PERCONA

Databases run better with Percona

PERCONA

PostgreSQL HA

High Availability delivered

JOBIN AUGUSTINE (47)

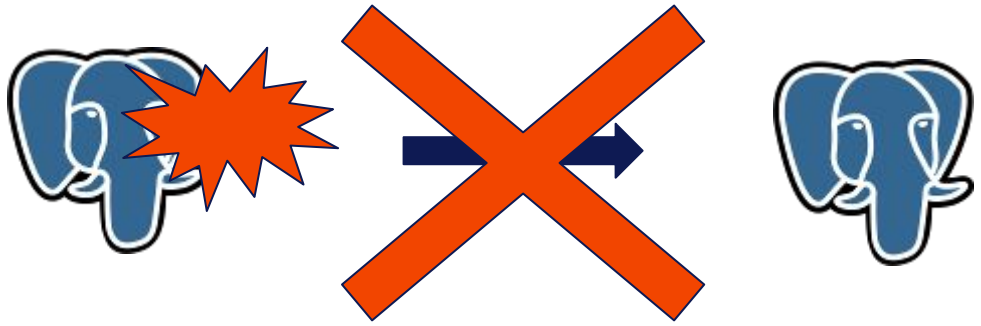
Topics

- What is HA (What is Not HA)
- Common misconceptions
- Lessons learned over 2-3 decades
- Disasters - Split Brain, Data loss
- Recommended topology
- Distribution, Patroni and tools
- Good practices

High Availability - The value

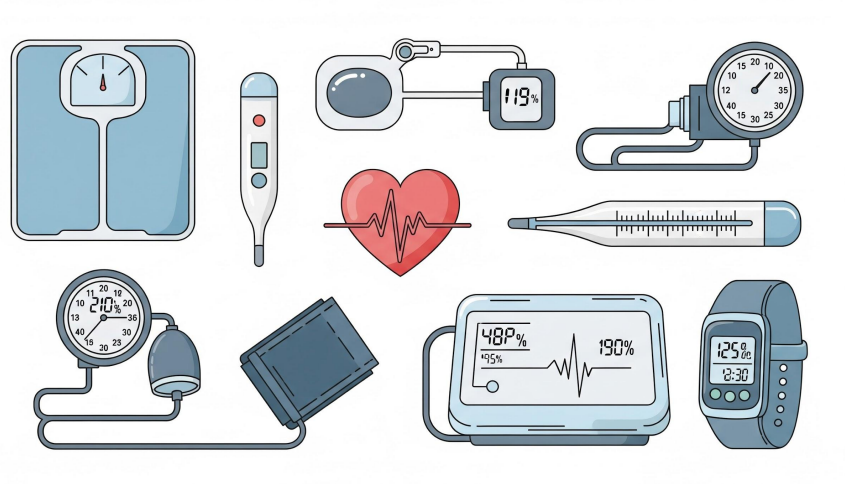
- Misconceptions by Proprietary Software marketing

What is HA

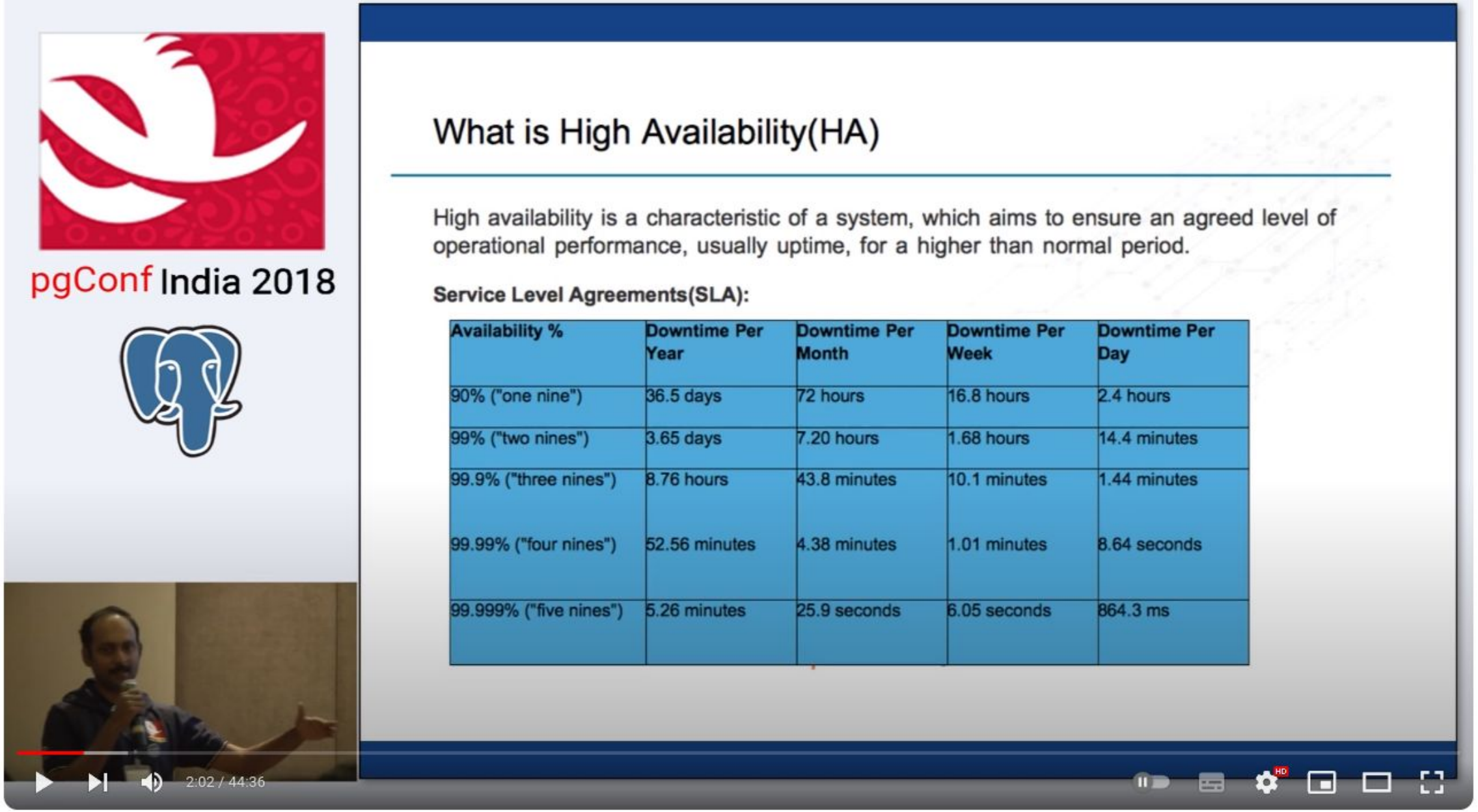


- Reduce the chance of failures - Reduce the failure points
- Elimination of single-point-of-failure (SPOF)
- Automatic detection of failures - Continuous monitoring
- Automatic action to reduce the impact
- Replacement of failed component then and there.

