JPMorganChase

The Migration Blueprint to EDB Postgres

Gianni Ciolli
VP, Practice Solutions
January 2026

Speaker Introduction

Gianni Ciolli

VP, Practice Solutions

Gianni is Vice President at EDB, with 30 years of open-source expertise, contributing to PostgreSQL, BDR, repmgr and TPA.
Lead author of the PostgreSQL Administration Cookbook, he specializes in High Availability and Active-Active replication.

He holds a PhD in Mathematics and formerly led Professional Services at 2ndQuadrant.

Connect on Linkedin

Agenda

Today’s Migration Blueprint Journey

Setting the Stage (5 min.)

- Why EDBPostgres:TheMigrationAccelerator
- EDB Migration Framework: ACTIONS Overview

Deep Dive: Oracle to EDB and Migration Framework (20 min.)

- Oracle Migration Complexity
- Automated Conversion Strategies
- EDB Migration Factory Toolkit
- Critical Gotchas and Solutions
- EDB Migration Framework: ACTIONS
- Ensuring Success

Breaking Cloud Lock-in (8 min.)

- AWSRDS/AuroraEscape Patterns
- Cloud-Native to Portable Architecture

Q&A (10–12 mins.)

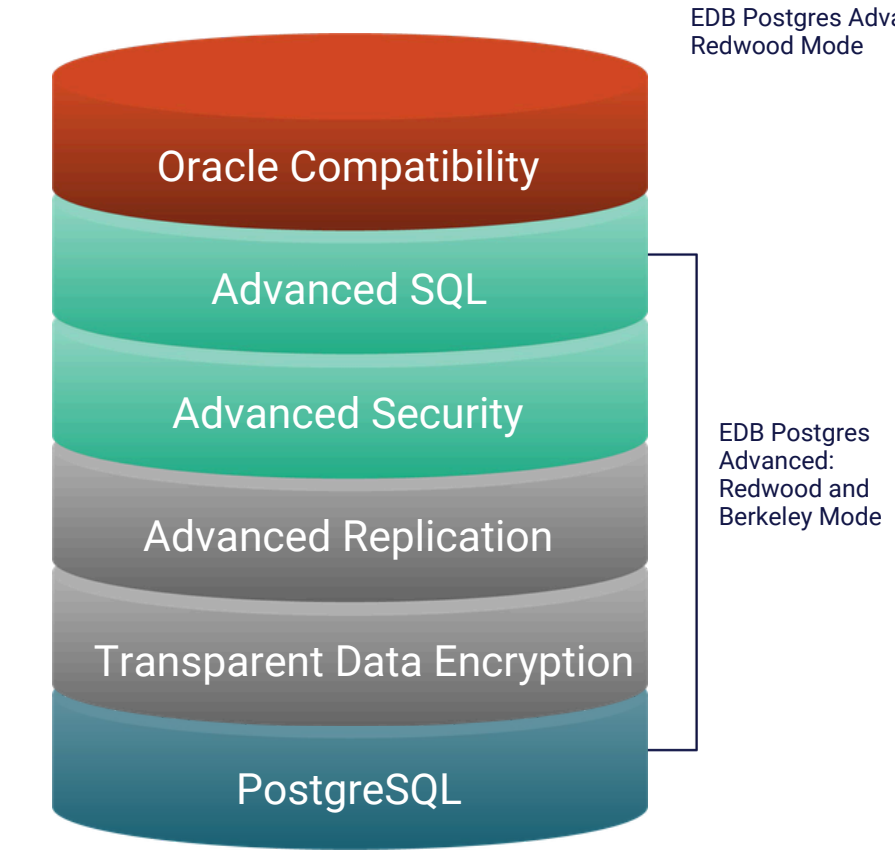
1. Setting the Stage

1.1. **Why EDB Postgres:** The Migration Accelerator

EDB's Oracle compatibility advantage

Built-in Oracle features (no code changes)

- PL/SQL execution: Procedures,functions,packages run as-is
- Oracle packages: DBMS_OUTPUT, UTL_FILE, DBMS_JOB suppor ted
- Data types: VARCHAR2, NUMBER, DATE with time preserved
- SQL syntax: CONNECT BY, ROWNUM, DUAL table work natively



