NoSQL Databases

Vincent.Garonne@cern.ch, Mario.Lassnig@cern.ch, Donal.Zang@cern.ch
CERN-PH-ADP

Maxim.Potekhin@bnl.gov
Physics Applications Software Group - BNL

ATLAS Software&Computing Workshop, April, 2011

Acknowledgements

Database administrators

- M. Blaszczyk
- L. Canali
- G. Dimitrov
- F. Tique Aires Viegas

Outline

- Relational Database Management System (RDBMS)
- The NoSQL complementarity
- First experiences and results
- Future plans

Relational Databases (RDBMS)

- Critical Oracle dependency for PanDA and $\mathcal{DQ}_2$
- Great for enforcing data integrity in distributed applications
  - Atomicity/transactions
  - Prevent orphan records/duplicates
  - Constraint with primary keys
  - Primary/Secondary indexes
  - Quick retrieval of data
- Cost for normalizing data
  - More tables, keys and indexes, table joins

RDBMS Limitations

- Hard to scale with data warehousing applications
  - As the databases grow larger, the queries start taking longer and longer
  - Non linear query execution time
  - Unstable query plans
  - Static schema

- Possible solutions
  - De-normalization
  - Flat schema
  - Data partitioning
  - New indexes

- Requirements to query extremely large datasets/logs/archives with fast query speeds

Example – $\mathcal{DDM}$

Grid Tracer service

- Record relevant information about data Access and Usage
- Key and critical component for ATLAS
  - Automatic cleaning of grid storages based on popularity
- $\sim$ 70 traces/second, $\sim$ 90 millions traces/month

Oracle issues

- Too static schema to store a lot of new metrics
  - Rate of requests/failures/transfer/etc.
  - Period: hour, day, month, year
  - Granularity: site, remotesite-localsite, users, etc
- Long query time for accounting report
  - Plots for management always for yesterday ☺

NoSQL Databases

- Google, Facebook, Amazon, Yahoo!
  - Process large amount of data at a petabyte scale
- NoSQL vs. RDBMS: Apples and Oranges ?
- Complementary technology to RDBMS
  - Eventual consistency (Brewer's CAP)
  - Schema-free
  - High throughput
  - Parallelism
  - Fault tolerant
  - Replication support
- Open source projects
  - Wide Column / Document / Key-Value Store
  - Cassandra vs Hadoop Hbase vs MongoDB vs Simpledb vs Dynamo vs Couchdb vs Hypertable vs Riak vs etc.

Experiences with NoSQL

Limited common test-bed cluster

- 4 VM nodes, 2 Intel/Xeon 2.27GHz, 4G
- Trivial deployment of NoSQL software

Applications relevant for NoSQL

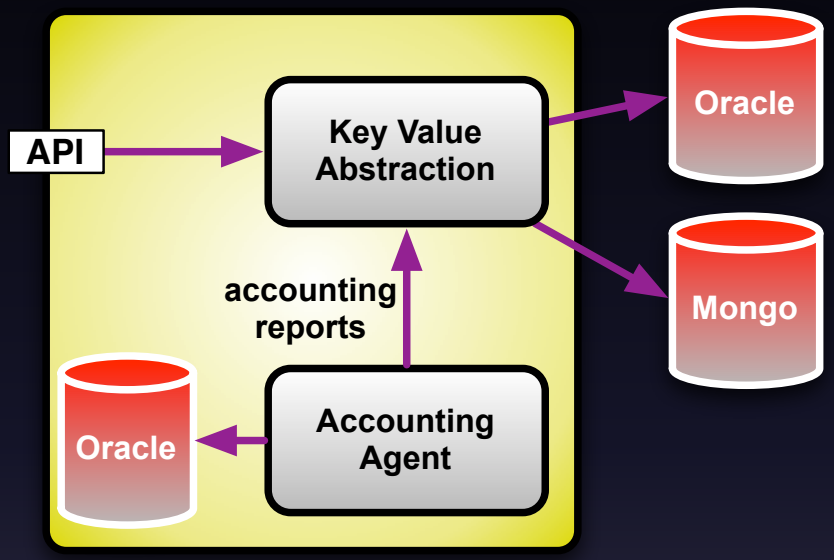
Application	NoSQL	Who
$\mathcal{DQ}_2$ Accounting	Hbase/MongoDB	M.Lassnig
PanDA historical data	Cassandra	M.Potekhin
$\mathcal{DQ}_2$ Tracer	Cassandra	D.Zang