How to measure



- **MTBF - Mean Time between Failures - repairable**
- MTTR - Mean Time to Respond
- MTTR - Mean time to repair / recover / restore
- MTTF - Mean time to failure - non-repairable
- Uptime - Availability 9s
- RPO & RTO



Mean time between failures

From Wikipedia, the free encyclopedia

Mean time between failures (**MTBF**) is the predicted elapsed time between inherent failures of a mechanical or electronic system during normal system operation. MTBF can be calculated as the arithmetic mean (average) time between failures of a system. The term is used for repairable systems while **mean time to failure** (**MTTF**) denotes the expected time to failure for a non-repairable system.[1]

The definition of MTBF depends on the definition of what is considered a failure. For complex, repairable systems, failures are considered to be those out of design conditions which place the system out of service and into a state for repair. Failures which occur that can be left or maintained in an unrepaired condition, and do not place the system out of service, are not considered failures under this definition.[2] In addition, units that are taken down for routine scheduled maintenance or inventory control are not considered within the definition of failure.[3] The higher the MTBF, the longer a system is likely to work before failing.

Single Point of Failure (SPOF)

Vs

Frequent and **Common** Point of Failure

Database Connection

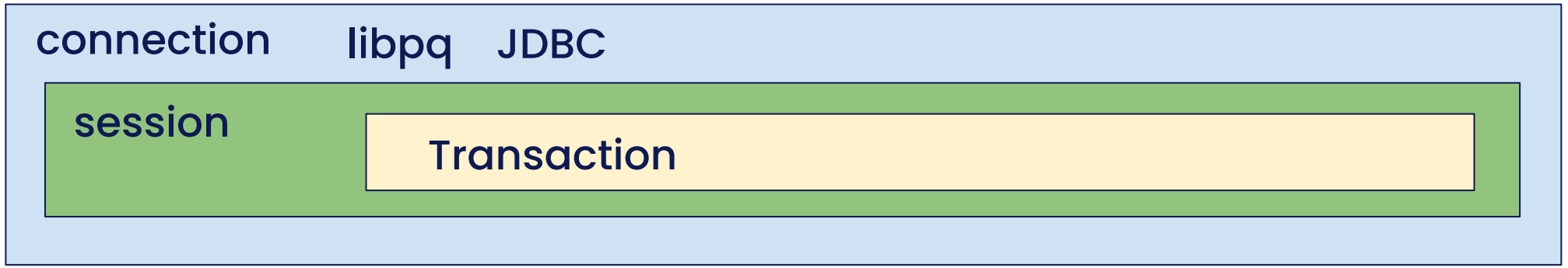
A database connection is the network **link** established between a client application and the PostgreSQL database server.

Vs

Database Session

A database session is the logical **context** within a connection that defines the scope of a client's interactions with the database

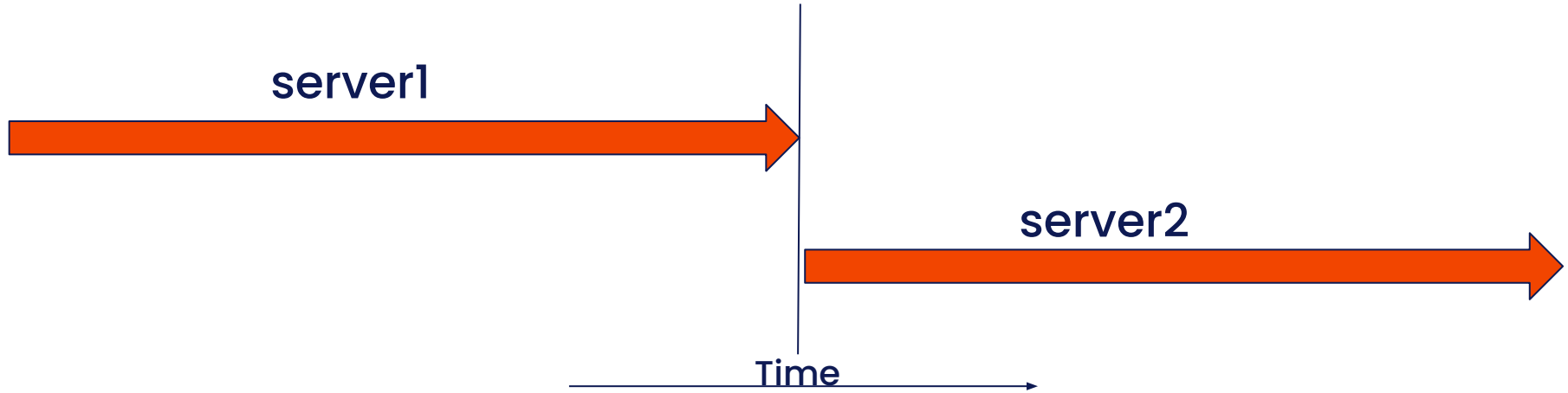
Client



Session Level

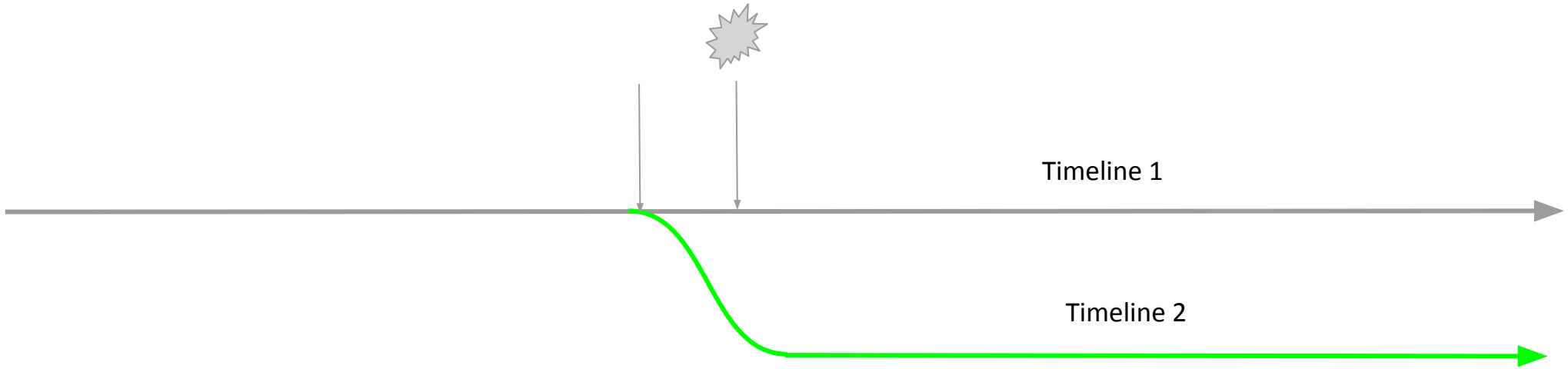
- client_encoding
- timezone
- prefered collation
- Prepared statements
- Temporary tables
- Cursors

