POSTGRES HA

The “why” most books skip | Summer Academy

EDB POSTGRES AI

AGENDA

WHAT WE'LL COVER

45 minutes, one thesis: everything is HA, or none is.

01

FOUNDATIONS

CAP & PACELC — what we're actually solving for.

02

ARCHITECTURES

Active/Passive vs. Active/Active, and why each exists.

03

THE REAL MESSAGE

Weakest link, quorum math, minimum topology.

04

FAILURE MODES

Split-brain, fencing, and the failover that makes things worse. A real-world example.

05

OBSERVABILITY & TESTING

The signals and drills that turn hope into confidence.

06

BEYOND THE DATABASE

Security and backup — the parts books skip.

FORMAT Plus 5 minutes of intro and 10 minutes of Q&A — 60 minutes total. No demos - this is a talk about issues, not specific products.

01 — FOUNDATIONS

What are we actually solving for?

FOUNDATIONS

THE CAP THEOREM

During a network partition, you can only keep two of three.

CONSISTENCY

Every read returns the most recent write, everywhere.

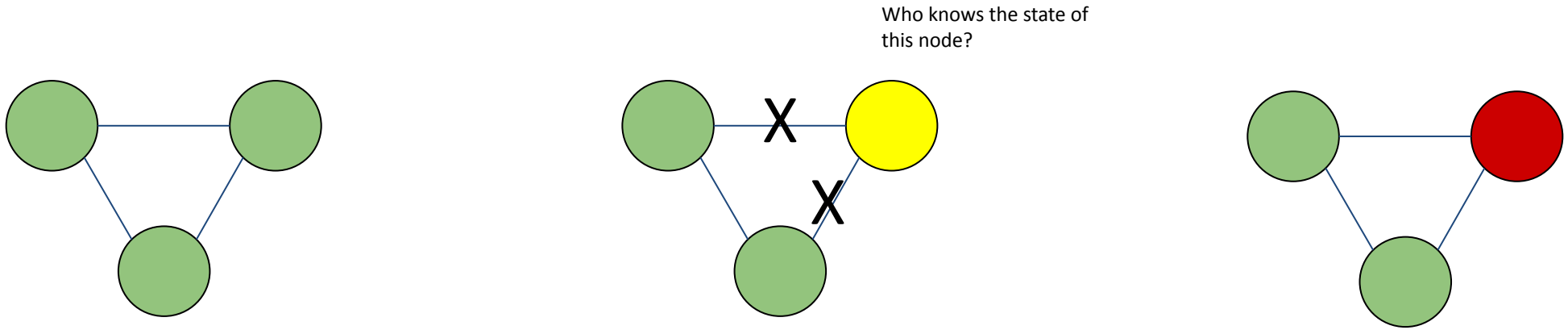
AVAILABILITY

Every request gets a response — no timeouts, no refusals.

PARTITION TOLERANCE

The system keeps working even when the network splits.

TAKEAWAY Real networks partition. The practical everyday choice is Consistency vs. Availability.



THE EXTENSION: PACELC

What governs behavior when the network is healthy — which is nearly always?

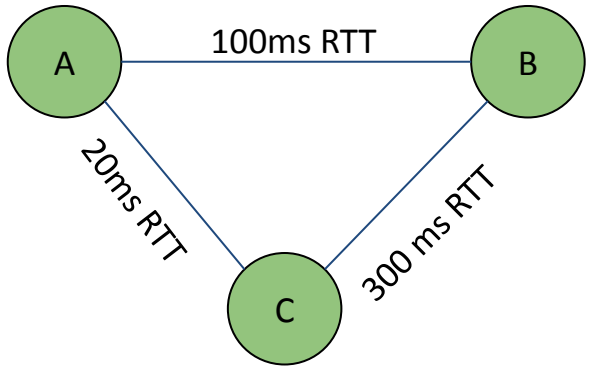
DURING A PARTITION (P)

- Same choice as CAP: **A**vailability or **C**onsistency.
- Rare in absolute terms — but the failure mode everyone plans for.

WHEN HEALTHY (ELSE)

- New tradeoff: **L**atency or **C**onsistency.
- This is the state your cluster is in 99%+ of the time.

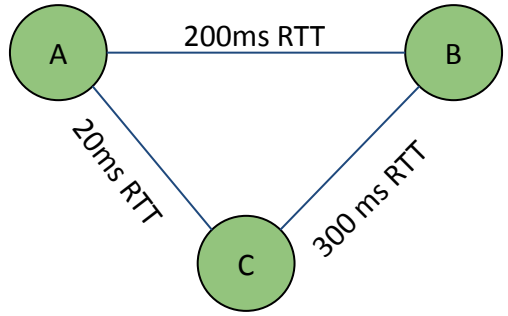
SO WHAT This is your synchronous vs. asynchronous replication decision, made concrete.



