EDB DAYS · TRACK 03

EPAS Technical Deep Dive:
Architecting for Portability, Performance, and Resiliency

Rahul Saha · Principal Resident Architect · 2026

ABOUT THE SPEAKER

Rahul Saha

Principal Resident Architect, EDB

CONNECT ON LINKEDIN

Scan to view profile

Embedded with Strategic Customers

Designs, hardens, and operates mission-critical Postgres estates directly alongside customer teams.

EPAS & Oracle Modernization

Focus areas: EPAS and Oracle modernization, EDB Postgres Distributed architectures, and performance engineering.

Resilient Operations

Working hands-on across the full stack covered today — geo-distributed PGD clusters and Barman-based recovery.

Today's Focus

Architecture & Oracle compatibility, intelligent observability, global scale, and resilience.

SPEAKER INTRODUCTION

Anil Kumar

Head of Product Management, Database Portfolio

CONNECT ON LINKEDIN

Scan to view profile

Leading the Database Portfolio

Owns product across EPAS / PGE, Postgres Distributed (PGD), the ops stack, and the new in-database AI layer (KernelIQ, SemanticIQ).

Two Decades of Systems Experience

Built mission-critical platforms across Postgres, NoSQL, and cloud-native infrastructure — at EDB, Couchbase, Microsoft, and Intel.

Work spans across:

Distributed databases, high availability, Kubernetes-native operators, and AI-ready data platforms.

SESSION PLAN

AGENDA

Four chapters, six new deep-dive topics woven in.

01 ARCHITECTURE & ORACLE COMPATIBILITY

What's new in EPAS · compression & partitioning · performance tuning · cross-platform deployment

02 PERFORMANCE & INTELLIGENT OBSERVABILITY

EDB Hybrid Manager · single pane of glass · AI-driven tuning recommendations · anomaly & plan-drift detection

03 UNLOCKING GLOBAL SCALE — PGD

Geo-distributed active-active clusters · HA & conflict resolution · horizontal sharding · S3 tiered storage · zero-downtime major upgrades & standby rebuilds

04 Demo

HM Observability

NEW THIS SESSION **EPAS 18 · compression & partitioning · PGD 6 sharding · S3 integration · zero-downtime upgrades · Barman table restore**

01

ARCHITECTURE & ORACLE COMPATIBILITY

How EPAS extends PostgreSQL into an enterprise, Oracle-compatible platform.

IN THIS SECTION

EPAS architecture
What's new in EPAS
Compression & partitioning
Performance tuning toolkit
Cross-platform deployment

ENTERPRISE POSTGRES — EDB POSTGRES ADVANCED SERVER

Six capability layers on top of open-source PostgreSQL.

EDB'S ORACLE COMPATIBILITY ADVANTAGE

Redwood mode: Oracle behavior built into the engine — no code changes.

BUILT-IN ORACLE FEATURES

- **PL/SQL execution** — procedures, functions, and packages run as-is.
- **Oracle packages** — DBMS_OUTPUT, UTL_FILE, DBMS_JOB, DBMS_SCHEDULER supported.
- **Data types** — VARCHAR2, NUMBER, DATE with time preserved.
- **SQL syntax** — CONNECT BY, ROWNUM, and DUAL work natively.
- **Transparent Data Encryption** — encrypt data at rest in the engine.

CRITICAL MIGRATION ENABLERS

- **Hierarchical queries** — CONNECT BY PRIOR works natively.
- **edb_redwood_strings** — handles Oracle NULL / empty-string behavior.
- **Oracle-style triggers** — NEW / OLD syntax compatibility.
- **Global temporary tables** — session-based temp table support.
- **Redwood or Berkeley mode** — choose Oracle- or Postgres-native behavior per database.

BOTTOM LINE **Preserve Oracle investments (70–80% code compatibility) and gain Postgres freedom — without a rewrite.**

POSTGRES EXTENSIONS — MODERN FUNCTIONALITY

The ecosystem that ships certified with EPAS 18.

EXTENSION	WHAT IT ADDS
Apache AGE	Graph database inside Postgres, fully ACID; SQL and Cypher side-by-side
pgvector / VectorChord	Vector similarity search for AI; VectorChord adds more dimensions & faster indexing
EDB Analytics Accelerator	Query Postgres + lakehouse data together; transparent tiered storage; Spark-scale
pg_search	Full-text search, loosely based on Apache Lucene
PostgreSQL Anonymizer	Declarative static & dynamic data redaction, generalization
EDB Wait States · Query Advisor · Stat Monitor	The observability trio feeding Hybrid Manager's intelligent recommendations

CERTIFIED **All EDB components and extensions are verified and certified against EPAS 18.**

WHAT'S NEW IN EPAS 18 — THE POSTGRESQL 18 CORE

NEW

EPAS 18 (GA Nov 2025) inherits the full PostgreSQL 18 feature set.

PERFORMANCE

- **Asynchronous I/O subsystem** — io_method (worker / io_uring) speeds seq scans, bitmap scans & vacuum — up to ~3x on cold reads.
- **B-tree skip scan** — multicolumn indexes now serve queries that skip leading columns.
- **Faster joins & GROUP BY** — hash-join memory + self-join elimination, OR-to-array transforms.
- **Parallel GIN builds & smarter vacuum** — eager freezing reduces aggressive-vacuum spikes.

DEVELOPER & SQL

- **uuidv7()** — time-ordered UUIDs in core — index-friendly primary keys.
- **Virtual generated columns** — computed on read; now the default for generated columns.
- **Temporal constraints** — PK/UNIQUE WITHOUT OVERLAPS, FK ... PERIOD.
- **RETURNING OLD/NEW** — in INSERT / UPDATE / DELETE / MERGE.
- **EXPLAIN upgrades** — BUFFERS on by default; index lookup counts; CPU/WAL in VERBOSE.

OPERATIONS & SECURITY

- **pg_upgrade keeps planner statistics** — no post-upgrade ANALYZE cliff; plus --jobs parallel checks and --swap.
- **OAuth 2.0 authentication** — modern identity integration; MD5 passwords deprecated.
- **Data checksums on by default** — for newly initialized clusters.
- **pg_createsubscriber --all** — logical replicas of every database in one command.

