



Recorded Future

UNLOCK THE PREDICTIVE POWER OF THE WEB

MySQL in the Clouds

(aka Databases in the Clouds)

- Real world database management with large data

Anders Karlsson

Database Architect, Recorded Future

anders@recordedfuture.com

Agenda

- About Anders Karlsson
- About Recorded Future
- What's the deal with the Cloud
- What is Recorded Future doing in the cloud?
- Challenges? Fixes? What works? What doesn't?
- Questions? Answers?



About Anders Karlsson

- Database Architect at Recorded Future
- Former Sales Engineer and Consultant with Oracle, Informix, MySQL / Sun / Oracle etc.
- Has been in the RDBMS business for 20+ years
- Has also worked as Tech Support engineer, Porting Engineer and in many other roles
- Outside Recorded Future I build websites (www.papablues.com) develop Open Source software (MyQuery, mycleaner etc) am a keen photographer and has an affection for English Real Ales



About Recorded Future

- US / Swedish startup
- R&D in Sweden
- Funded by VC capital, among them Google Ventures, IA Ventures and others
- Sales currently mostly in the US
- Currently customers are mainly in the Finance and Intelligence markets



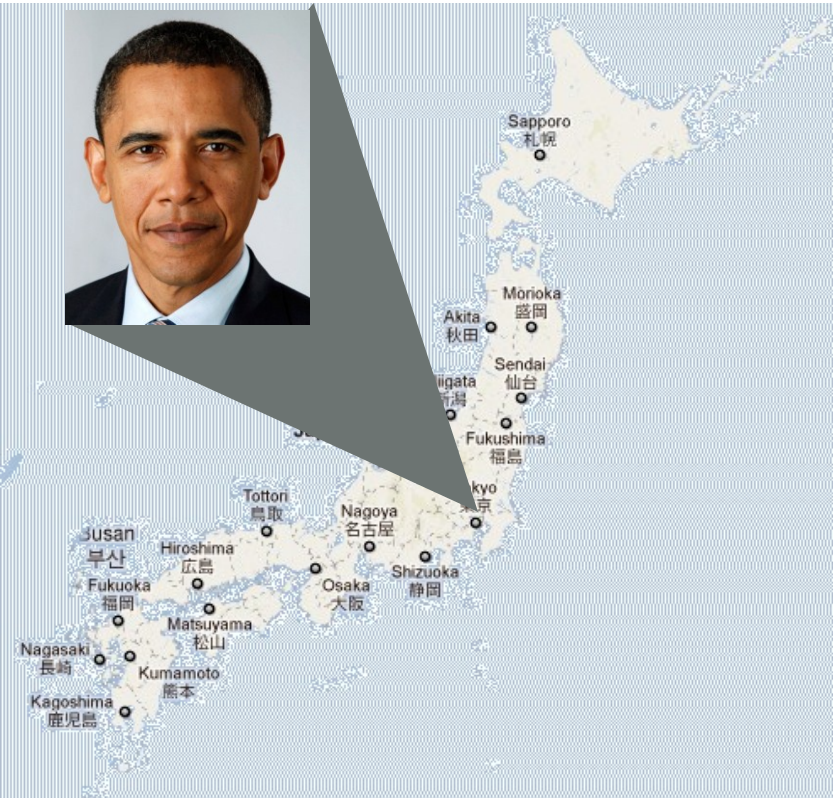
About Recorded Future

- Recorded Future does “Predictive Analysis” (Tagline: “Unlock the predictive power of the web”)
 - By scanning Twitters, Blogs, HTML, PDF, historical content and more
 - Add semantic, linguistic and orthogonal analysis to this content and compute a “momentum” to an entity
- Make this content searchable and compute a relevance
- Strong emphasis on temporal data / timeline

Web content analysis

Fact: *US President visits Tokyo*

Query: “*Barack Obama Japan*”



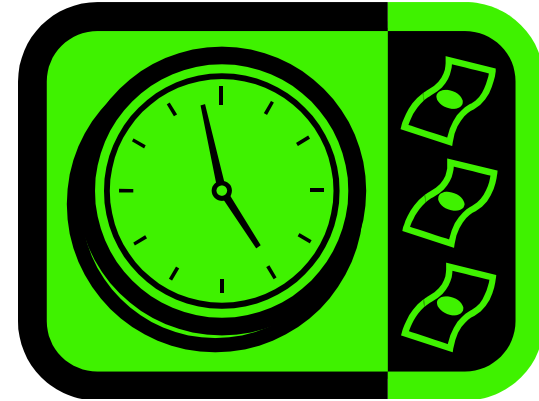
Will *Alta Vista* find this?

Google?