$\Rightarrow$ No transactions and relaxed consistency

$\Rightarrow$ Schemaless, multi dimensional queries, lot of data and fast query speeds

MongoDB

$\mathcal{DQ}_2$ Accounting service



- Storage space and usage information
- Break down volumes by metadata information
  - E.g. location, datatype, custodiality
- Reports generated from Oracle
- One year of reports stored in two backends

	Retrieval	Setup	Used space	Dev. time
Oracle	0.3s	ADCR	38 GB	5 weeks + DBAs
MongoDB	0.4s	8 Cores/16G	42 GB	4 hours

Oracle	4 Tables, 243 Indexes, 2 Functions, 365 Partitions/Y+hints
MongoDB	1 Table, 1 Index

$\Rightarrow$ Schemaless, multi dimensional queries, lot of data and fast query speeds?

Cassandra

Overview

- Distributed database with no master node
- Automatic replication
- Large user community and commercial support
- Good responsiveness of developer team
- Infrastructure monitoring tools for free

Data model

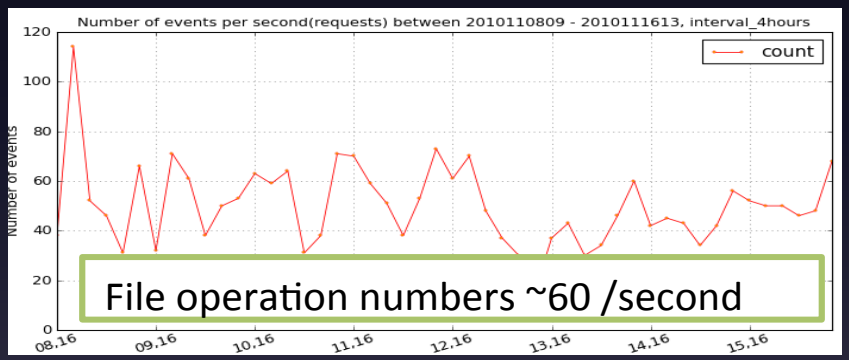
- Column, Key, Column family
- Analogy with persistent dictionnary
- E.g. data model for $\mathcal{DQ}_2$ Tracer monitoring

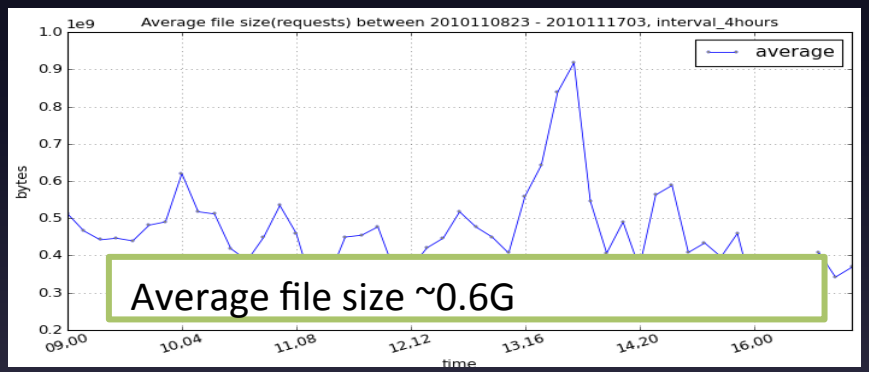
```
{'201011102105':
 {local_read:{count:11436,min:0.0,max:5507.0,avg:518.07}}
}
```

