Advanced Data Management (CSCI 490/680)

Review

Dr. David Koop
Final Exam

• Tuesday, May 5 from 4-5:50pm
• Online
• Similar format to Test 2
• Comprehensive but with more focus on last few weeks of class
• Contact me with questions:
  - Email
  - Setup a time to talk via Blackboard
Data systems rely on algorithms

[S. Idreos, 2019]
Data structures define performance

As time goes by, data structures become ever more critical for data driven applications. Data structures define performance. 

Jim Gray, Turing Award 1998

register = this room

memory = nearby city

caches = this city

disk = Pluto

[S. Idreos, 2019]
Every data structure design is simply a point in the design space of possible solutions. There is no perfect design. Every design balances the fundamental tradeoffs of Read, Update, and Memory amplification. For example, Read amplification is defined as the excess data an algorithm needs to read on top of the data it wants to read. Typically a data structure would have some kind of metadata or navigation data that help locate the actual data, e.g., the internal nodes of a B-tree. Reading this navigation data is an excess cost, adding to read amplification. Creating a data structure without any navigation data would suffer update or even more read amplification. For example, we could choose to not have any structure in the data at all. Then every query would have to touch all the data. The other extreme would be to sort all data which effectively provides an implicit structure. But then updates get expensive. Overall, there is no perfect design.

[S. Idreos, 2019]
Many efforts in the field have been motivated by the vision of generating tailored systems for a specific scenario. In fact, even traditional databases are architected with this vision in mind. A generic database system can optimize a plan on the fly to match the query needs, it can choose from different storage and indexing options, etc. This is how generic database systems can be used in a wealth of applications! And then recent research has tried to push the boundaries of tailored designs by rethinking parts of the stack of a database system.

"Traditional" Database Research

[S. Idreos, 2019]
Learned Data Structures and Algorithms
B-Tree

Key
(e.g., spoon #1)

Model

[T. Kraska, 2019]
Model to Predict Data's Location on Disk

- Frequency Distribution
- Cumulative Distribution Function (CDF)

\[ P(X<2017-11-27) \times N \]

[Diagram shows a graph with a red line representing the frequency distribution over dates.]

[T. Kraska, 2019]
Challenges

Traditional model architectures do not work

Frameworks are not designed for nano-second execution

Overfitting can be good

ML+System Co-Design

Overfitting desired

[Overfitting is not desired]
Recursive Model Index (RMI)

2-Stage RMI with Linear Model

\[
\begin{align*}
pos_0 &= a_0 + b_0 \times \text{key} \\
pos_1 &= m_1[pos_0].a + m_1[pos_0].b \times \text{key} \\
\text{record} &= \text{local-search(}\text{key, } pos_1\text{)}
\end{align*}
\]

[T. Kraska, 2019]
Sandwiched Bloom Filter

Is This **Key** In My Set?

- **No**
- **Maybe**
- **Yes**

- **Model**
  - **Maybe No**
  - **Maybe Yes**
  - **No**

[Michael Mitzenmacher, 2018 via T. Kraska, 2019]
Sorting

(a) CDF Model Pre-Sorts

(b) Compact & local sort

[T. Kraska, 2019]
Sorting

(a) CDF Model Pre-Sorts

(b) Compact & local sort

32-bit ints; normal distribution ($\mu=0$, $\sigma=1e6$)
More…

Tree, Multi-Dim Index, Bloom-Filter, Sorting, Scheduling, Range-Filter, Hash-Map

Data Cubes, DNA-Search, SQL Query Optimizer, Cache Policy, Join, Nearest Neighbor

[T. Kraska, 2019]
Review
What did we do this semester?
What's involved in dealing with data?

<table>
<thead>
<tr>
<th>Data Acquisition</th>
<th>Data Analysis</th>
<th>Data Curation</th>
<th>Data Storage</th>
<th>Data Usage</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Structured data</td>
<td>• Stream mining</td>
<td>• Data Quality</td>
<td>• In-Memory DBs</td>
<td>• Decision support</td>
</tr>
<tr>
<td>• Unstructured data</td>
<td>• Semantic analysis</td>
<td>• Trust / Provenance</td>
<td>• NoSQL DBs</td>
<td>• Prediction</td>
</tr>
<tr>
<td>• Event processing</td>
<td>• Machine learning</td>
<td>• Annotation</td>
<td>• NewSQL DBs</td>
<td>• In-use analytics</td>
</tr>
<tr>
<td>• Sensor networks</td>
<td>• Information extraction</td>
<td>• Data validation</td>
<td>• Cloud storage</td>
<td>• Simulation</td>
</tr>
<tr>
<td>• Protocols</td>
<td>• Linked Data</td>
<td>• Human-Data Interaction</td>
<td>• Query Interfaces</td>
<td>• Exploration</td>
</tr>
<tr>
<td>• Real-time</td>
<td>• Data discovery</td>
<td>• Top-down/Bottom-up</td>
<td>• Scalability and Performance</td>
<td>• Visualisation</td>
</tr>
<tr>
<td>• Data streams</td>
<td>• 'Whole world’ semantics</td>
<td>• Community / Crowd</td>
<td>• Data Models</td>
<td>• Modeling</td>
</tr>
<tr>
<td>• Multimodality</td>
<td>• Ecosystems</td>
<td>• Human Computation</td>
<td>• Consistency,</td>
<td>• Control</td>
</tr>
<tr>
<td></td>
<td>• Community data analysis</td>
<td>• Curation at scale</td>
<td>Availability,</td>
<td>• Domain-specific usage</td>
</tr>
<tr>
<td></td>
<td>• Cross-sectorial data analysis</td>
<td>• Incentivisation</td>
<td>Partition-tolerance</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Automation</td>
<td>• Security and Privacy</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Interoperability</td>
<td>• Standardization</td>
<td></td>
</tr>
</tbody>
</table>

[Big Data Value Chain, Curry et al., 2014]
Python!

