Cloudant

The Do More Data Layer

NOSQL

The WTF Edition

with Sam Bisbee

Before We Begin

- I am focusing on operational databases
  - This essentially strikes out all Hadoop based tech (HBase, Cassandra, etc.)
- If you want to build a data warehouse, go use Hadoop / HBase
- Feel free to ask me why I think data warehouses are a bad idea after this talk

What is NOSQL?

- ”Not Only SQL”
- 1870: Modern study of set theory begins
- 1970: Codd writes ”A Relational Model of Data for Large Shared Data Banks”
- 1970 – 1980: Commercial implementations of Codd's theory are released

What is NOSQL?

- 1970 – ~2000 the same sorts of databases were being made (sans a few niche productions)
- Dot-Com Bubble forced the same problems but at a new scale (Amazon), forcing innovation
- 2000 – present: innovations are becoming open source and ”main stream” (Hadoop)

So What is NOSQL Really?

New ways of looking at dynamic data storage

and querying for larger scale systems.

(scale = concurrent users and data size)

What's Wrong with Rows & Columns?

- Rows and columns are for 2D data structures
  - Multi-dimensional structures were solved with JOIN
  - Mapping these 2D structures requires tightly coupled mapping structures (3rd NF, etc.)
  - This was optimal for how computers ”think”, not humans
  - This was suboptimal for developers – change is hard
- Humans think in loosely coupled relationships like ”documents” and in >2D

Why Not Repurpose SQL?

- SQL is an implementation of relational algebra
- Relational algebra assumes set theory
- Most NOSQL databases are not based on set theory
  - CouchDB, MongoDB, Hadoop, etc.
- All attempts to square-peg-round-hole this problem have reproduced programming languages

If Not SQL, then What?

- Primarily...
  - K/V
  - MQL
  - MapReduce
  - Incremental MapReduce
- Relatively young area of dev/study that was distracted by Google's MapReduce white paper
- More interesting is the integrations with alt. solutions

Which Solution Should I Use?

Which Solution Should I Use?

What am I, your database shrink?

Which Solution Should I Use?

- What types of questions do you need to ask your database and how long can you wait for answers?
- Does all your data fit in RAM?
  - You can have 100s of GBs of RAM per box now
  - **AWS's cr1.8xlarge has 244GB of RAM**
- What are your consistency and durability needs?

Consistency?

- Do all your servers in a data center need to agree?
  What about across >1 data center (multi-geo)?

- Answer: probably not.

- CAP Theorem

  (Consistency, Availability, Partition Tolerance)

- Embrace eventual consistency

Durability?

- Can you lose data? If yes, how much?
- If yes, do write transactions to RAM
  (MongoDB, CouchBase, Redis, etc.)
- If no, then write transactions to disk
  (CouchDB, Cloudant, HBase, etc.)
- When using disk, make sure it's FAST
  - SSD prices dropping faster than 15k RPM SAS