WHY IT MATTERS **Faster I/O-bound workloads, index-friendly keys, and the smoothest major upgrade Postgres has ever shipped.**

WHAT'S NEW IN EPAS 18 — ADVANCED SERVER ADDITIONS

EDB engineering on top of the PG18 merge (18.1 → 18.4 train).

SECURITY & COMPLIANCE

- **Automated inactive account locking** — password profiles gain INACTIVE_ACCOUNT_TIME — dormant accounts lock automatically.
- **Password redaction in server logs** — edb_log_redact_password_statements reliably masks ALTER USER-style commands.
- **postgres_fdw password obfuscation** — no more plaintext secrets in PG_USER_MAPPING.
- **Deeper edb_audit** — audit nested statements inside functions/procedures, COPY commands, and system catalog access.

ORACLE COMPATIBILITY & DEV

- **NLSSORT() support** — locale-aware, collation-correct sorting for migrated code.
- **UTL_SMTP with AUTH + STARTTLS** — secure, authenticated email from the database.
- **DBMS_SCHEDULER calendar upgrades** — FREQ=SECONDLY, INTERVAL keyword, negative BYMONTHDAY.
- **ALTER [PUBLIC] DATABASE LINK** — modify links in place — no drop/recreate of dependencies.
- **ECPGPlus enhancements** — indicator arrays for Oracle & ANSI Dynamic SQL Method 4.

RELEASE TRAIN 18.1 (Nov '25) features · 18.2/18.3 (Feb '26) upstream merges & fixes · 18.4 (May '26) CVE hardening.

COMPRESSION & PARTITIONING ENHANCEMENTS

Shrink storage and tame very large tables — across EPAS, PGD, and backups.

COMPRESSION

- **LZ4 / ZSTD TOAST compression** — default_toast_compression = lz4 — faster than pglz; zstd for WAL via wal_compression.
- **Compressed backups end-to-end** — gzip/lz4/zstd in pg_basebackup & Barman (backup_compression) — smaller, faster restores.
- **Block-level incremental backups** — PG17+/Barman: only changed blocks; pg_combinebackup builds synthetic fulls.
- **Columnar cold tier** — Iceberg/Delta Parquet on object storage delivers order-of-magnitude compression for history.

PARTITIONING

- **Oracle-style partitioning in EPAS** — interval and automatic list partitioning — partitions appear as data arrives.
- **EPAS interval partitions on PGD** — 6.1: AutoPartition-created partitions are consistent and visible cluster-wide.
- **PGD AutoPartition retention control** — drop_after_retention_period=false detaches instead of dropping — safe archives.
- **Tiered partitions** — aged partitions auto-offload to Iceberg/Delta, staying queryable in place.

DESIGN PATTERN **Partition by time → compress the hot tier → tier cold partitions to object storage. Storage follows data value.**

PERFORMANCE TUNING — THE EPAS TOOLKIT

From workload capture to guided fixes, inside the engine.

TOOL	WHAT IT DOES	USE IT FOR
Optimizer hints	Oracle-style hints steer plans when the planner needs guidance	SELECT /*+ INDEX(t idx) */ …
SQL Profiler	capture & compare workload traces; find the top offenders	on-demand or scheduled traces
EDB Query Advisor	recommends indexes & multicolumn statistics from the live workload	what-if virtual index costing
EDB Wait States	historical wait-event sampling per session / per query	root-cause lock & I/O stalls
EDB Stat Monitor	aggregated query statistics with plan capture	detects execution-plan drift
Resource Manager	throttle CPU and I/O at session level; dynamic resource tuning	protect SLAs from noisy jobs

PG18 BONUS **EXPLAIN ANALYZE now shows buffers by default and index-lookup counts — richer evidence with zero setup.**

POSTGRESQL vs EPAS vs AMAZON AURORA — VALUE VIEW

Where EDB Postgres Advanced Server earns its keep.

	POSTGRESQL	EDB POSTGRES ADVANCED SERVER	AURORA POSTGRESQL
Deployment flexibility	Any cloud, bare metal, K8s	Most flexible — public/private cloud, bare metal, K8s	Platform lock-in; no self-managed
TCO	$ — OSS + EDB support	$$ — up to 5x lower TCO than Aurora	$$$ — premium for enterprise features
High availability	Standard HA — 99.99%	99.999% with PGD,	99.99%; DSQL trades PG compatibility
Security	Needs add-on tooling	Built-in: TDE, redaction, SQLi protection, password policies	Needs add-on tooling
Modernization	MTK with limited compatibility	Fastest Oracle on-ramp: Redwood mode + Migration Portal AI	DMS/SCT; no Oracle-compatibility mode
Postgres support	Community	True Postgres IQ — 24/7 break-fix on the database itself	Service-level support only

POSTGRESQL vs EPAS vs AURORA — PLATFORM VIEW

Architecture and operations under the hood.

	POSTGRESQL	EDB POSTGRES ADVANCED SERVER	AURORA POSTGRESQL
Architecture	Coupled compute & storage	Coupled or distributed via PGD multi-master mesh	Shared-storage, decoupled compute
Replication & HA	Physical + logical (sync/async); OSS failover	Full bidirectional replication (PGD); EFM & Patroni supported	Shared storage + physical sync; built-in failover
Maintenance	Scripted OSS tools	OSS tools + TPA Ansible automation + Hybrid Manager	Managed patching, backups, blue-green
Extensibility	OSS extensions	OSS + EDB extensions (Query Advisor, Wait States, AIDB…)	Limited approved extension list
Version currency	Community cadence	EPAS 18 GA Nov 2025 — day-one PG18 parity	Trails community releases

CROSS-PLATFORM DEPLOYMENT — RUN EPAS ANYWHERE

One engine, one experience — sovereignty in your environment of choice.

HYBRID SOFTWARE

A single, sovereign software installation delivers a consistent experience and cloud agility across hybrid and multi-cloud environments.

BEST FOR
Bare metal · VMs · Kubernetes (CNPG) · any cloud

SOVEREIGN DATA & AI FACTORY

Pre-configured EDB Postgres AI + Supermicro systems bring production-ready Postgres with 99.999% HA into your data center.