• Just assign expressions to variables, no typing
  
  ```python
  a = 12
  a = "abc"
  b = a + "de"
  ```

• Functions defined using `def`, called using parenthesis:
  
  ```python
  def hello(name1="Joe", name2="Jane"):
      print(f"Hello {name1} and {name2}"
  hello(name2="Mary")
  ```

• Always indent blocks (if-else-elif, while, for, etc.):
  
  ```python
  z = 20
  if x > 0:
      if y > 0:
          z = 100
  else:
      z = 10
  ```
Python Containers

- **List:** [1, "abc", 12.34]
- **Tuple:** (1, "abc", 12.34)
- **Indexing/Slicing:**
  - x[0], x[:-1], x[1:2], x[::2]
- **Set:** {1, "abc", 12.34}
- **Dictionary:** {'x': 1, 'y': "abc", 'z': 12.34}
- **Mutable vs. Immutable**
- **Stored by reference**
- **Iterators:** objects that traverse containers, just know how to get next element
- **You cannot index/slice an iterator** (d.values()[-1] doesn't work)
Comprehensions

• List Comprehensions:
  - squares = [i**2 for i in range(10)]

• Dictionary Comprehensions:
  - squares = {i: i**2 for i in range(10)}

• Set Comprehensions:
  - squares = {i**2 for i in range(10)}

• Comprehensions allow filters:
  - squares = [i**2 for i in range(10) if i % 2 == 0]
JupyterLab

- An interactive, configurable programming environment
- Supports many activities including notebooks
- Runs in your web browser
- Notebooks:
  - Originally designed for Python
  - Supports other languages, too
  - Displays results (even interactive maps) inline
  - You decide how to divide code into executable cells
  - Shift+Enter to execute a cell
Figure 4-1. Indexing elements in a NumPy array

In multidimensional arrays, if you omit later indices, the returned object will be a lower dimensional ndarray consisting of all the data along the higher dimensions. So in the 2×2×3 array \( \text{arr3d} \):

\[
\text{arr3d} = \begin{bmatrix}
\begin{bmatrix} 1, & 2, & 3 \\
4, & 5, & 6 \\
7, & 8, & 9 \\
10, & 11, & 12 \\
\end{bmatrix}
\end{bmatrix}
\]

\[
\text{arr3d}[0] \text{ is a 2×3 array:}
\]

\[
\text{arr3d}[0] = \begin{bmatrix}
1, & 2, & 3 \\
4, & 5, & 6 \\
\end{bmatrix}
\]

Both scalar values and arrays can be assigned to \( \text{arr3d}[0] \):

\[
\text{old_values} = \text{arr3d}[0].\text{copy}()
\]

\[
\text{arr3d}[0] = 42
\]

\[
\text{arr3d}
\]

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

• names == 'Bob' gives back booleans that represent the element-wise comparison with the array names

• Boolean arrays can be used to index into another array:
  - data[names == 'Bob']

• Can even mix and match with integer slicing

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

• Note: or and and do not work with arrays

• We can set values too! data[data < 0] = 0
What is Data?

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

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

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

- **Geometry (Spatial)**
  - Position

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

- **Trees**

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

- **Quantitative**
- **Ordinal**
- **Categorical**

<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>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>7/17/07</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>0.59</td>
<td>12/23/06</td>
<td></td>
</tr>
<tr>
<td>70</td>
<td>12/18/06</td>
<td>5-Low</td>
<td>0.59</td>
<td>12/23/06</td>
<td></td>
</tr>
<tr>
<td>96</td>
<td>4/17/05</td>
<td>2-High</td>
<td>0.55</td>
<td>4/19/05</td>
<td></td>
</tr>
<tr>
<td>97</td>
<td>1/29/06</td>
<td>3-Medium</td>
<td>0.38</td>
<td>1/30/06</td>
<td></td>
</tr>
<tr>
<td>129</td>
<td>11/19/08</td>
<td>5-Low</td>
<td>0.37</td>
<td>11/28/08</td>
<td></td>
</tr>
<tr>
<td>130</td>
<td>5/8/08</td>
<td>2-High</td>
<td>0.37</td>
<td>5/9/08</td>
<td></td>
</tr>
<tr>
<td>130</td>
<td>5/8/08</td>
<td>2-High</td>
<td>0.38</td>
<td>5/10/08</td>
<td></td>
</tr>
<tr>
<td>130</td>
<td>5/8/08</td>
<td>2-High</td>
<td>0.6</td>
<td>5/11/08</td>
<td></td>
</tr>
<tr>
<td>132</td>
<td>6/11/06</td>
<td>3-Medium</td>
<td>0.6</td>
<td>6/12/06</td>
<td></td>
</tr>
<tr>
<td>132</td>
<td>6/11/06</td>
<td>3-Medium</td>
<td>0.6</td>
<td>6/14/06</td>
<td></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>0.55</td>
<td>9/14/07</td>
<td></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>
### Pandas and Data Frames

- Data Frames are tables with many database-like operations
- Index shared across all columns
- Can select, project, merge (join), and more
- Read and write many file formats

```
In [1]: import pandas as pd
In [2]: # read the dataset using pandas
df = pd.read_csv("Food_Inspections.csv")
In [ ]: # look at the dataset, nice table formatting
df
In [ ]: # just the beginning of the dataset
df.head()
```