Database Availability Myth



- All Database connections are **stateful, session oriented, end-to-end** connections
- Sessions can't failover or move from one machine to another
- Running transactions will be aborted and rolled back
- Heavily dependent on application's capability to handle the exception

Database Availability Myth



There is potential data loss. even if it is very minimal

Fully Synchronous standby is not practical

Database Availability Myth

- Database Availability doesn’t imply application availability
- Aggressive failovers increases risk of application unavailability
- Failovers often has the risk of data loss.
- Common misunderstanding of Disaster Recovery (DR) with High Availability (HA)

Rules of High Availability is different for session oriented, stateful applications like databases

- if there is a failover, All sessions dies, transactions rolls back.
- Majority of applications crashes and misbehaves
- Outage

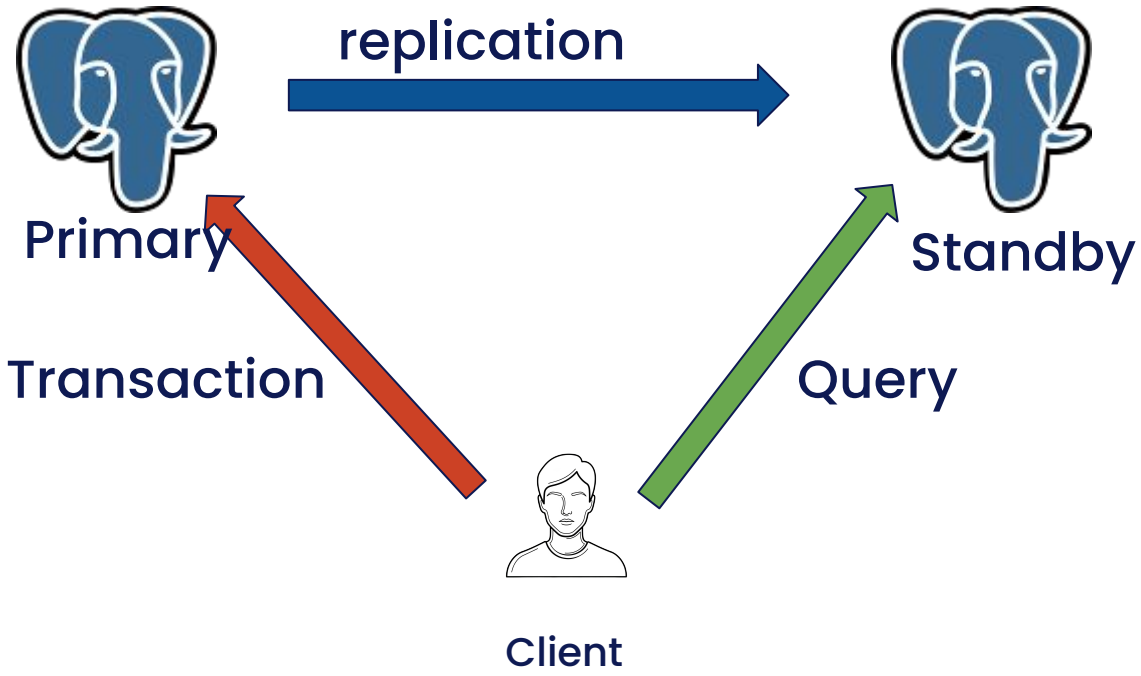
Availability & Performance

If performance requirements are not met, it is considered as unavailability.