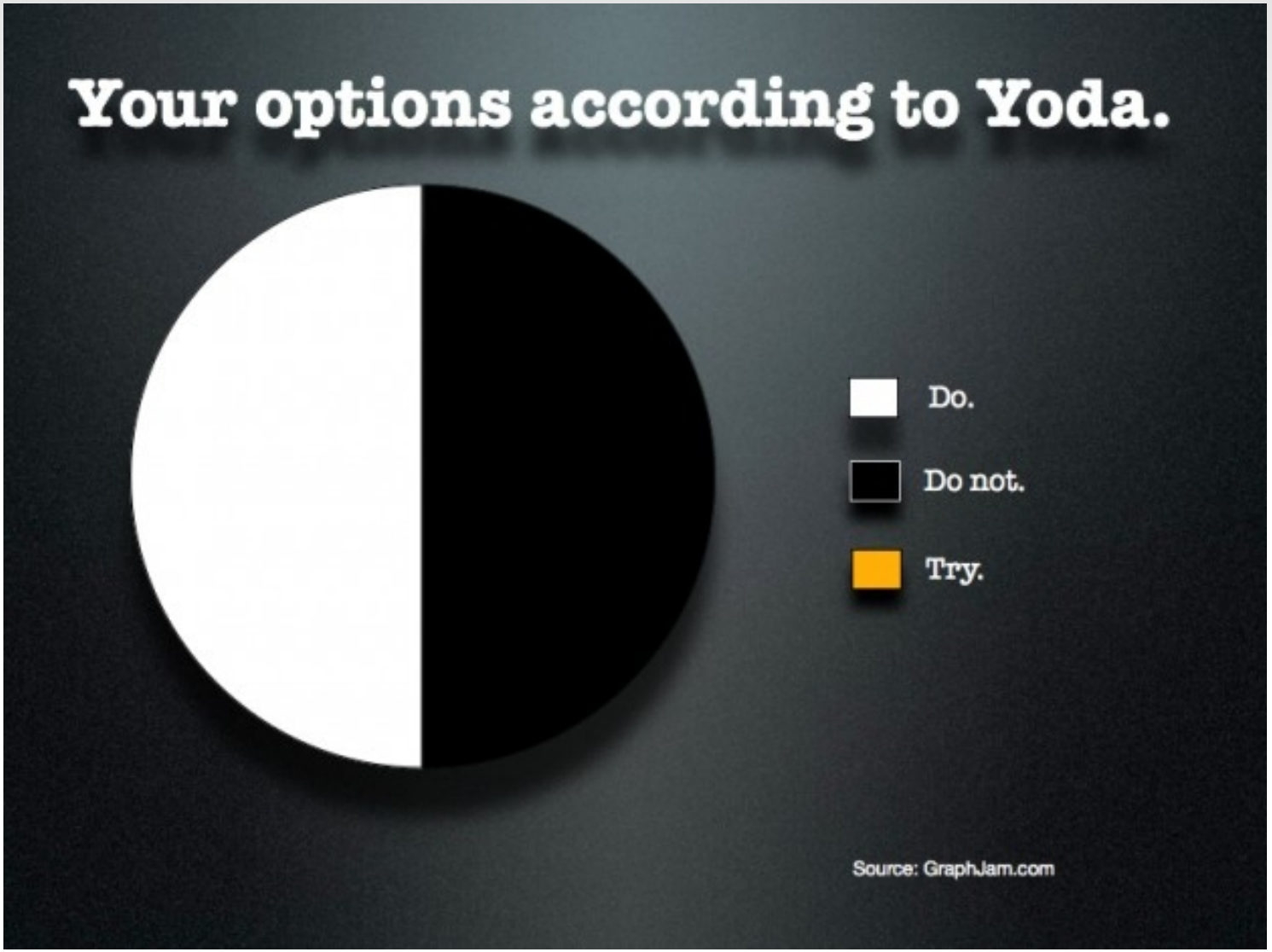
Concurrent Connections, Your DB's Heart Attack

- Web and mobile differentiate themselves by the amount of IO generated by a given user
- [Tell those 2 customer stories]
- You are now dealing with ”classic” scaling issues that beg for horizontal scaling solutions
- Most databases don't scale horizontally very well
  - Look at BigCouch, Cloudant, and Riak for hor. scale

Where and How am I Deploying My DB?

- Don't.
  - as-a-Service is a *good* thing
  - Focus on your core competency – if it's databases then you are likely bored right now
- Do.
  - There are tech heavy cultures that ”require” DIY
  - Watch these orgs carefully – they're probably reinventing wheels instead of building product
  - If they built their own database, might want to fire them (there are exceptions)

Where and How am I Deploying My DB?



How Do I DIY in "The Cloud (TM)(C)"

- Assuming your data size and TPS is non-trivial
- Virtual was optimized for app servers
- EC2 has horrible networking
  - Impacts clustered database environments
  - All disks sit behind a network connection, heavily impacting storage IOPS
- Joyent is a great contender w/ disks local to the rack

When Should I Use Relational?

- All of the reasons are cultural, not technical
- If all your developers know one relational database, don't want to learn something new, and/or you're launching in <4 months
- New database tech is not a good reason to rewrite an application – find a better reason or new application/project


Cloudant
The Do More Data Layer