44.72	No	740	Flat



# Tabular data (Simple structured data)

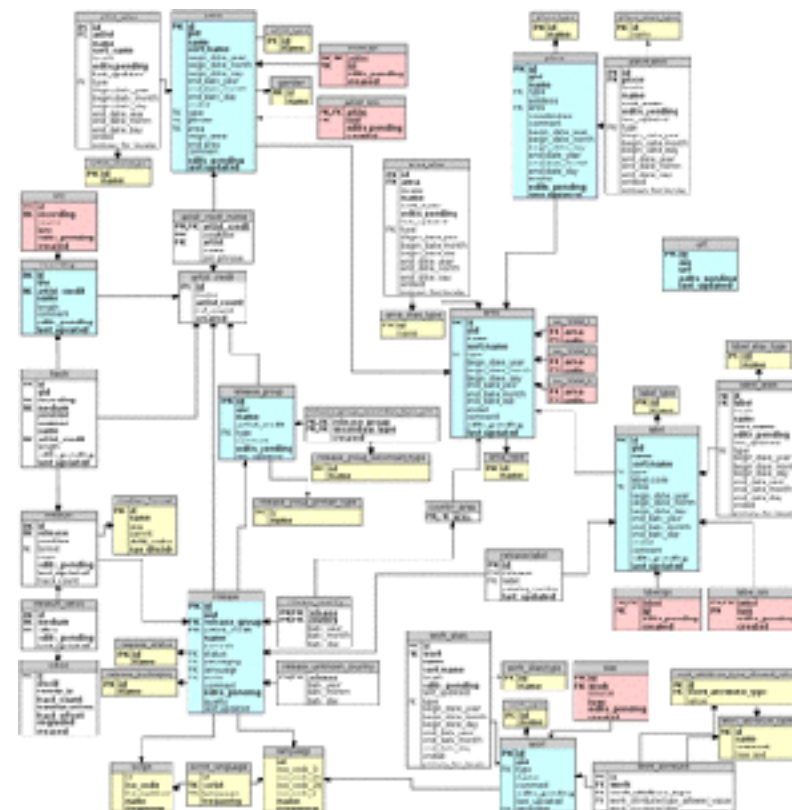
Collection of records, each of which consists of a fixed set of attributes

position #	time [UTC]	time [GMT]	Longitude [deg]	Latitude [deg]	Altitude [m]	Status	Main [V]	Beacon [V]	Temperature [°C]
1	2014-01-01 00:00:44	2014-01-01 02:00:44	32.24420	-27.80097	119	val GPS-00	3.36	2.00	25
2	2014-01-01 01:00:44	2014-01-01 03:00:44	32.24422	-27.88077	110	val GPS-30	3.36	2	24
3	2014-01-01 02:00:44	2014-01-01 04:00:44	32.24427	-27.88084	113	val GPS-30	3.36	2	24
4	2014-01-01 03:00:71	2014-01-01 05:00:71	32.24487	-27.88001	166	val GPS-30	3.36	1.84	23
5	2014-01-01 04:00:27	2014-01-01 06:00:27	32.24103	-27.87450	121	GPS-10	3.36	1.92	23
6	2014-01-01 05:00:54	2014-01-01 07:00:54	32.23727	-27.87759	121	GPS-20	3.36	2.21	26
7	2014-01-01 06:01:35	2014-01-01 08:01:35	32.23747	-27.87750	122	GPS-20	3.36	2.32	26
8	2014-01-01 07:01:20	2014-01-01 09:01:20	32.23740	-27.87713	124	GPS-20	3.36	2.64	28
9	2014-01-01 08:01:17	2014-01-01 10:01:17	32.23733	-27.87693	97	GPS-30	3.36	2.73	29
10	2014-01-01 09:01:14	2014-01-01 11:01:14	32.24147	-27.87177	80	val GPS-10	3.36	2.72	29
11	2014-01-01 10:01:20	2014-01-01 12:01:20	32.24537	-27.87179	84	GPS-20	3.36	2.4	27
12	2014-01-01 11:01:01	2014-01-01 13:01:01	32.24460	-27.87202	83	GPS-20	3.36	2.24	26
13	2014-01-01 12:01:30	2014-01-01 14:01:30				No Fix	3.36	2.32	26
14	2014-01-01 13:02:34	2014-01-01 15:02:34	32.24191	-27.87203	80	val GPS-30	3.36	2.64	29
15	2014-01-01 14:00:00	2014-01-01 16:00:00	32.24068	-27.87272	81	val GPS-10	3.36	2.74	26
16	2014-01-01 15:01:02	2014-01-01 17:01:02	32.23930	-27.87517	91	GPS-00	3.36	2.61	29
17	2014-01-01 16:01:13	2014-01-01 18:01:13	32.23813	-27.87820	104	val GPS-30	3.36	2.64	29
18	2014-01-01 17:00:10	2014-01-01 19:00:10	32.23720	-27.87853	100	val GPS-30	3.36	2.32	26
19	2014-01-01 18:01:38	2014-01-01 20:01:38	32.23773	-27.87838	113	val GPS-30	3.36	2.16	25
20	2014-01-01 19:00:02	2014-01-01 21:00:02	32.24650	-27.87858	136	val GPS-10	3.36	2	24
21	2014-01-01 20:00:50	2014-01-01 22:00:50	32.24410	-27.87817	103	val GPS-10	3.36	1.92	23