<table>
<thead>
<tr>
<th>Inspection ID</th>
<th>DBA Name</th>
<th>AKA Name</th>
<th>License #</th>
<th>Facility Type</th>
<th>Risk</th>
<th>Address</th>
<th>City</th>
<th>State</th>
<th>Zip</th>
<th>Inspection Date</th>
<th>Inspection Type</th>
<th>Results</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>UNCOOKED LLC</td>
<td>UNCOOKED LLC</td>
<td>2709319.0</td>
<td>NaN</td>
<td>All</td>
<td>210 N CARPENTER ST</td>
<td>CHICAGO</td>
<td>IL</td>
<td>60607.0</td>
<td>01/13/2020</td>
<td>License</td>
<td>Not Ready</td>
</tr>
<tr>
<td>1</td>
<td>MOJO 33 NORTH LASALLE LLC</td>
<td>MOJO 33 NORTH LASALLE LLC</td>
<td>2689550.0</td>
<td>Restaurant</td>
<td>Risk 1 (High)</td>
<td>33 N LA SALLE ST</td>
<td>CHICAGO</td>
<td>IL</td>
<td>60602.0</td>
<td>01/13/2020</td>
<td>License-Inspection</td>
<td>Pass</td>
</tr>
<tr>
<td>2</td>
<td>LA BIZNAGA #2</td>
<td>LA BIZNAGA #2</td>
<td>2708992.0</td>
<td>NaN</td>
<td>Risk 1 (High)</td>
<td>2949 W BELMONT AVE</td>
<td>CHICAGO</td>
<td>IL</td>
<td>60618.0</td>
<td>01/10/2020</td>
<td>License</td>
<td>Not Ready</td>
</tr>
<tr>
<td>3</td>
<td>LAS TABLAS</td>
<td>LAS TABLAS</td>
<td>1617900.0</td>
<td>Restaurant</td>
<td>Risk 1 (High)</td>
<td>4920 W IRVING PARK RD</td>
<td>CHICAGO</td>
<td>IL</td>
<td>60641.0</td>
<td>01/09/2020</td>
<td>Canvas</td>
<td>Pass</td>
</tr>
<tr>
<td>4</td>
<td>GIORDANO’S OF BEVERLY</td>
<td>GIORDANO’S OF BEVERLY</td>
<td>2074456.0</td>
<td>Restaurant</td>
<td>Risk 1 (High)</td>
<td>9613 S WESTERN AVE</td>
<td>CHICAGO</td>
<td>IL</td>
<td>60643.0</td>
<td>01/09/2020</td>
<td>Canvas</td>
<td>Pass</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>199687</td>
<td>PANDA EXPRESS #236</td>
<td>PANDA EXPRESS #236</td>
<td>1801495.0</td>
<td>Restaurant</td>
<td>Risk 1 (High)</td>
<td>77 W JACKSON BLVD</td>
<td>CHICAGO</td>
<td>IL</td>
<td>60604.0</td>
<td>02/18/2010</td>
<td>Suspected Food Poisoning</td>
<td>Pass</td>
</tr>
<tr>
<td>199688</td>
<td>UNCLE JOE’S</td>
<td>81083.0</td>
<td>Restaurant</td>
<td>Risk 1 (High)</td>
<td>1453 E HYDE PARK BLVD</td>
<td>CHICAGO</td>
<td>IL</td>
<td>60615.0</td>
<td>02/08/2010</td>
<td>Complaint</td>
<td>Pass</td>
<td></td>
</tr>
<tr>
<td>199689</td>
<td>Cafe Marbella</td>
<td>Cafe Marbella</td>
<td>2016764.0</td>
<td>Restaurant</td>
<td>Risk 1 (High)</td>
<td>5527-5531 N Milwaukee AVE</td>
<td>CHICAGO</td>
<td>IL</td>
<td>60630.0</td>
<td>01/28/2010</td>
<td>License-Inspection</td>
<td>Pass</td>
</tr>
<tr>
<td>199690</td>
<td>WALGREENS # 07876</td>
<td>WALGREENS # 07876</td>
<td>2004292.0</td>
<td>Grocery Store</td>
<td>Risk 3 (Low)</td>
<td>7544 S STONY ISLAND AVE</td>
<td>CHICAGO</td>
<td>IL</td>
<td>60649.0</td>
<td>02/18/2010</td>
<td>TASK FORCE LIQUOR 1474</td>
<td>Pass</td>
</tr>
<tr>
<td>199691</td>
<td>YSABEL’S FILIPINO CUISINE</td>
<td>YSABEL’S GRILL ASIAN CUISINE</td>
<td>2013419.0</td>
<td>Restaurant</td>
<td>Risk 1 (High)</td>
<td>4908 W Irving Park RD</td>
<td>CHICAGO</td>
<td>IL</td>
<td>60641.0</td>
<td>01/12/2010</td>
<td>License-Inspection</td>
<td>Pass</td>
</tr>
</tbody>
</table>

199692 rows x 17 columns
FINDINGS

we got about the future of the data science, the most salient takeaway was how excited our respondents were about the evolution of the \textit{CHOG7KH\FLWHGWKLQJVLQWKHLURZQSUDFWLFHKRZ} they saw their jobs getting more interesting and less repetitive, all while expressing a real and broad enthusiasm about the value of the work in their organization.

$VGDWDVFLHQFHEHFRPHVPRUHFRPPRQSODFHDQG$ \textit{VLPXOWDQHRXVO\DELWGHP\VWLĆHGZHH}[SHFWWKLV\textit{WUHQGWRFRQWLQXHDVZHOO$IWHUDOOODVW\HDUèV$ respondents were just as excited about their \textit{ZRUNDERXWZHUHêVDWLVĆHGëRUEHWWHU} How a Data Scientist Spends Their Day

$+$HUHèVZKHUHWKHSRSXODUYLHZRIGDWDVFLHQWLVWVGLYHUJHVSUHWW\VLJQLĆFDQWO\IURPUHDOLW\*HQHUDOO\ZHWKLQNRIGDWDVFLHQWLVWVEXLOGLQJDOJRULWKPVH[SORULQJGDWDDQGGRLQJSUHGLFWLYHDQDO\VLV7KDWèV$ actually not what they spend most of their time doing, however.

