My first 90 days with Vitess

Morgan Tocker

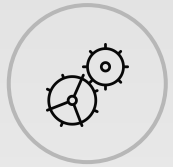
October 2019

Agenda

1. What is Vitess?
2. Terminology Essentials
3. My Questions (MySQL Compatibility, Consistency Model..)
4. Other Quirks and Features
5. The Best Use Case
6. Where Vitess Could Improve

What is Vitess?

Not a straightforward single-category answer



Middleware/Proxy

- Sits in between your application and MySQL
- Provides Routing, Query Consolidation, Connection Pooling

Orchestration

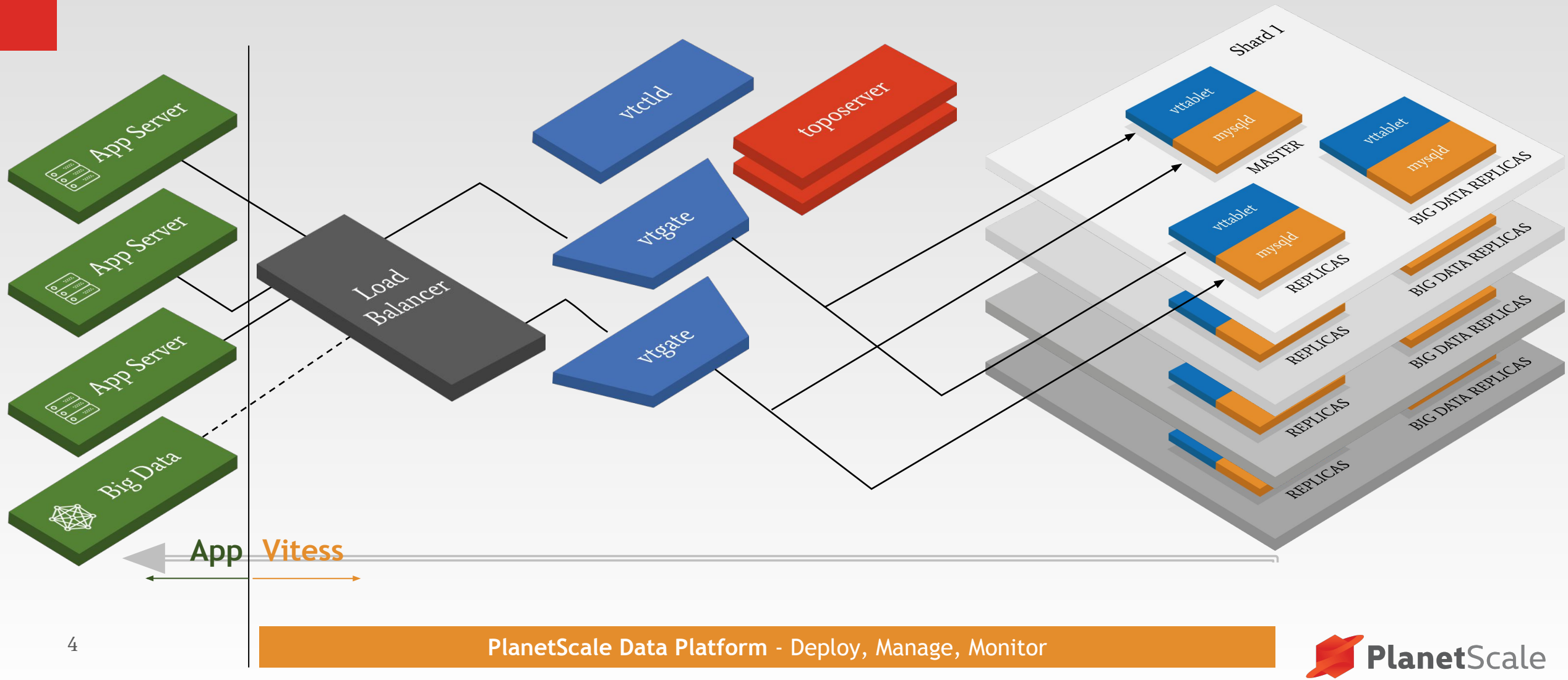
- Includes Monitoring & Backup
- Integrates with Orchestrator and provides failover

And More...

