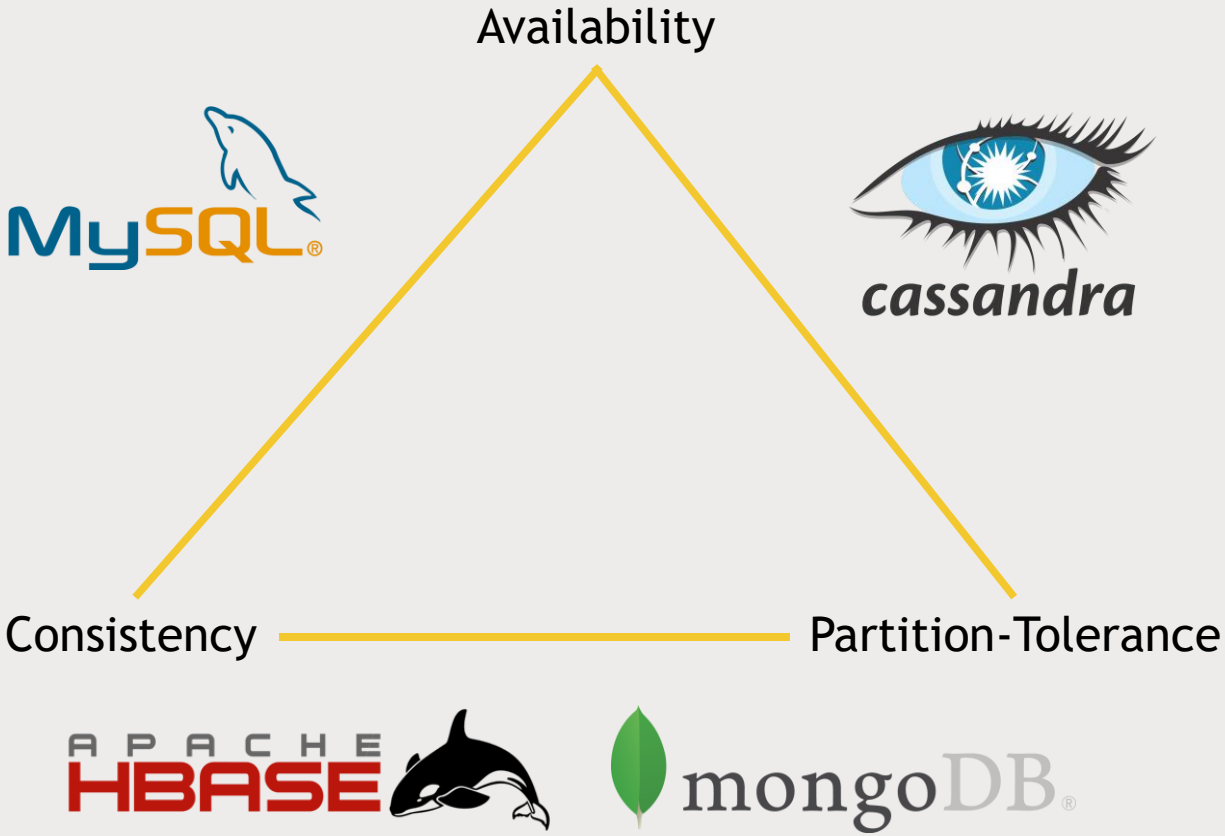
MONGODB

Managing HuMONGOus data

Where are we?


Availability
MySQL
cassandra
Consistency
Partition-Tolerance
APACHE HBASE
mongoDB

Document-based data model

Looks like JSON. Example:

```
{
        “_id” : ObjectID(“7b33e366ae32223aee34fd3”),
        “title” : “A blog post about MongoDB”,
        “content” : “This is a blog post about MongoDB”,
        “comments”: [
                {
                        “name” : “Frank”,
                        “email” : fkane@sundog-soft.com,
                        “content” : “This is the best article ever written!”
                        “rating” : 1
                }
        ]
}
```

No real schema is enforced.

- You can have different fields in every document if you want to
- No single “key” as in other databases
  - *But you can create indices on any fields you want, or even combinations of fields.*
  - *If you want to “shard”, then you must do so on some index.*
- Results in a lot of flexibility
  - *But with great power comes great responsibility*