FOUNDATIONS

FROM THEORY TO CONFIG

PACELC isn't academic — it's a parameter you set today.



WHERE THIS SHOWS UP IN POSTGRES

APPROACH	WHAT YOU GET	WHAT IT COSTS
Async replication (default)	Fast commits	Possible data loss on failover
Sync replication	No data loss	Higher commit latency

PATTERN Same lever, two layers: replica-level (sync/async) or cluster-level (Quorum Commit).

02 — ARCHITECTURES

Two families — and the tool choice is downstream of the decision.

ACTIVE/PASSIVE: Failover Managers

One writer, standbys promoted on failure. Tools differ in HOW consensus works.

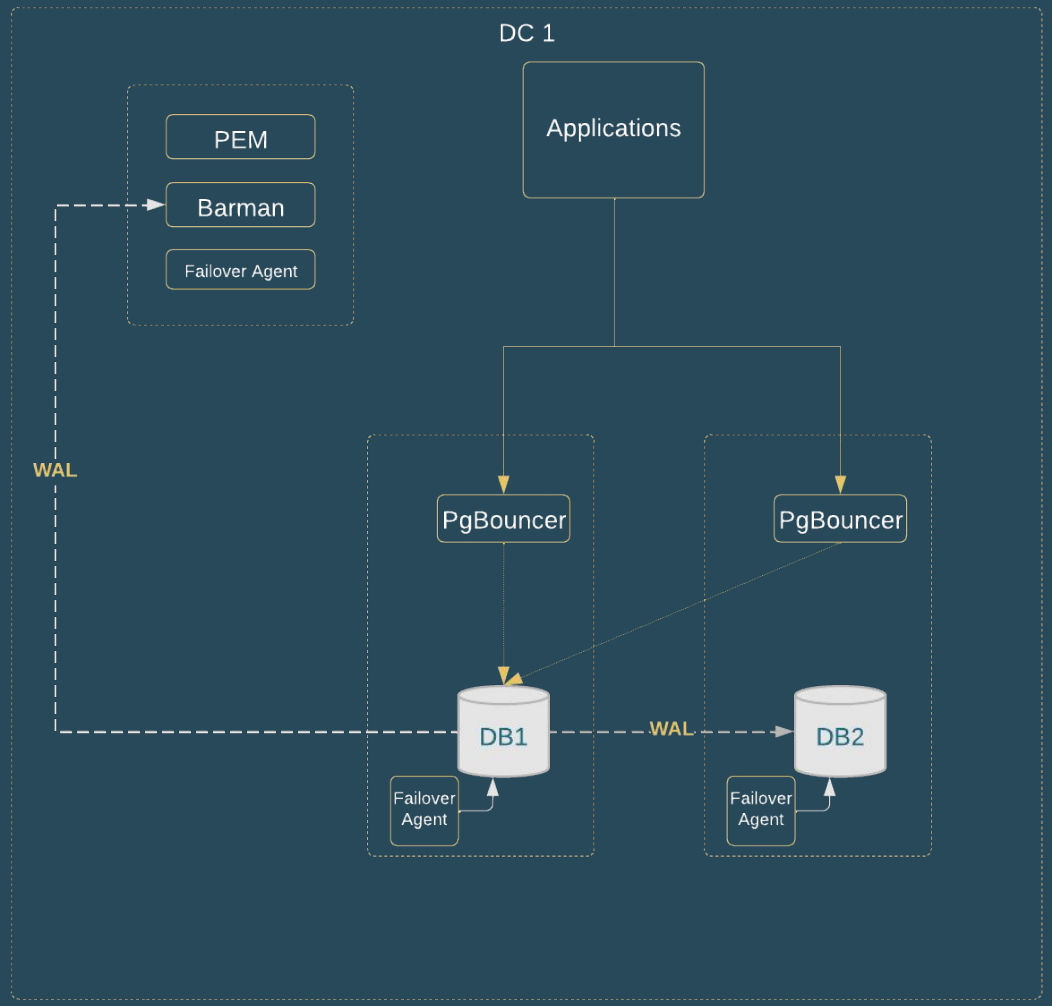
	EFM	repmgr	Patroni	CloudNativePg
FAILURE DETECTION	Agent-based cluster monitor, no external DCS required	repmgrd daemons monitor each other directly	DCS heartbeat via etcd, Consul, ZooKeeper, or Kubernetes	Kubernetes Operator, Auto-pilot maturity
CONSENSUS SOURCE	EFM's own cluster protocol reaches majority agreement	Daemon-to-daemon voting. note: consensus is off by default for backward compatibility	Leader election lives in the DCS, not in Patroni itself	Kubernetes API.
SPLIT-BRAIN PROTECTION	Built in via EFM cluster majority vote; override with fencing script	Partial quorum based fencing	Guaranteed by the DCS's leader lease mechanism	K8S API + isolation checks etc.
CLIENT REDIRECTION	Virtual IP directly supported; PgBouncer and PgPool-II via failover scripts	PgBouncer via failover scripts	REST API driving HAProxy.	Kubernetes Services for -rw, -ro, -r