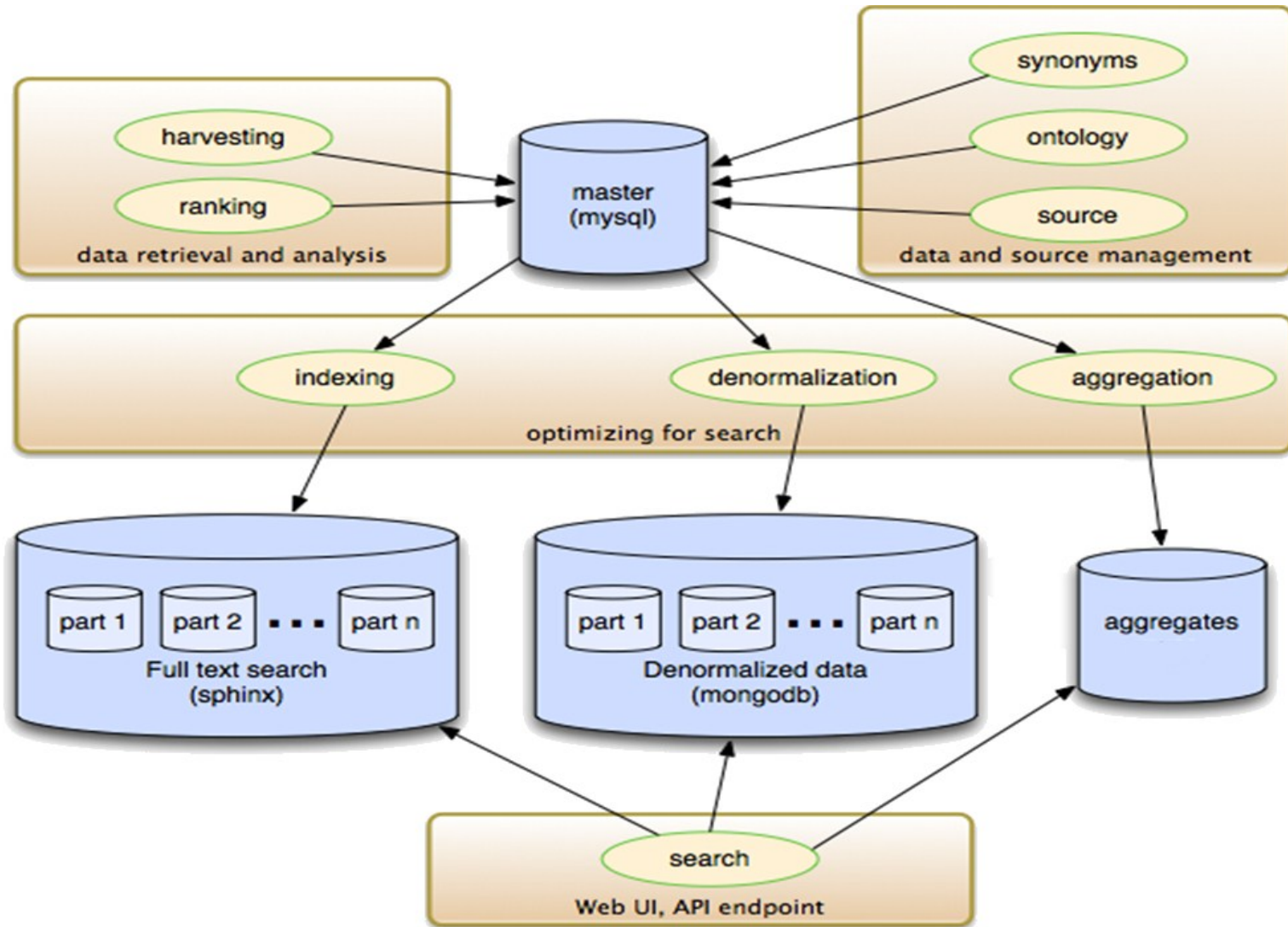
Recorded Future?

Temporal – It's about time!

- How is data ordered and searched?
 - Google – Ordered by “rank” (number of links to it and link “value”). No search by date / time except as text
 - Alta Vista – Who knows?
 - Recorded Future – Ordered by time of event, search by actual time (“travels by Barack Obama in 1997”)
- Search results order by time of actual event
 - “From 2009 and on, can I see Apples pressreleases in the order they were issued”?
 - “Can I see a timeline over events in Libya in the last year”?



