

Vitess

{mike | sougou}@youtube

code.google.com/p/vitess

Briefly

- introduction & motivation
- long term vision
- implementation strategy
- current tools
- questions, answers?

Begin at the end

- MySQL-esque*
- Self-managing without magic**
- Increased efficiency***
 - memory usage, throughput
- External replication

Labor vs Management

- Automated reparenting
- Online schema apply*
 - alters, rebuilds
- Auto-sharding
 - incremental provisioning

Assumptions/Constraints

- keyspaces / keyspace_id
- range-based shards
- "shards x replicas" structure
- no cross-shard transactions*
- eventual consistency

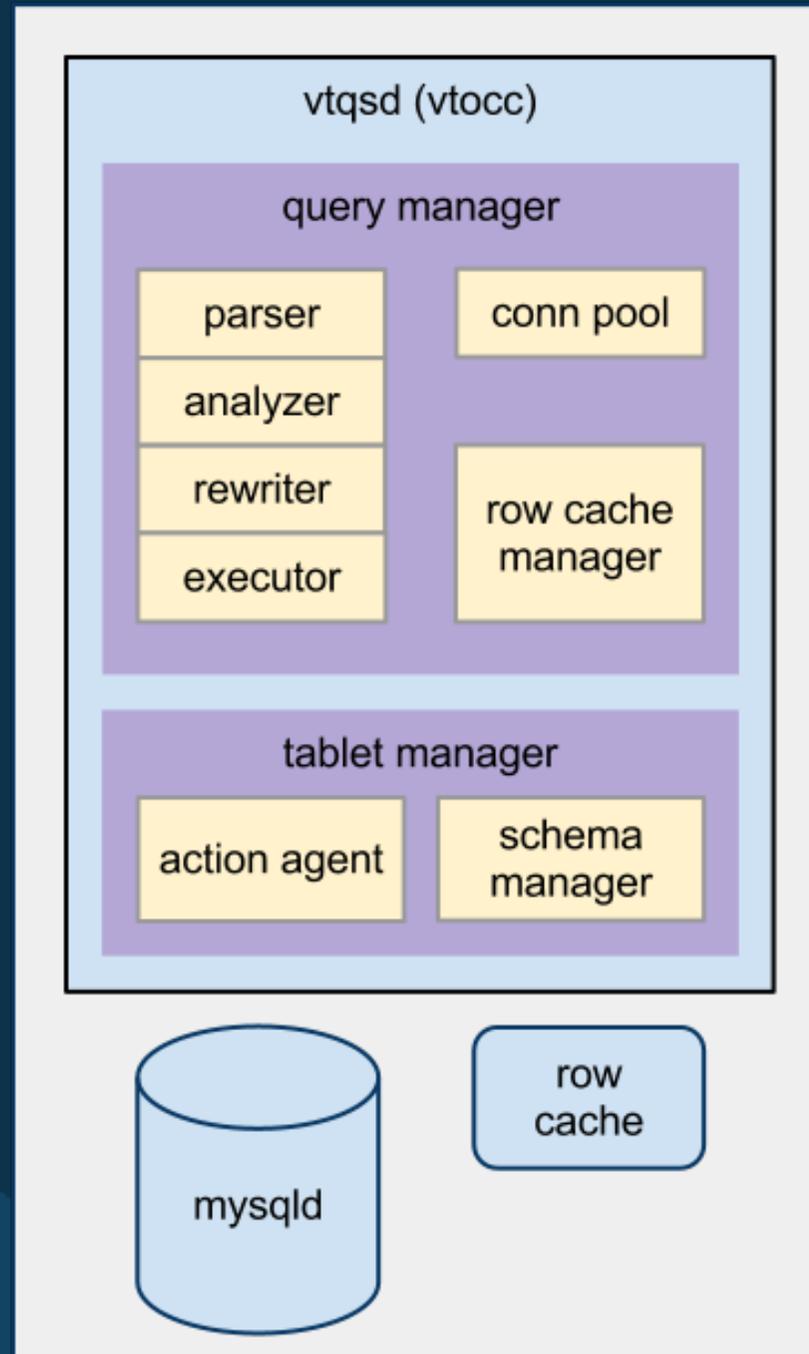
The Tao of Youtube

"choose the simplest
solution with the
loosest guarantees that
are practical"