BEST FOR
Engineered system · air-gapped ready

MANAGED PLATFORM

Unmatched Postgres expertise and 24×7 support, delivering mission-critical performance and availability for your hybrid environment.

BEST FOR
EDB-operated · your VPC or ours

ONE ENGINE **Same EPAS binaries, extensions, and tooling everywhere — workloads move without re-platforming.**

02

PERFORMANCE & INTELLIGENT OBSERVABILITY

One pane of glass — and an AI copilot — for every Postgres you run.

IN THIS SECTION

Hybrid estate challenges
EDB Hybrid Manager
Monitoring pipelines
AI-driven tuning recommendations
Anomaly & plan-drift detection

KEY CHALLENGES OF HYBRID DATABASE MANAGEMENT

Why fragmented estates drain money, time, and momentum.

01
DEPLOYMENT FLEXIBILITY

- Public-cloud DBaaS locks you into one provider's ecosystem.
- Legacy infrastructure can't keep pace with modern OSS stacks.
- Consumption pricing escalates quickly at scale.

02
COMPLEX MANAGEMENT

- Data lives across clouds, on-prem, and the edge.
- Sprawl multiplies tooling, skills, and operational overhead.
- Updates, backups, security, and tuning differ per platform.

03
AGILITY

- Problematic queries stifle performance; visibility is siloed.
- Root-cause analysis is slow across fragmented systems.
- Cloud is costly; on-prem alone lacks elasticity.

THE COST **Operational nightmares limit innovation — teams firefight instead of shipping.**

EDB POSTGRES AI — PLATFORM ARCHITECTURE

Hybrid Manager - A sovereign platform from clients to deployment, optimized for AI workloads.

CLIENTS	Agents · Applications · Users
WORKLOADS	Transactional (relational · document · vector · graph · time series) · Analytical (warehouse · lakehouse · real-time · batch) · Agentic (runtime · RAG · semantic search · ML)
PLATFORM SERVICES	Governance · Security · Data integration · Migrations · Observability · Lifecycle · HA/DR
DATA LAYER	PostgreSQL databases · Lakehouse / object storage · Streams & event data
DEPLOYMENT	Private data center · Engineered system · Any Kubernetes · Any cloud

HYBRID MANAGER is the control plane for the whole stack — the platform-services layer is where this section lives.

EDB HYBRID MANAGER — FEATURE OVERVIEW

Hybrid DBaaS: deploy, automate, observe, and optimize from one console.

- **Unified management console** — deploy and manage hundreds of clusters across public and private cloud from a single interface.
- **Database automation** — backups, point-in-time recovery, provisioning, activity logs, user management, and alerting.
- **Enhanced observability** — monitor and respond to performance and security issues in real time with 200+ metrics.
- **Intelligent recommendations & query diagnostics** — surface missing indexes, misconfigurations, and bottlenecks in seconds — no deep expertise required.
- **Sovereign migrations, AI-ready data** — built-in migration experience with AI-driven recommendations, secure data sync, and full Oracle compatibility.
- **Runs on your Kubernetes** — OpenShift, RKE2, EKS, GKE — cloud, on-prem, or hybrid.

DO MORE WITH LESS **Consolidates workloads and services with automation and observability — cutting cost and admin time.**

HYBRID OBSERVABILITY — ONE PANE OF GLASS

Every Postgres you run, whoever runs it, in one consistent view.