**Collection of data items, each with varying properties, as well as relations among the items, e.g., relational databases, graph data (organization of information is specified by data schema/model)**





# Semi-structured data

Collection of data items with less formal structure than relational data, but with some tags or markers to delineate semantic elements of items (i.e., properties), e.g., HTML, XML, or JSON

```
▼<user-agents>
  ▼<user-agent>
    <ID>id_a_f_3</ID>
    <String>!Susie {http://www.sync2it.com/susie}</String>
    <Description>Sync2It bookmark management & clustering engine</Description>
    <Type>C R</Type>
    <Comment/>
    <Link1>http://www.sync2it.com</Link1>
    <Link2/>
  </user-agent>
  ▼<user-agent>
    <ID>id_a_f_6</ID>
    ▼<String>
      <a href='http://www.unchaos.com/'> UnChaos </a> From Chaos To Order Hybrid
    </String>
    <Description>UnCHAOS search robot</Description>
    <Type>R</Type>
    <Comment>Site is dead</Comment>
    <Link1>http://www.unchaos.com/</Link1>
    <Link2/>
  </user-agent>
```

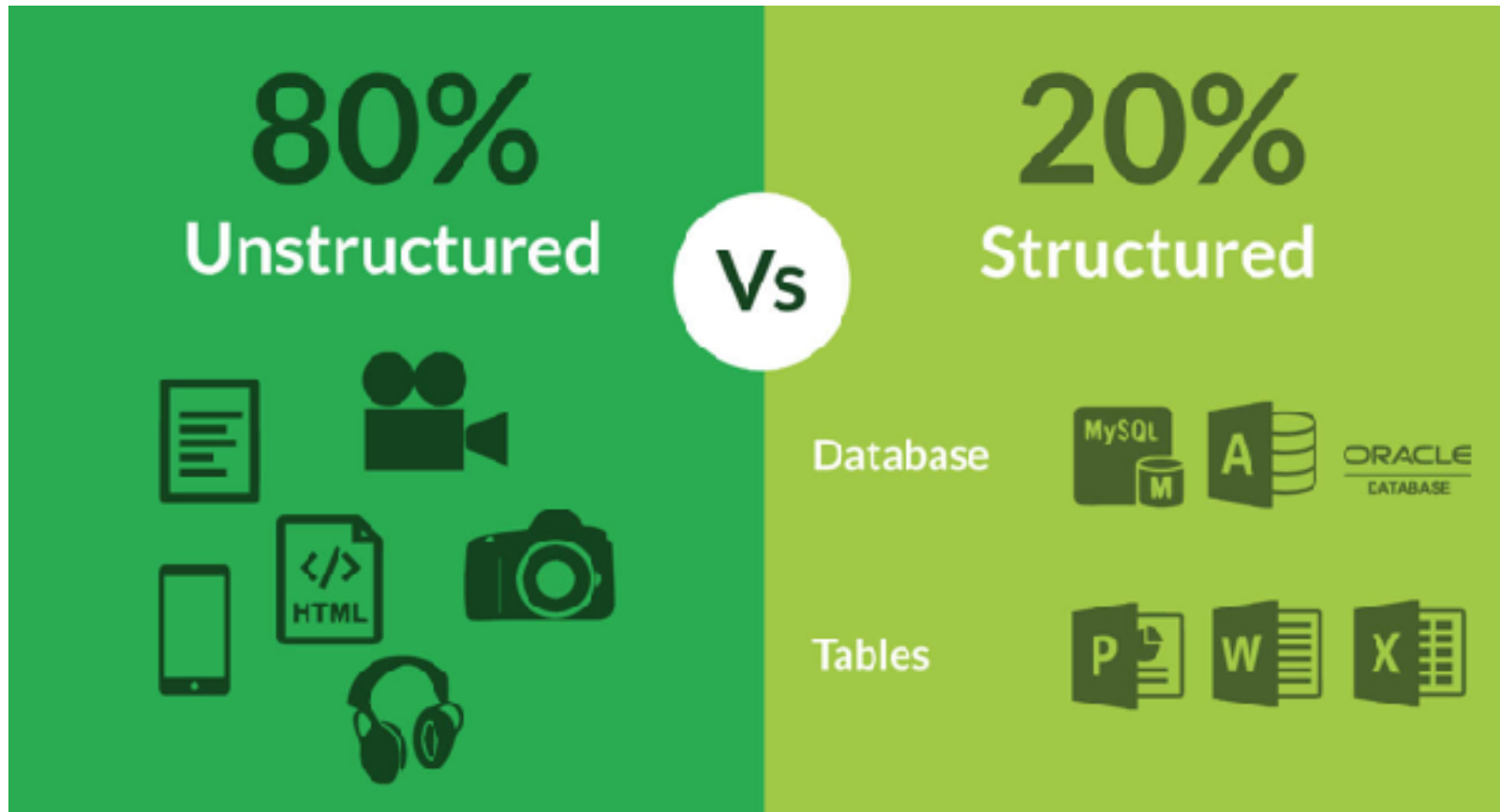