KEY POINTS

All tools are viable options, if properly configured. HA must be tested and validated at architecture level, as it depends on external conditions.

Varying level of control and need for user-supplied scripting across the various tools

Active/Passive



Active + passive database with Barman WAL streaming

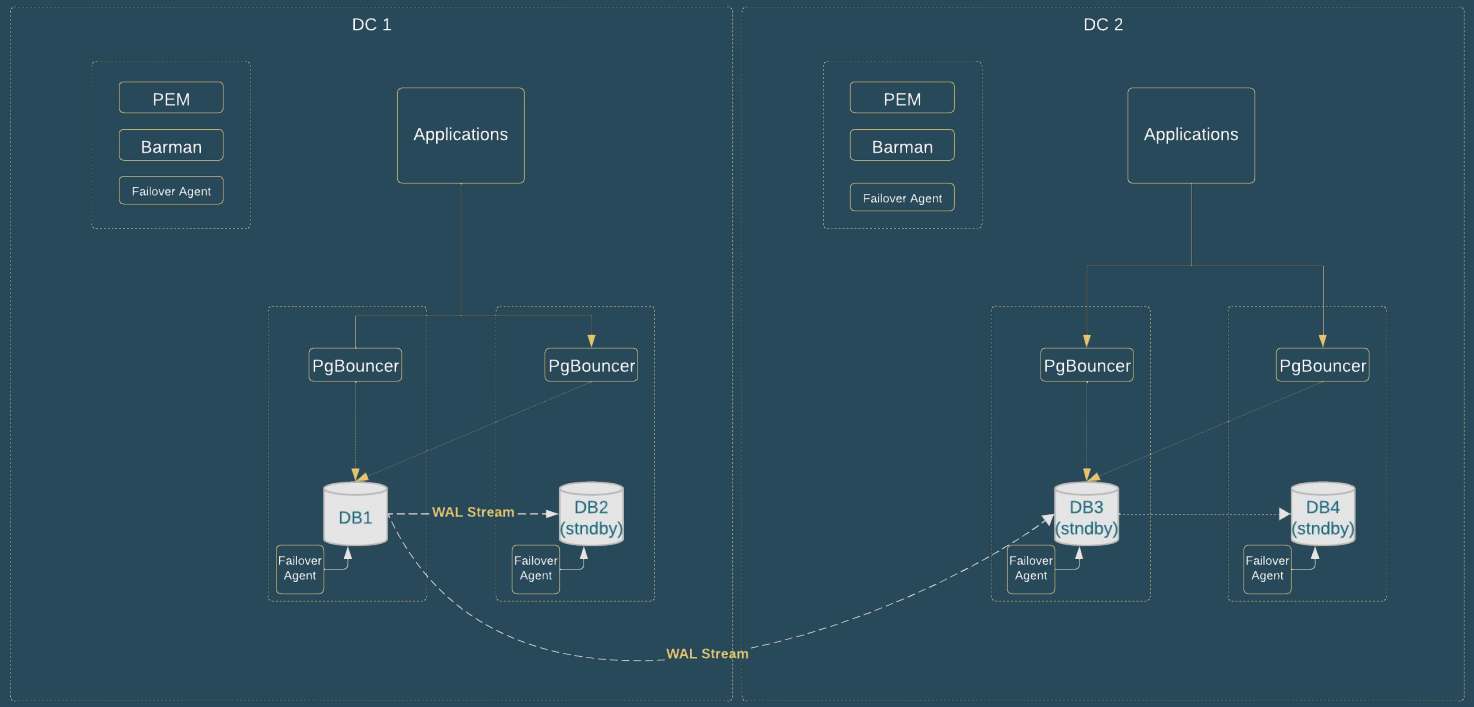
Features:

- **RTO**: 30 - 120 sec (99.99%)
- **RPO**: ~0 (with sync commit)

Notes:

Replication and synchronous commit to both Barman and standby are possible options.

Active/Passive with DR



Active + three passive databases with Barman WAL streaming. Replicated setup for DR site.