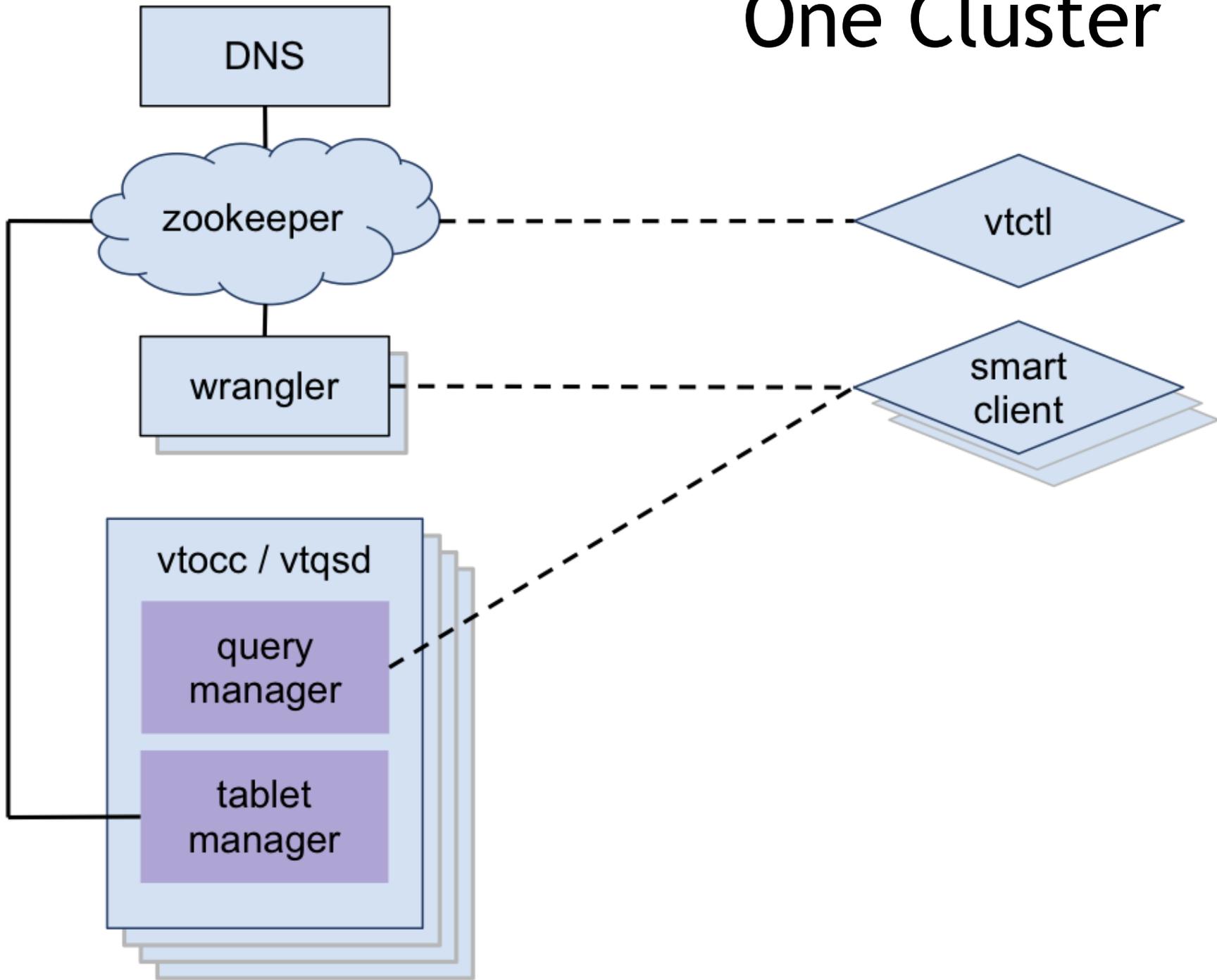
Implementation Strategy

- minimal changes to MySQL
- external query shaper
- external tablet manager
- coordinate in Zookeeper*

