



EDB Postgres Distributed Always On Architectures

AUTHORED BY:

Gianni Ciolli

VP, Solutions Architecture

Kelly Poole

VP, Product Management

Petr Jelinek

VP, Chief Architect

POWER TO POSTGRES

Contents

Introduction	03
1. Overview of the Always On Architectures	04
2. Architecture Details	06
3. How to Choose the Right Architecture	10
4. Deployment and sizing considerations	12
5. Adding flexibility to the standard architectures	14
6. Conclusion	16

Introduction

This whitepaper describes the blueprint for Always On architectures for Postgres. These architectures reflect EDB's recommended practices and help customers achieve highest possible service availability in multiple different configurations, ranging from single-location architectures all the way to complex distributed systems that protect from hardware failures and data center failures. The architectures leverage EDB Postgres Distributed's multi-master capability and its ability to achieve 99.999% availability including maintenance operations.

This whitepaper builds on [EDB Postgres Distributed: The Next Generation of PostgreSQL High Availability](#) and [The End of the Reign of Oracle RAC: EDB Postgres Distributed Always On](#).

EDB Postgres Distributed can be used for architectures beyond the examples described in this document. Use-case specific variations have been successfully deployed in production; however, such variations must undergo rigorous architecture review first, and EDB's standard deployment tool for Always On architectures - Trusted Postgres Architect (TPA) - should be enabled to support the variations before they can be supported in production environments.

1. Overview of the Always On Architectures



Overview of the Always On Architectures

EDB has identified a set of standardized architectures to support single or multi-location deployments with varying levels of redundancy depending on your RPO and RTO requirements.

EDB Postgres Distributed consists of two key components:

- Bi-Directional Replication (BDR) - a Postgres extension that orchestrates the distributed cluster, including transaction replication, durability, node management, and more.
- PGD-Proxy - a connection router that makes sure the application is connected to the right PGD nodes.

All Always On architectures protect against a range of failure modes. Depending on the required fault tolerance, Always On architectures can be reconfigured and expanded to tolerate an increasing range of failures. For example, a single active location with 2 data nodes protects against local hardware failure, but does not provide protection from location failure (data center or region). Extending that architecture with a backup at a different location ensures some protection in case of the catastrophic loss of a location, but the database still has to be restored from backup first which may violate recovery time objective (RTO) requirements. By adding a second active location, users may ensure that service remains available even in case a location goes offline. Finally, subscriber only nodes can be added to any architecture to offload read workload and meet reporting, archival and analytic needs.

Each configuration can provide zero recovery point objective (RPO), as data can be streamed synchronously to at least one local copy, thus guaranteeing zero data loss in case of local hardware failure.

Increasing the availability guarantee always drives additional cost for hardware and licenses, networking requirements, and operational complexity. Thus it is important to carefully consider the availability and compliance requirements before choosing an architecture.

2. Architecture Details



Architecture Details

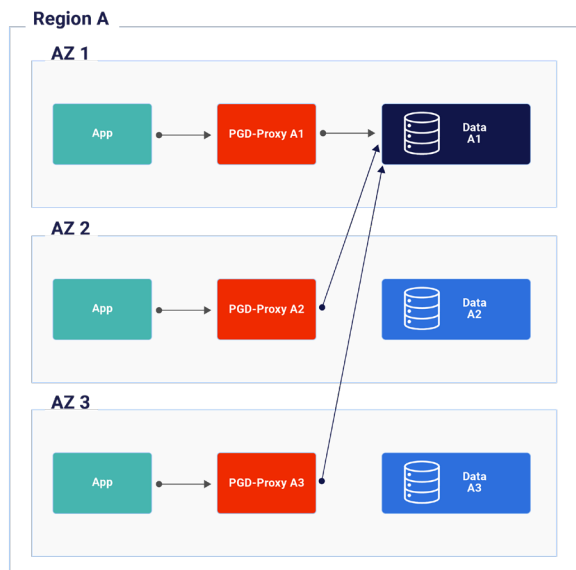
By default, application transactions do not require cluster-wide consensus for DML (selects, inserts, updates, and deletes) allowing for lower latency and better performance. However, for certain operations such as generating new global sequences or performing distributed DDL, EDB Postgres Distributed requires an odd number of nodes to make decisions using a Raft (<https://raft.github.io>)based consensus model. Thus, even the simpler architectures always have three nodes, even if not all of them are storing data.