Features:

- **RTO**: 30 - 120 sec (99.99%)
- **RPO**: ~0 (with sync commit)

Geo-redundancy

Notes:

Local failover is automated.

Global failover can also be automated, but users prefer to keep it manual, because a full site loss is a major event that requires decision ownership.

ARCHITECTURES

ACTIVE / ACTIVE: EDB POSTGRES DISTRIBUTED

Multi-writer. Currently PGD 6.5 — built on a decade-hardened BDR lineage.

QUORUM COMMIT

- Raft-style cross-node consistency
- Closes the “eventual consistency” gap

BEST FOR: TIER-1 FINANCIAL WORKLOADS

CONNECTION MANAGER

- Native, built-in connection routing
- No separate proxy layer required

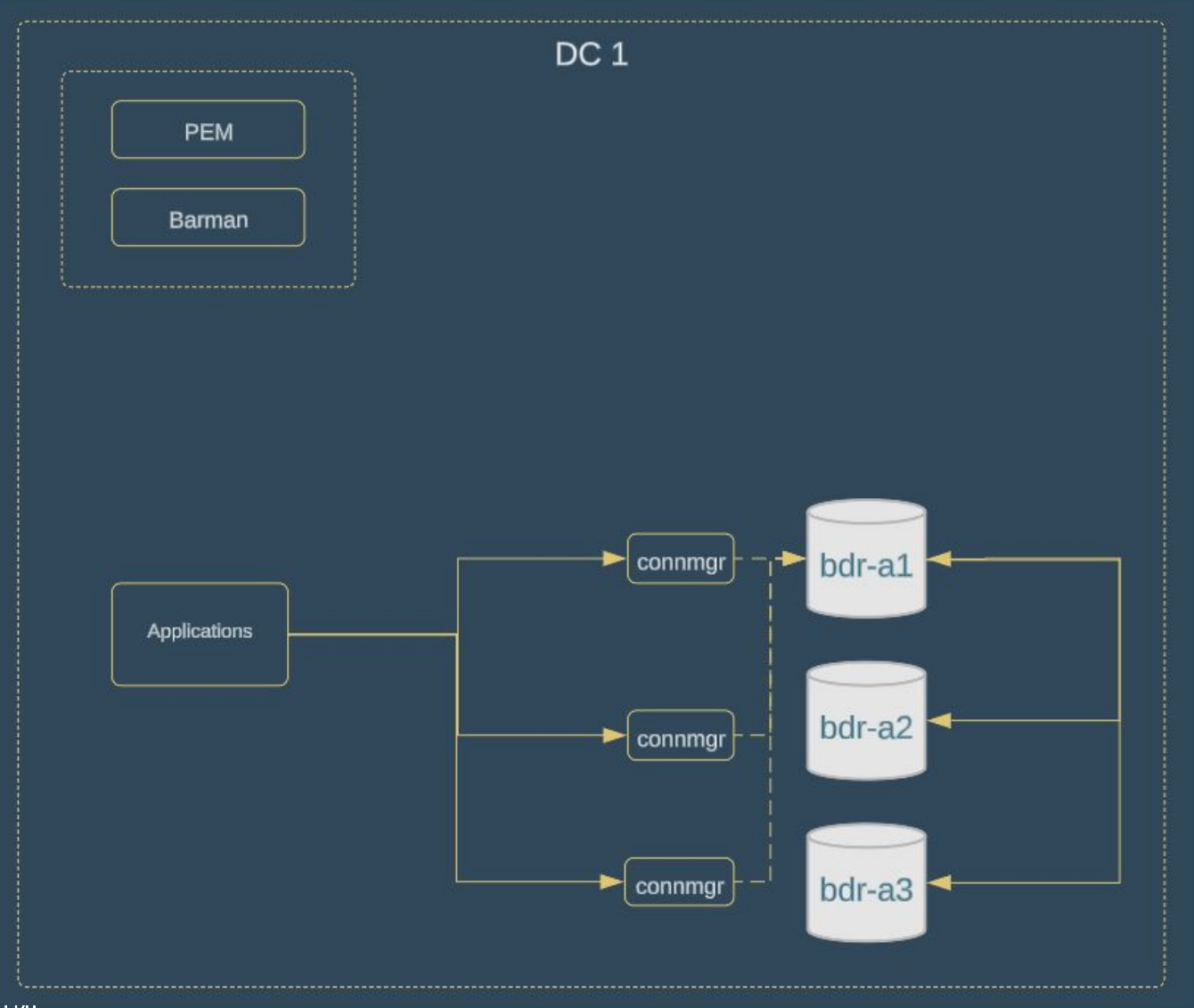
BEST FOR: SIMPLER OPS FOOTPRINT

GROUP COMMIT

- Degrade to Async based on latency limits
- provides most flexibility in managing PACELC tradeoffs