$\textit{V}	extit{RXFDQVHHIURPWKHFKDUWDERYHRXWRIHYHU\GDWDVFLHQWLVWèVZRUNćRZ\textit{QGQHDUO\VDLGWKH\VLPSO\VSHQWWRRPXFK}$

$\textit{time doing it.}$

\textbf{Data scientist job satisfaction}

<table>
<thead>
<tr>
<th>Activity</th>
<th>%</th>
</tr>
</thead>
<tbody>
<tr>
<td>Building training sets</td>
<td>3</td>
</tr>
<tr>
<td>Cleaning and organizing data</td>
<td>60</td>
</tr>
<tr>
<td>Collecting data sets</td>
<td>19</td>
</tr>
<tr>
<td>Mining data for patterns</td>
<td>9</td>
</tr>
<tr>
<td>Refining algorithms</td>
<td>4</td>
</tr>
<tr>
<td>Other</td>
<td>5</td>
</tr>
</tbody>
</table>

\textbf{How do data scientists spend their time?}

\textbf{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%

[CrowdFlower Data Science Report, 2016]
# Data Wrangling

## Customer Analysis

### Customer Random

<table>
<thead>
<tr>
<th>Preview</th>
<th>IMSI</th>
<th>CONTRACT_END</th>
<th>CONTRACT_START</th>
<th>SUBSCRIBER_AGE</th>
<th>STATUS</th>
</tr>
</thead>
<tbody>
<tr>
<td>- 310178226812721</td>
<td>6/4/16</td>
<td>7/29/89</td>
<td>ACTIVE</td>
<td></td>
<td></td>
</tr>
<tr>
<td>- 310180980766790</td>
<td>3/28/15</td>
<td>16/6/13</td>
<td>ACTIVE</td>
<td></td>
<td></td>
</tr>
<tr>
<td>- 31017854622541</td>
<td>9/23/16</td>
<td>1/9/87</td>
<td>7</td>
<td></td>
<td></td>
</tr>
<tr>
<td>- 310005423849230</td>
<td>5/29/15</td>
<td>1/24/81</td>
<td>13</td>
<td></td>
<td></td>
</tr>
<tr>
<td>- 3100269379121005</td>
<td>9/11/15</td>
<td>9/18/10</td>
<td>4</td>
<td></td>
<td></td>
</tr>
<tr>
<td>- 310062915466952</td>
<td>8/27/15</td>
<td>3/13/86</td>
<td>8</td>
<td></td>
<td></td>
</tr>
<tr>
<td>- 310178484724861</td>
<td>1/16/16</td>
<td>5/11/84</td>
<td>ACTIVE</td>
<td></td>
<td></td>
</tr>
<tr>
<td>- 310187875644871</td>
<td>85-Jul-2011</td>
<td>9/11/86</td>
<td>INACTIVE</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- 310260318424556 | 12/24/15 | 3/28/81 | 13 |
- 31058384595817 | 3/6/15 | 7/26/08 | 14 |
- 310184644252516 | 9/25/15 | 4/4/84 | 10 |
- 310284358780772 | 4/38/16 | 9/8/84 | 10 |
- 310288195729676 | 1/16/15 | 1/3/84 | 11 |
- 31006261822880 | 8/13/13 | 11/23/80 | INACTIVE |
- 310005678802848 | 8/4/16 | 18/22/14 | INACTIVE |
- 31018784820184 | 1/22/15 | 18/19/14 | INACTIVE |
- 31018877267782 | 11/21/15 | 12/28/14 |
- 31017812654014 | 21-Sept-2011 | 9/29/89 | INACTIVE |
- 31006211842337 | 5/29/15 | 3/29/85 | 9 |
- 310026881676970 | 11/17/16 | 5/21/87 | 7 |
- 310084436363016 | 9/15/16 | 7/24/11 | ACTIVE |
- 31028239999542 | 2/27/15 | 6/29/11 | 3 |
- 31018877394729 | 4/28/16 | 6/15/84 | 10 |
- 310032925559214 | 2/7/15 | 3/24/12 | 2 |
- 3101821258888547 | 13-Jan-2009 | 12/18/85 | 3 |
- 31012387860694 | 18/1/16 | 18/25/11 | 3 |

### Pattern Details CONTRACT_END

#### Hide Example Values

- 9/19/15
- 6/13/15
- 5/21/15
- 12/13/15
- 1/16/16

#### 12.65k
- 5.73k
- 1.99k

---

[Trifacta]
Foofah: Programming by Example

Motivation

Background knowledge for data transformation

Our goal: minimize depth of data transformation knowledge from the user.

Most real-world data transformation tasks often requires a user's effort and reduce the required effort and reduce the required knowledge from the user.