Enterprise - EDB Postgres Advanced Server

1.2. **Why EDB Postgres:** The Migration Accelerator

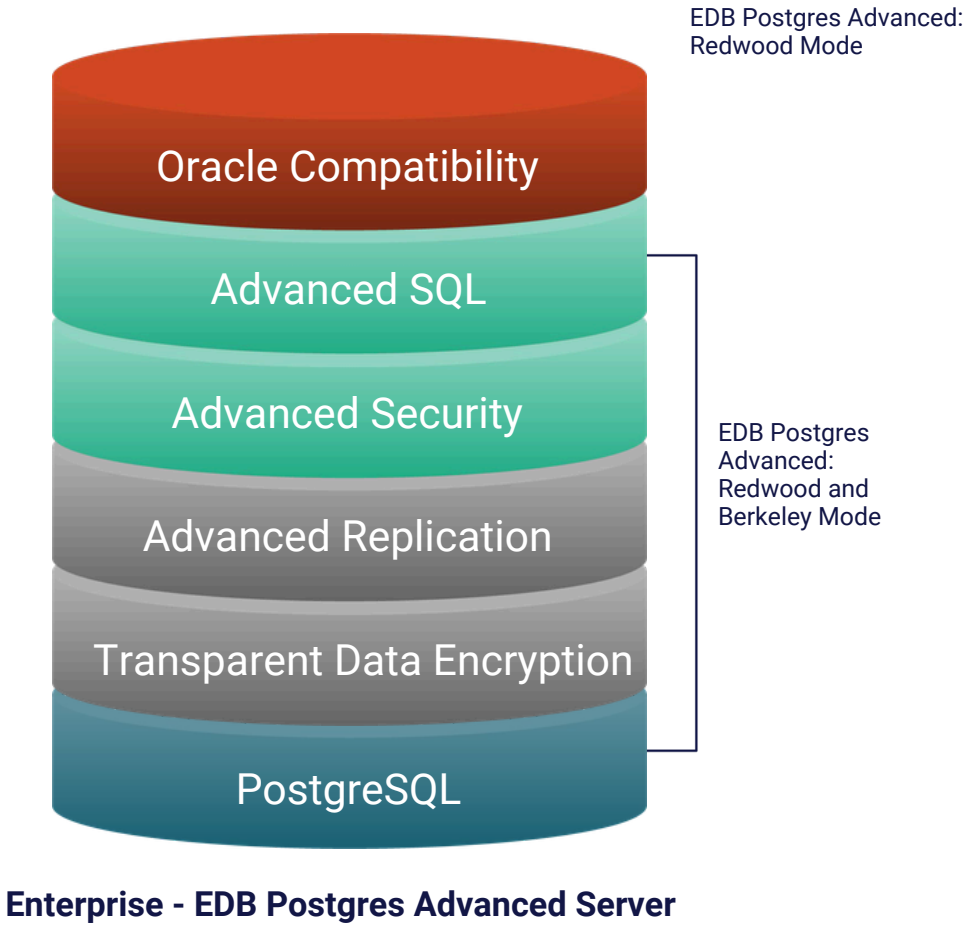
EDB's Oracle compatibility advantage

Critical migration enablers

- Hierarchicalqueries: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

EDB Migration Toolkit Suite

- Migration Portal: Cloud-based assessment (no installation)
- Migration Toolkit: Automated schema and data transfer
- SPL Check: PL/SQL code validation
- LiveCompare: Zero-downtime data validation



Enterprise - EDB Postgres Advanced Server

The bottom line: Preserve Oracle investments (70%–80% code compatibility). reduce migration risk with proven tools, Achieve PostgreSQL benefits without complete rewrite. Oracle compatibility + open source freedom = EDB Postgres

1.2. EDB Migration Framework: ACTIONS Overview

7 phases to database freedom

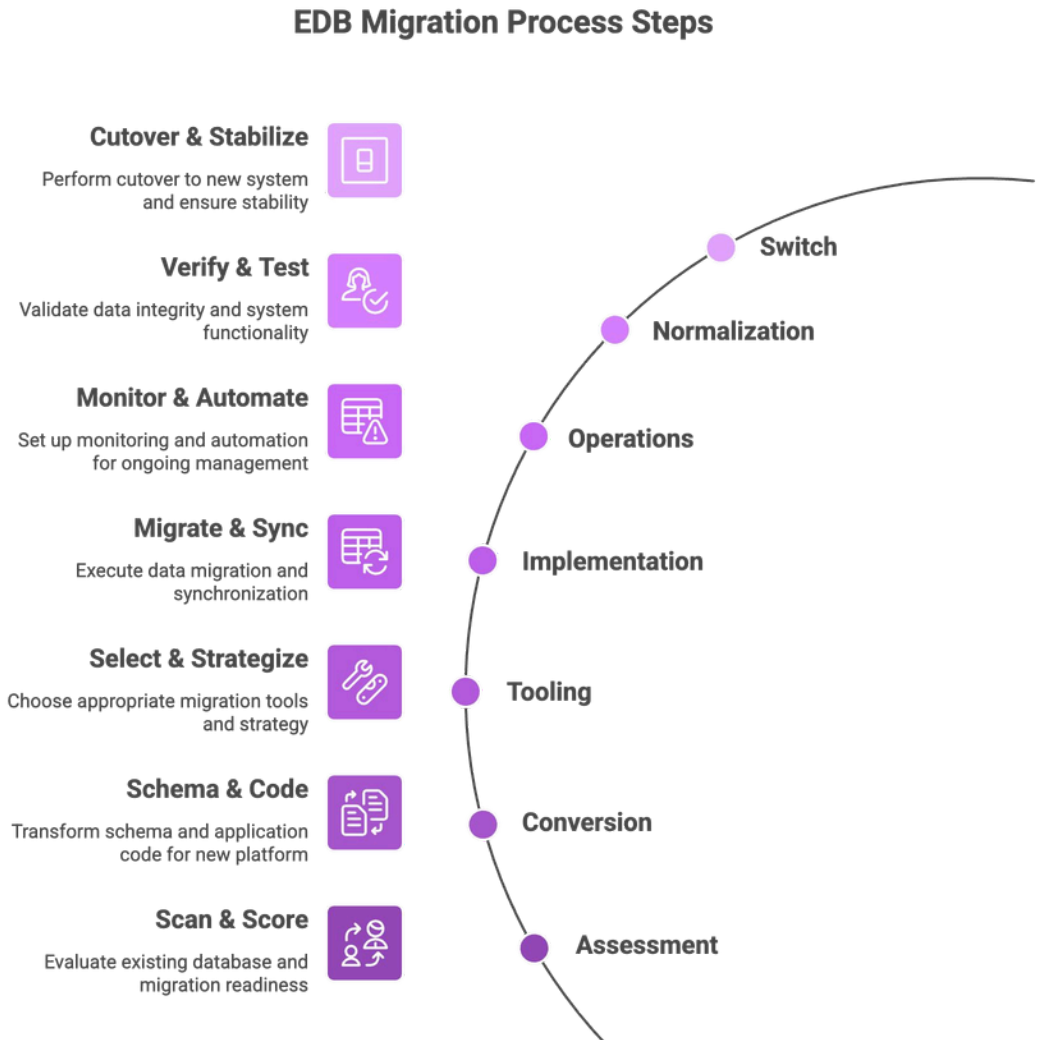
A - Assessment	Scan -> Score -> Size
C - Conversion	Schema -> Code -> Validation
T - Tooling	Right tools -> Right strategy
I - Implementation	Migrate -> Sync -> Optimize
O - Operation	Monitor -> Automate -> Manage
N - Normalization	Verify -> Test -> Certify
S - Switch	Cutover -> Stabilize-> Succeed

