FOLLOW MY LEADER

[Re]connecting clients after Postgres server failover

Alastair Turner
Community Technologist

Percona Live, Mountain View, May 2026

PERCONA®

Managed cutovers, or even failovers, should not be a drama

But they can be, so let's read this like a Greek Tragedy

- What, where and who
  - The scene
  - The tasks
  - The characters
- The tradeoffs

In the beginning...

Application connects to its backend database.

Once for each and every user session

PERCONA

In a bit more detail

Both app and database server have components running in process, or alongside

For the app there's at least a database driver, maybe framework[s] managing database access

The server's network stack may play a part and some other network bits may run alongside the database server

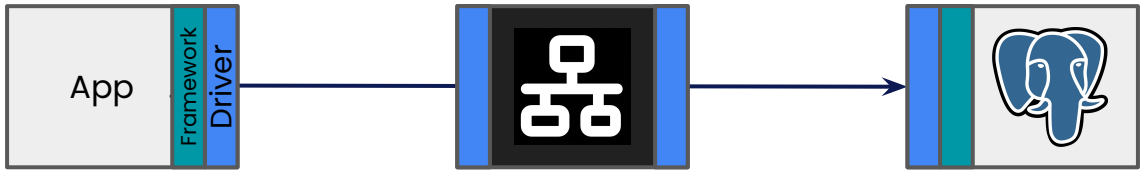
PERCONA

… and a bit more

Almost certainly a network between the client and the database server

Services may run on the network between the nodes

Mainly a black box



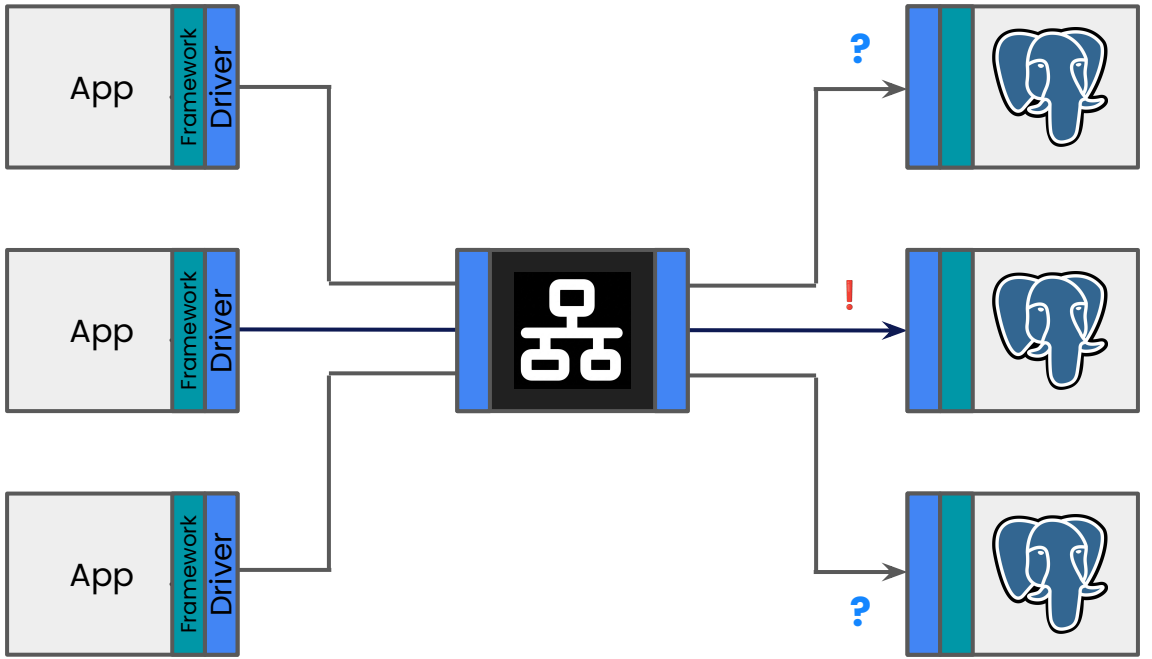
PERCONA

More of everything

Application may scale out

Database may have replicas for failover

Read replicas for throughput



PERCONA

The scene

Application

- Avoid application logic changes
- Data frameworks like Hibernate, Hikari, Django, Rails
- Database connection driver