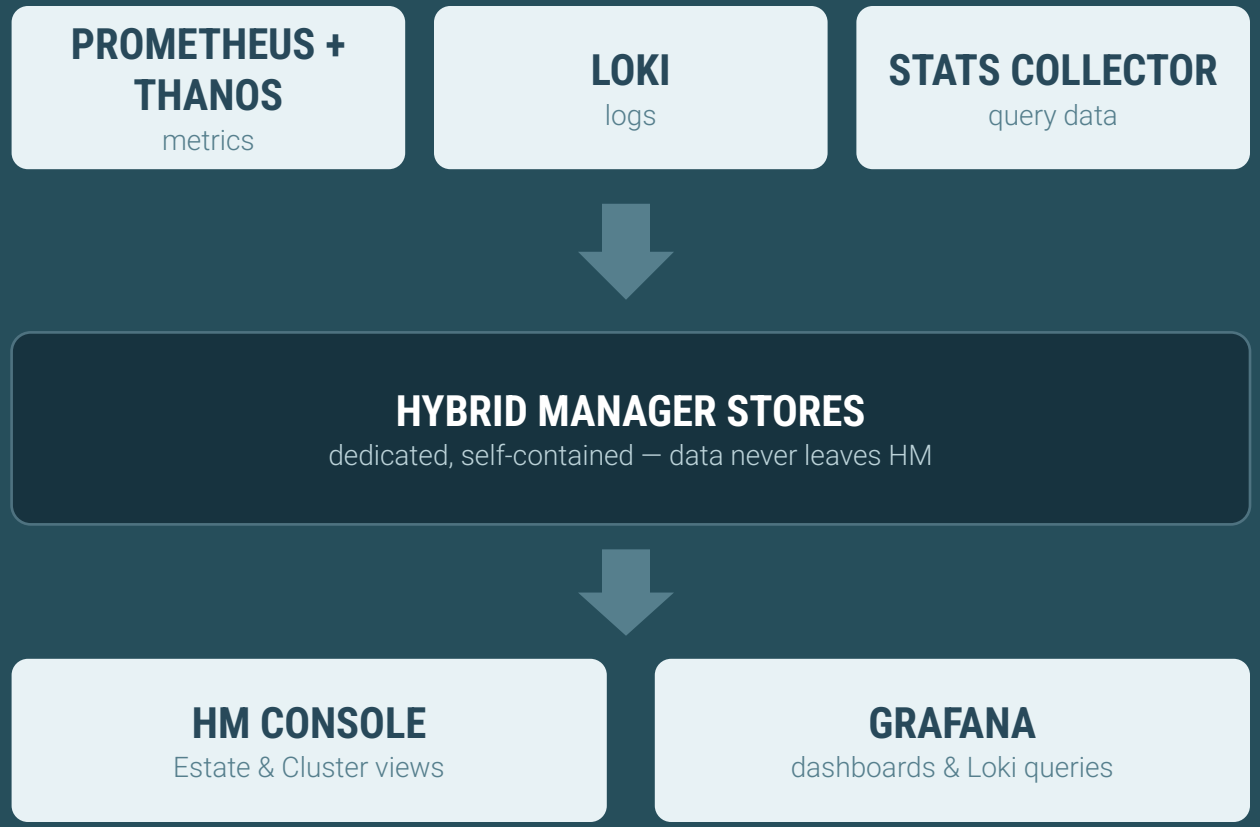
WHAT YOU GET

- **Metrics, alerts, logs** — host, Kubernetes, and 200+ Postgres metrics.
- **Query diagnostics** — top queries by wait, latency, and execution.
- **Intelligent recommendations** — indexes, stats, security, configuration, workload.

SUPPORTED POSTGRES ESTATES

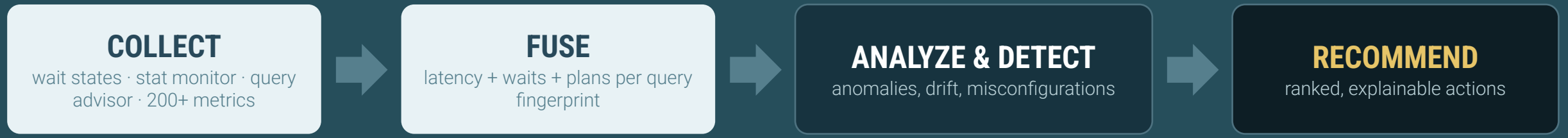
- **HM-managed clusters** — provisioned by Hybrid Manager on K8s.
- **Self-managed** — bare metal, VMs, K8s — community & EDB, CloudNativePG.
- **Cloud providers** — AWS RDS via API-based collection.

THREE INTERNAL PIPELINES



AI-DRIVEN TUNING & ANOMALY DETECTION

From raw telemetry to ranked fixes — in seconds, not war rooms.



INTELLIGENT RECOMMENDATIONS

- **Missing indexes & statistics** — what-if costed by EDB Query Advisor.
- **Configuration & workload advice** — memory, autovacuum, connection tuning.
- **Security posture findings** — risky settings surfaced before auditors find them.

ANOMALY & DRIFT DETECTION

- **Execution-plan drift** — EDB Stat Monitor compares plans over time and flags regressions.
- **Query Diagnostics** — wait-state history fused with latency/execution data pinpoints the moment things changed.
- **Alerting on 200+ metrics** — thresholds in the console; custom rules via Prometheus/Grafana.

NO EXPERTISE REQUIRED **Fix issues in seconds, even without manual analysis — expanding from HM-managed to self-managed estates.**

HYBRID OBSERVABILITY vs POSTGRES ENTERPRISE MANAGER

Where the new stack goes further — and what PEM still does uniquely.

	PEM (CLASSIC)	HYBRID MANAGER (HM)
Provisioning & upgrades	None	Full DBaaS provisioning & upgrades in the UI
Query-level insight	Top queries via Wait States; SQL Profiler tracing	Wait States fused with latency & execution from EDB Stat Monitor
Tuning recommendations	None	Indexes, statistics, security, configuration, workload
Dashboards	Predefined + custom in UI	One predefined; custom via Grafana integration
Alerts	Large default set; per-object thresholds; history	Focused default set; custom via Prometheus; shown in UI
Log analysis	Collection only	Collection + live streaming in UI; querying via Grafana/Loki
Scheduled jobs	SQL / shell jobs on monitored instances	Not supported — use platform schedulers

GUIDANCE **New estates start on Hybrid Manager; PEM remains supported where scheduled jobs or custom metrics rule.**

03

UNLOCKING GLOBAL SCALE WITH PGD 6.x

Geo-distributed, active-active Postgres — five 9s without the primary bottleneck.

IN THIS SECTION

Why distributed Postgres
PGD 6 platform & Connection Manager
Commit scopes & conflict resolution
Always-On architectures & geo-routing
Horizontal sharding
S3 tiered storage
Zero-downtime major upgrades

WHY DISTRIBUTED POSTGRES

Three problems streaming replication alone can't solve.

EXTREME HIGH AVAILABILITY

Minutes of downtime cost real money. Fault-tolerant clustering and redundant, multi-writer architecture eliminate single points of failure.

Outages · operational errors · hardware & software failures

ROLLING MAINTENANCE

Maintenance is the single largest source of downtime: 1 major release/year, 4 security releases/year, unscheduled patches, parameter & hardware changes.

No other Postgres solution eliminates upgrade downtime

GEOGRAPHIC DISTRIBUTION

Put data next to users and inside borders. Reduce latency and meet data-localization and cyber-security requirements.

Local writes · data residency · regional resilience

PGD **Highly resilient architecture keeps mission-critical applications safe from unplanned outages and planned work alike.**

COMPONENTS OF AN HA POSTGRES ARCHITECTURE

Two building blocks every design must answer for.

DATA REPLICATION

- **PostgreSQL native physical streaming** — byte-level standby copies.
- **PostgreSQL native logical** — row-level, selective, cross-version.
- **EDB Postgres Distributed (PGD)** — optimized bidirectional logical replication — up to 5x native throughput.
- **Key decisions** — synchronous vs asynchronous (consistency/durability vs latency); unidirectional vs bidirectional.

CLUSTER MANAGEMENT

- **Highly available access** — route apps to writable node(s) at all times.
- **Split-brain protection** — consensus (Raft) so only one truth exists.
- **Cluster awareness** — replication state & lag, node states, monitoring.
- **Cluster operations** — switchover, configuration, topology, maintenance.
- **Key decisions** — primary-standby vs multi-master; failover time (RTO).

HIGH-AVAILABILITY ARCHITECTURES FOR POSTGRES

Pick the deployment architecture, then the management layer follows.

