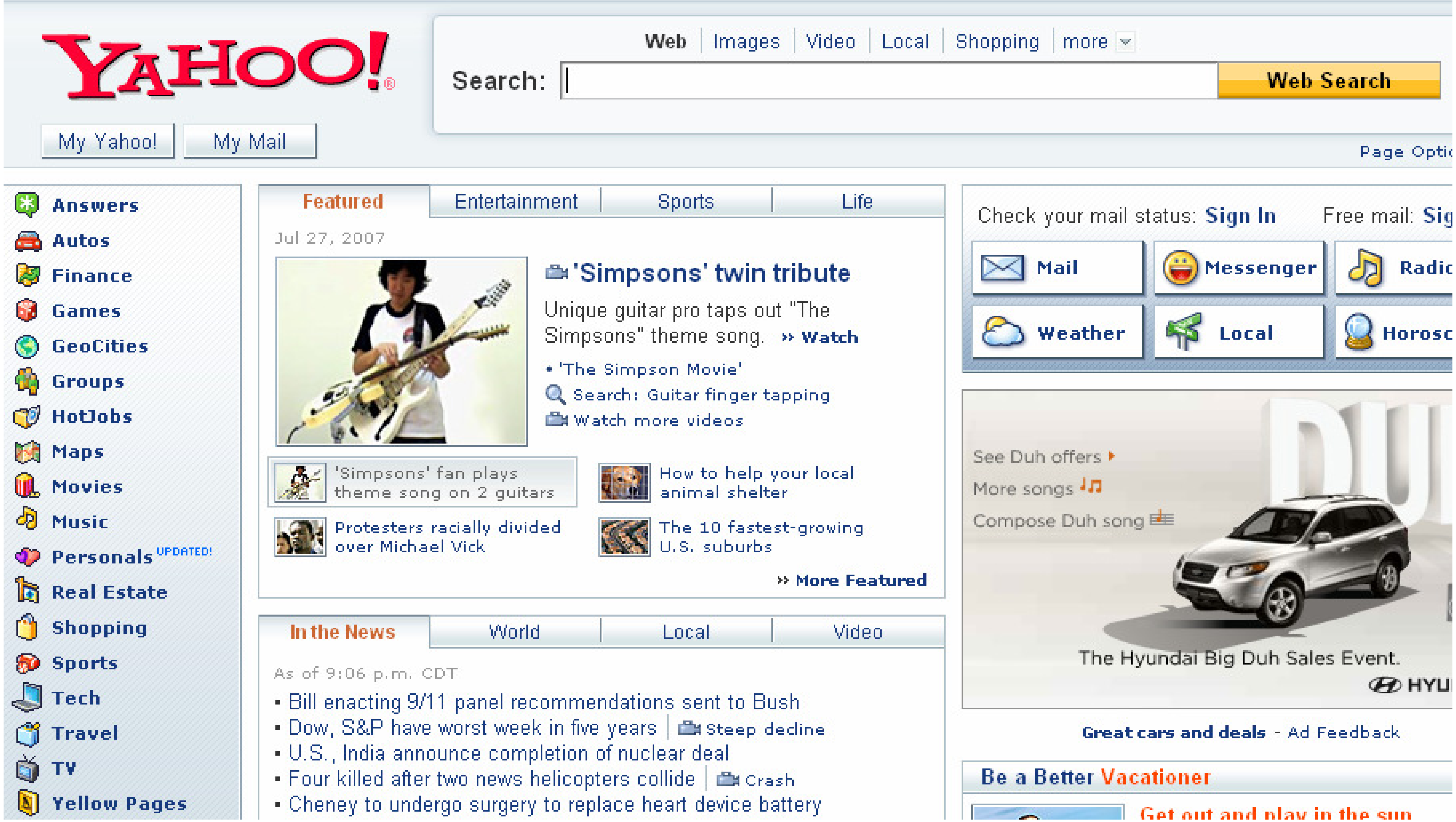
# A Relational Approach to Incrementally Extracting and Querying Structure in Unstructured Data