Network

- May host some services

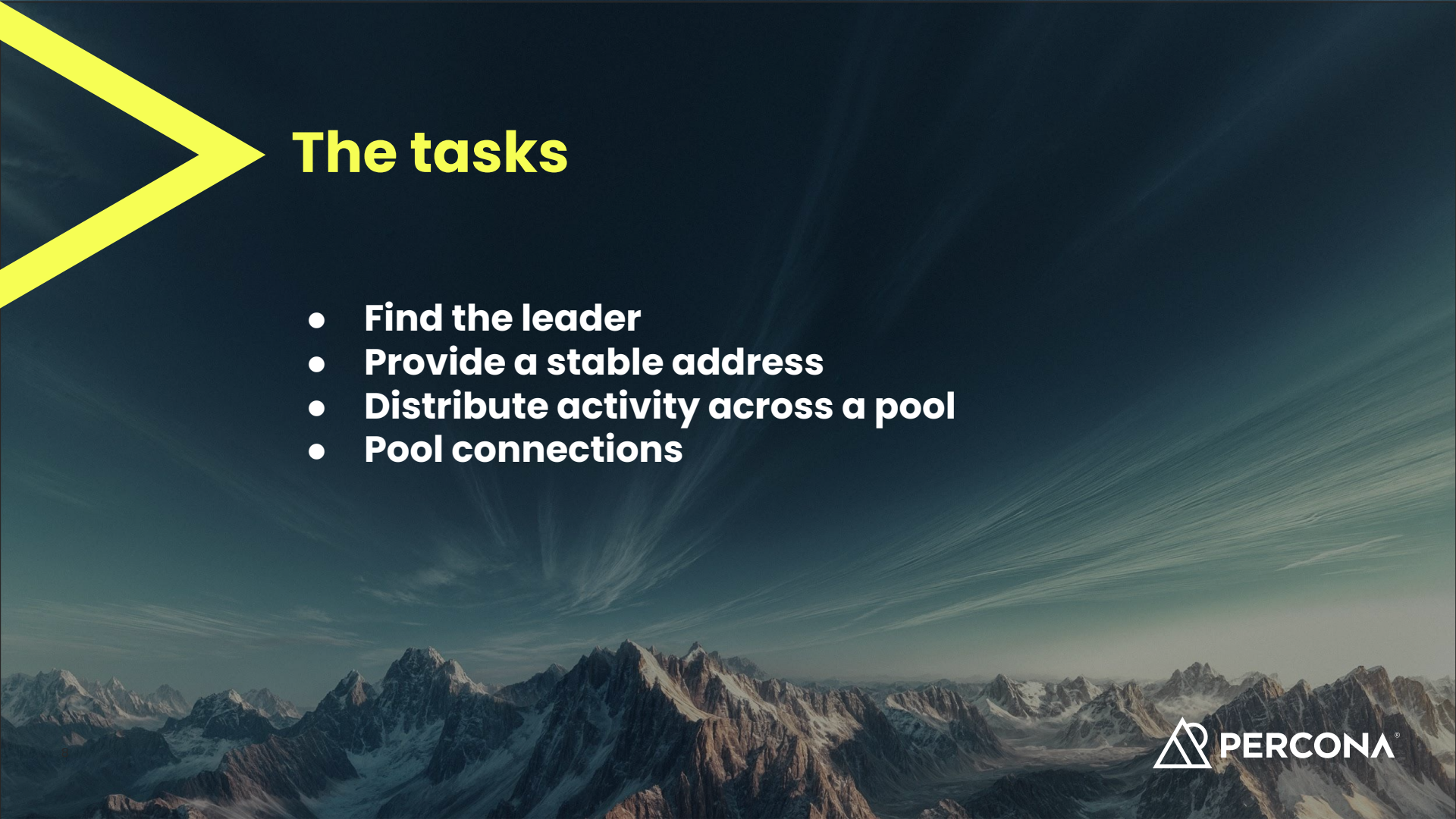
Server

- Network configuration
- May host some services

PERCONA

The tasks

- Find the leader
- Provide a stable address
- Distribute activity across a pool
- Pool connections