```
{
  "business_id": "PK6a5izckHFWk8i0oxt5DA",
  "full_address": "400 Waterfront Dr E\nHomestead\nHomestead, PA 15120",
  "hours": {},
  "open": true,
  "categories": [
    "Burgers",
    "Fast Food",
    "Restaurants"
  ],
  "city": "Homestead",
  "review_count": 5,
  "name": "McDonald's",
  "neighborhoods": [
    "Homestead"
  ],
  "longitude": -79.910032,
  "state": "PA",
  "stars": 2,
  "latitude": 40.412086,
  "attributes": {
    "Take-out": true,
    "Wi-Fi": "free",
    "Drive-Thru": true,
    "Good For": {
```

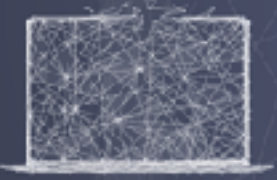


# Unstructured data

**Data that is not organized according to a predefined structure (e.g., free text, videos, photos, music, messages, etc.)**

Lack of structure makes it more difficult to search for and analyze patterns

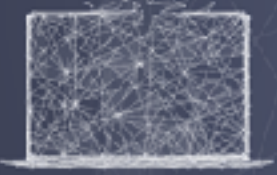




# Common data formats

- CSV (comma separate values)
- JSON (Javascript object notation)
- HTML/XML (hypertext markup language / extensible markup language)
- SQL and NoSQL databases (SQL=structured query language)