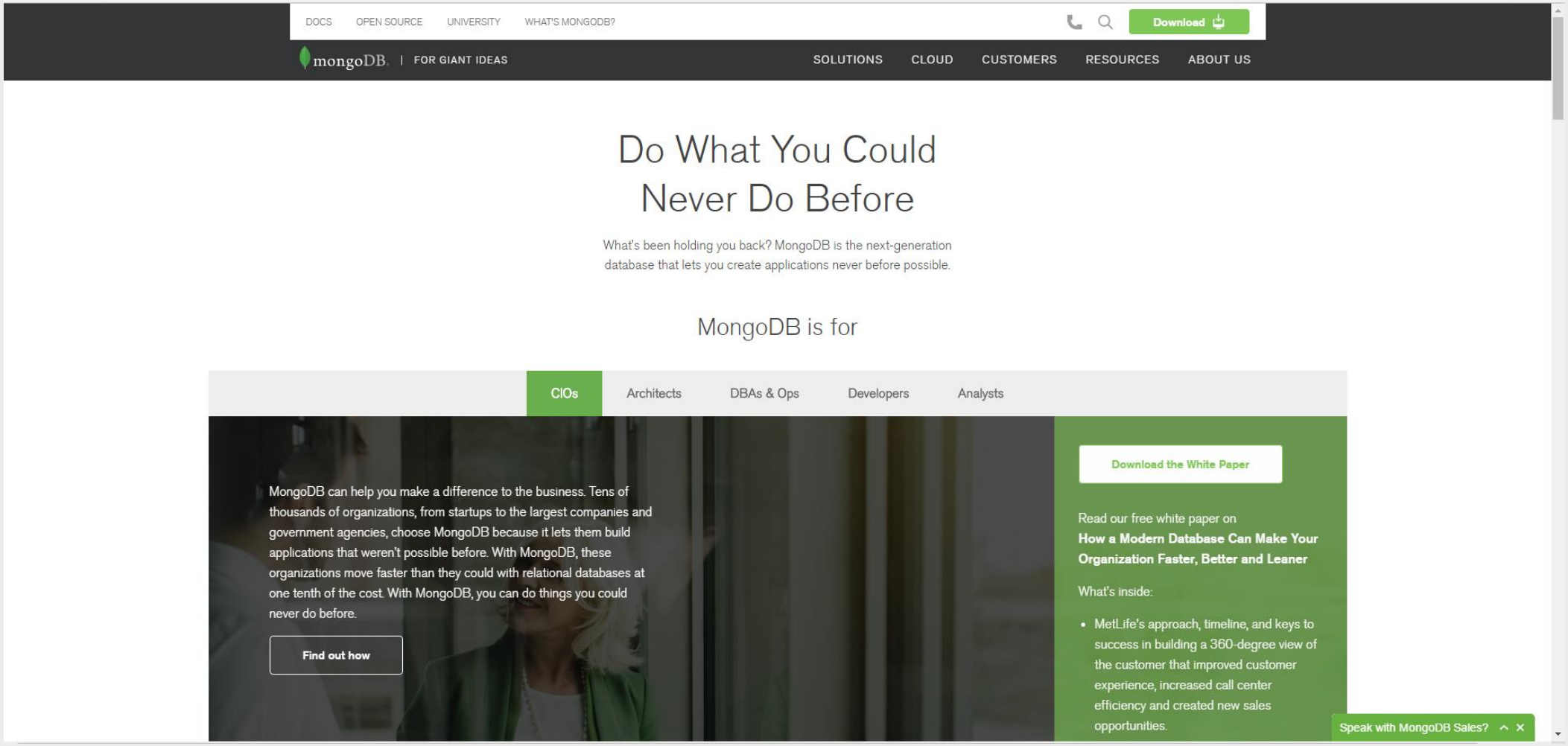
partitioning

design db based on queries that support requirement

MongoDB terminology

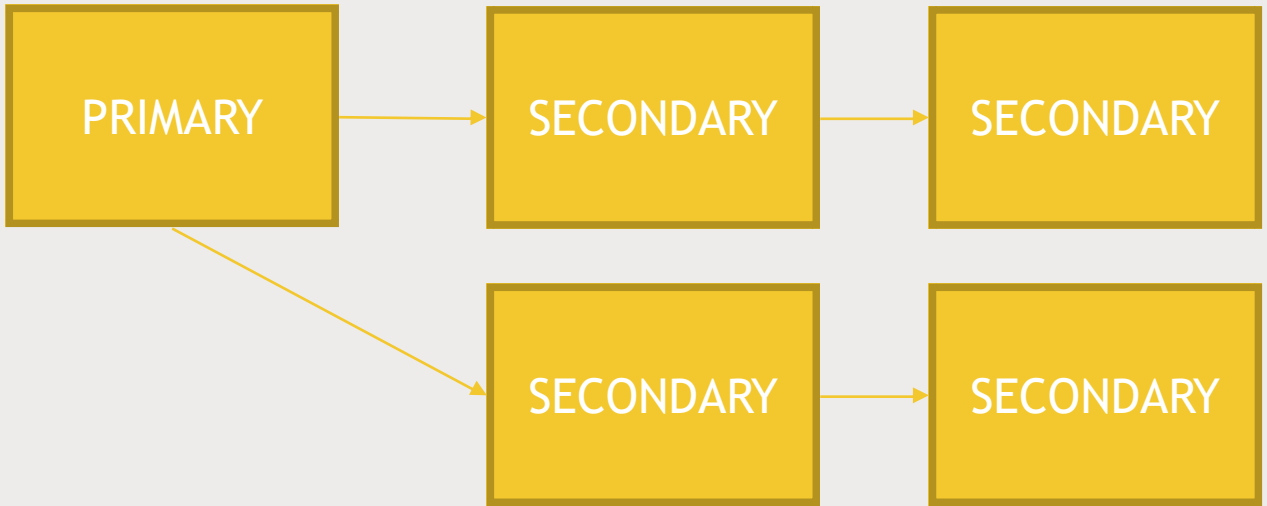
- Databases — dababases
- Collections — tables
- Documents — rows