PERCONA®

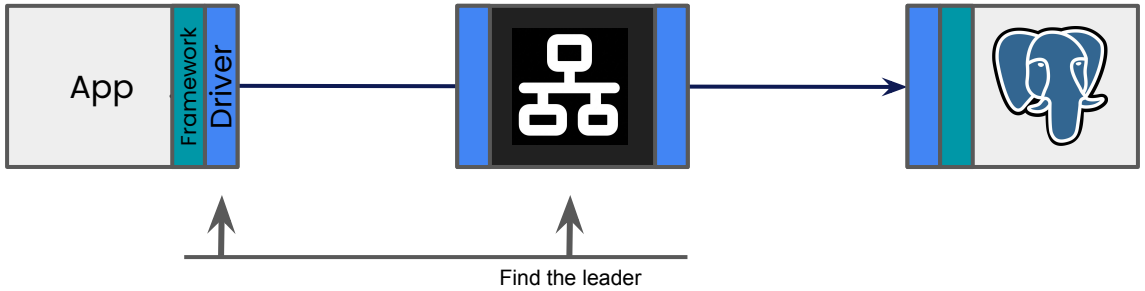
Find the leader

Client library config

- Application must support the configuration
- Stability and change control

Service on the network

- ~~DNS~~
- Proxy



PERCONA

Provide stable address

Client has one server address

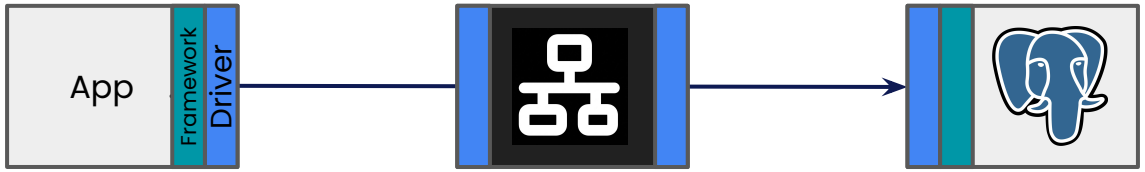
Belongs to server or proxy

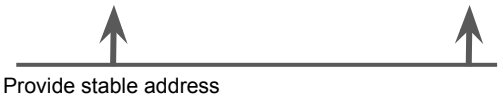
Move the address to whichever system is live