Example

• Program to synthesize:
  • Raw Data:
  • Transformed Data:

Input Example

Output Example

Test Data

Transformed Data

Synthesized Data Transformation Program in Python

[Z. Jin et al., 2017]
TDE: Transform Data by Example

<table>
<thead>
<tr>
<th>C</th>
<th>D</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Transaction Date</strong></td>
<td><strong>output</strong></td>
</tr>
<tr>
<td>Wed, 12 Jan 2011</td>
<td>2011-01-12-Wednesday</td>
</tr>
<tr>
<td>Thu, 15 Sep 2011</td>
<td>2011-09-15-Thursday</td>
</tr>
<tr>
<td>Mon, 17 Sep 2012</td>
<td>2012-09-17-Monday</td>
</tr>
<tr>
<td>2010-Nov-30 11:10:41</td>
<td>2010-11-30-Tuesday</td>
</tr>
<tr>
<td>2011-Jan-11 02:27:21</td>
<td>2011-01-11-Tuesday</td>
</tr>
<tr>
<td>2011-Jan-12</td>
<td>2011-01-12-Wednesday</td>
</tr>
<tr>
<td>2010-Dec-24</td>
<td>2010-12-24-Friday</td>
</tr>
<tr>
<td>9/22/2011</td>
<td>2011-09-22-Thursday</td>
</tr>
<tr>
<td>7/11/2012</td>
<td>2012-07-11-Wednesday</td>
</tr>
<tr>
<td>2/12/2012</td>
<td>2012-02-12-Sunday</td>
</tr>
</tbody>
</table>

[Y. He et al., 2018]
### Tidy Data

<table>
<thead>
<tr>
<th></th>
<th>treatmenta</th>
<th>treatmentb</th>
</tr>
</thead>
<tbody>
<tr>
<td>John Smith</td>
<td>—</td>
<td>2</td>
</tr>
<tr>
<td>Jane Doe</td>
<td>16</td>
<td>11</td>
</tr>
<tr>
<td>Mary Johnson</td>
<td>3</td>
<td>1</td>
</tr>
</tbody>
</table>

**Initial Data**

<table>
<thead>
<tr>
<th></th>
<th>treatmenta</th>
<th>treatmentb</th>
</tr>
</thead>
<tbody>
<tr>
<td>John Smith</td>
<td>—</td>
<td>2</td>
</tr>
<tr>
<td>Jane Doe</td>
<td>16</td>
<td>11</td>
</tr>
<tr>
<td>Mary Johnson</td>
<td>3</td>
<td>1</td>
</tr>
</tbody>
</table>

**Transpose**

<table>
<thead>
<tr>
<th></th>
<th>name</th>
<th>trt</th>
<th>result</th>
</tr>
</thead>
<tbody>
<tr>
<td>John Smith</td>
<td>a</td>
<td>—</td>
<td></td>
</tr>
<tr>
<td>Jane Doe</td>
<td>a</td>
<td>16</td>
<td></td>
</tr>
<tr>
<td>Mary Johnson</td>
<td>a</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>John Smith</td>
<td>b</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>Jane Doe</td>
<td>b</td>
<td>11</td>
<td></td>
</tr>
<tr>
<td>Mary Johnson</td>
<td>b</td>
<td>1</td>
<td></td>
</tr>
</tbody>
</table>
MultIndex Row Access and Slicing

- Remember that `loc` uses the index values, `iloc` uses integers
- **Note**: `df.iloc[0]` gets the first row, **not** `df.iloc[0,0]`
- Can get a subset of the data using partial indices
  - `df.loc["Boston"]` returns both 2007 and 2008 data
- What about slicing?
  - `df.loc["Boston":"Cleveland"]` → ERROR! (Need sorted data)
  - `df = df.sort_index()`
  - `df.loc["Boston":"Cleveland"]` → inclusive!
  - `df.loc[(slice("Boston","Cleveland"),2007),:]`
Merges (aka Joins)

- Need to merge data from one DataFrame with data from another DataFrame
- Example: Football game data merged with temperature data

### Game

<table>
<thead>
<tr>
<th>Id</th>
<th>Location</th>
<th>Date</th>
<th>Home</th>
<th>Away</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Boston</td>
<td>9/2</td>
<td>1</td>
<td>15</td>
</tr>
<tr>
<td>1</td>
<td>Boston</td>
<td>9/9</td>
<td>1</td>
<td>7</td>
</tr>
<tr>
<td>2</td>
<td>Cleveland</td>
<td>9/16</td>
<td>12</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>San Diego</td>
<td>9/23</td>
<td>21</td>
<td>1</td>
</tr>
</tbody>
</table>

### Weather

<table>
<thead>
<tr>
<th>wld</th>
<th>City</th>
<th>Date</th>
<th>Temp</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Boston</td>
<td>9/2</td>
<td>72</td>
</tr>
<tr>
<td>1</td>
<td>Boston</td>
<td>9/3</td>
<td>68</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>7</td>
<td>Boston</td>
<td>9/9</td>
<td>75</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>21</td>
<td>Boston</td>
<td>9/23</td>
<td>54</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>36</td>
<td>Cleveland</td>
<td>9/16</td>
<td>81</td>
</tr>
</tbody>
</table>

No data for San Diego
## Inner Strategy

### Merged

<table>
<thead>
<tr>
<th>Id</th>
<th>Location</th>
<th>Date</th>
<th>Home</th>
<th>Away</th>
<th>Temp</th>
<th>wld</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Boston</td>
<td>9/2</td>
<td>1</td>
<td>15</td>
<td>72</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>Boston</td>
<td>9/9</td>
<td>1</td>
<td>7</td>
<td>75</td>
<td>7</td>
</tr>
<tr>
<td>2</td>
<td>Cleveland</td>
<td>9/16</td>
<td>12</td>
<td>1</td>
<td>81</td>
<td>36</td>
</tr>
</tbody>
</table>