Eric Chu, Akanksha Baid, Ting Chen,
AnHai Doan, Jeffrey Naughton

University of Wisconsin-Madison

# Querying Unstructured Data:
# Keyword Search + Browsing



Jul 27, 2007

'Simpsons' twin tribute

Unique guitar pro taps out "The Simpsons" theme song. » Watch

- 'The Simpson Movie'
- Search: Guitar finger tapping
- Watch more videos

'Simpsons' fan plays theme song on 2 guitars

How to help your local animal shelter

Protesters racially divided over Michael Vick

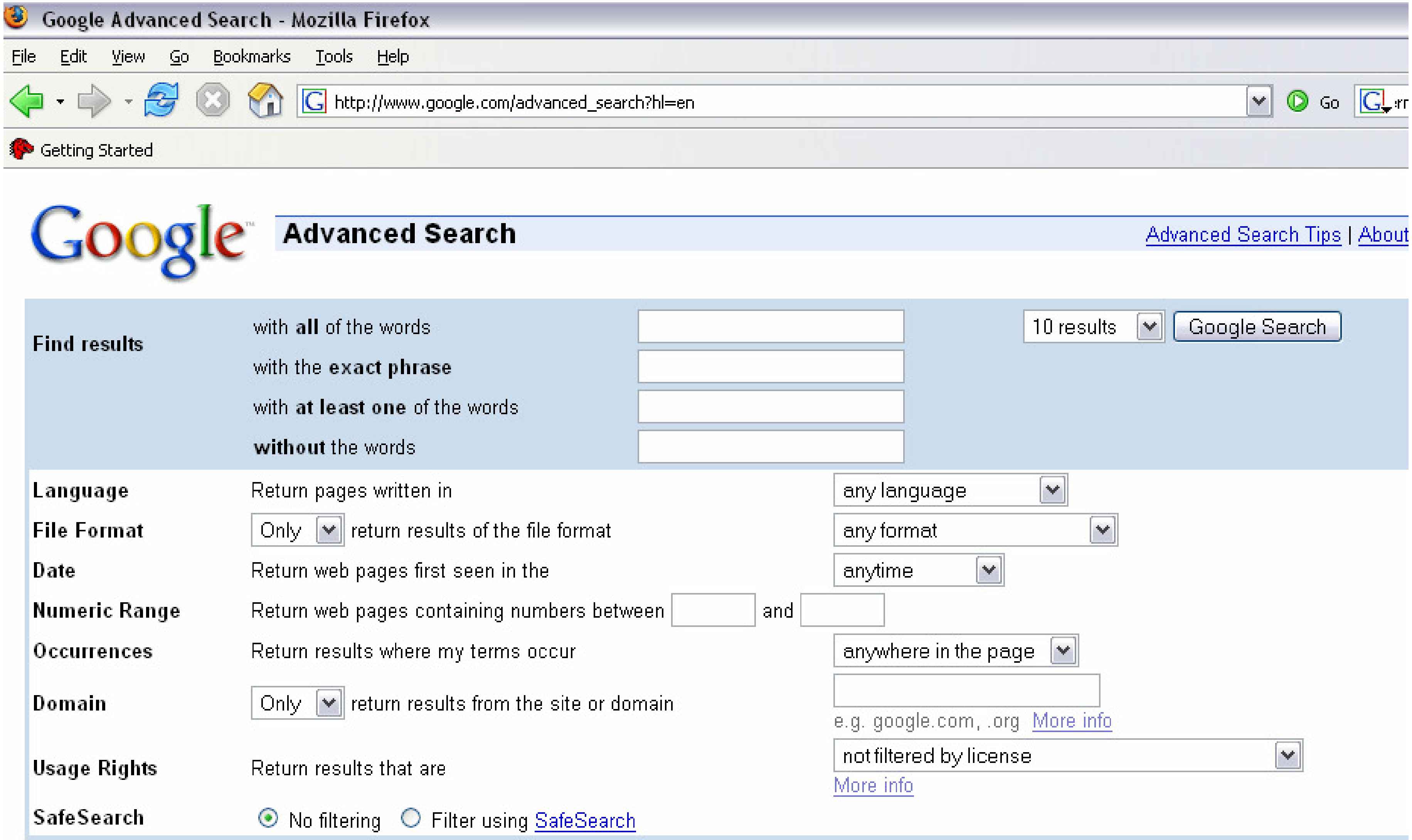
The 10 fastest-growing U.S. suburbs

» More Featured

As of 9:06 p.m. CDT

- Bill enacting 9/11 panel recommendations sent to Bush
- Dow, S&P have worst week in five years | Steep decline
- U.S., India announce completion of nuclear deal
- Four killed after two news helicopters collide | Crash
- Cheney to undergo surgery to replace heart device battery

