Advanced Data Management (CSCI 490/680)

Data and Pandas

Dr. David Koop
Arrays

What is the difference between an array and a list (or a tuple)?
Arrays

• Usually a fixed size—lists are meant to change size
• Are mutable—tuples are not
• Store only one type of data—lists and tuples can store anything
• Are faster to access and manipulate than lists or tuples
• Can be multidimensional:
  - Can have list of lists or tuple of tuples but no guarantee on shape
  - Multidimensional arrays are rectangles, cubes, etc.
Why NumPy?

- Fast **vectorized** array operations for data munging and cleaning, subsetting and filtering, transformation, and any other kinds of computations
- Common array algorithms like sorting, unique, and set operations
- Efficient descriptive statistics and aggregating/summarizing data
- Data alignment and relational data manipulations for merging and joining together heterogeneous data sets
- Expressing conditional logic as array expressions instead of loops with `if-elif-else` branches
- Group-wise data manipulations (aggregation, transformation, function application).

[W. McKinney, Python for Data Analysis]
NumPy Arrays

- `data1 = [6, 7.5, 8, 0, 1]`  
  `arr1 = np.array(data1)`

- **Zeros**: `np.zeros(10)`, **Ones**: `np.ones((4, 5))`,  
  **Empty**: `np.empty((2, 2))`

- # of dimensions: `arr2.ndim`, **Shape**: `arr2.shape`, **Type**: `arr2.dtype`

- **Types**: Each array has a fixed type unlike other variables in python
2D Array Slicing

How to obtain the blue slice from array `arr`?

[W. McKinney, Python for Data Analysis]
2D Array Slicing

How to obtain the blue slice from array `arr`?

Expression | Shape
---|---
`arr[:2, 1:]` | `(2, 2)`

[W. McKinney, Python for Data Analysis]
2D Array Slicing

How to obtain the blue slice from array `arr`?

<table>
<thead>
<tr>
<th>Expression</th>
<th>Shape</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>arr[:, 1:]</code></td>
<td><code>(2, 2)</code></td>
</tr>
<tr>
<td><code>arr[2]</code></td>
<td><code>(3,)</code></td>
</tr>
<tr>
<td><code>arr[2, :]</code></td>
<td><code>(3,)</code></td>
</tr>
<tr>
<td><code>arr[2:, :]</code></td>
<td><code>(1, 3)</code></td>
</tr>
</tbody>
</table>

How to obtain the blue slice from array `arr`?

<table>
<thead>
<tr>
<th>Expression</th>
<th>Shape</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>arr[0:2, 1:]</code></td>
<td><code>(2, 2)</code></td>
</tr>
<tr>
<td><code>arr[2]</code></td>
<td><code>(3,)</code></td>
</tr>
<tr>
<td><code>arr[2, :]</code></td>
<td><code>(3,)</code></td>
</tr>
<tr>
<td><code>arr[2:, :]</code></td>
<td><code>(1, 3)</code></td>
</tr>
</tbody>
</table>

[W. McKinney, Python for Data Analysis]
Boolean Indexing

- \texttt{names == 'Bob'} gives back booleans that represent the element-wise comparison with the array \texttt{names}

- Boolean arrays can be used to index into another array:
  - \texttt{data[names == 'Bob']}

- Can even mix and match with integer slicing

- Can do boolean operations (\&, \|) between arrays (just like addition, subtraction)
  - \texttt{data[(names == 'Bob') \| (names == 'Will')]}  

- Note: \texttt{or} and \texttt{and} do not work with arrays

- We can set values too! \texttt{data[data < 0] = 0}
Assignment 1

• Using Python for data analysis
• Analyze hurricane data (through 2018)
• Provided *a1.ipynb* file (right-click and download)
• Use basic python (+ collections module) for now to demonstrate language knowledge
• Use Anaconda or hosted Python environment (Colab, Azure Notebooks, etc.)
• Due Wednesday
• Turn *.ipynb* file in via Blackboard
Chicago Food Inspections Exploration

• Based on David Beazley's PyData Chicago talk
• YouTube video: https://www.youtube.com/watch?v=j6VSAsKAj98
• Our in-class exploration:
  - Python can give answers fairly quickly
  - Data analysis questions:
    • What is information is available
    • **Questions** are interesting about this dataset
    • How to decide on good follow-up questions
    • What the computations mean
Chicago Food Inspections Exploration
Data

• What is data?
  - Types
  - Semantics
• How is data structured?
  - Tables (Data Frames)
  - Databases
  - Data Cubes
• What formats is data stored in?
• Raw versus derived data
Data

• What is this data?