Recorded Future inside



Back-end process flow in short

- Data enters from many sources into the MySQL Master database
 - While data is entered into the database, certain preprocessing is done, as much as can be done at this stage
- Other processing is applied to the data after loading -
Some processing, such as momentum computation, is applied to larger parts of the data set



UI Database processing in short

- The Master database data is replicated to several slaves for further processing, and is then copied to user-focusing databases:
 - Searchable data that is loaded into Sphinx
 - Sphinx searches results in an ID being returned
 - Denormalized content that is loaded into Mongo
 - Sphinx provided ID is used for lookup
 - Aggregates are in another MongoDB database
 - Again, Sphinx ID is used for lookups

Our challenges!

- 10x+ data growth within a year
- Within 2 years 100 times! At least!
- We are going where no one else has gone before
 - We have to try things
 - We have to constantly redo what we did before and change what we are doing today
 - We are extracting and adding temporal data, synonyms, entities, events etc. all the time
- At the same time, keep the system ticking: We have paying customers you know!
- Also, we run on Amazon EC2!

What's the NOT deal with the Cloud?

- It's probably NOT what you think
- It not always about saving money
- It's not about better performance just like that
- It's not about VMWare or Xen!
 - And even less about Zones or Containers or stuff like that!

What IS the deal with the Cloud

- Unmatched flexibility!
- Scalability, sort of!
- A chance to change what you are doing right now and move to a more modern, cost-effective and performant environment
- It is also about a hardware environment shared with others! Yikes!
- It is about the sum of all those things, assuming you are prepared to change!

What IS the deal with the Cloud

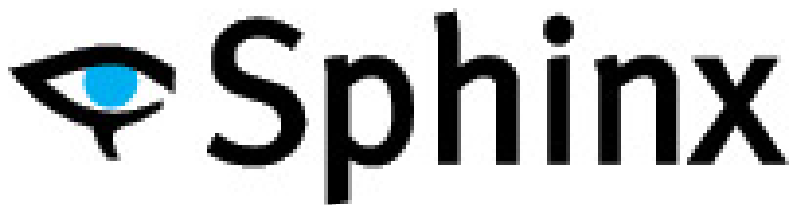
- Do not think of 25 servers. Or 13, or 5, or 58, or 279
 - Think of enough server to do the job today
- Think E! “The E is for elastic”
 - In hardware, infrastructure, applications, load etc.
- Think about massive scaling, up and down!
 - Today I need 5, by Christmas 87, when a run a special job 47 and tomorrow 2. Without downtime!
- Do NOT think 64Gb or 16Gb machines
 - Think more machines! Small or big, but more!

The Recorded Future Master database

- Runs MySQL 5.5
- Stores data in normalized form, but not enforced using Foreign Keys
- Not using sharding currently
- Runs on Amazon EC2 (not using Amazon RDS)
- Database currently has 71 tables and 392 columns, whereof there are 10 BLOB / TEXT columns
- Database size is about 2 Tb currently

The Search database

- The Search database is, as the name implies, used for searching
 - Uses Sphinx full-text search engine
 - Sphinx version 0.9.9
 - Sharded across 3 + 1 servers currently
 - Occupying some 500 Gb in size



Our MongoDB setup

- We are currently using MongoDB version 1.8.1 and 2.0.0
- Size of the MongoDB database is about 2.5 Tb
- We distribute the MongoDB database over multiple servers using MongoDB sharding
- We safeguard by using MongoDB Replica Sets and Amazon EC2 Snapshots for backups



Our Amazon EC2 setup

- We currently have some 74 EC2 instances running *Ubuntu*
- 663 EBS volumes are attached amounting to a total of 115 Tb
- Most of the instances are m1.large (2 cores 8 Gb). We have 33 of these right now
- MySQL Nodes are m2.4xlarge (8 Cores, 68 Gb RAM). We have 17 of these currently

Our Amazon EC2 setup

- We use LVM stripes across EC2 Volumes
- For snapshots we use EC2 snapshots, not LVM snapshots
- XFS is used as the file system for all database systems, allowing striped disk to be consistently backed up with EC2 snapshots
- XFS Freeze is a good tool for us here
- XFS is also a good choice of file system for databases in general

What works in EC2

- Amazon EC2 volumes are a great way of managing disk space
- The EC2 CLI is powerful and useful (albeit slow)
- The Instances has reasonably predictable CPU performance
- Backups through EC2 snapshots are great
- Data migration across machines is easy
- Integration with Operating System works reasonably well



What we are not so happy with in EC2

- Network performance is mostly OK, but varies way too much
- DNS lookups are probably smart but makes a mess of things, and some software doesn't like the way the network is set up too well
- Disk IO throughput is not great, latency even worse, and varies way too much!
- Disk writes are REAL slow and REAL unpredictable