	PRIMARY-STANDBY	MULTI-MASTER
Deployment architecture	Primary → Standby (one writer)	Multi-Master (every node writable)
Data replication	Unidirectional physical (sync or async)	Bidirectional logical (sync or async)
Cluster management	Patroni (OSS, EDB-supported) · EDB Failover Manager	EDB Postgres Distributed — Raft consensus built in
Failover behavior	Promotion event; apps reconnect after RTO	Route-around; no promotion — RTO under 5 seconds
Best when	Simplicity first; single region; modest RTO	Five-9s uptime, rolling maintenance, geo-distribution

CONSISTENCY DIAL **Async prioritizes performance & low latency; synchronous prioritizes consistency & durability — both are available in either model.**

THE PGD 6 PLATFORM — WHAT'S NEW IN THE 6.x TRAIN

PGD 6.1 is the session baseline; the train keeps accelerating.

PGD 6.0 — THE FOUNDATION

- **Built-in Connection Manager** — replaces PGD-Proxy: automatic routing, read/write + read-only ports, session pooling — no external software.
- **Predefined commit scopes** — durability profiles ready out of the box.
- **pgd node setup CLI** — create clusters and add nodes from one command.

PGD 6.1 — THIS SESSION

- **Direct upgrades from PGD 4.4+ / 5.9+** — first seamless in-place path to 6.
- **Replication to Apache Iceberg** — stream to the lakehouse; query it back from PGD.
- **Materialized views · leader DDL lock · EPAS interval partitions**

AND BEYOND (6.2 → 6.4)

- **6.2** — PostgreSQL 18 / EPAS 18 support; Connection Manager LDAP auth; one-command TDE adoption on upgrade.
- **6.3** — Connection Manager on subscriber-only nodes; Raft enable/disable via CLI.
- **6.4** — Quorum Commit — majority pre-commit agreement for a consistent global view; transaction-mode pooling; large-object replication.

SUPPORTED ENGINES

PostgreSQL · EDB Postgres Extended · EDB Postgres Advanced Server, versions 13–18

PGD AUTOMATIC REPLICATION — DML, DDL, SEQUENCES

The mesh keeps every node consistent without operator choreography.

DATA (DML)

- **Efficient** — logical replication, up to 5x native throughput.
- **Fast** — transaction streaming + Parallel Apply smooth large transactions.
- **Configurable** — async or sync per transaction via commit scopes.

SCHEMA (DDL)

- **Replicated automatically** — with fine-grained DDL locking.
- **Leader DDL lock (6.1)** — locks write-leaders only — no majority round-trip by default.
- **Table rewrites in transactions** — multiple ALTERs apply as one atomic unit.

SEQUENCES

- **Automatic handling** — sequences transform to distributed kinds on group join.
- **Multiple handlers** — Snowflake, Galloc, and offset strategies.
- **No manual ranges** — start values self-adjust past local usage.

ROLLING SYSTEM UPGRADES **Inter-node protocols renegotiate per link and stay backward-compatible — OS, Postgres, and PGD upgrade in place, node by node.**

DISTRIBUTED SEQUENCES — UNIQUE KEYS ACROSS EVERY NODE

Pick the generator that matches your key width and ordering needs.

KIND	HOW IT WORKS	PROPERTIES	BEST FOR
Snowflake	64-bit (19 digits): timestamp + node ID + local counter	Sortable by time; immune to replication lag; JS needs BIGINT care	bigint / bigserial keys
Galloc	Globally allocated ranges; each node holds current + future chunk	Continuous unique values, no manual range management	int2 / int4 / serial — 32-bit apps
UUID / KSUUID	Generated independently, no coordination; KSUUID embeds timestamp	128-bit storage & index overhead; KSUUID sorts by creation	natural keys, external IDs
Offset sequences	Classic Postgres sequences with per-node offsets	Self-managed, error-prone; not serializable cluster-wide	legacy compatibility only

PG18 TIP **uuidv7() in core gives time-ordered UUIDs — pair with PGD galloc OIDs for conflict-free large objects (6.4).**

CONSISTENCY & DURABILITY — COMMIT SCOPES + CONFLICTS

Dial consistency per cluster, per group, or per transaction.

COMMIT SCOPES

- **Synchronous Commit** — PostgreSQL-style sync replication via scope constructs.
- **Group Commit** — two-phase commit with control over which and how many nodes must agree.
- **CAMO** — Commit At Most Once — the client joins the consensus; no double-commits.
- **Lag Control** — throttle commits when replicas fall too far behind.
- **Quorum Commit (6.4)** — majority pre-commit agreement or rollback everywhere — no divergence.

CONFLICT MANAGEMENT

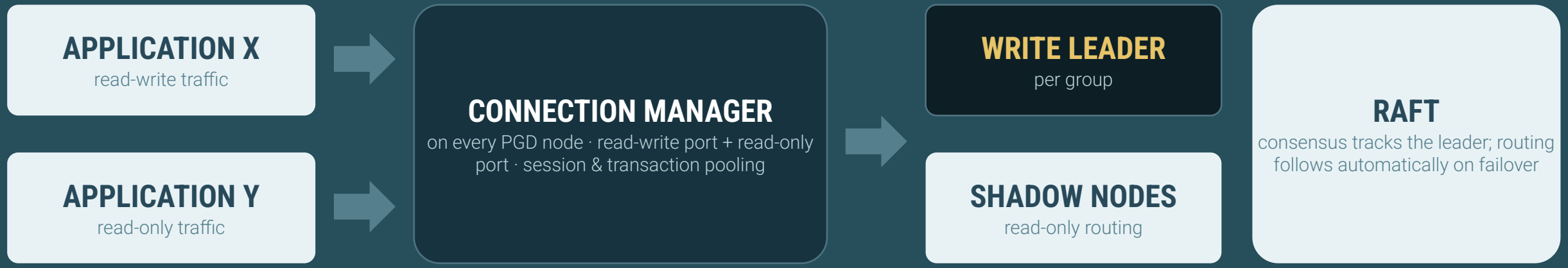
- **Conflicts are not errors** — automatic row-level detection with logged resolution.
- **Detectors + resolvers** — cover every conflict scenario; resolvers are customizable.
- **Column-level resolution & CRDTs** — conflict-free replicated data types for concurrent updates.
- **Largely avoided by design** — Always-On architectures keep writes local to a leader per location.