<table>
<thead>
<tr>
<th>R011</th>
<th>42ND STREET &amp; 8TH AVENUE</th>
<th>00228985</th>
<th>00008471</th>
<th>00000441</th>
<th>00001455</th>
<th>00000134</th>
<th>00033341</th>
<th>0071255</th>
</tr>
</thead>
<tbody>
<tr>
<td>R170</td>
<td>14TH STREET-UNION SQUARE</td>
<td>00224603</td>
<td>0011051</td>
<td>00000827</td>
<td>00003026</td>
<td>00000660</td>
<td>0089367</td>
<td>00199841</td>
</tr>
<tr>
<td>R046</td>
<td>42ND STREET &amp; GRAND CENTRAL</td>
<td>00207758</td>
<td>0007908</td>
<td>0000323</td>
<td>0001183</td>
<td>0003001</td>
<td>0040759</td>
<td>0096613</td>
</tr>
</tbody>
</table>

• Semantics: real-world meaning of the data
• Type: structural or mathematical interpretation
• Both often require metadata
  - Sometimes we can infer some of this information
  - Line between data and metadata isn’t always clear
# Data

<table>
<thead>
<tr>
<th>REMOTE</th>
<th>STATION</th>
<th>FF</th>
<th>Y</th>
<th>SEN/DIS</th>
<th>7-D ATLAS UNL</th>
<th>D ATLAS/RMF</th>
<th>JOIN T RR TKT</th>
<th>7-D UNL</th>
<th>30-D UNL</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>R011 42ND STREET &amp; 8TH AVENUE</td>
<td>00228985</td>
<td>00008471</td>
<td>00000441</td>
<td>00001455</td>
<td>00000134</td>
<td>00033341</td>
<td>00071255</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>R170 14TH STREET-UNION SQUARE</td>
<td>00224603</td>
<td>00011051</td>
<td>00000827</td>
<td>00003026</td>
<td>00000660</td>
<td>00089367</td>
<td>00199841</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>R046 42ND STREET &amp; GRAND CENTRAL</td>
<td>00207758</td>
<td>00007908</td>
<td>00000323</td>
<td>00001183</td>
<td>00003001</td>
<td>00040759</td>
<td>00096613</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>R012 34TH STREET &amp; 6TH AVENUE</td>
<td>00188311</td>
<td>00006490</td>
<td>00000498</td>
<td>00001279</td>
<td>00003522</td>
<td>00036527</td>
<td>00067483</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>R293 34TH STREET - PENN STATION</td>
<td>00168768</td>
<td>00006155</td>
<td>00000523</td>
<td>00001065</td>
<td>00005031</td>
<td>00030645</td>
<td>00054376</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>R033 42ND STREET/TIMES SQUARE</td>
<td>00159382</td>
<td>00005945</td>
<td>00000378</td>
<td>00001205</td>
<td>00000690</td>
<td>00058931</td>
<td>00078644</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>R022 34TH STREET &amp; 6TH AVENUE</td>
<td>00156008</td>
