

# Large Datasets on Amazon EC2

Anders Karlsson

Database Architect, Recorded Future

[anders@recordedfuture.com](mailto:anders@recordedfuture.com)

Recorded Future

# Agenda

- **About Anders Karlsson**
- **About Recorded Future**
- **What's the deal with the Cloud**
- **How Recorded Future Works**
- **How Recorded Future works in the Cloud**
- **What are our EC2 experiences so far**
- **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](http://www.papablues.com)), develop Open Source software (MyQuery, ndbtop etc), am a keen photographer and drives sub-standard cars, among other things



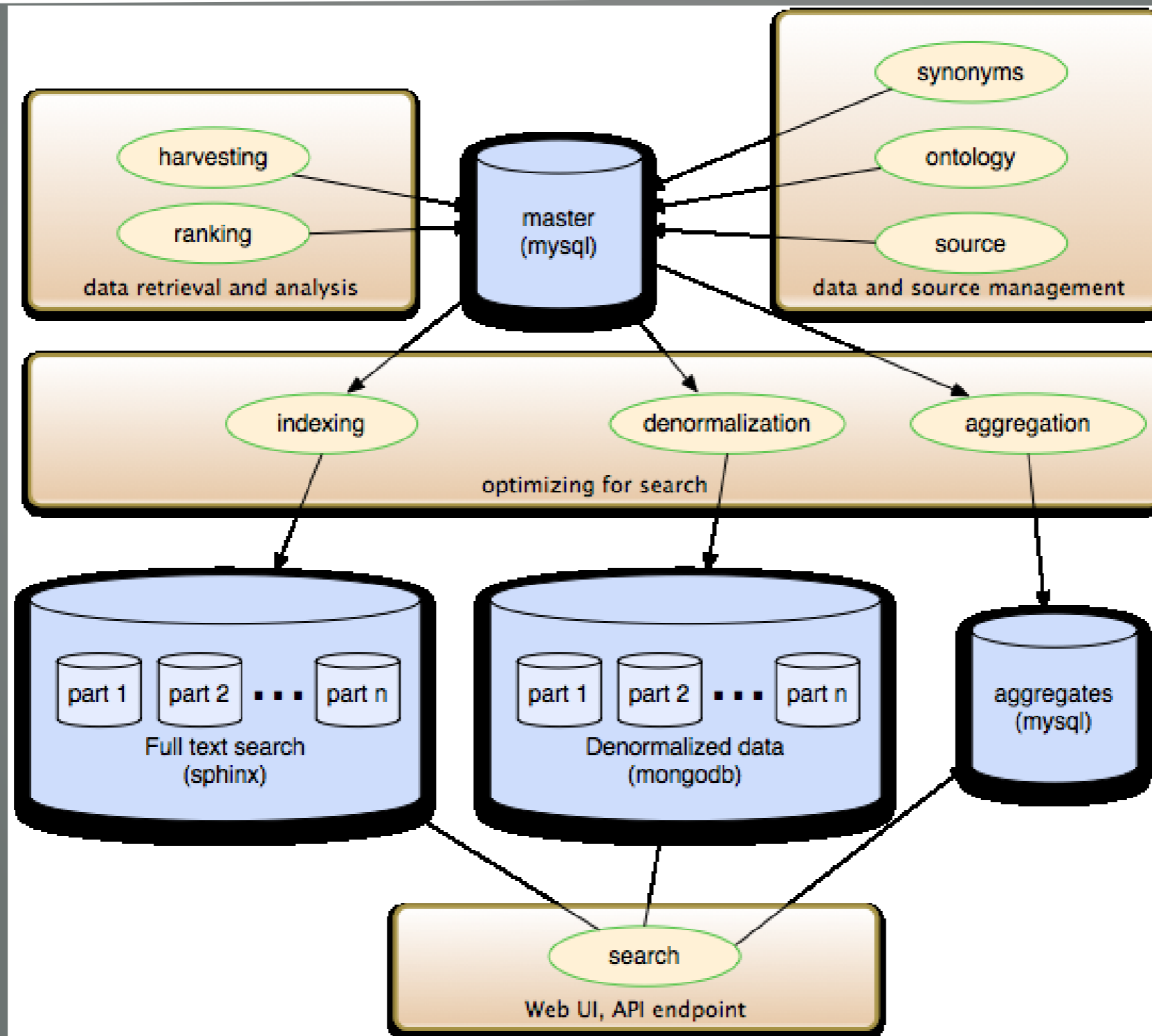
# About Recorded Future

- **US / Swedish company with R&D in Sweden**
- **Funded by VC capital, among them Google Ventures and others**
- **Sales mostly in the US**
- **Customers are mainly in the Finance and Intelligence markets, for example In-Q-Tel**