Built in for network load balancers

keepalived

vip-manager - specific to Patroni setups





PERCONA

Distribute activity

Client library config

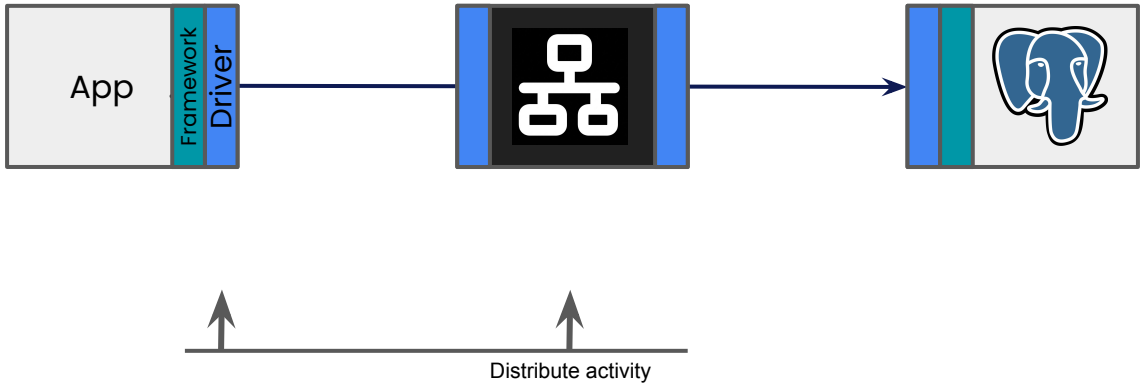
Service on the network

Spread traffic across healthy servers

Only primary is healthy for write service

All or replicas healthy for reads

Round robin or fewest connections



PERCONA

Pool connections

On client

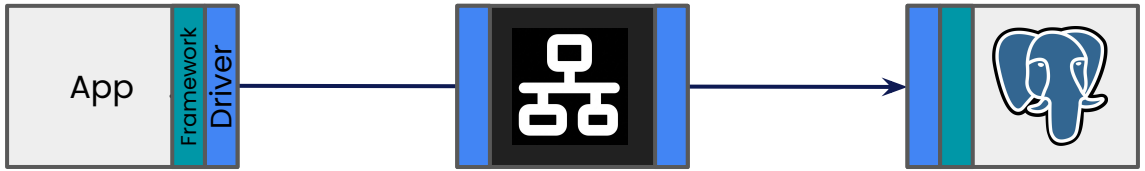
- Improve application performance

On network

- Protect server
- Client for finding leader and distributing activity
- Pause forwarding commands
- Turn downtime into latency