# Perhaps with predefined search options...


Google Advanced Search - Mozilla Firefox
File
Edit
View
Go
Bookmarks
Tools
Help
Go
Getting Started
Advanced Search Tips | About
Google Search

http://www.google.com/advanced_search?hl=en

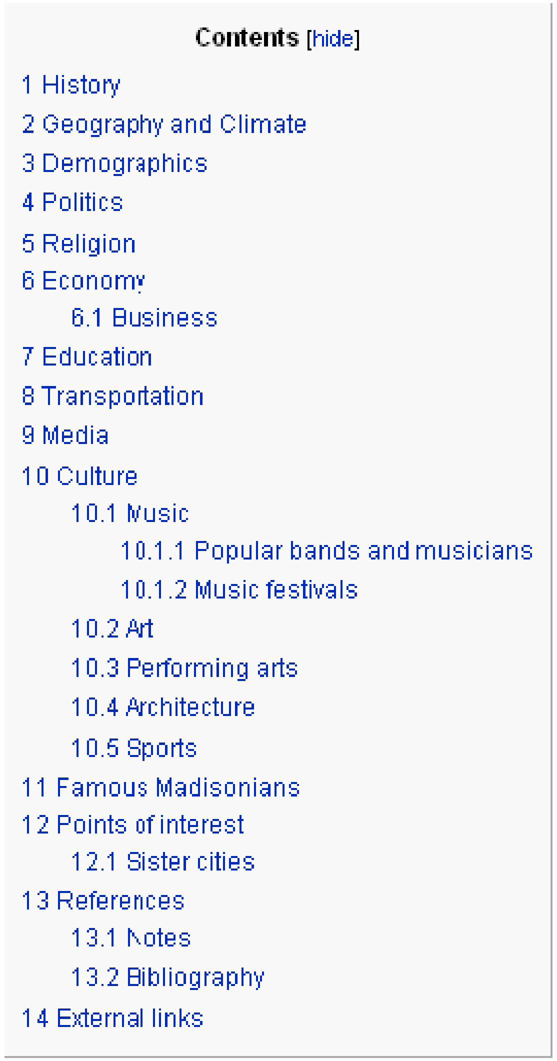
**Advanced Search**

| | | | |
|---|---|---|---|
| **Find results** | with **all** of the words | | 10 results |
| | with the **exact phrase** | | |
| | with **at least one** of the words | | |
| | **without** the words | | |
| **Language** | Return pages written in | | any language |
| **File Format** | Only | return results of the file format | any format |
| **Date** | Return web pages first seen in the | | anytime |
| **Numeric Range** | Return web pages containing numbers between | and | |
| **Occurrences** | Return results where my terms occur | | anywhere in the page |
| **Domain** | Only | return results from the site or domain | e.g. google.com, .org More info |
| **Usage Rights** | Return results that are | | not filtered by license<br>More info |
| **SafeSearch** | ● No filtering | ○ Filter using SafeSearch | |

# Structure in Text: Relationships

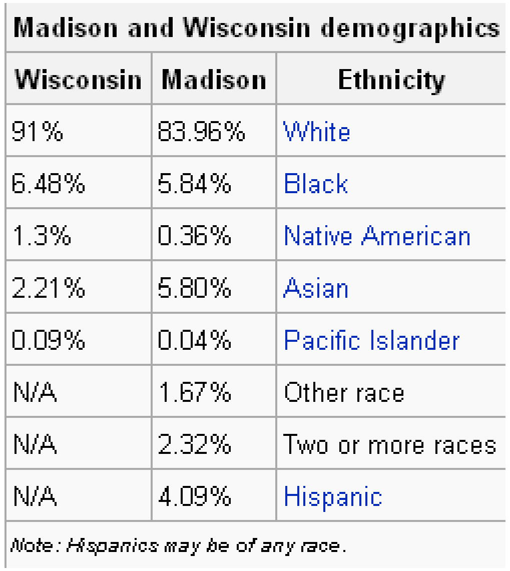
"Madison is also home to companies such as Broadcast Interactive Media, as well as the North American division of Spectrum Brands (formerly Rayovac), Alliant Energy, American Family Insurance, the Credit Union National Association, CUNA Mutual Group. Technology companies in the area include Netconcepts, TomoTherapy, Sonic Foundry, Raven Software, Human Head Studios, Renaissance Learning, Flame Front Software, Epic Systems Corporation, and Berbee Information Networks. ......"

