INTRODUCING DRUID:
FAST AD-HOC QUERIES ON BIG DATA

MICHAEL DRISCOLL - CEO
ERIC TSCHETTER - LEAD ARCHITECT
@ METAMARKETS
Metamarkets is the bridge from BI to Data Science, a new kind of real-time, analytics tool for the Big Data generation.
OUR SOLUTION...

Visualization

Analytics

Data Store

ETL
OUR SOLUTION... LEVERAGES OPEN SOURCE

Visualization (node.js, D3)

Analytics (R, Scala)

Data Store (Druid*)

ETL (Hadoop, Kafka)
1,000,000,000+ events daily
100,000+ queries daily
500 ms avg response

Data Store (Druid*)
Twitter
8 TB per day

Facebook
100 TB per day

US Library of Congress
235 TB

Global Businesses
1.8 ZB in 2011

Boeing
640 TB per flight

Wal-Mart
2.5 PB stored

A TIDAL WAVE OF DATA
JUST IN CASE THE WIFI FAILS US...
Past 24 hours vs. previous 24 hours

- Edits: 358k -49.8k (-12.2%)
- Average Added: 362 +30.1 (9.06%)
- Average Deleted: -174 -28.8 (19.9%)

Discovery Feed

- There was a small positive change in Edits for the Page: 2012 for 1 hour.
  - 2 hours ago

- Edits for the Page: Mozilla_Firefox increased by a small amount for 1 hour.
  - 2 hours ago

- Unique Users for the Page: Mozilla_Firefox experienced small increases for 1 hour.
  - 2 hours ago

- There was a small positive change in Edits for the Page: Lance_Armstrong for 1 hour.
  - 3 hours ago