Applications connect to the standard Always On architectures via multi-host connection strings, where each PGD-Proxy server is a distinct entry in the multi-host connection string. There should always be at least two proxy nodes in each location to ensure high availability. The proxy can be co-located with the database instance, in which case it's recommended to put the proxy on every data node.

Other connection mechanisms have been successfully deployed in production, but they are not part of the standard Always On architectures.



PGD5 Always On: One Location

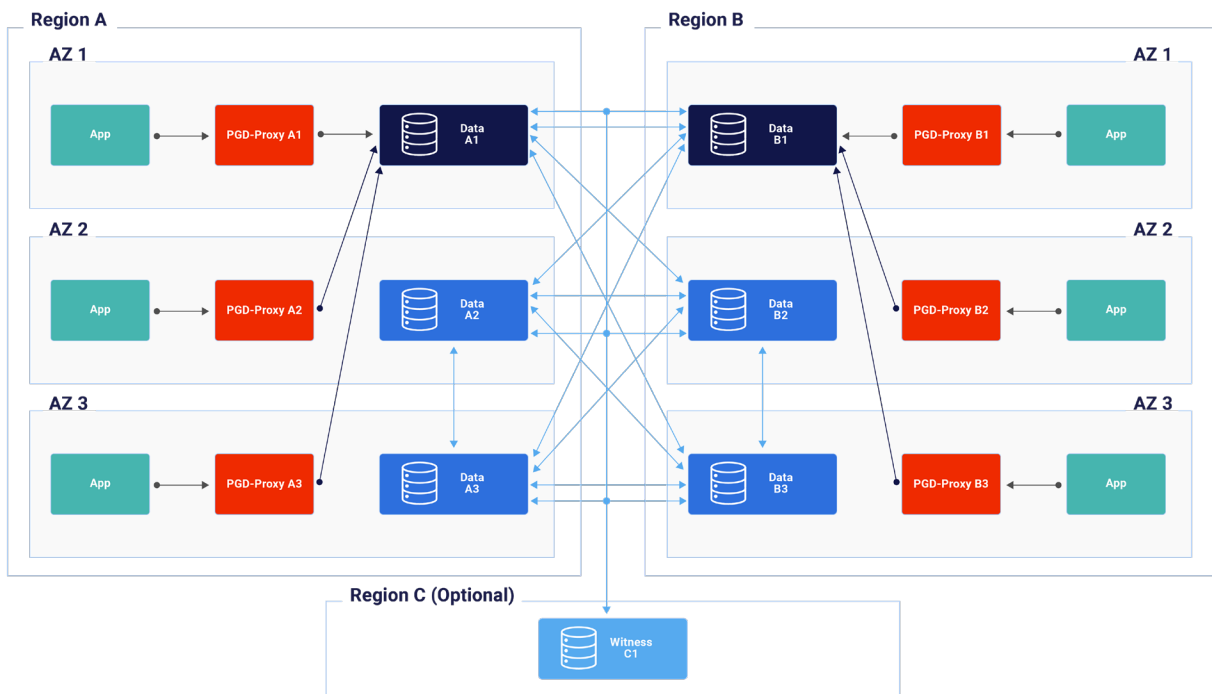


- Redundant hardware to quickly restore from local failures
 - 3 PGD nodes
 - ↪ could be 3 data nodes (recommended), or 2 data nodes and 1 witness which does not hold data (not depicted)
 - A PGD-Proxy for each data node with affinity to the applications
 - ↪ can be co-located with data node
- Replication between nodes 1 and 3 is not shown but occurs as part of the replication mesh

- Replication between nodes 1 and 3 is not shown but occurs as part of the replication mesh
- Barman for backup and recovery (not depicted)
 - Offsite is optional, but recommended
 - Can be shared by multiple clusters
- Postgres Enterprise Manager (PEM) for monitoring (not depicted)
 - Can be shared by multiple clusters