<td>00006276</td>
<td>00000487</td>
<td>00001543</td>
<td>00000712</td>
<td>00058910</td>
<td>00110466</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>R084 59TH STREET/COLUMBUS CIRCLE</td>
<td>00155262</td>
<td>00009484</td>
<td>00000589</td>
<td>00002071</td>
<td>00000542</td>
<td>00053397</td>
<td>00113966</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>R020 47-50 STREETS/ROCKEFELLER</td>
<td>00143500</td>
<td>00006402</td>
<td>00000384</td>
<td>00001159</td>
<td>00000723</td>
<td>00037978</td>
<td>00090745</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>R179 86TH STREET-LEXINGTON AVE</td>
<td>00142169</td>
<td>00010367</td>
<td>00000470</td>
<td>00001839</td>
<td>00000271</td>
<td>00050328</td>
<td>00125250</td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>R023 34TH STREET &amp; 6TH AVENUE</td>
<td>00134052</td>
<td>00005005</td>
<td>00000348</td>
<td>00001112</td>
<td>00000649</td>
<td>00031531</td>
<td>00075040</td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>R029 PARK PLACE</td>
<td>00121614</td>
<td>0004311</td>
<td>00000287</td>
<td>00000931</td>
<td>00000792</td>
<td>00025404</td>
<td>00065362</td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>R047 42ND STREET &amp; GRAND CENTRAL</td>
<td>00100742</td>
<td>00004273</td>
<td>00000185</td>
<td>00000704</td>
<td>00001241</td>
<td>00022808</td>
<td>00068216</td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>R031 34TH STREET &amp; 7TH AVENUE</td>
<td>00095076</td>
<td>00003990</td>
<td>00000232</td>
<td>00000727</td>
<td>00001459</td>
<td>00024284</td>
<td>00038671</td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>R017 LEXINGTON AVENUE</td>
<td>00094655</td>
<td>00004688</td>
<td>00000190</td>
<td>00000833</td>
<td>00000754</td>
<td>00020018</td>
<td>00055066</td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>R175 8TH AVENUE-14TH STREET</td>
<td>00094313</td>
<td>00003907</td>
<td>00000286</td>
<td>00001144</td>
<td>00000256</td>
<td>00038272</td>
<td>00074661</td>
<td></td>
</tr>
<tr>
<td>17</td>
<td>R057 BARCLAYS CENTER</td>
<td>00093804</td>
<td>00004204</td>
<td>00000454</td>
<td>00001386</td>
<td>00001491</td>
<td>00030113</td>
<td>00068119</td>
<td></td>
</tr>
<tr>
<td>18</td>
<td>R138 WEST 4TH ST-WASHINGTON SO</td>
<td>00093562</td>
<td>00004677</td>
<td>00000251</td>
<td>00000965</td>
<td>00000127</td>
<td>00031628</td>
<td>00074458</td>
<td></td>
</tr>
</tbody>
</table>
Dataset Types