- Edits for the Page: League_of_Legends experienced small gains over 1 hour.
<table>
<thead>
<tr>
<th>Language</th>
<th>User</th>
<th>Edits</th>
<th>Unique Users</th>
<th>Total Delta</th>
<th>Total Added</th>
<th>Total Deleted</th>
<th>Avg Delta</th>
<th>Avg Added</th>
<th>Avg Deleted</th>
<th>Variation</th>
<th>Avg Variation</th>
</tr>
</thead>
<tbody>
<tr>
<td>de</td>
<td>Gödeke</td>
<td>2k</td>
<td>1</td>
<td>50k</td>
<td>64k</td>
<td>-13k</td>
<td>32</td>
<td>40</td>
<td>-8.24</td>
<td>77k</td>
<td>48</td>
</tr>
<tr>
<td></td>
<td>WB</td>
<td>1k</td>
<td></td>
<td>190k</td>
<td>221k</td>
<td>-32k</td>
<td>128</td>
<td>150</td>
<td>-21.5</td>
<td>253k</td>
<td>171</td>
</tr>
<tr>
<td></td>
<td>Harry8</td>
<td>1k</td>
<td></td>
<td>19k</td>
<td>2,807k</td>
<td>-2,788k</td>
<td>13</td>
<td>2,016</td>
<td>-2,001</td>
<td>5,596k</td>
<td>4,017</td>
</tr>
<tr>
<td></td>
<td>Aka</td>
<td>1k</td>
<td></td>
<td>15k</td>
<td>20k</td>
<td>-5k</td>
<td>11</td>
<td>15</td>
<td>-3.43</td>
<td>25k</td>
<td>18</td>
</tr>
<tr>
<td></td>
<td>Åhhus</td>
<td>1k</td>
<td></td>
<td>54k</td>
<td>59k</td>
<td>-5k</td>
<td>43</td>
<td>47</td>
<td>-3.88</td>
<td>64k</td>
<td>50</td>
</tr>
<tr>
<td>en</td>
<td>ClueBot NG</td>
<td>18k</td>
<td></td>
<td>31,275k</td>
<td>33,200k</td>
<td>-1,925k</td>
<td>1,697</td>
<td>1,800</td>
<td>-104</td>
<td>35,125k</td>
<td>1,905</td>
</tr>
<tr>
<td></td>
<td>Silas S. Brown</td>
<td>8k</td>
<td></td>
<td>183k</td>
<td>183k</td>
<td>-0k</td>
<td>23</td>
<td>23</td>
<td>-0.01</td>
<td>183k</td>
<td>23</td>
</tr>
<tr>
<td></td>
<td>BrownHairedGirl</td>
<td>6k</td>
<td></td>
<td>396k</td>
<td>483k</td>
<td>-87k</td>
<td>69</td>
<td>84</td>
<td>-15.1</td>
<td>570k</td>
<td>99</td>
</tr>
<tr>
<td></td>
<td>Meno25</td>
<td>5k</td>
<td></td>
<td>-35k</td>
<td>26k</td>
<td>-61k</td>
<td>7</td>
<td>15</td>
<td>-11.4</td>
<td>87k</td>
<td>16</td>
</tr>
<tr>
<td></td>
<td>Waacstats</td>
<td>3k</td>
<td></td>
<td>74k</td>
<td>101k</td>
<td>-27k</td>
<td>21</td>
<td>29</td>
<td>-7.76</td>
<td>128k</td>
<td>37</td>
</tr>
<tr>
<td>es</td>
<td>Jkbw</td>
<td>1k</td>
<td></td>
<td>369k</td>
<td>448k</td>
<td>-79k</td>
<td>339</td>
<td>411</td>
<td>-72.3</td>
<td>527k</td>
<td>484</td>
</tr>
<tr>
<td></td>
<td>Canopus49</td>
<td>1k</td>
<td></td>
<td>-276k</td>
<td>162k</td>
<td>-438k</td>
<td>235</td>
<td>150</td>
<td>-40.4</td>
<td>550k</td>
<td>554</td>
</tr>
<tr>
<td></td>
<td>SuperBraulio13</td>
<td>1k</td>
<td></td>
<td>586k</td>
<td>674k</td>
<td>-88k</td>
<td>566</td>
<td>652</td>
<td>-85.4</td>
<td>763k</td>
<td>737</td>
</tr>
<tr>
<td></td>
<td>Helmy oved</td>
<td>1k</td>
<td></td>
<td>346k</td>
<td>490k</td>
<td>-144k</td>
<td>359</td>
<td>509</td>
<td>-149</td>
<td>634k</td>
<td>658</td>
</tr>
<tr>
<td></td>
<td>PatruBOT</td>
<td>1k</td>
<td></td>
<td>113k</td>
<td>176k</td>
<td>-63k</td>
<td>119</td>
<td>185</td>
<td>-66.3</td>
<td>239k</td>
<td>252</td>
</tr>
<tr>
<td>fr</td>
<td>Sebieouf</td>
<td>4k</td>
<td></td>
<td>61k</td>
<td>70k</td>
<td>-9k</td>
<td>17</td>
<td>19</td>
<td>-2.35</td>
<td>79k</td>
<td>21</td>
</tr>
<tr>
<td></td>
<td>Vlaam</td>
<td>3k</td>
<td></td>
<td>229k</td>
<td>272k</td>
<td>-43k</td>
<td>82</td>
<td>97</td>
<td>-15.2</td>
<td>315k</td>
<td>112</td>
</tr>
<tr>
<td></td>
<td>Polmars</td>
<td>2k</td>
<td></td>
<td>196k</td>
<td>234k</td>
<td>-38k</td>
<td>96</td>
<td>114</td>
<td>-18.3</td>
<td>271k</td>
<td>132</td>
</tr>
<tr>
<td></td>
<td>Thierry Caro</td>
<td>2k</td>
<td></td>
<td>90k</td>
<td>97k</td>
<td>-8k</td>
<td>44</td>
<td>48</td>
<td>-3.77</td>
<td>105k</td>
<td>52</td>
</tr>
<tr>
<td></td>
<td>Lomita</td>
<td>2k</td>
<td></td>
<td>2,288k</td>
<td>693k</td>
<td>-2,982k</td>
<td>344</td>
<td>344</td>
<td>-1,480</td>
<td>3,675k</td>
<td>1,825</td>
</tr>
<tr>
<td>ru</td>
<td>Fameowner</td>
<td>1k</td>
<td></td>
<td>52k</td>
<td>258k</td>
<td>-206k</td>
<td>36</td>
<td>180</td>
<td>-143</td>
<td>463k</td>
<td>324</td>
</tr>
<tr>
<td></td>
<td>Valda72</td>
<td>1k</td>
<td></td>
<td>256k</td>
<td>396k</td>
<td>-140k</td>
<td>244</td>
<td>377</td>
<td>-133</td>
<td>536k</td>
<td>510</td>
</tr>
<tr>
<td></td>
<td>Marchello3187</td>
<td>1k</td>
<td></td>
<td>134k</td>
<td>137k</td>
<td>-3k</td>
<td>186</td>
<td>191</td>
<td>-4.58</td>
<td>141k</td>
<td>196</td>
</tr>
<tr>
<td></td>
<td>AVB</td>
<td>1k</td>
<td></td>
<td>219k</td>
<td>335k</td>
<td>-116k</td>
<td>333</td>
<td>510</td>
<td>-177</td>
<td>452k</td>
<td>688</td>
</tr>
<tr>
<td></td>
<td>Ковплас</td>
<td>1k</td>
<td></td>
<td>38k</td>
<td>43k</td>
<td>-5k</td>
<td>66</td>
<td>75</td>
<td>-9.23</td>
<td>49k</td>
<td>84</td>
</tr>
</tbody>
</table>
ERIC TSCHETTER