PER TRANSACTION **Mix scopes in one cluster: payments run Quorum Commit while telemetry stays async — same tables, different guarantees.**

CONNECTION MANAGER — ROUTING WITHOUT PROXIES

PGD 6 builds routing, pooling, and read/write split into every node.

NEW



- **No external software** — replaces PGD-Proxy, PgBouncer, and HAProxy layers for most topologies; survives leader changes without dropping clients (6.2).
- **Read/write split by port** — apps can't accidentally write through the read-only port — pair with default_transaction_read_only=on client-side.
- **Transaction pooling (6.4)** — backend held only per transaction — relieves max_connections under high concurrency; LDAP auth (6.2); runs on subscriber-only nodes (6.3).

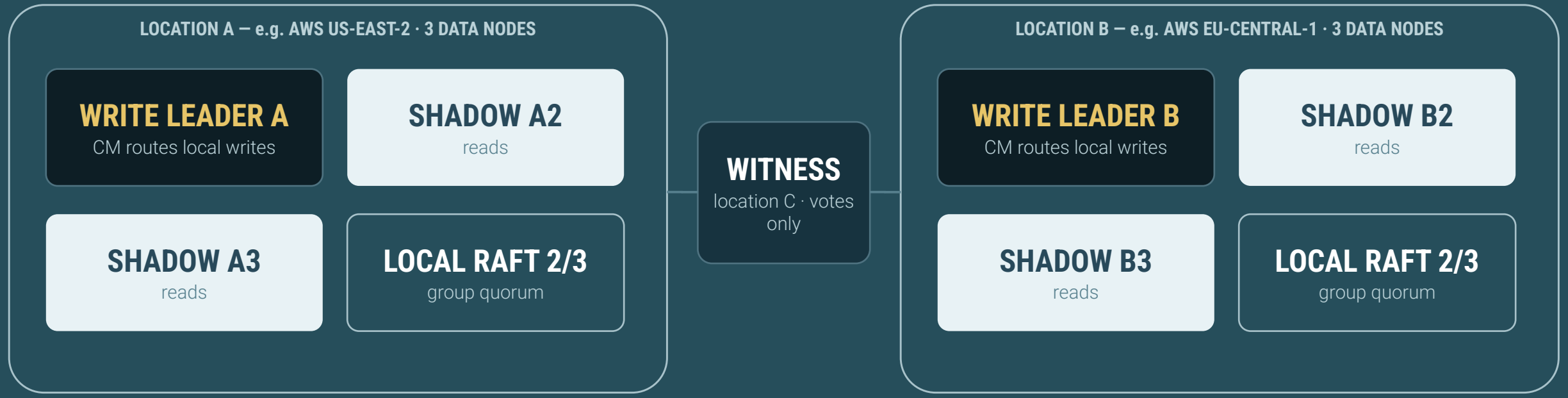
ALWAYS-ON — SINGLE LOCATION (ONE REGION)

Three data nodes across availability zones; apps connect anywhere.

- **Apps connect to any Connection Manager** — routing lands writes on the current write leader; shadows absorb reads.
- **Redundant hardware per zone** — resilience during local failures; Raft keeps one leader.
- **Barman in an alternate location** — provides the DR copy (not shown).

ALWAYS-ON — MULTIPLE LOCATIONS (CROSS-REGION)

Six data nodes in two regions plus a witness for global consensus.



- **Active/Active, Active/Passive, or Active/DR** — write leaders per location, or a single global leader — your choice.
- **Global Raft 4/7 across 7 voting nodes** — no location can split-brain the cluster; witness holds the tiebreak.
- **Replication lag budget** — ~90–105 ms cross-Atlantic is typical; commit scopes decide what waits for whom.

GEO-DISTRIBUTED CLUSTERS — PLANNING & ROUTING

The five decisions that shape a healthy multi-region PGD deployment.

01

CHOOSE A PATTERN

Active write locations, routing mode, and commit scope come first — everything else follows.

02

PLAN SUBGROUPS & QUORUM

Subgroup per location; no single site holds a Raft majority; add a witness location if needed.

03

CONFIGURE ROUTING

Local routing (write leader per location, default) or global routing (one cluster-wide leader); set route priorities.

04

SET COMMIT SCOPES

Match durability to the workload per location subgroup.

05

TUNE & VERIFY

Parallel apply + streaming for lag; distributed sequences for unique keys; measure inter-region latency.

- **Why geo-replicate** — location-failure protection · data residency · low-latency local reads · near-zero RTO disaster recovery.
- **Data residency by design** — pair subgroups with selective replication so regulated data never leaves its region — the bridge to sharding, next.

PGD 6 & HORIZONTAL SHARDING

Scale writes by placement: each shard is its own HA multi-master group.

ONE GLOBAL SCHEMA + REFERENCE TABLES
default replication set → replicated to every node

SHARD: EU SUBGROUP
EU tenants · repset 'eu' · 3 nodes, own write leader

SHARD: US SUBGROUP
US tenants · repset 'us' · 3 nodes, own write leader

SHARD: APAC SUBGROUP
APAC tenants · repset 'apac' · 3 nodes, own write leader

- **Selective replication with replication sets** — shard tables subscribe only within their subgroup; reference data replicates everywhere. Don't repurpose the default repset.
- **Each shard is highly available on its own** — a full PGD group with local Raft, Connection Manager routing, and rolling maintenance — sharding and HA in one layer.
- **Globally unique keys for free** — Snowflake / Galloc sequences prevent cross-shard collisions; cross-shard reads via Postgres FDW or the analytics tier.
- **Geo-sharding = compliance** — pin regulated data inside borders (geo-fencing) while keeping a unified global schema — the pattern behind sovereign, planet-scale estates.

VS HASH SHARDING **PGD shards by placement (tenant/region), not by hash — apps route naturally, and shards fail over independently.**

S3 STORAGE INTEGRATION WITH PGD

Tiered storage: hot rows on NVMe, history in Iceberg/Delta on object storage.