# About Recorded Future

- **Recorded Future is “predicting the future by analyzing the past” (Predictive Analytics)**
  - By scanning Twitters, Blogs, HTML, PDF, historical content and more
  - Add semantic and linguistic analysis to this content and compute a “momentum” to an entity
- **Make this content searchable and use the momentum to compute a relevance**

# Recorded Future inside



# The deal with Recorded Future

- You can subscribe to *Futures*. This is an email-based free service
- On-Line Web user interface is the second level of users. This is paid for by seat
- API access is more advanced, for users wanting to export data and possibly integrate it with their own data
- Local install is for users that want to apply their own data to our analytical tools.

# 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**

# 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 stored in another MySQL instance
    - 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
- **At the same time, keep the system ticking: We have paying customers you know!**

# What's the NOT deal with the Cloud?

---

- **It's probably NOT what you think**
- **It not about saving money (only)**
- **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 performance environment**
- **It is about 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**
  - 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 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 1Tb 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



# The Key-Value database

- **A Key-Value database is good for:**
  - “Here is a key, give me the value” type operation
  - Has limited functionality compared to an RDBMS
  - BUT: Distributed operation, scalability and performance compensates for all that
- **We currently use MongoDB as a KVS**

# Our MongoDB setup

- We are currently using MongoDB version 1.8.0
- Size of the MongoDB database is about 500 Gb
- We distribute the MongoDB database over 3 shards
- We do not use MongoDB replication



# Our Amazon EC2 setup

- **We currently have some 40 EC2 instances running *Ubuntu***
- **341 EBS volumes are attached amounting to a total of 38 Tb**
- **The majority of the instances are m1.large (2 cores 8 Gb). We have 16 of these**
- **MySQL Nodes are m2.4xlarge (8 Cores, 68 Gb RAM). We have 12 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 is also a good choice of file system for databases in general**

# Our application code

- **We use Java for the core of the application**
- **This is supported by Ruby, Python and bash scripting**
- **Among the supporting code is my Slavereadahead utility to speed to the slaves**
- **Data format through is JSON nearly everywhere**



# What works

- Amazon EC2 volumes are a great way of managing disk space
- The EC2 CLI is powerful and useful
- The Instances has reasonably predictable CPU performance
- Backups through EC2 snapshots are great
- Integration with Operating System works well



# What we are not so happy with

- 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



# 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

# How do we manage all this?

- **Hyperic is used for monitoring**
  - A mix of homebrew, modified and special agent scripts are used
  - Both Infrastructure components, such as databases and operating systems, as well as application specific data is monitored
  - This is still a work-in-progress, largely

# Things that we must fix!

- The single MySQL Master design has to be changed somehow
  - 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
  - We are on the lookout for other database technologies
    - AmbusDB looks cool, Drizzle could do us some good also, we are looking at InfoBright or similar for aggregates



# Things that will change!

- **We will manage A LOT more data**
  - +10 times more this year, at least!
  - +100 times more next year
- **We need to find a way to track usage of our data, and to balance frequently used data with not so frequently**
- **We must be become careful with how we manage disks and instances! This is getting expensive!**

# How to make good use of EC2

- **Do not think that Amazon EC2, or any other cloud, is just a Virtual Environment, and nutin' else**
- **Vendor asked for Cloud support: “Yeah, we run fine on VMWare”**
- **If you run it on a local Linux box today, it will almost certainly work on EC2, but:**
  - **It might be that it doesn't work well**
  - **It might well be that it's NOT cost-effective**

# 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
- **Prepare for an architecture that every service:**
  - Is stateless (web servers, app servers)
  - Can be sharded
- **Shared disk systems? Bad idea**
- **Relying on distributed locks in the network? Bad idea, unless some caution has been taken**

# The E is for Elastic! Really!

- **Don't for one second expect that Software vendors understand how proper cloud computing works. And that's pretty much OK!**
- **Don't expect Amazon folks to know or understand it**
  - They built a solid technical infrastructure, but how to reap the benefit of that, that is left to you!
- **Do not never, ever, assume it's cheaper just because it's in a cloud! It's more to it than that! Much more!**

# Questions and Answers

---

[anders@recordedfuture.com](mailto:anders@recordedfuture.com)