LEAD ARCHITECT @ METAMARKETS - @ZEDRUID
REAL-TIME ANALYTICS MEANS

- Fast aggregations over O(billions), high-dimensional events
- Flexible drill-downs with arbitrary depths
- Sub-second queries that reflect changes immediately
WHAT WE TRIED
WHAT WE TRIED

I. RDMBS - Relational Database
I. RDBMS - THE SETUP

- Star Schema
- Aggregate Tables
- Query Caching
I. RDBMS - THE RESULTS

- Queries that were cached
  - fast
- Queries against aggregate tables
  - fast to acceptable
- Queries against base fact table
  - generally unacceptable
## I. RDBMS - PERFORMANCE

<table>
<thead>
<tr>
<th>Description</th>
<th>Time/Rows</th>
</tr>
</thead>
<tbody>
<tr>
<td>Naive benchmark scan rate</td>
<td>~5.5M rows / second / core</td>
</tr>
<tr>
<td>1 day of summarized aggregates</td>
<td>60M+ rows</td>
</tr>
<tr>
<td>1 query over 1 week, 16 cores</td>
<td>~5 seconds</td>
</tr>
<tr>
<td>Page load with 20 queries over a week of data</td>
<td>long time</td>
</tr>
</tbody>
</table>
WHAT WE TRIED

I. RDMBS - Relational Database
WHAT WE TRIED

I. RDMBS - Relational Database
WHAT WE TRIED

I. RDMBS - Relational Database
II. NoSQL - Key/Value Store
II. NOSQL - THE SETUP

- Pre-aggregate all dimensional combinations
- Store results in a NoSQL store
II. NOSQL - THE RESULTS

- Queries were fast
  - range scan on primary key
- Inflexible
  - not aggregated, not available
- Not continuously updated
  - aggregate first, then display
- Processing scales exponentially
II. NOSQL - PERFORMANCE

- Dimensional combinations => exponential increase
- Tried limiting dimensional depth
  - still expands exponentially
- Example: ~500k records
  - 11 dimensions, 5-depth
    - 4.5 hours on a 15-node Hadoop cluster
  - 14 dimensions, 5-depth
    - 9 hours on a 25-node Hadoop cluster
WHAT WE TRIED

I. RDMBS - Relational Database
II. NoSQL - Key/Value Store
WHAT WE TRIED

I. RDMBS - Relational Database
II. NoSQL - Key/Value Store
WHAT WE TRIED

I. RDMBS - Relational Database
II. NoSQL - Key/Value Store
III. ???
WHAT WE LEARNED

- Problem with RDBMS: scans are slow
- Problem with NoSQL: computationally intractable
WHAT WE LEARNED