BEST FOR: DECLARATIVELY MANAGE PACELC TRADEOFFS FOR RPO~0

ACTIVE/ACTIVE (PGD v6)

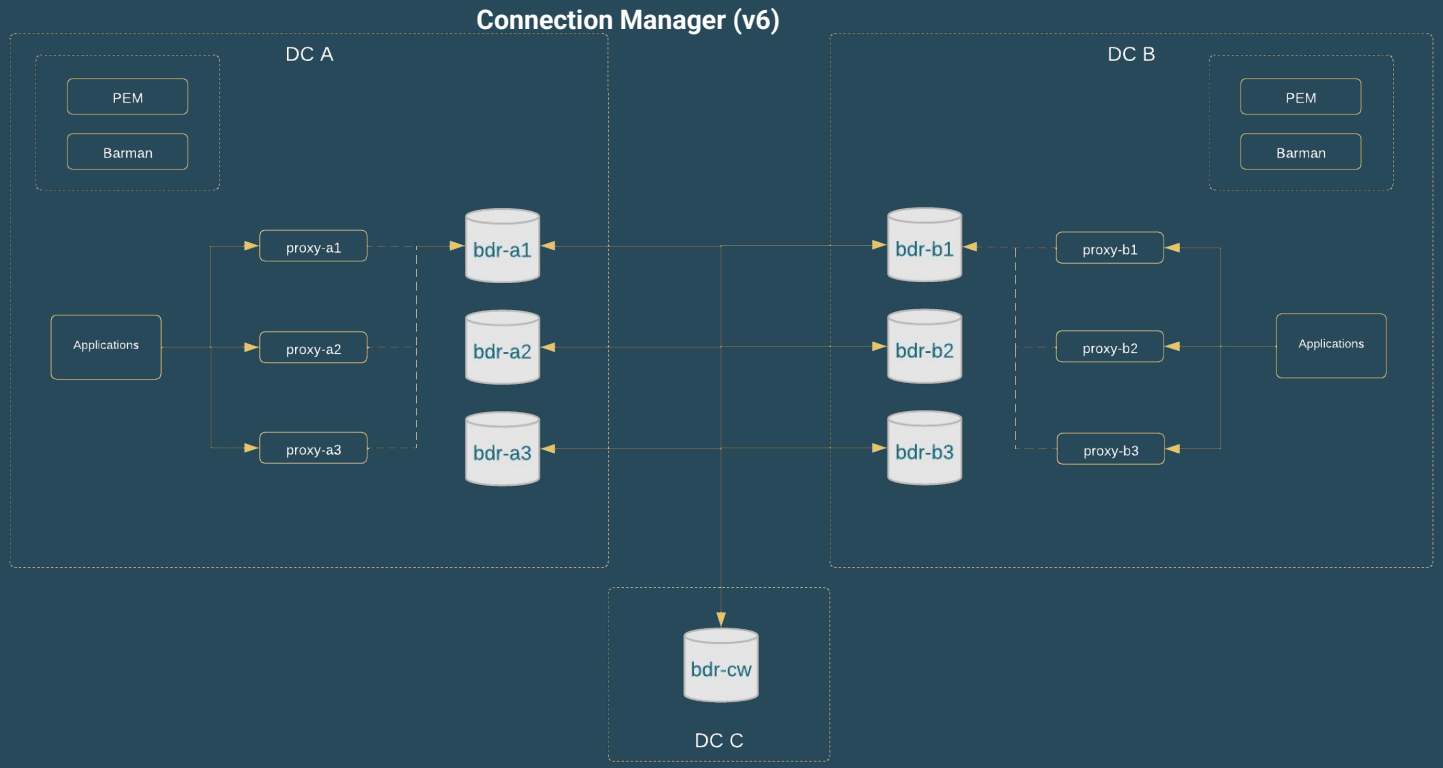


Tree active (PGD) databases

Features:

- **RTO**: <2 sec (99.999%)
- **RPO**: ~0 (with sync commit)
- Automatic conflict resolution
- Rolling system upgrades

Active/Active - 6 nodes



Six active (PGD) databases, one witness

Features:

- **RTO**: <2 sec (99.999%)
- **RPO**: ~0 (with sync commit)
- Automatic conflict resolution
- Rolling system upgrades possible
- Full geo-redundancy

Notes:

Witness db (bdr-cw) does not contain “real” data, but protects consensus against full loss of DC.
bdr-cw is optional - will require manual procedure.
Further split into smaller DCs is possible.
Same deployment method is used for 2, 4, 6 or more PGD nodes.

A COMMON MISCONCEPTION

"Active-active Postgres" is not one thing.