No San Diego entry
### Outer Strategy

#### Merged

<table>
<thead>
<tr>
<th>Id</th>
<th>Location</th>
<th>Date</th>
<th>Home</th>
<th>Away</th>
<th>Temp</th>
<th>wId</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Boston</td>
<td>9/2</td>
<td>1</td>
<td>15</td>
<td>72</td>
<td>0</td>
</tr>
<tr>
<td>NaN</td>
<td>Boston</td>
<td>9/3</td>
<td>NaN</td>
<td>NaN</td>
<td>68</td>
<td>1</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>1</td>
<td>Boston</td>
<td>9/9</td>
<td>1</td>
<td>7</td>
<td>75</td>
<td>7</td>
</tr>
<tr>
<td>NaN</td>
<td>Boston</td>
<td>9/10</td>
<td>NaN</td>
<td>NaN</td>
<td>76</td>
<td>8</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>NaN</td>
<td>Cleveland</td>
<td>9/2</td>
<td>NaN</td>
<td>NaN</td>
<td>61</td>
<td>22</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>2</td>
<td>Cleveland</td>
<td>9/16</td>
<td>12</td>
<td>1</td>
<td>81</td>
<td>36</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>3</td>
<td>San Diego</td>
<td>9/23</td>
<td>21</td>
<td>1</td>
<td>NaN</td>
<td>NaN</td>
</tr>
</tbody>
</table>
Data Integration

```
select title, startTime
from Movie, Plays
where Movie.title=Plays.movie AND
    location="New York" AND
    director="Woody Allen"
```

Sources S1 and S3 are relevant, sources S4 and S5 are irrelevant, and source S2 is relevant but possibly redundant.

[AH Doan et al., 2012]
Information Integration

Source A

Source B

Schema Mapping  Data Transformation  Duplicate Detection  Data Fusion

[L. Dong and F. Naumann, 2009]
Information Integration

[L. Dong and F. Naumann, 2009]

Source A

Source B

Schema Mapping

Data Transformation

Duplicate Detection

Data Fusion

Schema Integration

The diagram illustrates the process of information integration, involving schema mapping, data transformation, duplicate detection, and data fusion. It highlights the integration of data from different sources (Source A and Source B) through schema mapping and integration, followed by data transformation and subsequent steps to manage duplicate data and fuse the information.

The references to publications include:

1. Federated Database Systems
   - Title: Federated Database Systems for Managing Distributed, Heterogeneous, and Autonomous Databases
   - Author(s): Amit Sheth & James Larson
   - Year: 1990

2. Another publication
   - Title: Federated Database Systems
   - Author(s): Smith & Larson
   - Year: 1990
Information Integration

- Schema Mapping
- Data Transformation
- Duplicate Detection
- Data Fusion

Source A
- Federated Database Systems
- Authors: Amit Sheth, James Larson

Source B
- Federated Database Systems for Managing Distributed, Heterogeneous, and Autonomous Databases
- Authors: Scheth & Larson

Transformation queries or views

[L. Dong and F. Naumann, 2009]
Information Integration

Source A

Federated Database Systems

Schema Mapping

Data Transformation

Duplicate Detection

Data Fusion

Source B

Federated Database Systems for Managing Distributed, Heterogeneous, and Autonomous Databases

[L. Dong and F. Naumann, 2009]
Information Integration

Source A

Schema Mapping

Data Transformation

Duplicate Detection

Data Fusion

Preserve lineage

Source B

[L. Dong and F. Naumann, 2009]
Figure 1: An overview of the Dataset Search components. Google crawler collects the metadata from the Web; Dataset Search backend normalizes and reconciles the metadata; we then index the reconciled metadata and rank results for user queries.

We then look for the triples that use our vocabularies of interest, Schema.org and DCAT. Specifically, we collect all the triples for all the pages that have elements of specific types: http://schema.org/Dataset, http://schema.org/DataCatalog, and http://www.w3.org/ns/dcat#Dataset.

For a set of triples from each page, we traverse the graph to collect all the properties and related objects for each dataset in a protocol buffer. A dataset record can point to other records such as organizations that provided a dataset or a record describing the distribution of a dataset. A single Web page can have multiple dataset records on it.

The specification of the graph traversal captures the mapping from Schema.org and DCAT vocabularies to the corresponding elements in the protocol buffer definition (e.g., example fields in Figure 2). The schema of the protocol buffer for the metadata largely corresponds to http://schema.org/Dataset and therefore the transformation of metadata at this stage is rather small.

To improve scalability, we use the graph query independently on the triples from each individual page rather than try to extract information from a graph that includes all metadata triples on the Web. Because the links across different pages must specify objects on another page directly through a URL (e.g., a provider of this dataset on page A is described on page B), we can do this reconciliation post-hoc. So, essentially, each page corresponds to its own, possibly disconnected graph. At the same time, doing graph traversal only for a single page is dramatically more scalable.

The information that we extract through graph traversal constitutes the raw metadata, metadata that closely mimics the structure of Schema.org properties on the original page. In the next few steps, we describe how we create reconciled metadata for each dataset, accounting for the different levels of quality and variety of the modeling patterns used.

5.2 Normalizing and cleaning the metadata

As we mentioned in Section 4.1, we must assume that we will encounter every possible misuse and mis-interpretation of Schema.org properties when we operate at the scale of the whole Web. Thus, we perform a number of operations to normalize and clean up the metadata.

First, for the properties where we observe different patterns on the Web, we analyze the common patterns used and try to account for all of them. For instance Figure 2 shows the different patterns that we observed for defining downloads and distribution. In the figure, the first example of raw metadata defines the format of the dataset (CSV) at the level of the dataset itself and stores the download URL as the value of the http://schema.org/distribution property. Other examples in the figure deal with these two pieces of information differently. All these patterns are commonly used in our corpus. We mine these patterns by traversing either the initial graph or the resulting protocol buffer. Once we identify the patterns, we write adapters to convert all of them into the same modeling pattern in the reconciled metadata record. The right-hand side of Figure 2 shows this reconciled result.