- Problem with RDBMS: scans are slow
- Problem with NoSQL: computationally intractable
- Fixing RDBMS issue seems easier
INTRODUCING DRUID
DRUID – FIVE KEY FEATURES

1. Distributed Column-Store
2. Real-time Ingestion
3. Fast Data Scans
4. Fast Filtering
5. Operationally Simple
FEATURE 1 – DISTRIBUTED COLUMN-STORE

- Data chunked into “segments”
  - MVCC swapping protocol
- Converted to columnar format
- Coordinator oversees cluster
  - Data replication and balance
- Per-segment replication
  - Increase replication on hot spots
FEATURE 2 – REAL-TIME INGESTION
FEATURE 2 – REAL-TIME INGESTION

Broker Nodes

Historical Nodes

Query API

Query Rewrite
Scatter/Gather
FEATURE 2 – REAL-TIME INGESTION

Real-time Nodes

Broker Nodes

Historical Nodes

Query API

Query Rewrite Scatter/Gather
FEATURE 2 – REAL-TIME INGESTION

Real-time Nodes

Broker Nodes

Historical Nodes

Query API

Query Rewrite Scatter/Gather

Hand Off Data
FEATURE 3 – FAST DATA SCANS

- Column orientation
  - Only load/scan what you need
- Compression
  - Decrease size of data in storage (RAM/disk)
FEATURE 4 – FAST FILTERING

- Indexed
  - Bitmap indices
- Compressed Bitmaps
  - CONCISE compression
  - Operate on compressed form
  - Resolve filters without looking at data
FEATURE 5 – OPERATIONALLY SIMPLE

- Fault-tolerant
  - Replication
- Rolling deployments/restarts
  - All node types have failover/replication
- Grow == start processes
- Shrink == kill processes
DRUID IN ACTION
SELECT hour(timestamp), lang, hashtag, sum(tweets) AS tweets
FROM twitter-spritzer
GROUP BY hour(timestamp), lang, hashtag
WHERE country = 'US' AND timestamp BETWEEN '2012-10-01' AND '2012-10-02';
SELECT hour(timestamp), lang, hashtag, sum(tweets) AS tweets 
FROM twitter-spritzer 
GROUP BY hour(timestamp), lang, hashtag 
WHERE country = 'US' AND timestamp BETWEEN '2012-10-01' AND '2012-10-02';

{
  "queryType": "groupBy",
  "dataSource": "twitter-spritzer",
  "intervals": ["2012-10-01/2012-10-02"],
  "granularity": "hour",
  "dimensions": ["lang", "hashtag"],
  "aggregations": [
    { "type": "longSum", "fieldName": "tweets", "name": "tweets" }
  ],
  "filter": {"type": "selector", "dimension": "country", "value": "US"}
}
QUERY FEATURES

- Group By
- Time-series roll-ups
- Arbitrary boolean filters
- Aggregation functions
  - Sum, Min, Max, Avg, etc.
- Dimensional Search
  - Explore values in your dimensions
DRUID VERSUS OTHERS

• vs Google Dremel
  • No indexing structure
• vs Google PowerDrill
  • Close analog, all in-memory
• vs Hadoop+Avro+Hive(+Yarn)
  • Closer to Tenzig
  • Back-office use case
DRUID BENCHMARKS

- Scan speed
  - ~33M rows / second / core
- Realtime ingestion rate
  - ~10k records / second / node
- Cluster Examples
  - 100 nodes,
  - 6 TB Memory, 400 TB disk
- 1.4 seconds 6TB in-memory query
DRUID IS NOW OPEN SOURCE
DRUID IS NOW OPEN SOURCE

Learn more:

METAMARKETS.COM/DRUID

GITHUB.COM/METAMX/DRUID
DRUID IS NOW OPEN SOURCE

Learn more:

METAMARKETS.COM/DRUID
GITHUB.COM/METAMX/DRUID

eric tschetter - cheddar@metamarkets.com
@ zedruid