HOT TIER — PGD
local heap · sub-ms latency · full ACID

PGAA

COLD TIER — OBJECT STORAGE
Iceberg / Delta (Parquet) on S3 · GCS · Azure

QUERY EITHER — OR BOTH
one Postgres interface · vectorized engine · no ETL

THREE STRATEGIES

STRATEGY	HOW IT WORKS	USE FOR
Tiered tables	AutoPartition ages partitions past a threshold → bulk-copied to object storage as PGAA tables, local disk reclaimed, still queryable	high-volume time-series & audit history
Replicate to analytics	heap becomes HTAP: near-real-time analytical copy maintained via Iceberg Merge-on-Read — minimal write overhead	live BI on transactional data
Offload	one-time archive: truncate local heap, table access method flips to PGAA; restore back when writes return	surgical space reclamation

ALSO ON S3 **Barman cloud backups + WAL archive land on S3/Blob/GCS — one bucket strategy for analytics and recovery.**

MAJOR VERSION UPGRADES WITHOUT DOWNTIME

PGD rolling upgrades turn the biggest outage source into a non-event.

01
SHIFT TRAFFIC
Connection Manager moves the write leader off the target node; apps never notice.

02
UPGRADE IN PLACE
pgd node upgrade wraps pg_upgrade — Postgres (and PGD) jump majors on the same hardware.

03
REJOIN & CATCH UP
node returns in a mixed-version cluster; replication protocols stay backward-compatible.

04
REPEAT PER NODE
rotate through every node — and shadow-verify with LiveCompare as you go.

- **What this eliminates** — downtime for 1 major/year, 4 security releases/year, unscheduled patches, parameter & hardware changes.
- **PG18 makes each hop cheaper** — pg_upgrade preserves planner statistics — no post-upgrade ANALYZE cliff; --jobs runs checks in parallel.
- **Also covered** — PGD 6.1 upgrades directly from PGD 4.4+ / 5.9+, and pgd node upgrade can convert nodes to TDE in the same pass (6.2).

THE ROAD TO SCALE-OUT

Horizontal scale-out for PGD — write scalability by adding shard groups, with the HA guarantees you already run

PGD HORIZONTAL SCALING

TRANSPARENT HORIZONTAL SCALE — IN PHASES

Sharding that distributes data across PGD groups — linear write scale with per-shard HA

PHASE 1 · FOUNDATION

PGD 19 · FEB 2027

Hash sharding on any column — customer_id, token, tenant. Apps route optimally with a shard-key hint; cross-shard reads and writes handled by FDW + WAL replication, with read-your-writes consistency.

PHASE 2 · ELASTICITY

PGD 19 · FEB 2027

Online shard rebalancing: add a shard group, double-write while data migrates in the background, switch routing, clean up — zero downtime throughout. Scale when the business grows.

PHASE 3 · AUTOMATION

PGD 20 · NOV 2027

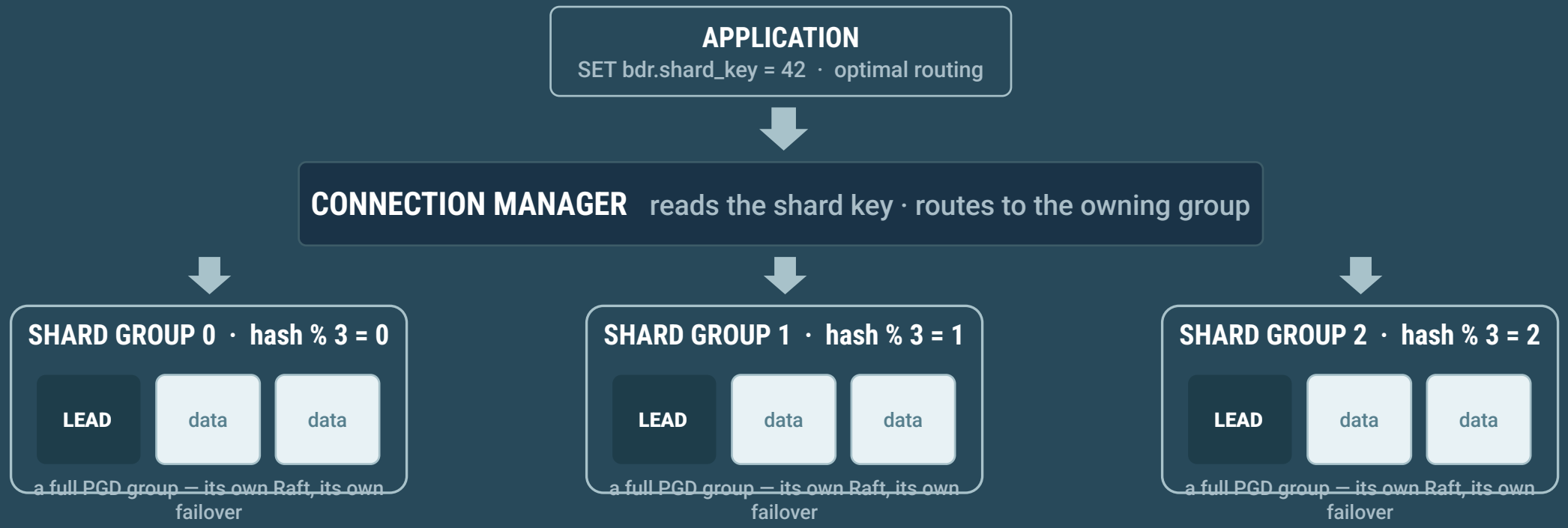
The SQL-aware router parses statements and bind values, routes every query to the owning shard, scatter-gathers when needed. Optimal performance becomes automatic — zero app changes.