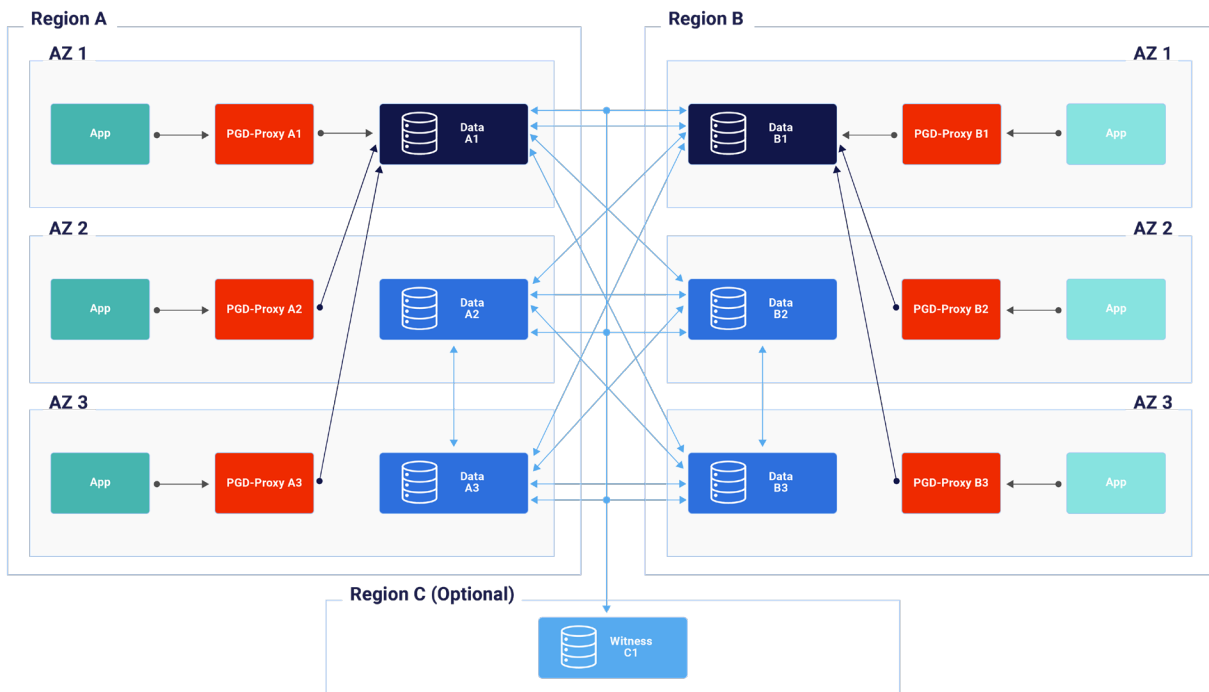


PGD5 Always On: Two Locations (Active/Active)





PGD5 Always On: Two Locations (Active/Passive)



- Application can be Active/Active in each location, or Active/Passive or Active DR with only one location taking writes
- Replication between nodes 1 and 3 within region is not shown but occurs as part of the replication mesh
- Redundant hardware to quickly restore from local failures
 - 6 PGD nodes total, 3 in each location
 - ➔ could be 3 data nodes (recommended), or 2 data nodes and 1 witness which does not hold data (not depicted)
 - A PGD-Proxy for each data node with affinity to the applications
 - ➔ can be co-located with data node
- Barman for backup and recovery (not depicted)
 - Can be shared by multiple clusters
- Postgres Enterprise Manager (PEM) for monitoring (not depicted)
 - Can be shared by multiple clusters
- An optional witness node should be placed in a third region to increase tolerance for location failure
 - Otherwise, when a location fails, actions requiring global consensus will be blocked such as adding new nodes and distributed DDL

3. How to Choose the Right Architecture



How to Choose the Right Architecture

All architectures provide the following:

- Hardware failure protection
- Zero downtime upgrades
- Support for availability zones in public/private cloud

This section discusses criteria that help in selecting the appropriate Always On Architecture

	Single Data Location	Two Data Locations	Two Data Locations + Witness	Three or More Data Locations
Locations needed	1	2	3	3+
Fast restoration of local HA after data node failure	Yes - if 3 PGD data nodes No - if 2 PGD data nodes	Yes - if 3 PGD data nodes No - if 2 PGD data nodes	Yes - if 3 PGD data nodes No - if 2 PGD data nodes	Yes - if 3 PGD data nodes No - if 2 PGD data nodes
Data protection in case of location failure	No (unless offsite backup)	Yes	Yes	Yes
Global consensus in case of location failure	N/A	No	Yes	Yes
Data restore required after location failure	Yes	No	No	No
Immediate failover in case of location failure	No - requires data restore from backup	Yes - alternate Location	Yes - alternate Location	Yes - alternate Location
Cross Location Network Traffic	Only if backup is offsite	Full replication traffic	Full replication traffic	Full replication traffic
License Cost	2 or 3 PGD data nodes	4 or 6 PGD data nodes	4 or 6 PGD data nodes	6+ PGD data nodes

This general progression can continue to 3 or 5 data locations by continually adding more 3 node groups to the cluster. The addition of witness only location is only recommended with 2 data locations.