Implications of Multiple machines

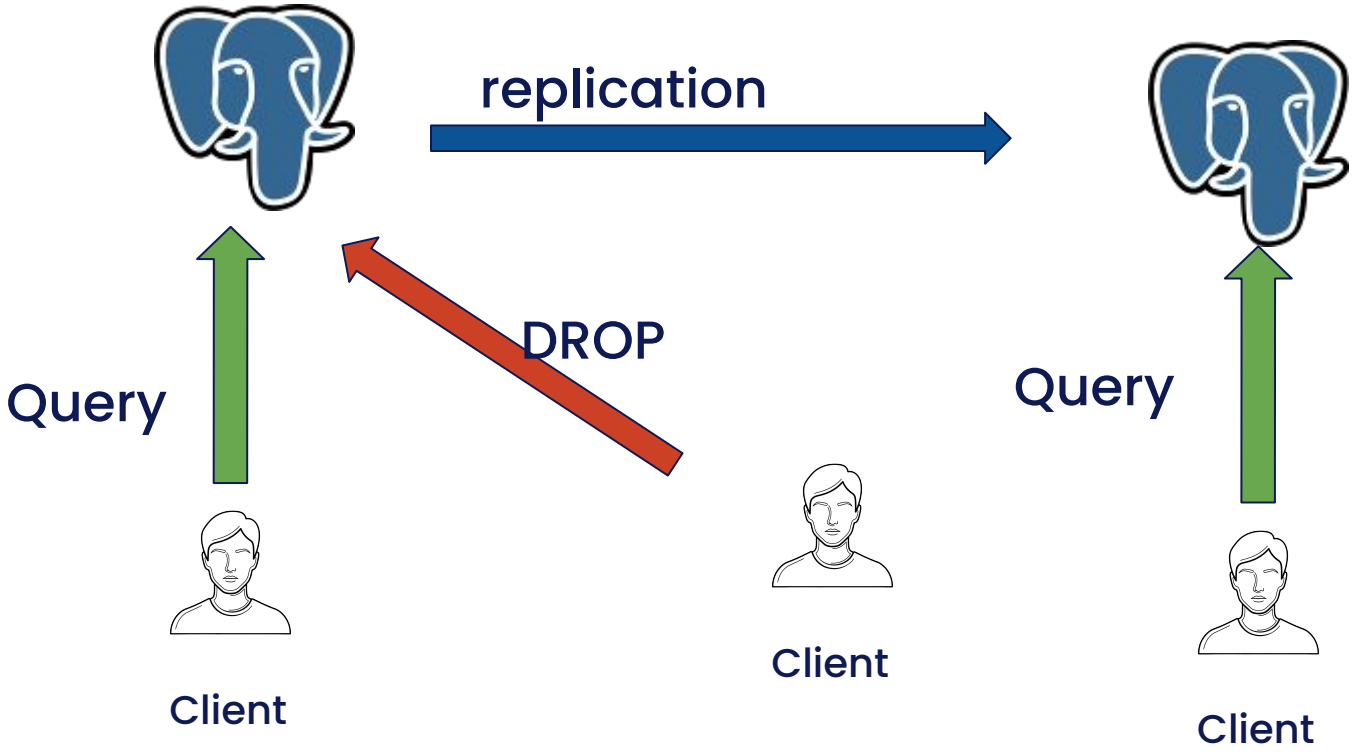
Reading from Standby

Eventually replicated. Information on the standby may be obsolete



Reading from Standby

Standby Cannot Participate in ACID transactions in the Primary



Reading from Standby

- Session on Standby cannot place a lock on Primary
  - No Locks ACCESS SHARE, ROW SHARE etc possible
- Sessions on **Standby and Primary can conflict**
- Conflicts are possible with even vacuum workers.
- Query cancellation on the standby might be the remedy if a conflict happens
  - Else delay the replication

Reading from Standby - Points

- Data from Standby could be inconsistent
- Standby might not see transactions which is committed on Primary
- Standby might see transactions which are yet to be committed
- Session on Standby might get into conflict with session on Primary
- Expect Query cancellation on Standby

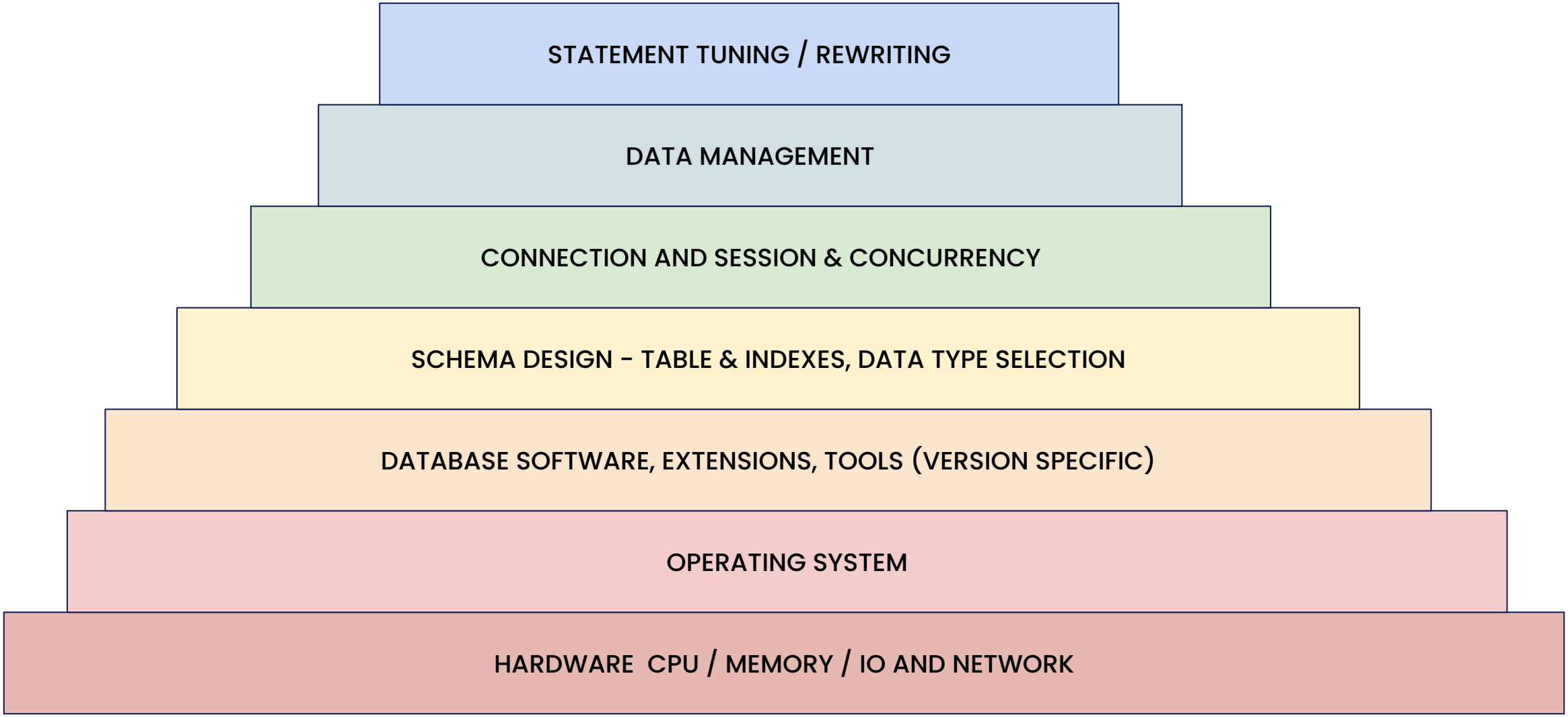
Reading from Standby - Intermediaries

- SELECT Doesn’t mean “READ”
  - SELECT can execute functions behind.

How to achieve High Availability

How HA is achieved

- Ensure the replacements are ready to take over
- Eliminate all SPOF
- No ambiguity - Split-Brain
- Failure Detection.
- New Leader promotion.



Hardware / Platform

- Reliable and Low latency network.
- Right sized host machine
- **Nothing shared architecture**

Software Stack

- Protection from Abuses
  - Long running transactions / statement
  - Connection bursts
- Active maintenance
  - Monitor the usage
  - Obsolete Objects
  - Vacuum tuning and custom jobs
- Expect performance cliffs
  - Data retention policies and implementation