WHAT PEOPLE ASSUME

- All multi-master tools are built the same way.
- It's just logical replication with conflict handling bolted on.
- "BDR" means the original BDR 1.0 architecture.

WHAT'S ACTUALLY TRUE

- PGD descends from BDR 1.0, with extra 10 years of work.
- Full control of latency and consistency across nodes and locations using features such as Commit Scopes and Lag Control.

PGD is the Rolls-Royce of High Availability.

It is not "just" Logical replication + conflict resolution. PGD includes advanced control of commit scopes, automatic degrade from synchronous to asynchronous commit. Lag control and Commit-at-most-once for optimistic async commits.

PGD does not mean distributed eventual consistency. It means granular control of commit for availability vs latency and network partitioning - per transaction if needed.

PGD 6

The latest version

SIMPLIFICATION TREND

- External proxy process replaced by built-in Connection Manager
- External consensus already internalized in PGD 4
- PGD 6 is just an extension turning Postgres into PGD

WHY?

- Fewer moving parts to deploy and manage
- Fewer failure scenarios to detect and address
- Less variability improves test quality by reducing the surface that needs testing

LESSON: Sometimes, Less is More.

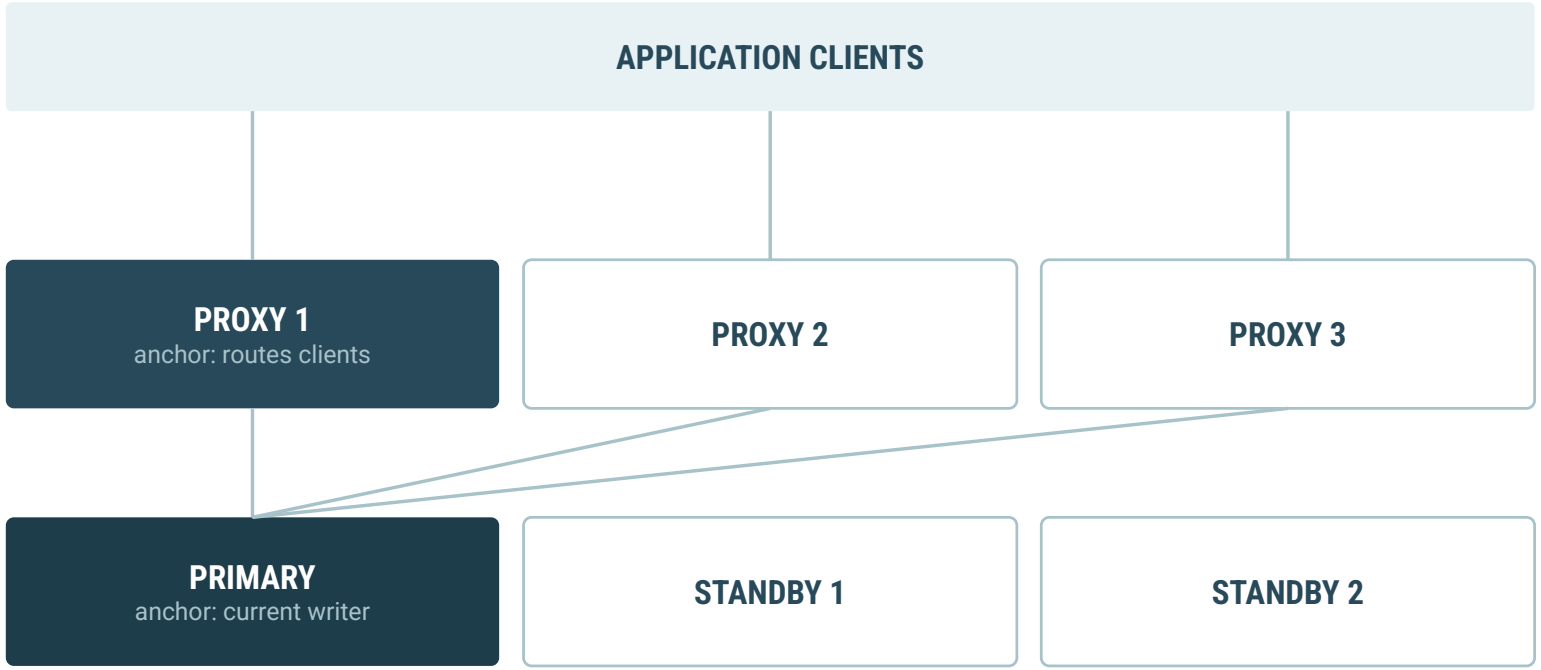
03 — THE MESSAGE

Everything is HA, or none is.