# Initial data tasks

## Read data into internal memory data structure

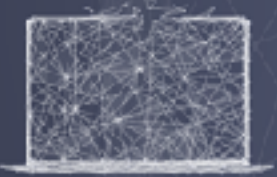
- Read in data from file, parse, and store in internal data structure
- Data structures for tabular data:
  - Single example can be stored in:
    - Vector or list
    - Dictionary (key=attribute name, value=attribute value)
  - Set of examples can be stored in:
    - Matrix or list of lists/dictionaries
    - Dictionary (key=example id, value=example data)
    - Data frame (table-like data structure in Pandas)



- ```

Process  "NounName" "Name" "Handles" "VM" "WS" "PM" "NPM" "Path" "Company" "CPU" "FileVersion" "ProductVersion"
Process  "AcDeskBand" "pr" "27" "53273720" "10047483" "4352240" "14224" "C:\Program Files (x86)\Lai
Process  "AcPr-MarsSvc" "167" "72732672" "11051008" "4956150" "7120"
Process  "AcSvc" "411" "127533056" "17551360" "10117120" "26060"
Process  "amsvc" "54" "15129728" "4509696" "1601535" "8528"
Process  "btwdvcs" "127" "60632092" "7720960" "3743744" "10056"
Process  "CamMute" "111" "40837120" "5312512" "1789952" "9410"
Process  "ComExec" "1433" "329342576" "89608192" "58417152" "58432"
Process  "conhost" "35" "25186304" "4136960" "2048000" "5158"
Process  "conhost" "45" "52461568" "6311936" "325390" "6350" "C:\Windows\System32\conhost.exe" "M
Process  "csrss" "1143" "51078912" "5923808" "2977792" "17768"
Process  "csrss" "628" "86055296" "17383424" "20799488" "28704"
Process  "CvAudMpeg" "98" "57937520" "6680576" "7385088" "8832"
Process  "daemonu" "413" "73506816" "8957952" "5935104" "20132"
Process  "DcaSvc" "642" "64344064" "13956032" "10173560" "27904"
Process  "dcaarray" "644" "220614656" "35368960" "33372960" "34628" "C:\Program Files (x86)\directa
Process  "dwm" "108" "84504576" "10313728" "5709824" "10028" "C:\Windows\System32\Dwm.exe" "Microso
Process  "HvKlang" "284" "105017344" "20892000" "3966496" "22248"
Process  "EXCEL" "384" "316968960" "42233855" "27745304" "48584" "C:\Program Files (x86)\Microsoft
Process  "explorer" "1153" "357845824" "75698176" "52066624" "84040" "C:\Windows\explorer.exe" "Mi
Process  "frapp" "30" "53719040" "6185576" "3743744" "6960" "C:\Program Files\CONEXANT\ForceConfig
Process  "fwcAgent" "515" "59432960" "12111372" "9289728" "22248"
Process  "hkcmd" "84" "71577600" "8040448" "3984928" "8048" "C:\Windows\System32\hkcmd.exe" "Intel
Process  "ilmpmvs" "61" "44564480" "4255840" "2301352" "5568"
Process  "Idle" "0" "24576" "0" "0" "0" "0" "0" "0"
Process  "iexplore" "774" "304881664" "79273080" "77525083" "82152" "C:\Program Files (x86)\Intern
Process  "iexplore" "639" "157421568" "22075160" "12051403" "38648" "C:\Program Files (x86)\Intern
Process  "iexplore" "203" "83292160" "10544512" "5844952" "10808" "C:\Windows\System32\iexplore.ex
Process  "LMS" "112" "41300160" "5484544" "2117632" "5600"
Process  "lsass" "1545" "75075384" "27377664" "18477056" "4721"
Process  "lsass" "113" "13779584" "5664768" "3756032" "7600"
Process  "micnote" "110" "48323704" "5672960" "6272572" "11120"
Process  "mscscsvw" "105" "73936896" "14417920" "8912896" "11184"
Process  "MsttB32A" "118" "45617352" "8012896" "4481024" "10328"
Process  "MsttCtsvc" "712" "565702656" "33861632" "40435712" "41284"
Process  "MstTpmSvc" "84" "43318296" "10129408" "6975488" "7568"
Process  "MstEng" "503" "238940160" "87953108" "117288960" "47112"
Process  "MSCIbSvc" "612" "106561536" "21938176" "14983168" "27648"
Process  "MSCIbSvc" "72" "36134912" "4841472" "1621440" "3080"
Process  "mssecet" "341" "156336128" "23216128" "10706944" "75232" "C:\Program Files\Microsoft seci
Process  "NisSvc" "261" "80003072" "4771840" "9138176" "18000"
Process  "nusb3mon" "89" "75804672" "5914624" "2224128" "10328" "C:\Program Files (x86)\Panasas E