Criteria for Selection of HA Framework

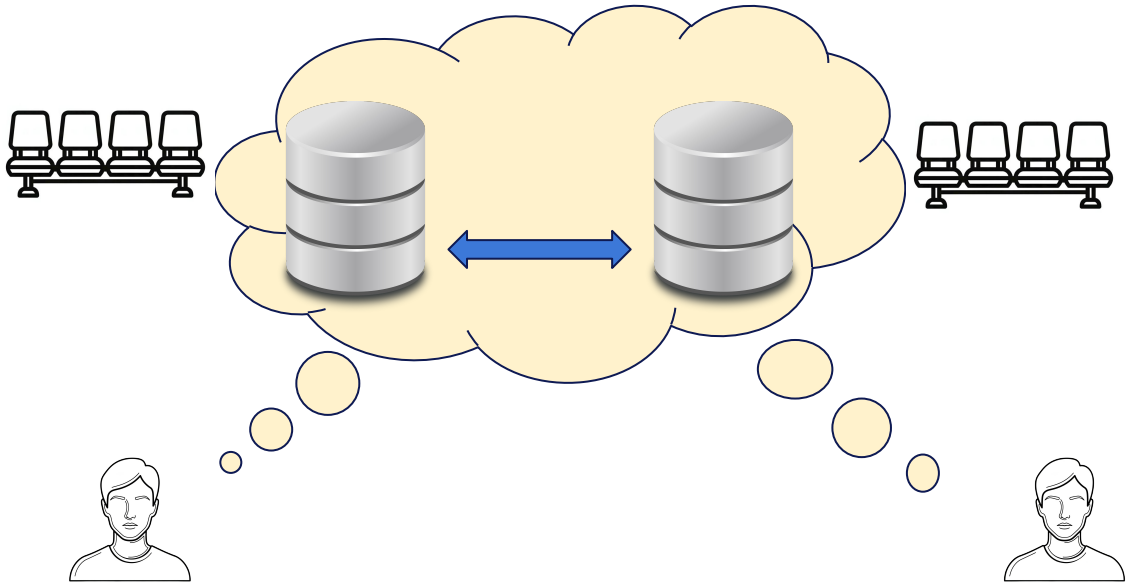
- Intelligent
  - Shouldn’t cause any damage
- Take decisions and take care of side effects also
- Easy auto-pilot on/off
- Maintain identical system/nodes
- Meet the expectation of Availability numbers
- Same location / same latency

What is **NOT** HA

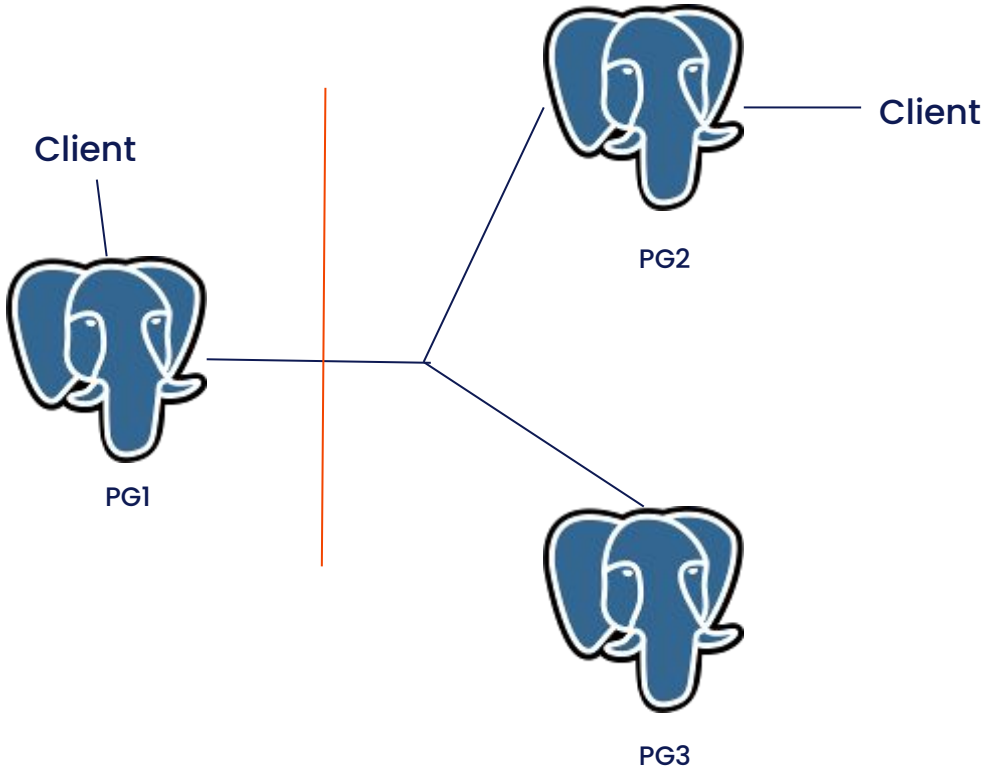
- Multi-Master , Active - Active
- Remote location - DR site
- Database Backups

Multi-Master, Active-Active Myths

- Marketing lies, propaganda
- Database is a persistent storage of data. Not CPU
- CAP Theorem and its implication.
- Pessimistic locking Vs Optimistic “conflict” resolution