- **Tables**
  - Attributes (columns)
  - Items (rows)
  - Cell containing value

- **Networks**
  - Link
  - Node (item)

- **Fields (Continuous)**
  - Grid of positions
  - Attributes (columns)
  - Cell
  - Value in cell

- **Geometry (Spatial)**
  - Position

- **Multidimensional Table**
  - Key 1
  - Key 2
  - Attributes
  - Value in cell

- **Trees**

[Munzner (ill. Maguire), 2014]
Data Terminology

• Items
  - An item is an individual discrete entity
  - e.g., a row in a table

• Attributes
  - An attribute is some specific property that can be measured, observed, or logged
    - a.k.a. variable, (data) dimension
  - e.g., a column in a table
# Tables

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>S</th>
<th>T</th>
<th>U</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Order ID</td>
<td>Order Date</td>
<td>Order Priority</td>
<td>Product Container</td>
</tr>
<tr>
<td>3</td>
<td>6</td>
<td>7</td>
<td>7</td>
<td>7</td>
<td>7</td>
</tr>
<tr>
<td>32</td>
<td>32</td>
<td>32</td>
<td>32</td>
<td>32</td>
<td>32</td>
</tr>
<tr>
<td>65</td>
<td>65</td>
<td>65</td>
<td>65</td>
<td>65</td>
<td>32</td>
</tr>
<tr>
<td>66</td>
<td>66</td>
<td>66</td>
<td>66</td>
<td>66</td>
<td>32</td>
</tr>
<tr>
<td>70</td>
<td>70</td>
<td>70</td>
<td>70</td>
<td>70</td>
<td>32</td>
</tr>
<tr>
<td>96</td>
<td>96</td>
<td>96</td>
<td>96</td>
<td>96</td>
<td>32</td>
</tr>
<tr>
<td>97</td>
<td>97</td>
<td>97</td>
<td>97</td>
<td>97</td>
<td>32</td>
</tr>
<tr>
<td>129</td>
<td>129</td>
<td>129</td>
<td>129</td>
<td>129</td>
<td>32</td>
</tr>
<tr>
<td>130</td>
<td>130</td>
<td>130</td>
<td>130</td>
<td>130</td>
<td>32</td>
</tr>
<tr>
<td>132</td>
<td>132</td>
<td>132</td>
<td>132</td>
<td>132</td>
<td>32</td>
</tr>
<tr>
<td>134</td>
<td>134</td>
<td>134</td>
<td>134</td>
<td>134</td>
<td>32</td>
</tr>
<tr>
<td>135</td>
<td>135</td>
<td>135</td>
<td>135</td>
<td>135</td>
<td>32</td>
</tr>
<tr>
<td>166</td>
<td>166</td>
<td>166</td>
<td>166</td>
<td>166</td>
<td>32</td>
</tr>
<tr>
<td>193</td>
<td>193</td>
<td>193</td>
<td>193</td>
<td>193</td>
<td>32</td>
</tr>
<tr>
<td>194</td>
<td>194</td>
<td>194</td>
<td>194</td>
<td>194</td>
<td>32</td>
</tr>
</tbody>
</table>

- **Item**: The row or column in the table.
- **Cell**: The specific cell within the table.
- **Attribute**: The table header or column heading.
Tables

Flat
- Data organized by rows & columns
  - row ~ item (usually)
  - column ~ attribute
  - label ~ attribute name
- Key: identifies each item (row)
  - Usually unique
  - Allows join of data from 2+ tables
  - Compound key: key split among multiple columns, e.g. (state, year) for population

Multidimensional
- Split compound key

[Munzner (ill. Maguire), 2014]
Attribute Types

- Categorical
  - [Icons: +, ., □, △]