WHY OURS IS DIFFERENT **every shard is a PGD group — write scale inherits the AAA failover, routing, and durability you already operate.**

ROADMAP — DIRECTIONAL **Subject to change. Not a commitment to deliver features or dates.**

SHARDING, ON THE PLATFORM YOU RUN

Hash-based shard groups behind Connection Manager — hot rows on the shards, history in open formats



cross-shard reads via FDW · remote writes via WAL replication + MoR buffer — read-your-writes preserved

AND ANALYTICS COMES FREE **sharded tables tier transparently to Iceberg / object storage — one virtual petabyte table, no ETL, no second platform.**

ROADMAP — DIRECTIONAL **Subject to change. Not a commitment to deliver features or dates.**

SCALE-OUT, WITHOUT THE TRADE-OFFS

What distributed-SQL and NoSQL alternatives give up — and PGD doesn't

	EDB PGD	Citus	NoSQL	CRDB / YB	Cloud-only DSQL
True PostgreSQL engine	✓	✓	✗	✗ / ~	~
Per-shard HA, Raft consensus	✓	~	~	✓	✓
No coordinator bottleneck / SPOF	✓	✗	✓	✓	✓
ACID + joins across shards	✓	~	✗	✓	✓
On-prem · hybrid · air-gapped	✓	~	✓	~	✗
Analytics on sharded data, no ETL	✓	~	✗	✗	✗

✓ full · ~ partial / limited · ✗ not supported | cloud-only engines: lock-in, egress, and no air-gapped path for regulated estates

THE POSITION **Sharding + enterprise HA + in-place analytics — on real Postgres, on-prem and cloud. Nobody else holds all three.**

ONE PRODUCT · ONE VERSION · ONE UPGRADE

PGD BECOMES A DISTRIBUTION

You gave us the feedback. PGD 6 simplified it. Now it ships as one product — Postgres included.

PGD 6 · 2025 – 2026 → **PGD 6.5 · Q4 2026** → **PGD 19 · FEB 2027 · ON PG 19.2**

Simplified. Connection Manager built in, CLI and commit scopes unified, the quality bar raised to the Postgres standard — the operability gaps you flagged, closed.

The PGD UI ships in the box — visual cluster management, no external dependency. Final 6.x feature release.

The full distribution: Postgres + PGD + monitoring + backup + UI — one package, one version, one upgrade. Plus Hierarchical Mesh topology and next-generation logical replication.

PGD 19 = PG 19 · PGD 20 = PG 20 · PGD 21 = PG 21

The version number becomes the compatibility guarantee. One annual major aligned to Postgres; quarterly maintenance (Feb · May · Aug · Nov) carries fixes only — never surprise features.

NO FORCED MARCH

PGD 6 runs under its own lifecycle through November 2030. PGD 19 is the next step — on your certification timeline, not ours.

BUILT FOR YOUR OPERATING MODEL **when a major upgrade is a funded, multi-year program, fewer versions and a published calendar are the feature.**

ROADMAP — DIRECTIONAL **Subject to change. Not a commitment to deliver features or dates.**

THE INTELLIGENT ENGINE

PRODUCT INITIATIVE · EDB POSTGRES AI

KERNELIQ — THE AUTONOMOUS DATABASE

INITIATIVE

A database that watches itself, predicts what will break, and acts before it does

SENSE → **THINK** → **ACT**

kernel telemetry — query heat-maps, index drift, I/O saturation

predict the problem before it reaches users

non-blocking: reorg, tune, patch, fail over — supervised

1 DBA → 1,000 servers. The ambition — up from ~20 today.

The DBA stops administering and starts supervising — guardrails on, human in control.

YOUR PRINCIPLE, OUR TARGET no snowflakes, no manual patching, everything automated — an engine that upholds that on its own.

IT ACTS THROUGH WHAT ALREADY SHIPS

Query tuning — pg_plan_advice — PG 19

Reorg & capacity — VACUUM FULL CONCURRENTLY

Security posture — TDE · auditing · KMIP

Integrity & failover — amcheck · EFM · PGD

Not another external tool watching Postgres from outside — intelligence built on primitives inside the engine, where the signal actually lives.

PRODUCT INITIATIVE — IN DEVELOPMENT **Directional vision. Not committed, no dates, not a release plan. Shared to shape it with you — not for you to plan against.**

PRODUCT INITIATIVE · EDB POSTGRES AI

SEMANTICIQ — THE SOVEREIGN SEMANTIC LAYER

INITIATIVE

The database understands what a question means — and answers with the query the business approved

REGISTER → **RESOLVE** → **GOVERN**

business meaning, per domain, in a governed catalog

the question, in the caller's context

approved joins only — unknown terms refused

50% → 90% text-to-SQL accuracy

Same model, same data — the lift is the catalog. Target demo benchmark, not a field number.

ONE WORD, THREE MEANINGS

"Exposure" is one number to a banking desk, another to insurance, a third to wealth. A schema-only AI agent guesses; the semantic catalog knows — per domain, with governed join paths.

Meaning lives inside the database perimeter — not in an external tool that copies your data out.

WHY IT MATTERS TO A NETWORK

Governed, auditable AI on your own data, inside your own perimeter — the answer an architecture board can approve, and a regulator can inspect.

PRODUCT INITIATIVE — IN DEVELOPMENT **Directional vision. Not committed, no dates, not a release plan. We are validating this with a small set of enterprises — join us.**

THE CLOSE

WHY EDB — WE DON'T JUST SHIP POSTGRES. WE BUILD IT.

The roadmap you just saw is not a bet on someone else's work — it is our own engineers landing it in core

CORE TEAM

EDB employs multiple PostgreSQL core-team members and major committers — including the engineers behind today's roadmap

TOP CONTRIBUTOR

Consistently among the largest corporate contributors to every PostgreSQL release — features, reviews, fixes

UPSTREAM = ROADMAP

OAuth (PG18) · Pluggable Access Methods (PG18) · VACUUM FULL CONCURRENTLY (PG19) · pg_plan_advice (PG19) — all EDB-led

ONE ENGINE, YOUR OUTCOMES

Freedom from Oracle at lower TCO · always-on at network scale — your AAA, with quorum-grade consistency · compliance and sovereignty by architecture · and an engine growing intelligence from the inside.

ONE ENGINE. ALWAYS ON. SCALES OUT. ON YOUR SCHEDULE. *Keep telling us what to build.*

04

Demo

HM Observability

IN THIS SECTION

Observability

Agentic AI to resolve issues

THANK YOU

Questions?

RAHUL SAHA
Principal Resident Architect

ANIL KUMAR
Head of Product Management, Database Portfolio