Problem of split-brain



Old ideas and solutions

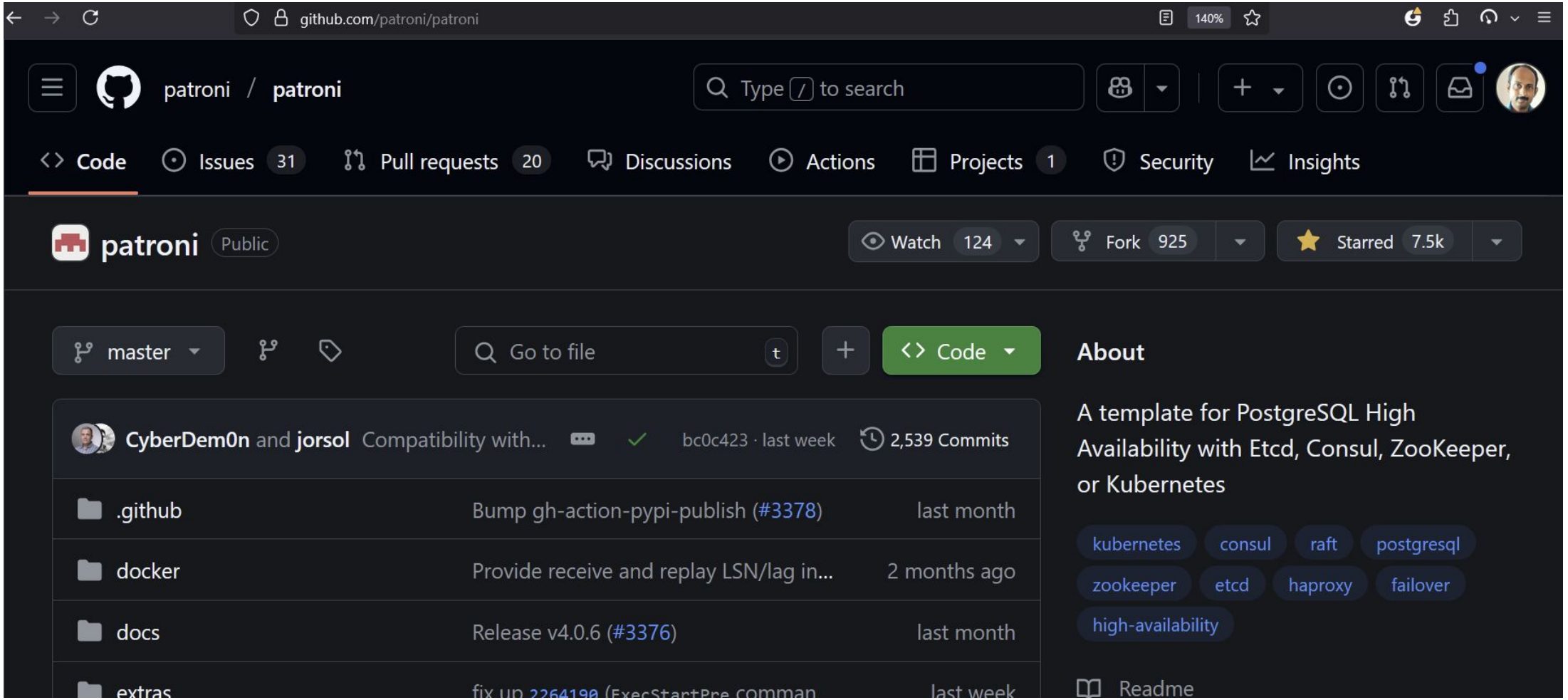
- Monitoring nodes
- Voting disk

Modern Solution - Distributed Consensus

- Raft
- Paxos

Major solutions for PostgreSQL

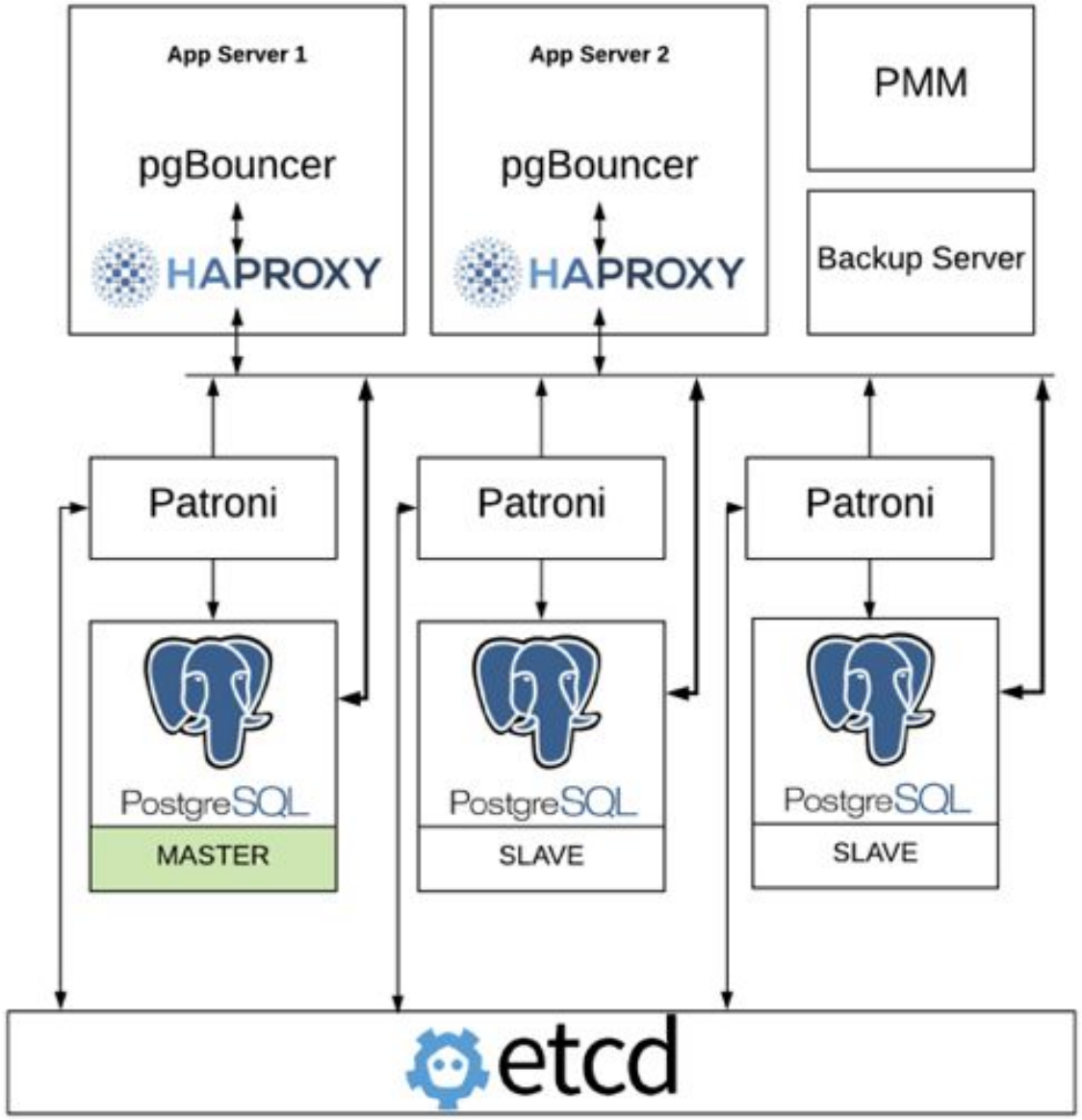
- pgpool-II — Popular as a pooling solution + Router
- repmgr — 2ndQuadrant Product
- Stolon — Not very popular
- pg_auto_failover — Microsoft (Past)
- Patroni — Zolando and Community, Now Microsoft



Patroni Features

- Great Protection from Split-Brain using Raft
- Reuses available solutions like ETCD, Consul, Zookeeper
- Integration with Linux Watchdog - Second level protection
- Easy switch off from auto-pilot (maintenance / pause mode)
- Easy takeover of existing cluster
- Easy Global configuration
- PostgreSQL parameter aware