# Structure in Text: Sections and Tables



Contents [hide]

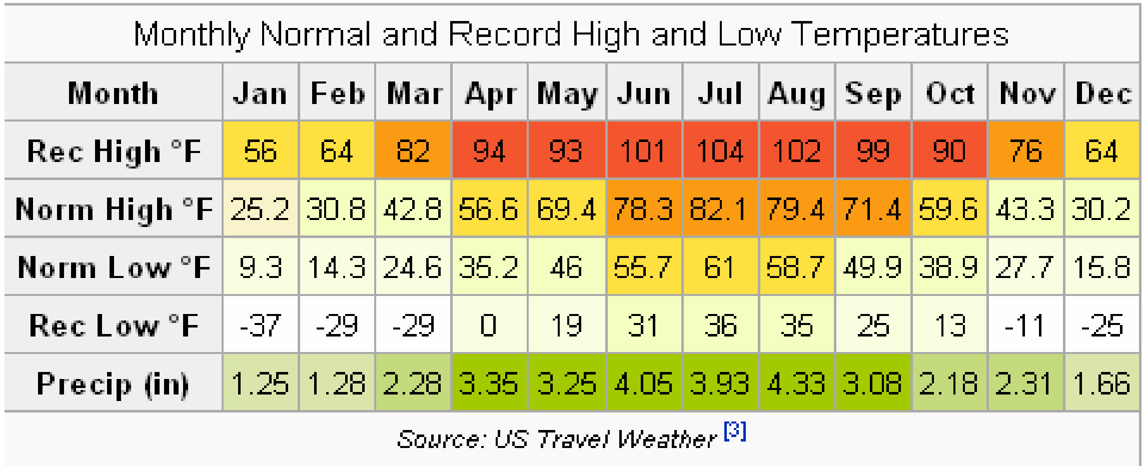
1 History
2 Geography and Climate
3 Demographics
4 Politics
5 Religion
6 Economy
    6.1 Business
7 Education
8 Transportation
9 Media
10 Culture
    10.1 Music
        10.1.1 Popular bands and musicians
        10.1.2 Music festivals
    10.2 Art
    10.3 Performing arts
    10.4 Architecture
    10.5 Sports
11 Famous Madisonians
12 Points of interest
    12.1 Sister cities
13 References
    13.1 Notes
    13.2 Bibliography
14 External links



**Madison and Wisconsin demographics**

| Wisconsin | Madison | Ethnicity |
|---|---|---|
| 91% | 83.96% | White |
| 6.48% | 5.84% | Black |
| 1.3% | 0.36% | Native American |
| 2.21% | 5.80% | Asian |
| 0.09% | 0.04% | Pacific Islander |
| N/A | 1.67% | Other race |
| N/A | 2.32% | Two or more races |
| N/A | 4.09% | Hispanic |

*Note: Hispanics may be of any race.*



Monthly Normal and Record High and Low Temperatures

| Month | Jan | Feb | Mar | Apr | May | Jun | Jul | Aug | Sep | Oct | Nov | Dec |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Rec High °F | 56 | 64 | 82 | 94 | 93 | 101 | 104 | 102 | 99 | 90 | 76 | 64 |
| Norm High °F | 25.2 | 30.8 | 42.8 | 56.6 | 69.4 | 78.3 | 82.1 | 79.4 | 71.4 | 59.6 | 43.3 | 30.2 |
| Norm Low °F | 9.3 | 14.3 | 24.6 | 35.2 | 46 | 55.7 | 61 | 58.7 | 49.9 | 38.9 | 27.7 | 15.8 |
| Rec Low °F | -37 | -29 | -29 | 0 | 19 | 31 | 36 | 35 | 25 | 13 | -11 | -25 |
| Precip (in) | 1.25 | 1.28 | 2.28 | 3.35 | 3.25 | 4.05 | 3.93 | 4.33 | 3.08 | 2.18 | 2.31 | 1.66 |

*Source: US Travel Weather* [3]

# Goal: Exploit the Structure!

- Improve keyword search and browsing

- Structured queries

  "What companies have their headquarters in Wisconsin?"