Similarly, we have developed adapters for other metadata fields: We understand a lot more representations of dates than the ISO standard required by the Schema.org specification (Section 4.1. We will pick up digital object identifiers (DOIs) for a dataset from a variety of fields, and not just http://schema.org/identifier. We will use a uniform field, provider, for the many different fields that dataset providers used to identify this property. As we collect more metadata, our set of such adapters grows. Our decisions in these steps are guided by two factors: (1) the frequent usage patterns that we observed in the data; and (2) our understanding of what we expect the users to see in Dataset Search results.
3.2 Shared-Nothing

A shared-nothing parallel system (Figure 3.2) is made up of a cluster of independent machines that communicate over a high-speed network interconnect or, increasingly frequently, over commodity networking components. There is no way for a given system to directly access the memory or disk of another system.

Shared-nothing systems provide no hardware sharing abstractions, leaving coordination of the various machines entirely in the hands of the DBMS. The most common technique employed by DBMSs to support these clusters is to run their standard process model on each machine, or node, in the cluster. Each node is capable of accepting client SQL requests and processing them in parallel.

[Hellerstein et al., Architecture of a Database System]
## Column Stores

<table>
<thead>
<tr>
<th>id</th>
<th>Title</th>
<th>Person</th>
<th>Genre</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Mrs. Doubtfire</td>
<td>Robin Williams</td>
<td>Comedy</td>
</tr>
<tr>
<td>2</td>
<td>Jaws</td>
<td>Roy Scheider</td>
<td>Horror</td>
</tr>
<tr>
<td>3</td>
<td>The Fly</td>
<td>Jeff Goldblum</td>
<td>Horror</td>
</tr>
<tr>
<td>4</td>
<td>Steel Magnolias</td>
<td>Dolly Parton</td>
<td>Drama</td>
</tr>
<tr>
<td>5</td>
<td>The Birdcage</td>
<td>Nathan Lane</td>
<td>Comedy</td>
</tr>
<tr>
<td>6</td>
<td>Erin Brokovich</td>
<td>Julia Roberts</td>
<td>Drama</td>
</tr>
</tbody>
</table>

Each column has a file or segment on disk

[J. Swanhart, Introduction to Column Stores]
CAP Theorem

Scalability: CAP Theorem

Remains accessible and operational at all times.

Availability

Traditional relational databases: PostgreSQL, MySQL, etc.

CA

AP

Voidemort, Riak, Cassandra, CouchDB, Dynamo-like systems

Pick Two!

Consistency
Commits are atomic across the entire distributed system.

C

CP

HBase
MongoDB
Redis
MemcacheDB
BigTable-like systems

Partition Tolerance
Only a total network failure can cause the system to respond incorrectly.

P

SQL

[E. Brewer]
Cassandra: Replication and Consistency

[Image of a diagram showing replication between DC 1 and DC 2]
Spanner: Google's NewSQL Cloud Database

- Which type of system is Spanner?
  - C: consistency, which implies a single value for shared data
  - A: 100% availability, for both reads and updates
  - P: tolerance to network partitions

- Which two?
  - CA: close, but not totally available
  - So actually CP
Data Curation

The DCC Curation Lifecycle Model

Conceptualise
Create or receive
Appraise and select
Ingest
Preservation planning
Preservation action
Preserve
Store
Access, use, and reuse
Transform
Representational information
Community watch and participation
Description
Preservation planning
Data (Digital Objects or Databases)

Dispose
Reappraise
Migrate

Data, any information in binary digital form, is at the centre of the Curation Lifecycle. This includes:

- Simple Digital Objects are discrete digital items; such as textual files, images or sound files, along with their related identifiers and metadata.
- Complex Digital Objects are discrete digital objects, made by combining a number of other digital objects, such as websites.
- Structured collections of records or data stored in a computer system.

Full Lifecycle Actions

Sequential Actions

- Conceptualise
- Create or receive
- Appraise and select
- Ingest
- Preservation planning
- Preservation action
- Preserve
- Store
- Access, use, and reuse
- Transform
- Representational information
- Community watch and participation
- Description

Occasional Actions

- Dispose
- Reappraise
- Migrate

Dispose of data, which has not been selected for long-term curation and preservation in accordance with documented policies, guidance or legal requirements. Typically data may be transferred to another archive, repository, data centre or other custodian. In some instances data is destroyed. The data’s nature may, for legal reasons, necessitate secure destruction.

Return data which fails validation procedures for further appraisal and reselection.

Migrate data to a different format. This may be done to accord with the storage environment or to ensure the data’s immunity from hardware or software obsolescence.
FAIR Principles

• Findable: Metadata and data should be easy to find for both humans and computers

• Accessible: Users need to know how data can be accessed, possibly including authentication and authorization

• Interoperable: Can be integrated with other data, and can interoperate with applications or workflows for analysis, storage, and processing

• Reusable: Optimize the reuse of data. Metadata and data should be well-described so they can be replicated and/or combined in different settings
Graph Databases focus on relationships

- Directed, labelled, attributed multigraph
- Properties are **key/value pairs** that represent metadata for nodes and edges

![Diagram of property graph data model](image-url)

*Figure 7: Property graph data model. The main characteristic of this model is the occurrence of properties in nodes and edges. Each property is represented as a pair property-name = "property-value".*

*Note that the expressiveness of a model is defined by ease of use, not by the limits of what can be modeled.*