- Ordered
  - Ordinal
  - Quantitative

[Munzner (ill. Maguire), 2014]
Categorial, Ordinal, and Quantitative

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>S</th>
<th>T</th>
<th>U</th>
</tr>
</thead>
<tbody>
<tr>
<td>Order ID</td>
<td>Order Date</td>
<td>Order Priority</td>
<td>Product Container</td>
<td>Product Base Margin</td>
<td>Ship Date</td>
</tr>
<tr>
<td>3</td>
<td>10/14/06</td>
<td>5-Low</td>
<td>Large Box</td>
<td>0.8</td>
<td>10/21/06</td>
</tr>
<tr>
<td>6</td>
<td>2/21/08</td>
<td>4-Not Specified</td>
<td>Small Pack</td>
<td>0.55</td>
<td>2/22/08</td>
</tr>
<tr>
<td>32</td>
<td>7/16/07</td>
<td>2-High</td>
<td>Small Pack</td>
<td>0.79</td>
<td>7/17/07</td>
</tr>
<tr>
<td>32</td>
<td>7/16/07</td>
<td>2-High</td>
<td>Jumbo Box</td>
<td>0.72</td>
<td>7/17/07</td>
</tr>
<tr>
<td>32</td>
<td>7/16/07</td>
<td>2-High</td>
<td>Medium Box</td>
<td>0.6</td>
<td>7/18/07</td>
</tr>
<tr>
<td>32</td>
<td>7/16/07</td>
<td>2-High</td>
<td>Medium Box</td>
<td>0.65</td>
<td>7/18/07</td>
</tr>
<tr>
<td>35</td>
<td>10/23/07</td>
<td>4-Not Specified</td>
<td>Wrap Bag</td>
<td>0.52</td>
<td>10/24/07</td>
</tr>
<tr>
<td>35</td>
<td>10/23/07</td>
<td>4-Not Specified</td>
<td>Small Box</td>
<td>0.58</td>
<td>10/25/07</td>
</tr>
<tr>
<td>36</td>
<td>11/3/07</td>
<td>1-Urgent</td>
<td>Small Box</td>
<td>0.55</td>
<td>11/3/07</td>
</tr>
<tr>
<td>65</td>
<td>3/18/07</td>
<td>1-Urgent</td>
<td>Small Pack</td>
<td>0.49</td>
<td>3/19/07</td>
</tr>
<tr>
<td>66</td>
<td>1/20/05</td>
<td>5-Low</td>
<td>Wrap Bag</td>
<td>0.56</td>
<td>1/20/05</td>
</tr>
<tr>
<td>69</td>
<td>6/4/05</td>
<td>4-Not Specified</td>
<td>Small Pack</td>
<td>0.44</td>
<td>6/6/05</td>
</tr>
<tr>
<td>69</td>
<td>6/4/05</td>
<td>4-Not Specified</td>
<td>Small Pack</td>
<td>0.6</td>
<td>6/6/05</td>
</tr>
<tr>
<td>70</td>
<td>12/18/06</td>
<td>5-Low</td>
<td>Large Box</td>
<td>0.59</td>
<td>12/23/06</td>
</tr>
<tr>
<td>70</td>
<td>12/18/06</td>
<td>5-Low</td>
<td>Medium Box</td>
<td>0.82</td>
<td>12/23/06</td>
</tr>
<tr>
<td>96</td>
<td>4/17/05</td>
<td>2-High</td>
<td>Small Box</td>
<td>0.55</td>
<td>4/19/05</td>
</tr>
<tr>
<td>97</td>
<td>1/29/06</td>
<td>3-Medium</td>
<td>Medium Box</td>
<td>0.38</td>
<td>1/30/06</td>
</tr>
<tr>
<td>129</td>
<td>11/19/08</td>
<td>5-Low</td>
<td>Large Box</td>
<td>0.37</td>
<td>11/28/08</td>
</tr>
<tr>
<td>130</td>
<td>5/8/08</td>
<td>2-High</td>
<td>Small Box</td>
<td>0.37</td>
<td>5/9/08</td>
</tr>
<tr>
<td>130</td>
<td>5/8/08</td>
<td>2-High</td>
<td>Medium Box</td>
<td>0.38</td>
<td>5/10/08</td>
</tr>
<tr>
<td>130</td>
<td>5/8/08</td>
<td>2-High</td>
<td>Small Box</td>
<td>0.6</td>
<td>5/11/08</td>
</tr>
<tr>
<td>132</td>
<td>6/11/06</td>
<td>3-Medium</td>
<td>Medium Box</td>
<td>0.6</td>
<td>6/12/08</td>
</tr>
<tr>
<td>132</td>
<td>6/11/06</td>
<td>3-Medium</td>
<td>Jumbo Box</td>
<td>0.69</td>
<td>6/14/06</td>
</tr>
<tr>
<td>134</td>
<td>5/1/08</td>
<td>4-Not Specified</td>
<td>Large Box</td>
<td>0.82</td>
<td>5/3/08</td>
</tr>
<tr>
<td>135</td>
<td>10/21/07</td>
<td>4-Not Specified</td>
<td>Small Pack</td>
<td>0.64</td>
<td>10/23/07</td>
</tr>
<tr>
<td>166</td>
<td>9/12/07</td>
<td>2-High</td>
<td>Small Box</td>
<td>0.55</td>
<td>9/14/07</td>
</tr>
<tr>
<td>193</td>
<td>8/8/06</td>
<td>1-Urgent</td>
<td>Medium Box</td>
<td>0.57</td>
<td>8/10/06</td>
</tr>
<tr>
<td>194</td>
<td>4/5/08</td>
<td>3-Medium</td>
<td>Wrap Bag</td>
<td>0.42</td>
<td>4/7/08</td>
</tr>
</tbody>
</table>
# Categorial, Ordinal, and Quantitative