On server

- Protect server



PERCONA

Identify request type

Request strings and resultsets

- SELECT could call a function which updates things

Proxy can parse and decide guess

- pgPool
- pgDog

pg_query at least uses the real Postgres parser

Application framework *knows*

PERCONA

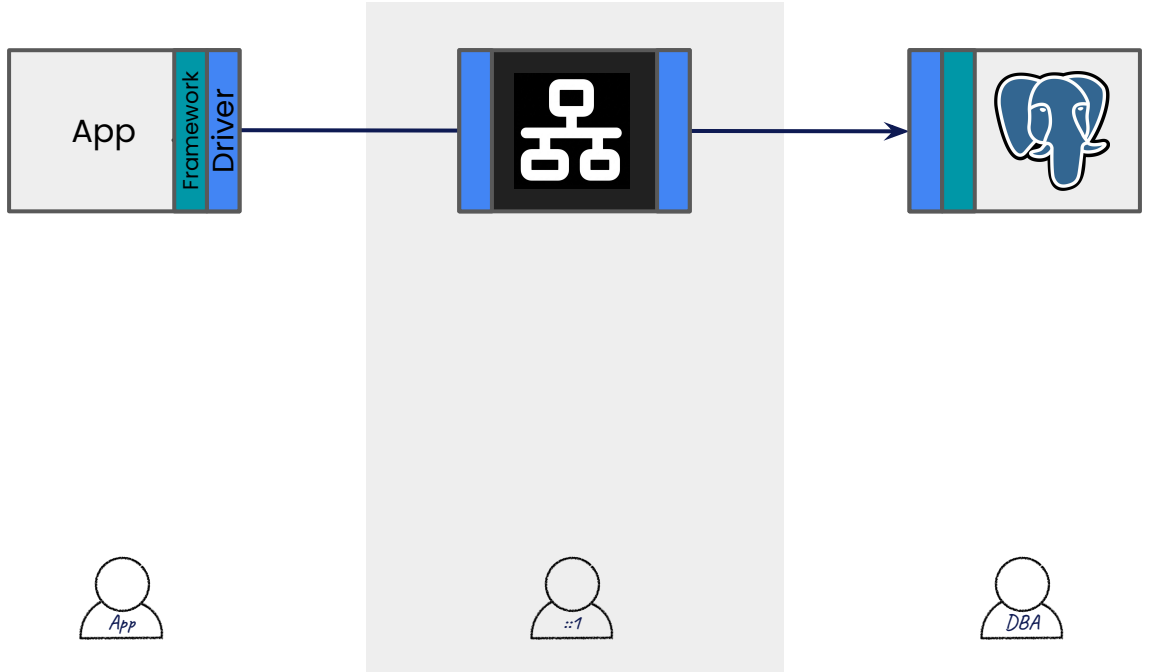
PERCONA

Identify request type

Application team
- The customer, or proxy for
- Low weight
- Move fast and…

Network team
- Mature tech and practice
- Massive installed base
- Issues are huge, but brief

Database team
- Pressure to move as fast as the app teams
- Once data is gone, it's gone