  "Which universities are in places that are very cold in winter?"

  Queries that keyword search and browsing are not good at answering

# Challenges beyond Information Extraction

Continual discovery and extraction of structure

- Manage this *evolving* structure *incrementally*
  - Little consensus on what system to use

- How to enable users to query using *as much structure as is currently known*?

# Our Proposal

A relational workbench that provides:

- A way to store an expanding set of documents and attributes
- Tools to incrementally process the data
- A way to exploit structure in queries

# Advantages

- Data always available for querying
- Supports *incremental* data processing
- Can pose increasingly sophisticated queries over time
- Exploit strengths of a RDBMS

# Relational Workbench

- Data Storage
- Data Processing
- Case Study: Swikipedia

# Extracting Structure from Text

Set of extracted attributes:

- Keeps evolving
- Heterogeneous
- Some may refer to same real-world concept

No good schema!

# Storage: Wide Table

- A single table
- One document per row
- Discovery of new attribute: create new column in the table

# Example

- Seattle and Madison pages in Wikipedia

| **DocTitle** | **DocContent** | **Official flower** |
|---|---|---|
| Madison, Wisconsin | “Madison is the capital of the U.S. state of Wisconsin ...” | *null* |
| Seattle, Washingon | “Seattle is the largest city in the Pacific Northwest ...” | Dahlia |

# The Wide Table Grows

- Longer when we insert new documents
- Wider when we create new columns for the new attributes we extract
- Also increasingly sparse

  No overhead for storing nulls if we use interpreted storage (Chu et al., SIGMOD ’07) or a column-oriented database (Abadi, CIDR ’07)

# Problem with 1NF

- A document can have set-valued attributes
  - E.g., “Lake Mendota” and “Lake Monona” from Madison, and “Lake Washington” and “Lake Union” from Seattle

- Structure can be complex
  - E.g., the weather wiki table

| Madison | Jan | Feb |
|---|---|---|
| Avg High Temp °F (° C) | 23 (-5) | 29 (-2) |
| Avg Low Temp °F (° C) | 6 (-14) | 12 (-11) |
| Mean Temp °F (° C) | 15 (-9) | 20 (-7) |
| Avg Precipitation in (cm) | 1.14 (2.9) | 1.14 (2.9) |

# Proposal: Complex Attributes

- Allow attributes to have internal structure

| DocTitle | DocContent | official flower | headquarter(city, company) |
|---|---|---|---|
| Madison, Wisconsin | “Madison is the capital of the U.S. state of Wisconsin ...” | | [(Madison, Raven Software),<br>(Madison, Human Head Studios), ...] |
| Seattle, Washingon | “Seattle is the largest city in the Pacific Northwest ...” | dahlia | [(Seattle, Starbucks),<br>(Seattle, Amazon.com), ...] |

# Summary of Wide Table

- Each document corresponds to a row
- Each attribute corresponds to a column
- Table can get very wide and sparse
- Attributes can have internal structure

# Implementing Results of Integration: Mapping Table

- Store mappings for different attributes that correspond to the same real-world concept

| host id | host name | mappings |
|---|---|---|
| a6 | temp (°F) | {a6 = a7 * 9/5 + 32} |
| a7 | temperature (°C) | {a7 = 5/9 * (a6 – 32)} |

- Query evaluation:
  - Look up mapping table
  - Rewrite query to include matching attributes

# Relational Workbench

- Data Storage
- Data Processing
- Case Study: Swikipedia

# Processing Data with the Workbench

- Workbench does not decide how to process data
- Provides three basic operators:
  - Extract
  - Integrate
  - Cluster
- DBAs decide what operators to use and set the parameters

# Operators VS User Defined Functions

Advantages of defining operators:

- Ease of use
- Performance optimization
- Synergistic interaction among operators

# Whole is greater than the sum of parts (1)

An operator often improves the performance of its following operators

Example: finding new input for extraction

- Extract: “address” => “city,” “state,” and “zip code”
- Integrate: “address” = “sent-to”
- Extract: “sent-to” => “city,” “state,” and “zip code”