Patroni



Percona Distribution

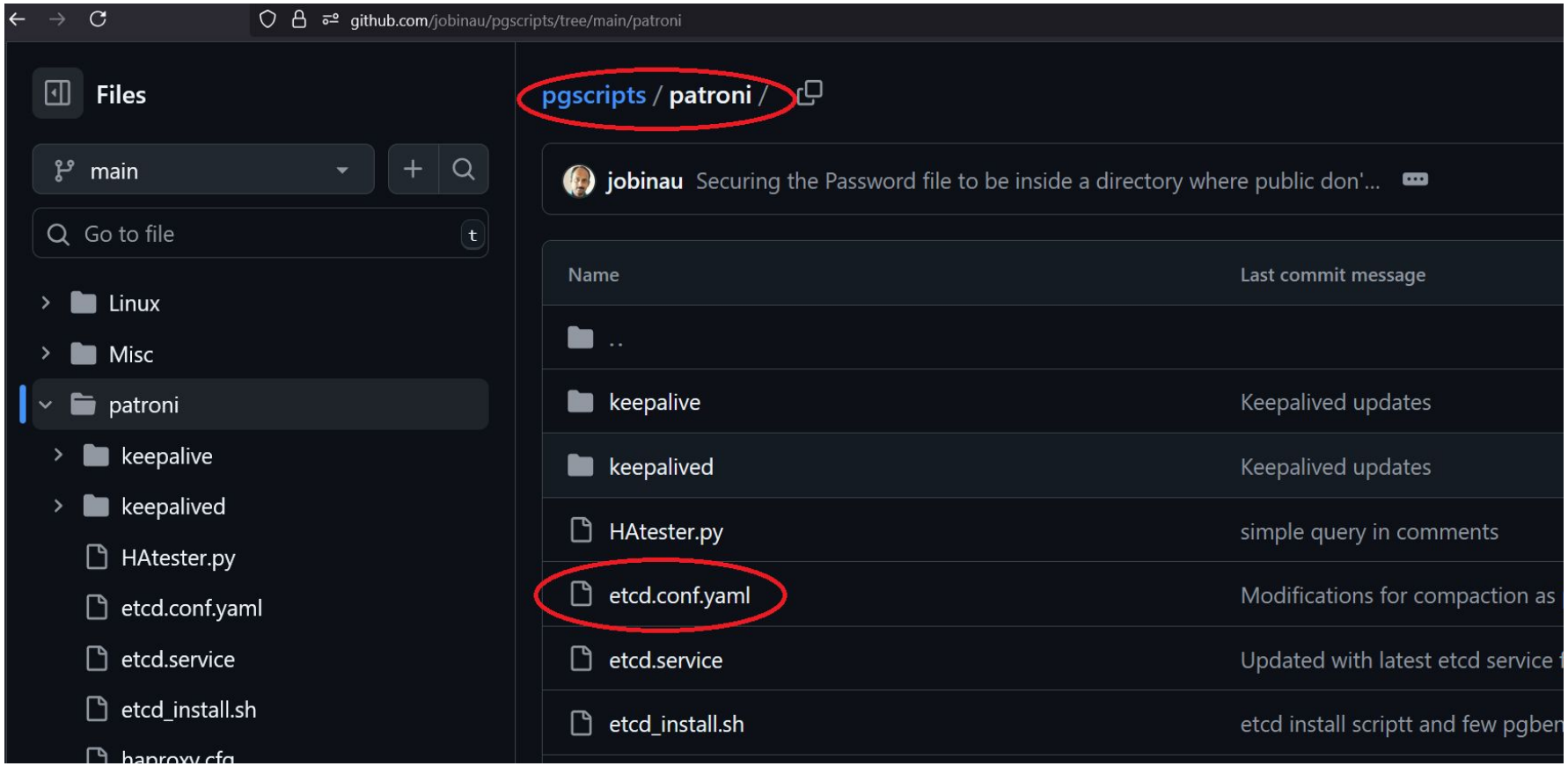
- Well tested packages
- Easy installation of ETCD and services
- Patroni dependencies are provided

Engineers with Years of Knowledge on the subject

Patroni Setup

Reference Scripts

https://github.com/jobinau/pgscripts/



etcd - The DCS

2 Objectives

- Distributed Consensus Store
  - Raft - The fool proof method of leader election
  - Leader key store
- Distributed Configuration Store
  - Centrally manage all parameter configuration

