Druid pour l'analyse de données en temps réel

Yann Esposito

7 Avril 2016

Intro

Plan

- Introduction; why?
- How?

Experience

- Real Time Social Media Analytics

Real Time?

- Ingestion Latency: seconds
- Query Latency: seconds

Demande

- Twitter: `20k msg/s`, `1msg = 10ko` pendant 24h
- Facebook public: 1000 à 2000 msg/s en continu

En pratique

- Twitter: 400 msg/s en continu, pics à 1500

Origine (PHP)

Introduction

- Traitement de donnée gros volume + faible latence
- Typiquement `pulse`

DEMO

Pre Considerations

Discovered vs Invented

Try to conceptualize (events)

Scalable + Real Time + Fail safe

- timeseries
- alerting system
- top N
- etc...

In the End

Druid concepts are always emerging naturally

Druid

Who

Metamarkets

Goal

Druid is an open source store designed for real-time exploratory analytics on large data sets.

hosted dashboard that would allow users to arbitrarily explore and visualize event streams.

Concepts

- Column-oriented storage layout
- distributed, shared-nothing architecture
- advanced indexing structure

Features

- fast aggregations
- flexible filters
- low latency data ingestion

arbitrary exploration of billion-row tables tables with sub-second latencies

Storage

- Columnar
- Inverted Index
- Immutable Segments

Columnar Storage

Index

- Values are dictionary encoded

```
{"USA" 1, "Canada" 2, "Mexico" 3, ...}
```

- Bitmap for every dimension value (used by filters)

```
"USA" -> [0 1 0 0 1 1 0 0 0]
```

- Column values (used by aggergation queries)

```
[2,1,3,15,1,1,2,8,7]
```

Data Segments

- Per time interval
- skip segments when querying
- Immutable
- Cache friendly
- No locking
- Versioned
- No locking
- Read-write concurrency

Real-time ingestion

- Via Real-Time Node and Firehose
- No redundancy or HA, thus not recommended
- Via Indexing Service and Tranquility API
- Core API
- Integration with Streaming Frameworks
- HTTP Server
- **Kafka Consumer**

Batch Ingestion

- File based (HDFS, S3, ...)

Real-time Ingestion

```
Task 1: [   Interval   ][ Window ]
Task 2:                 [              ]
---------------------------------------->
                                   time
```

Minimum indexing slots $=$
Data Sources $\times$ Partitions $\times$ Replicas $\times$ 2

Querying

Query types

- Group by: group by multiple dimensions
- Top N: like grouping by a single dimension
- Timeseries: without grouping over dimensions
- Search: Dimensions lookup
- Time Boundary: Find available data timeframe
- Metadata queries

Tip

- Prefer `topN` over `groupBy`
- Prefer `timeseries` over `topN`
- Use limits (and priorities)

Query Spec

- Data source
- Dimensions
- Interval
- Filters
- Aggregations
- Post Aggregations
- Granularity
- Context (query configuration)
- Limit

Example(s)

TODO

Caching

- Historical node level
- By segment
- Broker Level
- By segment and query
- `groupBy` is disabled on purpose!
- By default - local caching

Load Rules

- Can be defined
- What can be set

Components

Druid Components

- Real-time Nodes
- Historical Nodes
- Broker Nodes
- Coordinator
- For indexing:
- Overlord
- Middle Manager
- Deep Storage
- Metadata Storage
- Load Balancer
- Cache