<table>
<thead>
<tr>
<th>Order ID</th>
<th>Order Date</th>
<th>Order Priority</th>
<th>Product Type</th>
<th>Product Base Margin</th>
<th>Ship Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>10/14/06</td>
<td>Low</td>
<td>Large Box</td>
<td>0.8</td>
<td>10/21/06</td>
</tr>
<tr>
<td>6</td>
<td>2/21/08</td>
<td>Not Specified</td>
<td>Small Pack</td>
<td>0.55</td>
<td>2/22/08</td>
</tr>
<tr>
<td>32</td>
<td>7/16/07</td>
<td>High</td>
<td>Small Pack</td>
<td>0.79</td>
<td>7/17/07</td>
</tr>
<tr>
<td>32</td>
<td>7/16/07</td>
<td>High</td>
<td>Jumbo Box</td>
<td>0.72</td>
<td>7/17/07</td>
</tr>
<tr>
<td>32</td>
<td>7/16/07</td>
<td>High</td>
<td>Medium Box</td>
<td>0.6</td>
<td>7/18/07</td>
</tr>
<tr>
<td>32</td>
<td>7/16/07</td>
<td>High</td>
<td>Medium Box</td>
<td>0.65</td>
<td>7/18/07</td>
</tr>
<tr>
<td>35</td>
<td>10/23/07</td>
<td>Not Specified</td>
<td>Wrap Bag</td>
<td>0.52</td>
<td>10/24/07</td>
</tr>
<tr>
<td>35</td>
<td>10/23/07</td>
<td>Not Specified</td>
<td>Small Box</td>
<td>0.58</td>
<td>10/25/07</td>
</tr>
<tr>
<td>36</td>
<td>11/3/07</td>
<td>Urgent</td>
<td>Small Box</td>
<td>0.55</td>
<td>11/3/07</td>
</tr>
<tr>
<td>65</td>
<td>3/18/07</td>
<td>Urgent</td>
<td>Small Pack</td>
<td>0.49</td>
<td>3/19/07</td>
</tr>
<tr>
<td>66</td>
<td>1/20/05</td>
<td>Low</td>
<td>Wrap Bag</td>
<td>0.56</td>
<td>1/20/05</td>
</tr>
<tr>
<td>69</td>
<td>6/4/05</td>
<td>Not Specified</td>
<td>Small Pack</td>
<td>0.44</td>
<td>6/6/05</td>
</tr>
<tr>
<td>69</td>
<td>6/4/05</td>
<td>Not Specified</td>
<td></td>
<td>0.6</td>
<td>6/6/05</td>
</tr>
<tr>
<td>70</td>
<td>12/18/06</td>
<td>Low</td>
<td></td>
<td>0.59</td>
<td>12/23/06</td>
</tr>
<tr>
<td>70</td>
<td>12/18/06</td>
<td>Low</td>
<td></td>
<td>0.59</td>
<td>12/23/06</td>
</tr>
<tr>
<td>96</td>
<td>4/17/05</td>
<td>High</td>
<td></td>
<td>0.55</td>
<td>4/19/05</td>
</tr>
<tr>
<td>97</td>
<td>1/29/06</td>
<td>Medium</td>
<td></td>
<td>0.38</td>
<td>1/30/06</td>
</tr>
<tr>
<td>129</td>
<td>11/19/08</td>
<td>Low</td>
<td></td>
<td>0.37</td>
<td>11/28/08</td>
</tr>
<tr>
<td>130</td>
<td>5/8/08</td>
<td>High</td>
<td>Small Box</td>
<td>0.37</td>
<td>5/9/08</td>
</tr>
<tr>
<td>130</td>
<td>5/8/08</td>
<td>High</td>
<td>Medium Box</td>
<td>0.38</td>
<td>5/10/08</td>
</tr>
<tr>
<td>130</td>
<td>5/8/08</td>
<td>High</td>
<td>Small Box</td>
<td>0.6</td>
<td>5/11/08</td>
</tr>
<tr>
<td>132</td>
<td>6/11/06</td>
<td>Medium</td>
<td>Medium Box</td>
<td>0.6</td>
<td>6/12/06</td>
</tr>
<tr>
<td>132</td>
<td>6/11/06</td>
<td>Medium</td>
<td>Jumbo Box</td>
<td>0.69</td>
<td>6/14/06</td>
</tr>
<tr>
<td>134</td>
<td>5/1/08</td>
<td>Not Specified</td>
<td>Large Box</td>
<td>0.82</td>
<td>5/3/08</td>
</tr>
<tr>
<td>135</td>
<td>10/21/07</td>
<td>Not Specified</td>
<td>Small Box</td>
<td>0.64</td>
<td>10/23/07</td>
</tr>
<tr>
<td>166</td>
<td>9/12/07</td>
<td>High</td>
<td>Small Box</td>
<td>0.55</td>
<td>9/14/07</td>
</tr>
<tr>
<td>193</td>
<td>8/8/06</td>
<td>Urgent</td>
<td>Medium Box</td>
<td>0.57</td>
<td>8/10/06</td>
</tr>
<tr>
<td>194</td>
<td>4/5/08</td>
<td>Medium</td>
<td>Wrap Bag</td>
<td>0.42</td>
<td>4/7/08</td>
</tr>
</tbody>
</table>
Attribute Types