PERCONA

PERCONA

Delegated client configuration

Fewest moving parts

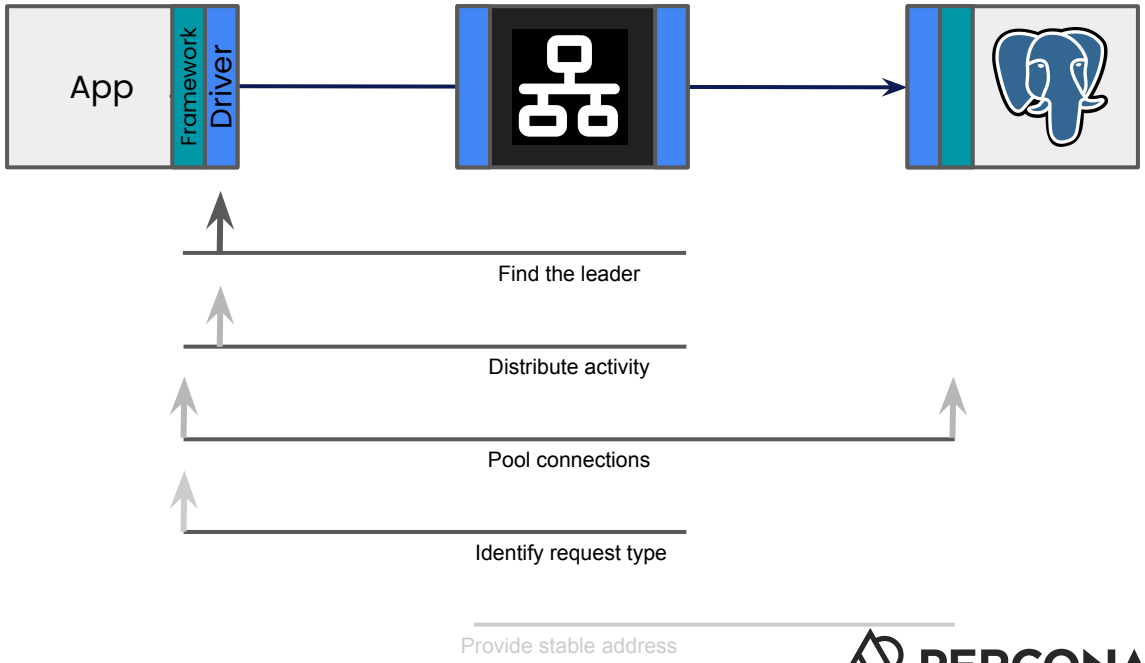
Stable database deployments

Collaborative between application and database teams

Support from client libraries

Together on PaaS or K8s

Optional pooler in DB deployment



PERCONA

Use what the network provides

No requirements on application config or client libraries

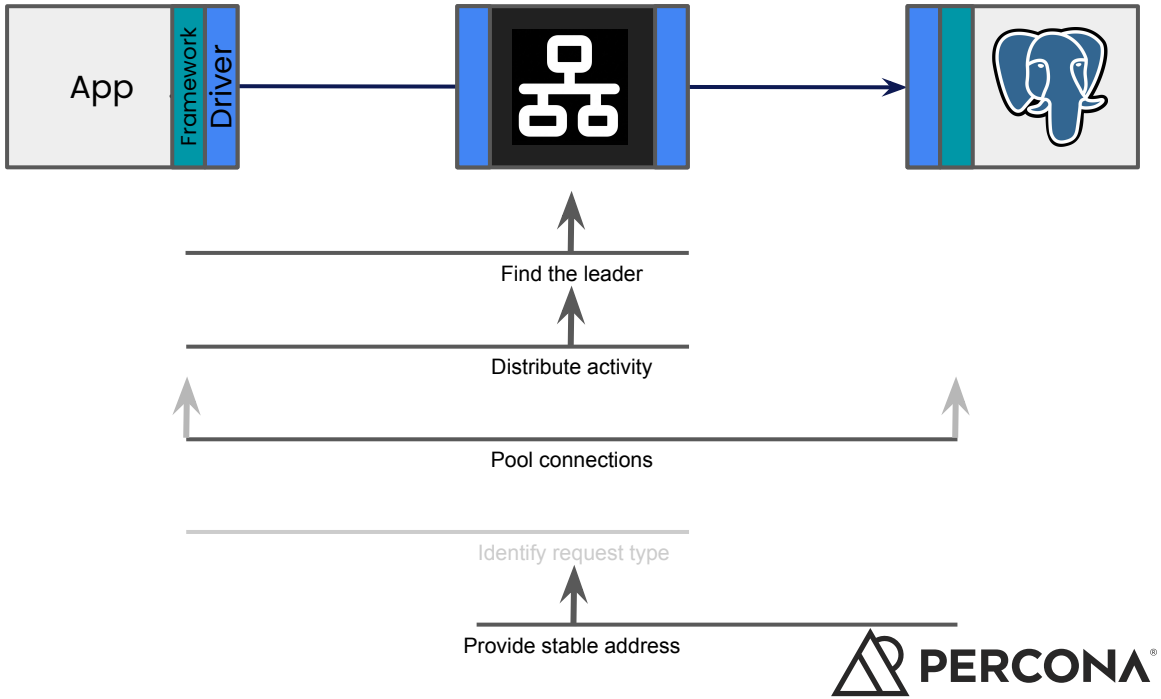
Everyone does what they're good at

How K8s DBaaS ends up

Possible costs

Automations may not integrate

Optional pooler in DB deployment



PERCONA

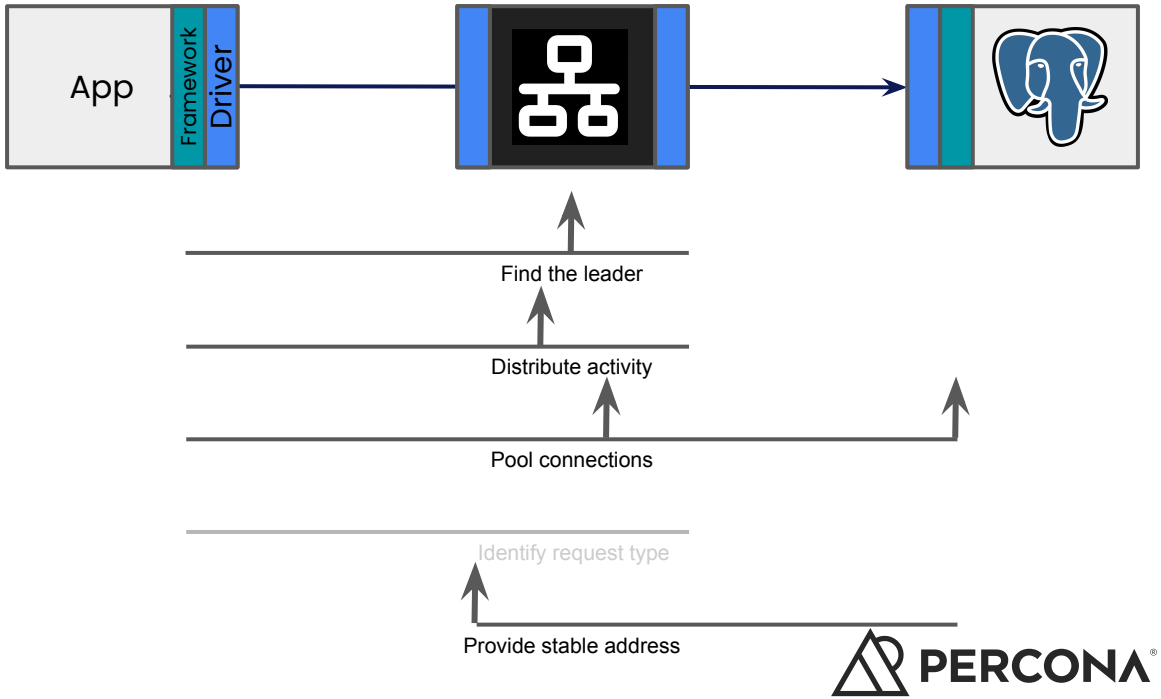