```



# XML/HTML Files

- The main format for the web
- XML files contain hierarchical content delineated by tags
- HTML is syntactically like XML, but sometimes tags are not closed and tags are primarily used to describe appearance of page
- There are a number of Python parsers for XML/HTML. We will use the BeautifulSoup library.

```
▼<user-agents>
  ▼<user-agent>
    <ID>id_a_f_3</ID>
    <String>!Susie (http://www.sync2it.com/susie)</String>
    <Description>Sync2It bookmark management & clustering engine</Description>
    <Type>C R</Type>
    <Comment/>
    <Link1>http://www.sync2it.com</Link1>
    <Link2/>
  </user-agent>
  ▼<user-agent>
    <ID>id_a_f_6</ID>
    ▼<String>
      <a href='http://www.unchaos.com/'> UnChaos </a> From Chaos To Order Hybrid
    </String>
    <Description>UNCHAOS search robot</Description>
    <Type>R</Type>
    <Comment>Site is dead</Comment>
    <Link1>http://www.unchaos.com/</Link1>
    <Link2/>
  </user-agent>
```



# JSON Files

- JSON originated as a way of encapsulating Javascript objects
- JSON data looks much like a dictionary in Python, with keys and value stores
- Python has built in methods to parse JSON, but Pandas can also be used to read and parse JSON

```
{
  "business_id": "PK6a5izckHPWk0i0xt5DA",
  "full_address": "400 Waterfront Dr E\\nHomestead\\nHomestead, PA 15120",
  "hours": {},
  "open": true,
  "categories": [
    "Burgers",
    "Fast Food",
    "Restaurants"
  ],
  "city": "Homestead",
  "review_count": 5,
  "name": "McDonald's",
  "neighborhoods": [
    "Homestead"
  ],
  "longitude": -79.910032,
  "state": "PA",
  "stars": 2,
  "latitude": 40.412086,
  "attributes": {
    "Take-out": true,
    "Wi-Fi": "free",
    "Drive-Thru": true,
    "Good For": {
      "dessert": false,
      "latenight": false,
      "lunch": false,
      "dinner": false,
      "breakfast": false,
      "brunch": false
    },
    "Caters": false,
    "Noise Level": "average",
    "Takes Reservations": false,
    "Delivery": false
  }
}
```





# JSON Files

- JSON originated as a way of encapsulating Javascript objects
- JSON data looks much like a dictionary in Python, with keys and value stores
- Python has built in methods to parse JSON, but Pandas can also be used to read and parse JSON

```
{
  "business_id": "PK6a5izckHPWk0i0xt5DA",
  "full_address": "400 Waterfront Dr E\\nHomestead\\nHomestead, PA 15120",
  "hours": {},
  "open": true,
  "categories": [
    "Burgers",
    "Fast Food",
    "Restaurants"
  ],
  "city": "Homestead",
  "review_count": 5,
  "name": "McDonald's",
  "neighborhoods": [
    "Homestead"
  ],
  "longitude": -79.910032,
  "state": "PA",
  "stars": 2,
  "latitude": 40.412086,
  "attributes": {
    "Take-out": true,
    "Wi-Fi": "free",
    "Drive-Thru": true,
    "Good For": {
      "dessert": false,
      "latenight": false,
      "lunch": false,
      "dinner": false,
      "breakfast": false,
      "brunch": false
    },
    "Caters": false,
    "Noise Level": "average",
    "Takes Reservations": false,
    "Delivery": false
  }
}
```