# Whole is greater than the sum of parts (2)

Example: improving clustering via iteration

- Extract section names from wikipedia pages
- Cluster pages based on section names
  - Problem: short pages have no sections
- Extract and cluster pages based on other attributes

# Case Study: Swikipedia

- Simulated a workbench for Wikipedia
- Core dump of ~4 million pages (XML)
- Toy data set (882 files)
  - 3 domains: cities (254), universities (255), tennis players (373)

# Stage 1: Initial Loading

- Parsed and loaded XML files to wide table
- 5 columns: PageId, PageText, RevisionId, ContributorName, LastModificationDate
- Can do keyword search over PageText immediately

# Stage 2: Extracting Sections

| Doc Title | Doc Content | History | Demo-graphics | Religion | Cityscape | Points of Interest |
|---|---|---|---|---|---|---|
| Madison, Wisconsin | “Madison is the capital of the U.S. state of Wisconsin ...” | “Madison was created in…” | “As of the census…” | “…” | “…” | *null* |
| Seattle, Washingon | “Seattle is the largest city in the Pacific Northwest ...” | “What is now Seattle has been…” | “As of the census…” | “…” | *null* | “…” |

# Why extract sections?

- For doing future extraction more efficiently
- For “focused” keyword search

  Example: “world no. 1 player”

  - Over PageText column:

    return 83 pages, 23 correct
  - Over Introduction column:

    return 67 pages, 21 correct

# Explosion of New Columns

- Wide table now had 1,253 new columns!
  - Each row had only 13 non-null attributes
- Integrate found many aliases
  - 350 of all attributes belonged to 1 of 14 attribute groups
  - E.g., campus, famous people, tournament titles, etc.

# Handling Aliases with the Mapping Table

Wide table

| **DocTitle** | **DocContent** | **History** | **Cityscape** | **Points of Interest** |
|---|---|---|---|---|
| Madison, Wisconsin | “......” | “..... “ | “Madison …” | ***null*** |
| Seattle, Washingon | “.......” | “......” | ***null*** | “Seattle …” |

Mapping table

| **host id** | **host name** | **mappings** |
|---|---|---|
| a8 | Cityscape | a9 |
| a9 | Points of interest | a8 |

# Handling Aliases: An Alternative

- Collapse aliases into one column in the wide table.

| **DocTitle** | **DocContent** | **History** | **Cityscape,**<br>**Points of Interest** |
|---|---|---|---|
| Madison, Wisconsin | “......” | “..... “ | “Madison …” |
| Seattle, Washingon | “.......” | “......” | “Seattle …” |

# Stage 3: Attribute Clustering

- Grouped together attributes (i.e., section names) that were either both null or both non-null in a row
- Found three clusters
  - Used a column to store cluster IDs
- Views on clusters
  - ~25 ms for each cluster
  - Wide table: 44 sec

# Stage 4: Extracting Wiki Tables

**temperature_wiki**

| City | Month | Low_F | Low_C | High_F | ... |
|---|---|---|---|---|---|
| Madison, Wisconsin | 1 | 6 | -14 | 23 | ... |
| Madison, Wisconsin | 2 | 12 | -11 | 29 | ... |
| Madison, Wisconsin | 12 | 13 | -11 | 29 | ... |
| Seattle, Washington | 1 | 36 | 2 | 46 | ... |

# Examples

"Find average temperature of Madison during winter."

```
SELECT     AVG(Low_F)
FROM       temperature_wiki as T
WHERE      T.city = 'Madison, Wisconsin' AND
           T.Month = 1 OR T.Month = 2 OR T.Month = 12;
```

"Which universities are in places that can be very cold?"

```
SELECT     T1.ID
FROM       WideTable T1, temperature_wiki T2
WHERE      T1.location = T2.city AND T2.month =
           1 AND Low_F < 32
```

# Summary of Case Study

- Only done basic data processing
- Incremental approach promising
  - Pay (and get rewarded) as you go
  - Flexible
- Set of attributes could evolve in size and complexity very quickly
- Multiple ways to process the data

# Current and Future Work

- Prototype for Swikipedia
- Query construction, evaluation, and optimization
- Changes to data and operators

# Conclusion

- Relational workbench to incrementally extract and query structure from unstructured data
  - Wide table
  - Mapping table
  - Operators

- Swikipedia

- Many problems ahead!