All we have is VMs - Maximalist

HAProxy load balancer on VM

keepalived

Pooler on VM or server

Blame culture or managed service SLA wars



PERCONA

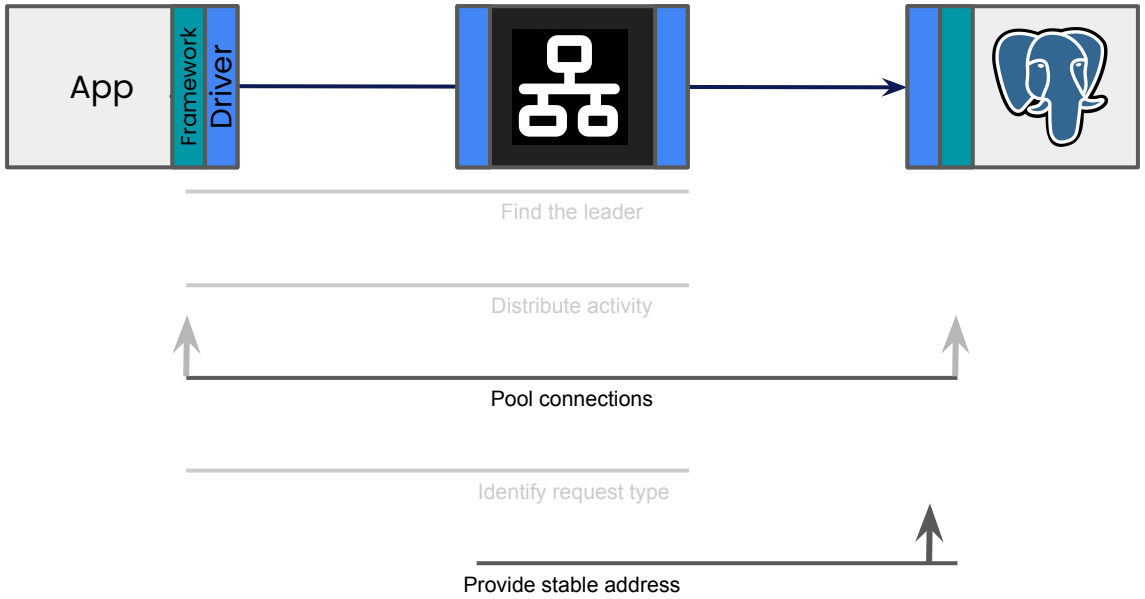
All we have is VMs - Minimalist

keepalived or vip-manager on server

Optional pooler on server

What SQL Server does

Common practice for MySQL



PERCONA

Where to next?

- Integration with service discovery
  - Update the list when scaling read replicas
  - Auto-wiring with frameworks like Spring
    - Patroni event callbacks
- Protocol extension for topology discovery
  - Long term goal :)

PERCONA


PERCONA®