Hypercubes in HBase

Fredrik Möllerstrand <fredrik@last.fm>
Hadoop User Group UK, April 14 2009
Hello.

- Per Andersson, Fredrik Möllerstrand
- Chalmers University of Technology, Sweden
- Master thesis at last.fm
stats.last.fm

- Web statistics for in-house use.
- Served out of mysql.
stats.last.fm

- y-axis: pageviews.
- x-axis: time.
- also: countries.
SELECT pageviews, country
FROM webstatistics
GROUP BY country;
SQL: Star schema

- Facts table & dimension tables.
- Joins!
Data Cube Foundations

- n-dimensional cube
- attribute $\Rightarrow$ dimension
- attribute value $\Rightarrow$ measurement
Data Cube Foundations

- dimensionality reductions
- projections
Data Cube Foundations

- Aggregation: sum, count, average, &c.
- Data cubes modeled as:
  - in RDBMSs, modeled as star schemas.
  - in HBase, modeled with column-families.
Data Cubes in HBase

• Store projections, not distinct dimensions.

• Pre-compute *everything*. 
Data Cubes in HBase

- Rowkey: unit + time
  i.e. ‘pageviews-20090414’

- One column-family for every projection.
  i.e. ‘country-useragent’

- One qualifier per point in n-space.
  i.e. ‘US-safari’, ‘NO-opera’, &c.
The SQL-DB Problem

- Too much data to keep in memory.
- Plenty of joins makes queries complex.
- Can’t serve at mouse click rate.
The Solution;
A data store that is:

• Distributed
• Multi-dimensional
• Magnetic(!)
• Just general enough
Enter: Zohmg.
Zohmg;
A data store that is:

- Distributed
- Multi-dimensional
- Time-series-based
- Magnetic(!)
- Just general enough
Tech

- Rides on the back of Dumbo.
- Stores aggregates in HBase.
- Serves JSON.
Zohmg

- `>$ setup.py`  
  # create hbase database.

- `>$ import.py --mapper weblogs.py`  
  # run dumbo job.

- `>$ serve.py`  
  # start web server.
Developers, developers.

- Configuration - yaml.
- Mapper - python.
User’s configuration.

project_name: webmetrics

dimensions:
  - country
  - domain
  - useragent
  - usertype

units:
  - pageviews

projections:
  country:
    - country
  domain-usertype:
    - domain
    - usertype
  country-domain-useragent-usertype:
    - country
    - domain
    - useragent
    - usertype
def map(key, value):
    from lfm.data.parse import web

    log = web.parse(value)

    dimensions = {'country' : geoip(log.host),
                  'domain'   : log.domain,
                  'useragent' : classify(log.useragent),
                  'usertype'  : ('user', 'anon')[log.userid == None]
                }

    values = {'pageviews' : 1}

    yield log.timestamp, dimensions, values
Example.
Dimensions in HBase

• Column-family: 
  country-useragent-domain

• Qualifier: 
  US-firefox-last.fm
Questions?