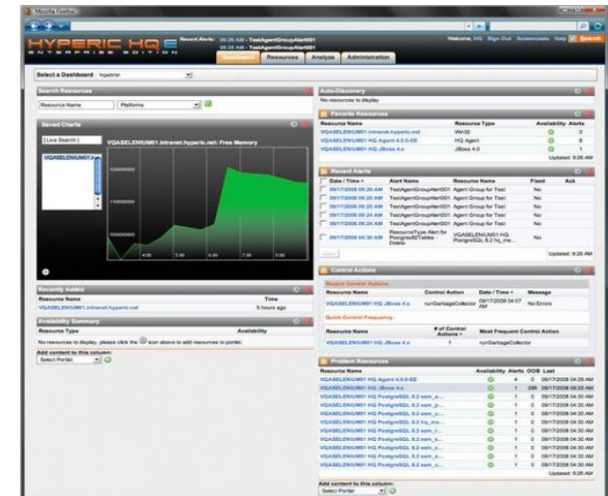


How do we manage all this?

- OpsCode / Chef is used for managing the servers and most of the software
 - We have done a lot of customization to the standard chef recipes, and many are written from scratch
 - My personal opinion: chef is a good idea, but I'm not so sure about the implementation. I like it better now than when I first started using it
 - Do you need a configuration management solution at all?
You bet!

How do we manage all this?

- Hyperic is used for monitoring
 - Is it better than, say, Nagios? Possibly, but not by a lot if that is the case.
 - Will it scale with the rest of our infrastructure? Nope, far from it.
 - Do we have an alternative? Nope, not yet.
 - Monitoring real big setups is difficult!



Things we are fixing!

- The single MySQL Master design has to be changed somehow (I'll let you in on a secret here. Listen up now!)
 - This is more difficult for us than in many other cases, as our processing does a lot of references to the database, and there is no good natural sharding key. Also, we are write intensive! Also, we need both throughput (for the backend) and low latency (for the frontend)
 - We are on the lookout for other database technologies
 - NuoDB looks cool, Drizzle could do us some good also, we are also looking at InfoBright or similar for aggregates



Things we are fixing!

- Better scaling! Much better scaling!
- Make better use of chef/opscode for automation
 - Better use of chef databags
 - Better use of chef environments
- More use of MongoDB!
- Backups using EC2 snapshots!
 - No, mysqldump / mongodump is **no good at all** for 10'ish Tb of data!



MySQL experiences

- MySQL OSS tools has been developed at Recorded Future
 - *slaveaccelerator* – Speeds up slaves. We need and use this, and it makes good sense.
 - *sqlstats* – MySQL 5.5 plugin to look at frequently executed SQL statements
 - *slaverestart* – Script to restart a slave automatically
 - *MyQuery* – Windows MySQL Query tool
 - *myCleaner* – Database cleanup tool

Will MySQL cut it?

- Maybe, but not for everything, we already use MongoDB and Sphinx for more and more things (these technologies have issues of their own)
- My main gripe with MySQL: It's not elastic. And few databases are!
- MySQL Cluster is possibly closer, but: can I add 2 more servers? Just like that? And get increased write performance? And then take them away when I no longer need them? Nope.

Will sharding cut it?

- This is something you hear all the time. Sharding! Automatic (as in MongoDB), application based (as with MySQL), etc. etc.
- Will it help then? Maybe, maybe not!
 - Scaling with sharding means rebalancing eventually, which can be slow and error-prone!
 - Sharding / balancing itself introduce an overhead. If that grows with scale (which it mostly does), then there is a limit to how far you can scale! And it might be closer up than you think!

Issues with scale / Big data

- Data consistency
 - Is it needed? Hopefully not!
 - Can it even be achieved at reasonable cost with 10's of Tb of data? Yes!
- What is the missing link?
 - Software! Hey, come on, I can add CPUs, memory and disk more or less on the fly, why can't I add another database shard on the fly?
- Are the vendors understanding Big Data and Cloud Environments? Nope. Mostly not (but if you feel differently, talk to me, I will listen)!

Things that will change!

- We will manage A LOT more data
 - +10 times more in a not too distant future!
 - +100 times more in the mid-term
- We need to find a way to track usage of our data, and to balance frequently used data with not so frequently
- We must become careful with how we manage disks and instances! This is getting expensive!



The E is for Elastic! And **don't you forget it!**

- Don't expect to solve performance problems by getting a bigger EC2 instance / server
 - Just don't do it, don't even **THINK** about it!
- Prepare for an architecture where every service either:
 - Is stateless (web servers, app servers)
 - Can be sharded
- Shared disk systems? **Beep! Wrong answer!**
- Relying on distributed locks in the network? Bad idea, unless some caution has been taken

Questions and Answers

anders@recordedfuture.com