If using only 2 data nodes and a witness, then recovery time locally involves the time to restore HA, includes the time to provision an additional Virtual Machine (VM) and approximately 60 minutes per 500GB of data being restored.

4. Deployment and sizing considerations






Deployment and sizing considerations

For production deployments, EDB recommends a minimum of 4 cores for each EDB Postgres Distributed data node. Witness nodes do not participate in the data replication operation and do not have to meet this requirement. Logical standbys should always be sized exactly like the EDB Postgres Distributed data nodes to avoid performance degradations in case of a node promotion. In production deployments PGD Proxy nodes require a minimum of 1 core, and as a rule of thumb they should scale incrementally in correlation with an increase in the number of database cores in approximately a 1:10 ratio. EDB recommends detailed benchmarking of your specific performance requirements to determine appropriate sizing based on your workload. The EDB Professional Services team is available to assist if needed.

For development purposes, PGD data nodes should not be assigned less than 2 cores. The sizing of Barman nodes depends on the database size and the data change rate.

PGD nodes, Barman nodes, and PGD Proxy nodes can be deployed on virtual machines or in a bare metal deployment mode. However, various affinity and anti-affinity properties must be maintained: deployment mode. However, various affinity and anti-affinity properties must be maintained:

-  Multiple PGD data and witness nodes should not be on VMs that are co-located on the same physical hardware, as that reduces resilience.
-  Multiple PGD Proxy nodes should not be on VMs that are co-located on the same physical hardware, as that reduces resilience.
-  Single PGD Proxy nodes can be co-located with single PGD data nodes when deployed as VMs.

5. Adding flexibility to the standard architectures



Adding flexibility to the standard architectures

The single location architecture can be deployed in as many locations as desired to provide the data resiliency needed and proximity to applications and users maintaining the data. While EDB Postgres Distributed has a variety of conflict handling approaches available, in general care should be taken to minimize the number of expected collisions if allowing write activity from geographically disparate locations.

The standard architectures can also be expanded with two additional types of nodes:

- Subscriber only nodes
- Logical standbys

Subscriber only nodes can be used to achieve additional read scalability and to have data closer to users when the majority of an application's workload is read intensive with infrequent writes. They can also be leveraged to publish a subset of the data for reporting, archiving, and analytic needs.

Logical standbys receive replicated data from another node in the PGD cluster but do not participate in the replication mesh or consensus. They contain all the same data as the other PGD data nodes, and can quickly be promoted to a master if one of the data nodes fails to return the cluster to full capacity/consensus. They can be used in environments where network traffic between data centers is a concern; otherwise 3 PGD data nodes per location is always preferred.

6. Conclusion



Conclusion

EDB Postgres Distributed provides standard Always On architectures to support a varying degree of availability requirements, ranging from single-location architectures to multi-location architectures that include redundant hardware components. The Always On architectures are adaptive based on the number of data centers or regions desired and extensible for faster recovery from failures and additional downstream needs. These architectures have been proven in production and leverage EDB Postgres Distributed's logical replication and mesh-based architecture to achieve industry-leading Postgres availability of up to 99.999% in public and private cloud deployments.

[Learn more about EDB Postgres Distributed](#)



About EDB

EDB provides enterprise-class software and services that enable businesses and governments to harness the full power of Postgres, the world's leading open source database. With offices worldwide, EDB serves more than 1,500 customers, including leading financial services, government, media and communications and information technology organizations. As one of the leading contributors to the vibrant and fast-growing Postgres community, EDB is committed to driving technology innovation. With deep database expertise, EDB ensures high availability, reliability, security, 24x7 global support and advanced professional services, both on premises and in the cloud. This empowers enterprises to control risk, manage costs and scale efficiently.

For more information, visit www.enterprisedb.com.



EDB Postgres Distributed Always On Architectures

© Copyright EnterpriseDB Corporation 2023

EnterpriseDB Corporation

34 Crosby Drive

Suite 201

Bedford, MA 01730

EnterpriseDB and Postgres Enterprise Manager are registered trademarks of EnterpriseDB Corporation. EDB, EnterpriseDB, EDB Postgres, Postgres Enterprise Manager, and Power to Postgres are trademarks of EnterpriseDB Corporation. Oracle is a registered trademark of Oracle, Inc. Other trademarks may be trademarks of their respective owners. Postgres

POWER TO POSTGRES