MongoDB
Horizontal scaling with Sharding
Simple Doc Store

Not even Mongo, just a file system
With replication, the database becomes more resilient to hardware problems, network outages, improved READ performance, and even geopolitical issues.
Vertical Scaling

Add more …

• storage
• memory
• CPU power

• = $$$$  

• Cloud providers only offer certain configurations
Sharding

- mongod runs replicas
- Config servers manage the shards
- mongos routes queries to shards
To Shard or not to Shard?

Sharing is complicated, make sure you really need it

- Running out of disk space and can’t add more?

- MongoDB works best with indexes and most common queries are kept resident in RAM.

- If you start running out of RAM, your queries will slow down and your memory accesses will begin to thrash
Sharding

- mongod runs replicas
- Config servers manage the shards
- mongos routes queries to shards
Load balancing with *chunk splits & migrations*

- A chunk are usually key ranges
- If one chunk is getting too big (>64MB), mongos will split it into multiples
- If one shard is getting too much traffic, mongos will migrate chunks between shards by updating the config server and moving the chunks.
Shard keys are forever

- Immutable - changing a shard key requires rebuilding the DB
- Must have sufficient cardinality - sharding across $n$ replies won’t work if the shard key cardinality is $< n$
- Unique attributes should be part of the shard key, since otherwise MongoDB can’t ensure the required uniqueness since shards are independent
- Shards only index on \_id and the shard key, since using any other field would require inter-shard communication
- If distribution of shard keys isn’t uniform (across shards), consider a multi-field composite shard key with the better key first.