• May be further specified for computational storage/processing
  - **Categorical**: string, boolean, blood type
  - **Ordered**: enumeration, t-shirt size
  - **Quantitative**: integer, float, fixed decimal, datetime

• Sometimes, types can be **inferred** from the data
  - e.g. numbers and none have decimal points → integer
  - could be incorrect (data doesn't have floats, but could be)
Ordering Direction

- Sequential
- Diverging
- Cyclic

[Munzner (ill. Maguire), 2014]
Sequential and Diverging Data

• Sequential: homogenous range from a minimum to a maximum
  - Examples: Land elevations, ocean depths
• Diverging: can be deconstructed into two sequences pointing in opposite directions
  - Has a zero point (not necessary 0)
  - Example: Map of both land elevation and ocean depth

[Rogowitz & Treinish, 1998]
Sequential and Diverging Data

• Sequential: homogenous range from a minimum to a maximum
  - Examples: Land elevations, ocean depths

• Diverging: can be deconstructed into two sequences pointing in opposite directions
  - Has a zero point (not necessary 0)
  - Example: Map of both land elevation and ocean depth

[Rogowitz & Treinish, 1998]
Cyclic Data
Cyclic Data

The left graph shows a bar chart of temperature over the months from January to December for three years: 2015, 2017, and 2019. The vertical axis represents temperature, while the horizontal axis represents the months.

The right graph is a radar chart that displays the temperature for each month across the years 2015, 2017, and 2019. The chart uses different colors for each year to distinguish them.
Semantics

- The meaning of the data
- Example: 94023, 90210, 02747, 60115
Semantics

• The meaning of the data
• Example: 94023, 90210, 02747, 60115
  - Attendance at college football games?
Semantics

• The meaning of the data
• Example: 94023, 90210, 02747, 60115
  - Attendance at college football games?
  - Salaries?
Semantics

- The meaning of the data
- Example: 94023, 90210, 02747, 60115
  - Attendance at college football games?
  - Salaries?
  - Zip codes?
- Cannot always infer based on what the data looks like
- Often require semantics to better understand data
- Column names help with semantics
- May also include rules about data: a zip code is part of an address that uniquely identifies a residence
Data Model vs. Conceptual Model

• Data Model: raw data that has a specific data type (e.g. floats):
  - Temperature Example: [32.5, 54.0, -17.3] (floats)

• Conceptual Model: how we think about the data
  - Includes semantics, reasoning
  - Temperature Example:
    • Quantitative: [32.50, 54.00, -17.30]
Data Model vs. Conceptual Model

• Data Model: raw data that has a specific data type (e.g. floats):
  - Temperature Example: $[32.5, 54.0, -17.3]$ (floats)

• Conceptual Model: how we think about the data
  - Includes semantics, reasoning
  - Temperature Example:
    • Quantitative: $[32.50, 54.00, -17.30]$
    • Ordered: [warm, hot, cold]