[R. Angles and C. Gutierrez, 2017]
Time Series Data

- **US Treasury bill contracts**
  - **Day** vs. **price**

- **Australian electricity production**
  - **Year** vs. **GWh**

- **Sales of new one-family houses, USA**
  - **Year** vs. **Total sales**

- **Annual Canadian Lynx trappings**
  - **Time** vs. **Number trapped**

[R. J. Hyndman]
Time Series Data

**Trend**

**US Treasury bill contracts**

**Australian electricity production**

**Sales of new one-family houses, USA**

**Annual Canadian Lynx trappings**

[R. J. Hyndman]
Time Series Data

**US Treasury bill contracts**
- Trend

**Australian electricity production**
- Trend + Seasonality

**Sales of new one-family houses, USA**

**Annual Canadian Lynx trappings**

---

[R. J. Hyndman]
Time Series Data

- **US Treasury bill contracts**: Trend
- **Australian electricity production**: Trend + Seasonality
- **Sales of new one-family houses, USA**: Seasonality + Cyclic
- **Annual Canadian Lynx trappings**: Trend + Seasonality

[R. J. Hyndman]
Time Series Data

- Trend
  - US Treasury bill contracts
  - US Treasury bill contracts
  - Australian electricity production
  - Australian electricity production

- Seasonality + Cyclic
  - Sales of new one-family houses, USA
  - Sales of new one-family houses, USA
  - Annual Canadian Lynx trappings
  - Annual Canadian Lynx trappings

- Trend + Seasonality

- Stationary

[R. J. Hyndman]
Split-Apply-Combine

![Diagram showing the process of Split-Apply-Combine]

Split: A 0, B 5, A 10
Apply: A 15, B 30, C 45
Combine: A 15, B 30, C 45

[W. McKinney, Python for Data Analysis]
Interactive Exploration of Spatial Data

```
SELECT lat, lng, (b4-b6)/(b4+b6) as ndsi
FROM modis_data
WHERE ndsi > 0.7
```
Interactive Exploration of Spatial Data

SELECT lat, lng, \((b4-b6)/(b4+b6)\) as ndsi
FROM modis_data
WHERE ndsi > 0.7

[Interactive Exploration of Spatial Data][1]

---

SLOW

DBMS

query

result

[Client]

[Server]

DBMS

[PostgreSQL, MySQL, Vertica, SciDB]

---

[L. Battle, 2017]
Spatial Data: NanoCubes and TopKube

Wikipedia: we can see that there were an unusual spike of activity during which we used to trace their location. The final dataset, with Anonymous edits contain the IP information of the user, (1.57 million unique ones). Figure 8 shows how exploration visualization of the dataset using T

OP

KUBE data structure.

The Yahoo! Flickr Creative Commons dataset [34] contains edit history for every article since its creation in 2005. Each record containing a set of user tags and geographical information. The dataset contains 84 million geolocated tags, which were taken by a Flickr user, along the West Coast of Africa. By highlighting the region, can be used to gain insight of unusual patterns in the data timeseries: a blue one covering the low activity days, and an orange one covering the high activity days. We can see that the high activity spike is mostly due to photos tagged with a few days in January. We create two different brushes in the exploration steps within January 2015 records. First we select a geographical area around Paris and find out an unusual activity on the west coast are from photos taken during a bike trip.

Fig. 7. Comparing the top edited articles in Nevada and Mississippi.

Fig. 8. Geolocated Flickr tags in Africa: the unusual activity on the west

OP

KUBE.

Microblogging: This dataset is comprised of publicly available geotagged microblog entries. From each post, we extracted the latitude, longitude, and hashtags from the blog. Freewheely.com is one of the most popular bicycle projects. Each commit was geolocated based on the location dotfiles, a project for sharing customized environment files on Unix-based operating systems. It is also interesting to notice how diversified open source community across the United States.

6.1 Use Cases

We can use T

OP

KUBE to explore the most popular hashtags in question was the terrorism attack at the Charlie Hebdo headquarters. To understand how the hashtags created for this event at the day of the attack faded in time, we further constrain our selection to just the hashtags related to the terrorism attack and see that those fade almost completely in 10. #rip 187

8. #lt 335

6. #charliehebdo 418

5. #gagnetaplace 447

4. #paris 607

3. #lrt 1,146

2. #charliehebdo 4,190

1. #jesuischarlie 4,456

In order to understand how trending topics vary over time in the current selection (i.e. Paris and Jan 7). The event in question was the terrorism attack at the Charlie Hebdo in Paris and Jan 7. The event select this peak we quickly find evidence of the event that was the terrorism attack at the Charlie Hebdo terrorism act in Paris.

Fig. 9. Microblog exploration using T

OP

KUBE: a temporal perspective of the top hashtags related to the the Charlie Hebdo terrorism act in Paris.

Fig. 10. GitHub projects with most commits in three large urban centers.

Reno, Nevada 303

Early Christianity 284

Comparison of the AK-47 and M1... 273

Las Vegas Academy 225

timel... 216

Las Vegas 204

Council of Jerusalem 192

Paul the Apostle 190

University of Nevada, Las Vegas... 189

Nevada 188

Antinomianism 188

[F. Miranda et al., 2017]
Provenance

Data Management

Visualization

Computation

Publishing
Prospective and Retrospective Provenance

• Recipe for baking a cake versus the actual process & outcome
• Prospective provenance is what was specified/intended
  - a workflow, script, list of steps
• Retrospective provenance is what actually happened
  - actual data, actual parameters, errors that occurred, timestamps, machine information
• Do not need prospective provenance to have retrospective provenance!
Using Provenance
Reproducibility

Reproducibility Spectrum

- Publication only
- Code
- Code and data
- Linked and executable code and data
- Full replication

Not reproducible — Gold standard

[R. D. Peng]
Machine Learning and Databases

[Image of a diagram illustrating various machine learning and database concepts]

D. Koop, CSCI 490/680, Spring 2020

T. Kraska, 2019
Final Exam

• Tuesday, May 5 from 4-5:50pm
• Online
• Similar format to Test 2
• Comprehensive but with more focus on last few weeks of class
• Contact me with questions:
  - Email
  - Setup a time to talk via Blackboard
Stay Safe