THE REAL MESSAGE

THE CHAIN HAS THREE LINKS

A SPOF in any tier is a SPOF in the whole system.



THE REAL MESSAGE

THE QUORUM MATH (FOR STATEFUL COMPONENTS)

Why the minimum is 3, not 2 — and definitely not 1.

2 NODES

- A node loss looks identical to a network partition.
- No majority exists — either side could be “right.”
- Any auto-promotion risks split-brain.

3 NODES

- A majority (2-of-3) exists even if 1 node is unreachable.
- The surviving majority can safely make a decision.

→ **Tolerates 1 failure, safely**

RULE Quorum = ⌈n/2⌉ + 1. Apply it to every tier — DB, proxy, and consensus store — not just the database.

EDB POSTGRES AI

THE REAL MESSAGE

THE MINIMUM TOPOLOGY

If you want genuine HA, this is the floor — not the ceiling.

3

DATABASE NODES

One primary, majority of standbys can safely detect and act on a failure.

One can be a witness.

3

PROXY / POOLING NODES

The layer clients actually connect to. One proxy = one SPOF, regardless of DB count.

3 is best, 2 can work if state is held in the database.

3

PATRONI: ETCD (DCS)

etcd itself is the Raft quorum implementation.

Alternatives: Consul, Zookeeper, …

Note: A 2-node DCS cluster is (almost) worse than 1. If one node is lost - DCS looks available, but can't make decisions (majority lost)

BOTTOM LINE Skip a tier and that tier becomes your real single point of failure. The "right" setup is a question of comfort level.

04 — FAILURE MODES

The scenarios that turn a passing test into a 2am page.

FAILURE MODES

FIELD REPORT: KILLED THE PRIMARY, PROMOTED NOBODY

A real PACELC related incident. The network failed — the outage was optional.

THE TRIGGER

– Network partition results in Primary becoming isolated through IP.

– Primary fails its health check — not reachable for EFM by IP adresses.

– EFM fences the primary. Correct action. Primary database and agent are isolated - risk that a promotion happens elsewhere, fencing is mandatory.

THE GAP

– The same network event makes the standbys unreachable too.

– No standby can be confirmed as promotion-safe.

– Nothing gets promoted. Zero primaries.

→ **A network configuration becomes SPOF.**

– Due to critical logs being unavailable a full RCA was not possible.

LESSONS

Fencing and promotion are two decisions drawn from the same shaky network path. A health-check failure means “I can’t see it,” not “it’s dead.” Gate the kill on the promotion, not the other way around.

HA must be tested and validated on the whole architecture, not just on the tool.

THE FIX: CLOSE THE GAP IN EFM

Kill and promote need to share one quorum check, not two independent network paths.

	TOPOLOGY	STOP.ISOLATED.PRIMARY	STOP.FAILED.PRIMARY + MINIMUM.STANDBYS
WHAT IT DOES	3+ EFM agents (primary, standby, witness), each on an independent network path or AZ. Always an odd number (>3).	Primary self-fences when it loses contact with the cluster majority — the same condition that lets the majority promote	A remote-triggered kill is paired with a safety floor on confirmed standbys before anything is promoted
WHY IT MATTERS HERE	No single link/AZ failure can deny majority to everyone at once	Ties the kill decision to the same majority check used for promotion — not an independent reachability guess	Prevents a remote kill from firing faster or looser than the promotion decision it depends on

KEY POINT This isn't an EFM flaw — every fencing-based tool can hit this if kill and promote aren't gated on the same quorum. EFM already ships the mechanism; the topology and defaults have to be set to use it.

FAILURE MODES

SPLIT-BRAIN ISN'T A QUORUM BUG, IT'S A FENCING GAP

Quorum decides who should be primary. It doesn't make the loser stop.

WHAT QUORUM GUARANTEES

– Only one node can win an election — a majority lock or lease can only be held by one side.
– The winning side knows it won.

WHAT QUORUM DOESN'T GUARANTEE

– That the losing node stops accepting writes on its own.

– That the gap between “elected” and “confirmed dead” closes instantly.

Split-brain lives in the gap between “new primary elected” and “old primary confirmed dead.”

RULE Close that gap with fencing, not hope.

EDB POSTGRES AI

FAILURE MODES

FENCE IT, DON'T ASSUME IT

An unreachable primary and a dead primary look identical from the outside.