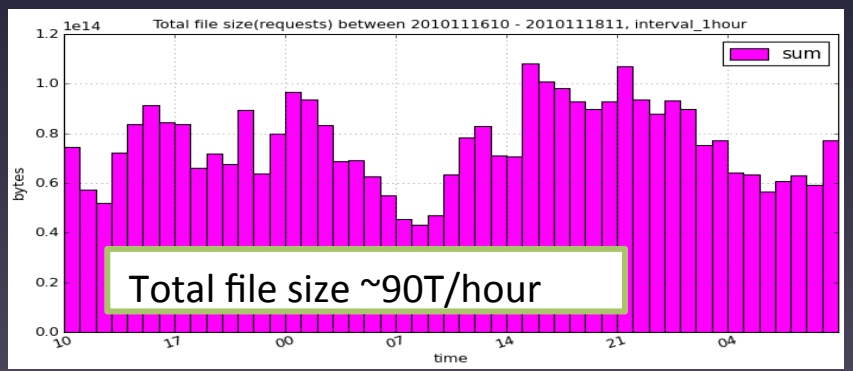
$\mathcal{DQ}_2$ Tracer monitoring

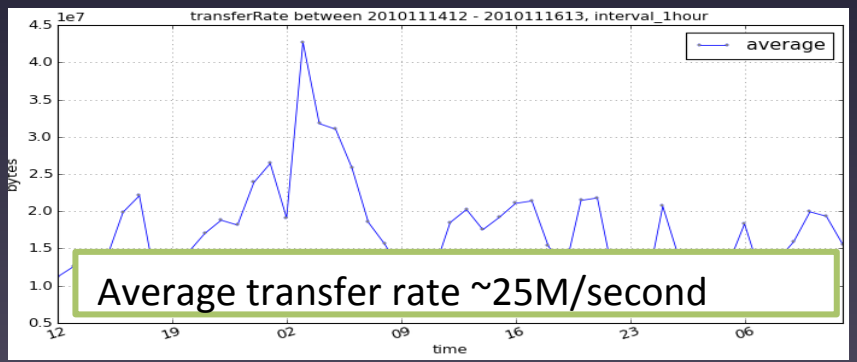
- Statistic metrics in Cassandra
- Generic monitoring on thousands of metrics

Tracer monitoring plots (based on statistic metrics in Cassandra)









Don Zang, PH-ADP/IHEP, 2011

Cassandra Scaling Tests

Test-bed issues

- Instability due to a not-so-optimal setup
- Too low-cost cluster
  - VM nodes
  - CPUs, Memory
  - No isolation between commit log and data file
- Good for learning and sharing experiences
  - Data design, queries
  - Cf. Maxim's talk - PanDA and NoSQL

Next steps

- Large insertion of data
- Large data analysis, e.g. map-reduce
- Best effort model for machines
  - $\sim$ 2/3 development machines, 8 Intel/Xeon 2.27GHz, 16G