One Machine



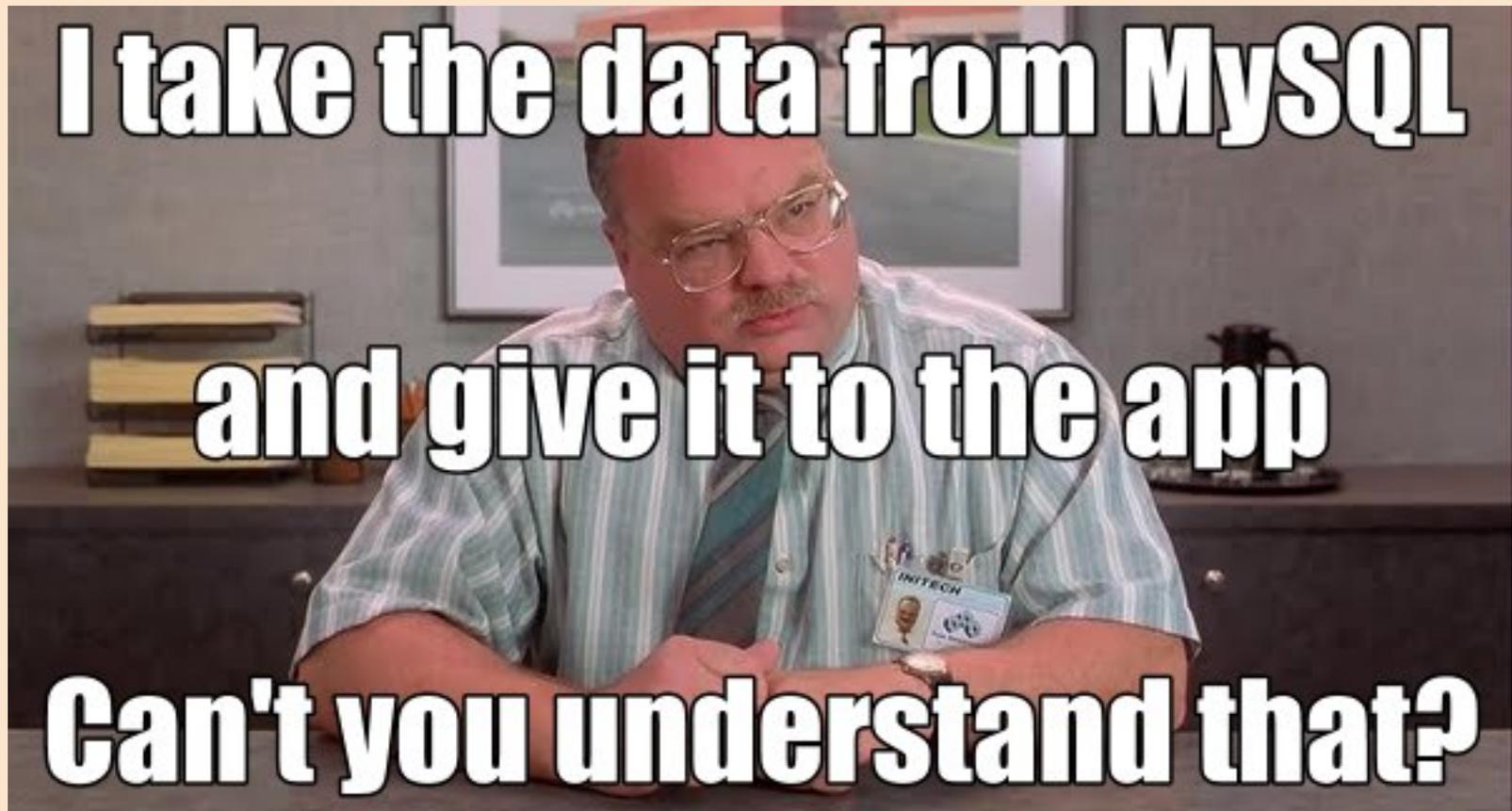
One Cluster



Elephants

- Why is this in Go?
- Why isn't this inside of MySQL?

vtocc



I take the data from MySQL

and give it to the app

Can't you understand that?

vtocc

- First usable piece of vitess
- RPC front-end to MySQL
- Connection pooling
- Transaction management
- DML annotations
- SQL parser

Production ready

- Serves all of YouTube's MySQL queries
- Months of crash-free & leak-free operation
- Zero downtime restarts
- Configuration, Logging & Statistics

Fail-safes

- Query consolidation
- Row count limit
- Transaction limit
- Query and transaction timeouts
- Easy to add more

Row Cache

- vs Buffer Cache
- CPU usage
- Primary Key fetches
- Results merging using subqueries
- Code complete, benchmark in progress

Other Upcoming features

- vtqsd
- Update stream
- Multi-user
- Reserved connections

Questions?