Coordinator

Manage Segments

Real-time Nodes

- Pulling data in real-time
- Indexing it

Historical Nodes

- Keep historical segments

Overlord

- Accepts tasks and distributes them to middle manager

Middle Manager

- Execute submitted tasks via Peons

Broker Nodes

- Route query to Real-time and Historical nodes
- Merge results

Deep Storage

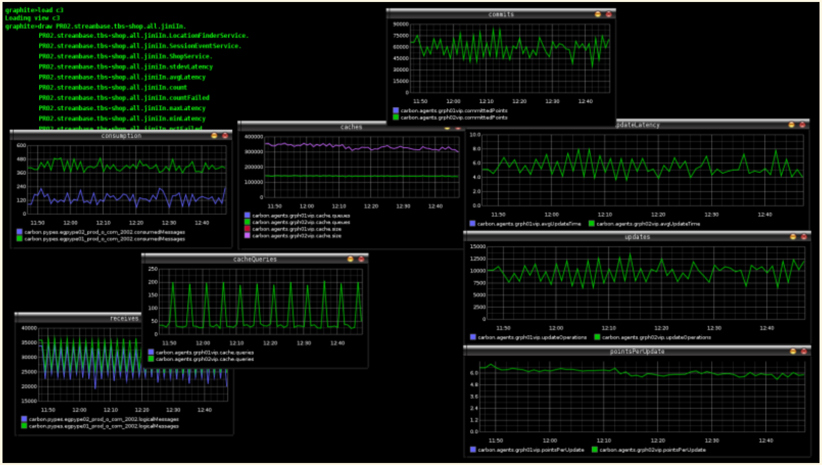
- Segments backup (HDFS, S3, ...)

Considerations & Tools

When *not* to choose Druid

- Data is not time-series
- Cardinality is *very* high
- Number of dimensions is high
- Setup cost must be avoided

Graphite (metrics)

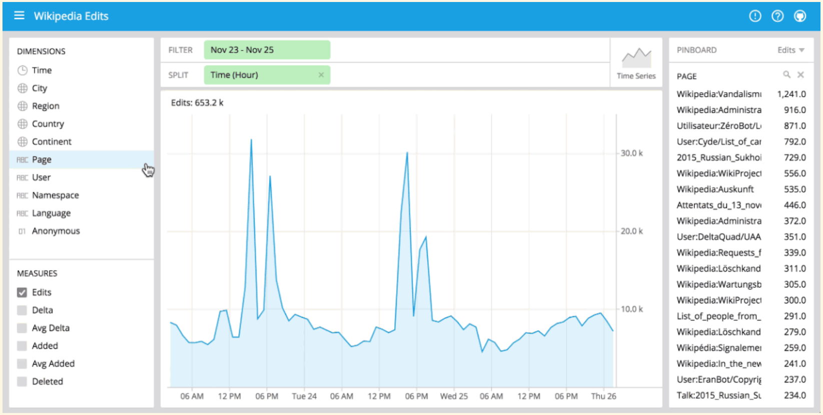


```
graphite>load c3
Loading view c3
graphite>draw PR02.streambase.tbs-shop.all.jiniIn.
          PR02.streambase.tbs-shop.all.jiniIn.LocationFinderService.
          PR02.streambase.tbs-shop.all.jiniIn.SessionEventService.
          PR02.streambase.tbs-shop.all.jiniIn.ShopService.
          PR02.streambase.tbs-shop.all.jiniIn.stdevLatency
          PR02.streambase.tbs-shop.all.jiniIn.avgLatency
          PR02.streambase.tbs-shop.all.jiniIn.count
          PR02.streambase.tbs-shop.all.jiniIn.countFailed
          PR02.streambase.tbs-shop.all.jiniIn.maxLatency
          PR02.streambase.tbs-shop.all.jiniIn.minLatency
          PR02.streambase.tbs-shop.all.jiniIn.pctFailed
```

commits
carbon.agents.grph01vip.committedPoints
carbon.agents.grph02vip.committedPoints
consumption
carbon.pypes.egpype02_prod_o_com_2002.consumedMessages
carbon.pypes.egpype01_prod_o_com_2002.consumedMessages
caches
carbon.agents.grph01vip.cache.queues
carbon.agents.grph02vip.cache.queues
carbon.agents.grph01vip.cache.size
carbon.agents.grph02vip.cache.size
updateLatency
carbon.agents.grph01vip.avgUpdateTime
carbon.agents.grph02vip.avgUpdateTime
cacheQueries
carbon.agents.grph01vip.cache.queries
carbon.agents.grph02vip.cache.queries
updates
carbon.agents.grph01vip.updateOperations
carbon.agents.grph02vip.updateOperations
receives
carbon.pypes.egpype02_prod_o_com_2002.logicalMessages
carbon.pypes.egpype01_prod_o_com_2002.logicalMessages
pointsPerUpdate
carbon.agents.grph01vip.pointsPerUpdate
carbon.agents.grph02vip.pointsPerUpdate