Insertion speed: First results

Workload

Concurrency: 10 threads
Run time: 600s
Ramp up: 5s
Row: Tracer event

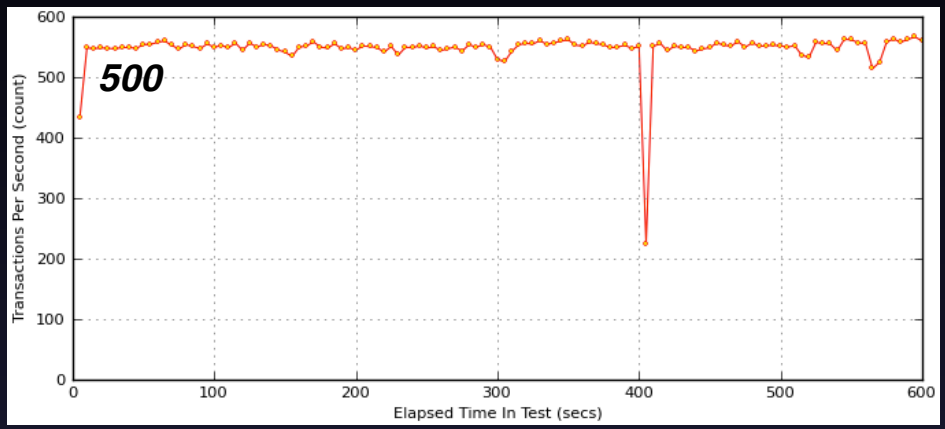
Python client

Oracle: cx_Oracle
MongoDB: pymongo
Cassandra: pycassa

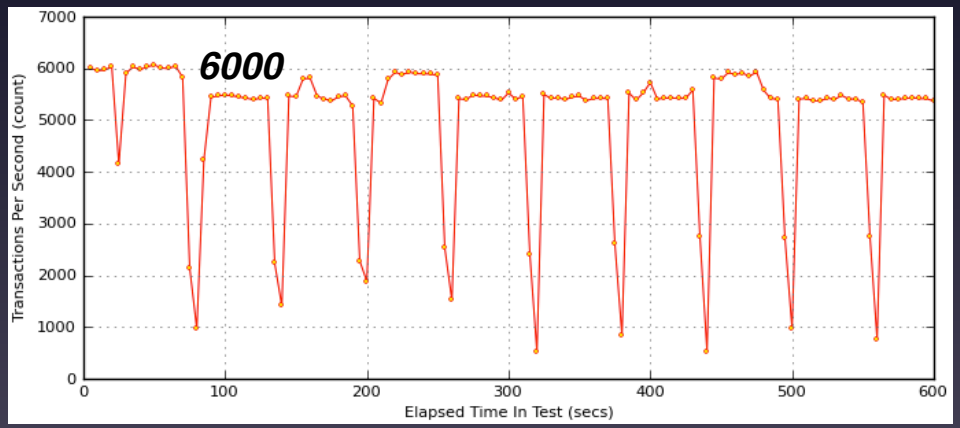
Setup

*Oracle: INTR 2 * 8 Cores 2.27HZ/ 24G*
*MongoDB: 3 * 8 Cores 2.27Hz/16G , 1 master, 3 slaves*
*Cassandra: 2 * 8 Cores 2.27Hz/16G, 2 replicas*

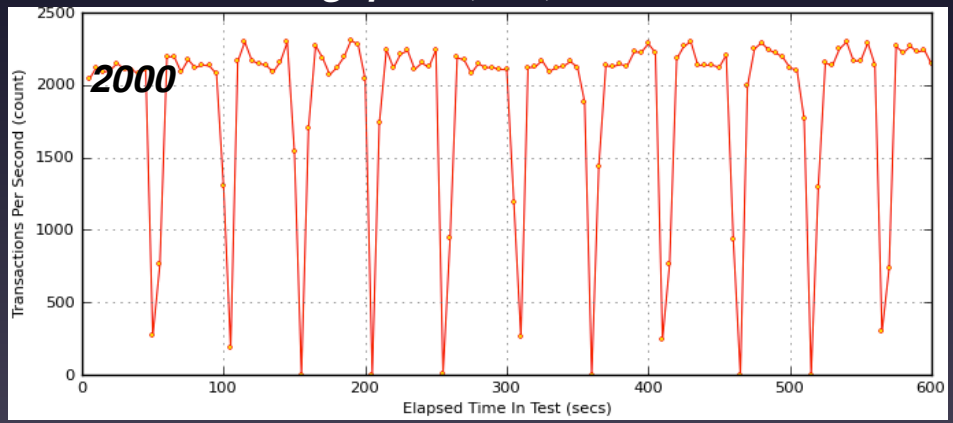
Oracle Throughput: 327,948 inserted rows



MongoDB Throughput: 29,900,053 inserted rows



Cassandra Throughput: 1,124,815 inserted rows



High write speed with NoSQL (buffering effect)

Next step: Read queries against dedicated hardware + hbase + tunings

NoSQL Summary

- Complementary to Oracle, like caching, for certain data warehousing applications
  - We do have such applications
  - More intuitive and flexible than Oracle
  - Save development time
- 3 NoSQL candidates: MongoDB, Cassandra, Hbase
- Need for a more powerfull test-bed to perform scaling tests
  - Large, random and I/Os intensive performance tests
  - Proposed test-bed setup [link]
- Happy to work with interested parties at CERN-IT
  - Expertise, testing facility and operational support