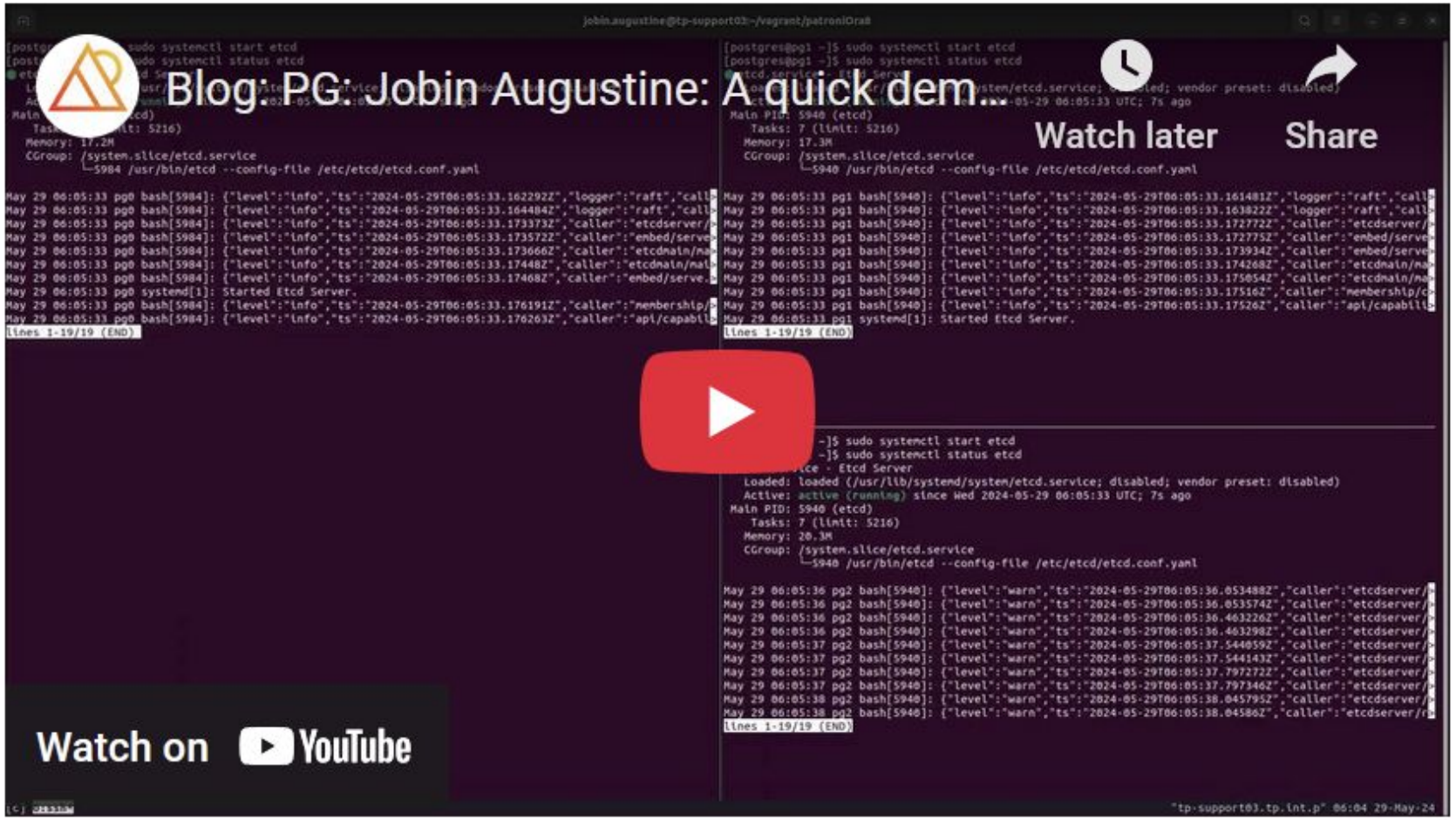
https://thesecretlivesofdata.com/raft/

etcd configuration

```
name: 'pg0'
data-dir: /var/lib/etcd
initial-advertise-peer-urls: http://172.20.0.2:2380
listen-peer-urls: http://172.20.0.2:2380
advertise-client-urls: http://172.20.0.2:2379
listen-client-urls: http://172.20.0.2:2379,http://localhost:2379
initial-cluster: pg0=http://172.20.0.2:2380,pg1=http://172.20.0.3:2380,pg2=http://172.20.0.4:2380
initial-cluster-state: new
initial-cluster-token: pgcluster
auto-compaction-retention: '15m'
auto-compaction-mode: 'periodic'
```

Environment Variables Vs YAML configuration file

2min demo



https://www.youtube.com/watch?v=j1B2RLsHWCU

etcd cluster

```
[root@node0 /]# etcdctl member list -w table
+------------------+---------+------+-----------------------+-----------------------+------------+
|        ID        | STATUS  | NAME |       PEER ADDRS      |      CLIENT ADDRS     | IS LEARNER |
+------------------+---------+------+-----------------------+-----------------------+------------+
| 4457dece7b809955 | started |  pg2 | http://172.20.0.4:2380 | http://172.20.0.4:2379 |      false |
| 5968f5c7c4cd9a98 | started |  pg0 | http://172.20.0.2:2380 | http://172.20.0.2:2379 |      false |
| 8a914bf5780754bf | started |  pg1 | http://172.20.0.3:2380 | http://172.20.0.3:2379 |      false |
+------------------+---------+------+-----------------------+-----------------------+------------+
[root@node0 /]# etcdctl endpoint status -w table
+----------------+------------------+---------+---------+-----------+------------+-----------+------------+--------------------+--------+
|    ENDPOINT    |        ID        | VERSION | DB SIZE | IS LEADER | IS LEARNER | RAFT TERM | RAFT INDEX | RAFT APPLIED INDEX | ERRORS |
+----------------+------------------+---------+---------+-----------+------------+-----------+------------+--------------------+--------+
| 127.0.0.1:2379 | 5968f5c7c4cd9a98 |  3.5.12 |   20 kB |     false |      false |         3 |         14 |                 14 |        |
+----------------+------------------+---------+---------+-----------+------------+-----------+------------+--------------------+--------+
[root@node0 /]#
```

Patroni

- Thin Wrapper as PostgreSQL Service Manager
- Can Bootstrap a new cluster
- Can takeover and existing cluster and manage
- Manages Leader key and PostgreSQL configuration in DCS
- Ensures configurations are propagated to all the nodes
- Backup tools like pgBackRest is well integratable

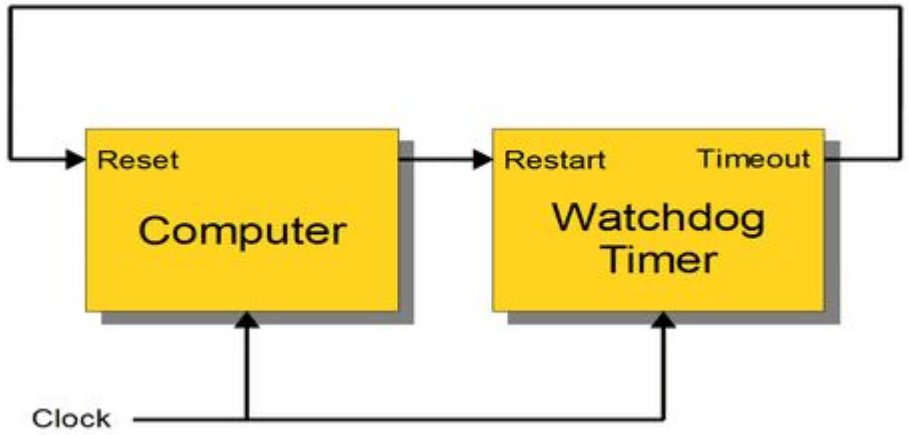
Patroni configuration

https://github.com/jobinau/pgscripts/blob/main/patroni/patroni.yml

Walk through

Linux Watchdog

Secondary level protection against Split-Brain



Connection routing

- Application connectors / Drivers
- External router - HAProxy
- VIP failover

HAProxy Configuration

```
listen primary
    bind *:5000
    option httpchk OPTIONS /primary
    http-check expect status 200
    default-server inter 3s fall 3 rise 2 on-marked-down shutdown-sessions
    server pg0 pg0:5432 maxconn 100 check port 8008
    server pg1 pg1:5432 maxconn 100 check port 8008
    server pg2 pg2:5432 maxconn 100 check port 8008

listen standbys
    bind *:5001
    option httpchk OPTIONS /replica
    http-check expect status 200
    default-server inter 3s fall 3 rise 2 on-marked-down shutdown-sessions
    server pg0 pg0:5432 maxconn 100 check port 8008
    server pg1 pg1:5432 maxconn 100 check port 8008
    server pg2 pg2:5432 maxconn 100 check port 8008
```

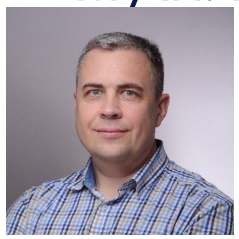
https://github.com/jobinau/pgscripts/blob/main/patroni/haproxy.cfg

Myths and Truths about Synchronous replication

- Myth: Transaction is committed after receiving confirmation from synchronous standby

  Fact : Transaction is committed locally.
- Myth: Synchronous replication guarantees Zero Recovery Point Objective (RPO) / no data loss

  Fact : Since transaction is committed on primary, it could become visible on primary if there is cancellation/disconnect. But standby won't see that and data will be lost if there is a failover
- Mythi: Reading from Synchronous standby is as safe as reading from Primary
- …
- Myth: Having a synchronous standby won't affect the performance of primary



https://www.postgresql.eu/events/pgconfde2025/sessions/session/6559/slides/663/Myths%20and%20Truths%20about%20Synchronous%20Replication%20in%20PostgreSQL.pdf

Thank You!