- VReplication
- Vitess Messaging
- Easier deployment path into Kubernetes

PlanetScale

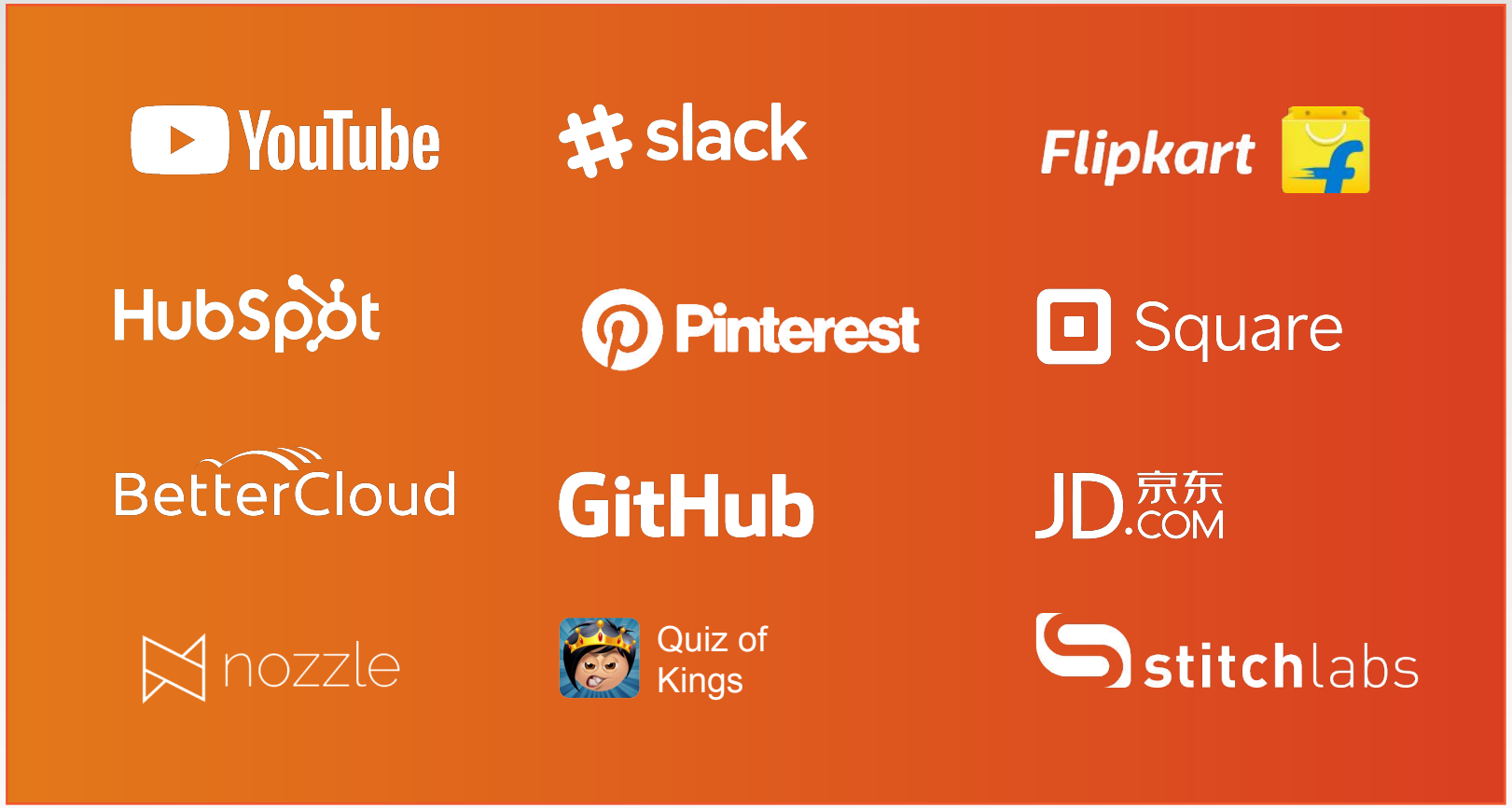
Vitess Architecture



PlanetScale Data Platform - Deploy, Manage, Monitor

PlanetScale

Vitess serves millions of QPS in production



PlanetScale

Terminology: The Essentials

VTGate: The Proxy that your applications connect to.

Topo Server: The etcd server containing meta data.

Tablet: A combination of a vttablet and a MySQL server.

Keyspace: a logical “MySQL database” (aka schema). Keyspaces can be sharded or unsharded.

Cell: A data center

https://vitess.io/docs/concepts/

My-MySQL Questions

- **What version of MySQL is it?**

  VTGate currently advertises itself as MySQL 5.5.

- **What versions of MySQL does it support?**

  MySQL 5.6+ and MariaDB 10.0+

  Requires RBR + GTIDs enabled

  Strongly recommends semi-sync

My Questions: What queries are supported?

- Depends on if keyspace is sharded
- Major Limitations:
  - `SET [SESSION] var = x;`
  - `GROUP BY key ORDER BY different_key;`

https://vitess.io/docs/faq/queries/#is-there-a-list-of-supported-unsupported-queries

Questions: Consistency Model?

- `READ-COMMITTED` for reads
- Atomic within a shard on update
  - 2PC also available but not recommended
- Has elements of opt-in eventual consistency (VReplication provides materialized views).

Quirks and Other Features

- All queries are SQL parsed
- They may be changed before routing to mysqld, but best attempt is made to preserve comments!

Quirks (cont.)

- The original designed used a gRPC protocol to VTGate instead of MySQL - it still exists!
- The original design also had type safety - very different from MySQL's history!
  - This pedantic design helps a lot
  - Easier to add flexibility later vs. remove it

Quirks (cont.)

- Scalability Philosophy = 250G shards
- Makes sense once explained
- Biggest benefit is fast recovery time

(Translucent) Sharding

- Still designing a VSchema
- Recommend modeling updates to be single shard
- vs. transparent:
  - Distributes data everywhere
  - Possible latency penalty; typically a new engine

Vitess Integrations

- Orchestrator
- ZooKeeper/etcd/Consul
- XtraBackup
- MySQL/MariaDB Replication

The “Best Use Case”

Signs you might be a fit include:

- You currently have schema-per-tenant
- You have a multi-tenant Application
- You need an upgrade path from MySQL to sharded MySQL
- Your monolithic databases are blocking your Kubernetes adoption

To Be Aware Of

Still MySQL underneath

- Including the MySQL Optimizer (good for OLTP)
- In some cases may improve analytics (VReplication + parallel scatter gather). Not All.

Born as an Internal Company Project

- Ease of use not Day 1 Priority
- It is a Priority for PlanetScale

Questions

Join the Vitess Slack Community!

https://vitess.io

PlanetScale

Features I've worked on

MySQL Flavor detection

Used to bootstrap MySQL Server correctly (mysql_install_db versus mysqld --initialize etc).

Go modules support

Things I would like to see improved

Reduce delta from MySQL default configuration

Add support for additional SQL syntax:

- NOOP where safe for connectors
- Reduce pedantic-ness of design for compatibility

Reduce shell scripting in examples (use ~/.vitess.cnf etc)

Embrace smaller footprint for development environments:

- VTCombo