	WATCHDOG / STONITH	SELF-FENCING	EXTERNAL FENCING SCRIPT
HOW IT WORKS	Kernel-level watchdog reboots the node if it can't renew its leadership lease (Patroni's approach)	The node detects it lost the lease itself and rejects writes / closes connections	A hook actively isolates or kills the old primary before the new one takes writes (EFM/repmgr fencing scripts)
TRIGGERED BY	Loss of lease renewal, decided by the node itself	Loss of lease/majority contact, decided by the node itself	A remote agent's decision, run before promotion completes
TRADE-OFF	Requires hardware/kernel support; a hard reset	Depends on the node correctly detecting its own isolation	Manual scripting; only as safe as its trigger conditions

KEY POINT "Assume it's dead" is how you get two primaries. Fence first, promote second.

FAILURE MODES

AUTOMATED FAILOVER CAN TURN A BLIP INTO AN OUTAGE

The failure isn't the failover — it's failing over on the wrong signal.

FLAPPING	PREMATURE PROMOTION	CASCADING FAILOVER
– A transient network blip trips failover, then trips back. – Repeated promotions and connection storms follow.	– A standby is promoted before the primary is confirmed down — only unreachable. – Risks split-brain if the “dead” primary is still writing.	– Proxy/pooler churn during an election storm. – One election becomes an app-visible outage across every tier.

PATTERN Tune failure-detection delay and confirmation thresholds, and default to pausing — not promoting — when confidence is low. A slower correct failover beats a fast wrong one.

EDB POSTGRES AI

05 — OBSERVABILITY & TESTING

The difference between hoping it works and knowing it does.

OBSERVABILITY & TESTING

METRICS THAT ACTUALLY PREDICT TROUBLE

By the time failover fails, it's too late to start watching.

REPLICATION & CONSENSUS HEALTH

– Replication lag (replay_lag / flush_lag) — your async data-loss exposure, live.
– DCS/consensus health (etcd, Patroni's /health) — your failover capability, live.

CLIENT-FACING HEALTH

– Proxy/pooler connection saturation — where clients feel it first.

– Promotion and election frequency — count it; frequency itself is a flapping signal.

SO WHAT Alert on lag and consensus health before a failover, not just on whether the failover succeeded.

OBSERVABILITY & TESTING

A REHEARSED FAILOVER IS ROUTINE. AN UNTESTED ONE IS A COIN FLIP.

Confidence in HA is earned in staging, not assumed in production.

GAME DAYS

– Scheduled failover drills, under realistic load, not just at 2am on a quiet Tuesday.
– Include the asymmetric cases — isolated primary, not just node-down.

CHAOS TESTING

– Kill the primary, partition the network, fill the disk — confirm fencing and quorum hold.
– Validate the whole path: client reconnect, stale connection eviction, measured RTO/RPO.

REMEMBER A team found their Patroni-based Postgres-on-Kubernetes clusters couldn't safely failover — during a game day, not an outage. Find it in a drill, not production.

06 — BEYOND THE DATABASE

Security & backup — the parts most HA talks skip.

BEYOND THE DATABASE

SECURITY: DETECT, DON'T JUST PREVENT

Audit logging doesn't stop a breach — it documents one.

AUDIT LOGGING (PGAUDIT, EDB Audit)

- Session- and object-level logging beyond log_statement.
- The record: what happened, who did it, when.
- Forensic value — doesn't prevent anything by itself.

EVENT TRIGGERS

- ddl_command_start / end, table_rewrite hooks.
- The tripwire: fires while a change is happening.
- Catches unauthorized schema changes on sensitive tables.

PAIR THEM Audit logging is the record; event triggers are the tripwire. Neither replaces access control.

BEYOND THE DATABASE

BACKUP, ESPECIALLY FOR VLDBS

Rethink what a “backup” is doing for you at scale.

STANDBYS = YOUR BACKUP

– For a VLDB, standbys are already warm and continuously tested by replication traffic.
– Restoring from a standby is fast — far faster than a physical restore.

WHEN PHYSICAL BACKUPS MATTER

– Human error — a dropped table replicates everywhere just as fast as a good write.
– Ransomware / corruption — you need a point BEFORE the damage replicated.
– Regulatory retention, independent of operational recovery.

REMEMBER Test your backups. An untested backup is not worth the bytes it uses.

WRAP-UP

THE THROUGH-LINE

One thesis, six consequences.

ARCHITECTURE

Choose active/passive vs. active/active by your consistency needs, not habit.

QUORUM MATH

Odd numbers, every tier — DB, proxy, and consensus store alike.

FAILURE MODES

Gate the kill on the promotion. Fence first, promote second.

OBSERVABILITY & TESTING

Watch leading indicators; rehearse failover before you need it.

SECURITY

Detective controls, not preventive ones. Pair logging with tripwires.

BACKUP

Standbys for speed. Physical backups for what replication can't save you from.

THE THESIS Everything is HA, or none is. A chain breaks at its weakest link.

THANK YOU

Questions? — 10 minutes