[via A. Lex, 2015]
Data Model vs. Conceptual Model

- Data Model: raw data that has a specific data type (e.g. floats):
  - Temperature Example: [32.5, 54.0, -17.3] (floats)
- Conceptual Model: how we think about the data
  - Includes semantics, reasoning
  - Temperature Example:
    - Quantitative: [32.50, 54.00, -17.30]
    - Ordered: [warm, hot, cold]
    - Categorical: [not burned, burned, not burned]
Derived Data
Derived Data

- Often, data in its original form isn't as useful as we would like
- Examples: Data about a basketball team's games
Derived Data

- Often, data in its original form isn't as useful as we would like
- Examples: Data about a basketball team's games
  - Example 1: $1stHalfPoints, 2ndHalfPoints$
    - More useful to know total number of points
    - $Points = 1stHalfPoints + 2ndHalfPoints$
Derived Data

- Often, data in its original form isn't as useful as we would like
- Examples: Data about a basketball team's games
- Example 1: 1stHalfPoints, 2ndHalfPoints
  - More useful to know total number of points
  - Points = 1stHalfPoints + 2ndHalfPoints
- Example 2: Points, OpponentPoints
  - Want to have a column indicating win/loss
  - Win = True if (Points > OpponentPoints) else False
Derived Data

- Often, data in its original form isn't as useful as we would like
- Examples: Data about a basketball team's games
- Example 1: 1stHalfPoints, 2ndHalfPoints
  - More useful to know total number of points
  - Points = 1stHalfPoints + 2ndHalfPoints
- Example 2: Points, OpponentPoints
  - Want to have a column indicating win/loss
  - Win = True if (Points > OpponentPoints) else False
- Example 3: Points
  - Want to have a column indicating how that point total ranks
  - Rank = index in sorted list of all Point values
pandas

- Contains high-level data structures and manipulation tools designed to make data analysis fast and easy in Python
- Built on top of NumPy
- Requirements:
  - Data structures with labeled axes (aligning data)
  - Time series data
  - Arithmetic operations that include metadata (labels)
  - Handle missing data
  - Merge and relational operations
Pandas Code Conventions

• Universal:
  - import pandas as pd

• Also used:
  - from pandas import Series, DataFrame
Series

- A one-dimensional array (with a type) with an **index**
- Index defaults to numbers but can also be text (like a dictionary)
- Allows easier reference to specific items
- `obj = pd.Series([7, 14, -2, 1])`
- Basically two arrays: `obj.values` and `obj.index`
- Can specify the index explicitly and use strings
  - `obj2 = pd.Series([4, 7, -5, 3],
    index=['d', 'b', 'a', 'c'])`
- Kind of like fixed-length, ordered dictionary + can create from a dictionary
  - `obj3 = pd.Series({'Ohio': 35000, 'Texas': 71000,
    'Oregon': 16000, 'Utah': 5000})`
Series

- Indexing: `s[1]` or `s['Oregon']`
- Can check for missing data: `pd.isnull(s)` or `pd.notnull(s)`
- Both index and values can have an associated name:
  - `s.name = 'population'; s.index.name = 'state'`
- Addition and NumPy ops work as expected and preserve the index-value link
- These operations **align**:

  In [28]: obj3
  Out[28]:
      Ohio   35000
      Oregon  16000
      Texas   71000
      Utah     5000
  dtype: int64

  In [29]: obj4
  Out[29]:
      California   NaN
      Ohio         35000
      Oregon      16000
      Texas       71000
  dtype: float64

  In [30]: obj3 + obj4
  Out[30]:
      California   NaN
      Ohio         70000
      Oregon      32000
      Texas      142000
      Utah          NaN
  dtype: float64

[W. McKinney, Python for Data Analysis]
Data Frame

- A dictionary of Series (labels for each series)
- A spreadsheet with column headers
- Has an index shared with each series
- Allows easy reference to any cell
- \( \text{df} = \text{DataFrame}(\{\text{'state': ['Ohio', 'Ohio', 'Ohio', 'Nevada'], 'year': [2000, 2001, 2002, 2001], 'pop': [1.5, 1.7, 3.6, 2.4]}\}) \)

- Index is automatically assigned just as with a series but can be passed in as well via index kwarg
- Can reassign column names by passing columns kwarg
Examples

• See ch05.ipynb