It’s kinda corporate-y



Replication Sets

- Single-master!
- Maintains backup copies of your database instance
  - *Secondaries can elect a new primary within seconds if your primary goes down*
  - *But make sure your operation log is long enough to give you time to recover the primary when it comes back…*

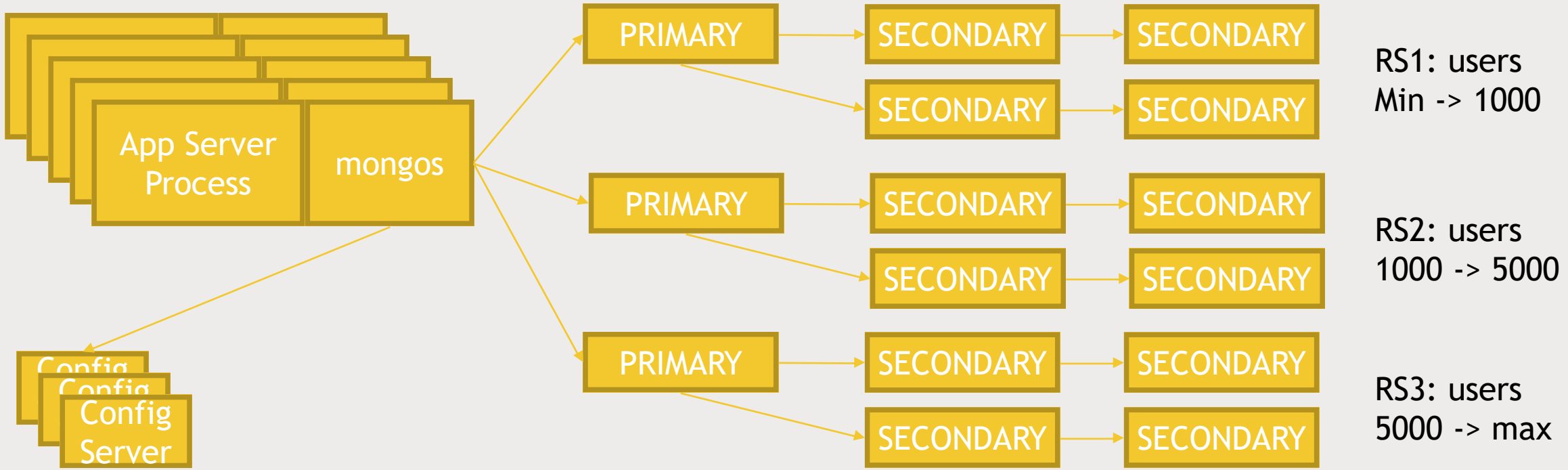


Replica Set Quirks

- A majority of the servers in your set must agree on the primary
  - *Even numbers of servers (like 2) don't work well*
- Don't want to spend money on 3 servers? You can set up an 'arbiter' node
  - *But only one*
- Apps must know about enough servers in the replica set to be able to reach one to learn who's primary
- Replicas only address durability, not your ability to scale
  - *Well, unless you can take advantage of reading from secondaries - which generally isn't recommended*
  - *And your DB will still go into read-only mode for a bit while a new primary is elected*
- Delayed secondaries can be set up as insurance against people doing dumb things

Sharding

- Finally – “big data”
- Ranges of some indexed value you specify are assigned to different replica sets



Sharding Quirks

limitation

- Auto-sharding sometimes doesn't work
  - *Split storms, mongos processes restarted too often*
- You must have 3 config servers
  - *And if any one goes down, your DB is down*
  - *This is on top of the single-master design of replica sets*
- MongoDB's loose document model can be at odds with effective sharding

Neat Things About MongoDB

- It’s not just a NoSQL database – very flexible document model
- Shell is a full JavaScript interpreter
- Supports many indices
  - *But only one can be used for sharding*
  - *More than 2-3 are still discouraged*
  - *Full-text indices for text searches*
  - *Spatial indices*
- Built-in aggregation capabilities, MapReduce, GridFS
  - *For some applications you might not need Hadoop at all*
  - *But MongoDB still integrates with Hadoop, Spark, and most languages*
- A SQL connector is available
  - *But MongoDB still isn’t designed for joins and normalized data really.*

Let’s Mess Around

- We’ll integrate MongoDB with Spark
- Then play around with the resulting database in the mongo shell