Graphite

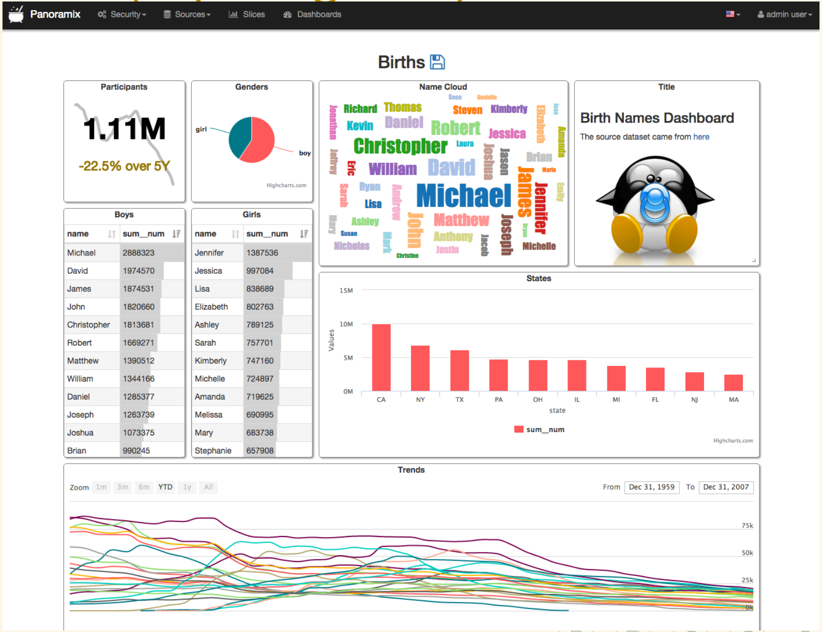
Pivot (exploring data)



PAGE	
Wikipedia:Vandalismı	1,241.0
Wikipedia:Administra	916.0
Utilisateur:ZéroBot/L	871.0
User:Cyde/List_of_car	792.0
2015_Russian_Sukhoi	729.0
Wikipedia:WikiProject	556.0
Wikipedia:Auskunft	535.0
Attentats_du_13_nov	446.0
Wikipedia:Administra	372.0
User:DeltaQuad/UAA	351.0
Wikipedia:Requests_f	339.0
Wikipedia:Löschkand	311.0
Wikipedia:Wartungsb	305.0
Wikipedia:WikiProject	300.0
List_of_people_from_	291.0
Wikipedia:Löschkand	279.0
Wikipédia:Signalemeı	259.0
Wikipedia:In_the_new	241.0
User:EranBot/Copyrig	237.0
Talk:2015_Russian_Su	234.0

Pivot

Caravel (exploring data)



Births

1.11M

-22.5% over 5Y

Title

Birth Names Dashboard

The source dataset came from here

Boys

name	sum__num
Michael	2888323
David	1974570
James	1874531
John	1820660
Christopher	1813681
Robert	1669271
Matthew	1390512
William	1344166
Daniel	1285377
Joseph	1263739
Joshua	1073375
Brian	990245

Girls

name	sum__num
Jennifer	1387536
Jessica	997084
Lisa	838689
Elizabeth	802763
Ashley	789125
Sarah	757701
Kimberly	747160
Michelle	724897
Amanda	719625
Melissa	690995
Mary	683738
Stephanie	657908

From Dec 31, 1959 To Dec 31, 2007