Why ACTIONS framework delivers

Traditional Migration	EDB ACTIONS Framework
6–12 months	2–4 months
40% failure rate	<1% rollback rate
Trial and error	Predictable phases
Multiple vendors	Integrated toolkit

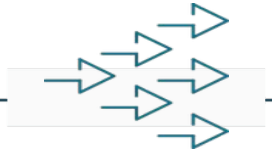
Key differentiators

Automatedassessmentidentifies issues before they become problems
Built-in validation at every phase reduces risk
Proven playbooks for Oracle and Cloud PostgreSQL scenarios



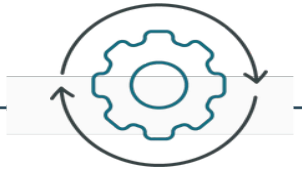
JPMC’s Three Target Use Cases

SECURELY BUILD, TEST, AND LAUNCH SOVEREIGN AI APPLICATIONS



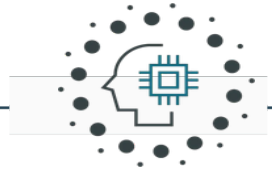
Modernization Use Case 1
New App OSS to EDB PG

1. Scenarios
   a. Tier 1-2 Triple A, Double A Same DC
   b. Tier 3-4 Active Passive DR
2. Targets
   a. Reference Architecture
   b. App Migration Process
   c. Differentiation & Benchmarks
   d. Decisioning App Modernization
3. Demonstration
   a. Efficiency
   b. Innovate Faster
   c. Revenue Growth
4. KPI Outcomes



Modernization Use Case 2
AWS RDS-Aurora to EDB PG

1. Scenarios
   a. Tier 1-2 Triple A, Double A Same DC
   b. Tier 3-4 Active Passive DR
2. Content per Scenario
   a. Reference Architecture
   b. App Migration Process
   c. Differentiation & Benchmarks
   d. Decisioning App Modernization
3. Demonstration
   a. Efficiency
   b. Innovate Faster
   c. Revenue Growth
4. KPI Outcomes



Modernization Use Case 3
Database to EDB PG

1. Scenarios
   a. Tier 1-2 Triple A, Double A Same DC
   b. Tier 3-4 Active Passive DR
   c. Start with Oracle - Next: SQL Server, Cockroach, MS SQL, Sybase, DB2
2. Content per Scenario
   a. Reference Architecture
   b. App Migration Process
   c. Differentiation & Benchmarks
   d. Decisioning App Modernization
3. Demonstration
   a. Efficiency
   b. Innovate Faster
   c. Revenue Growth
4. KPI Outcomes

2. Deep Dive: Oracle to EDB and Migration Framework

JPMorganChase

2.1. Oracle Migration Complexity: The Real Challenges

What makes Oracle migrations complex

Technical debt

- 15+ yearsofPL/SQL accumulation
- 10,000+ stored procedures and packages
- Deep application dependencies on Oracle features
- RAC clusters with active-active requirements

Oracle lock-in features

- Advancedqueuing: Message processing
- DBMS packages: Business logic embedded
- Materialized views: Fast refresh dependencies
- Flashback: Temporal queries in applications
- RAC: High availability requirements

Business constraints

- Zero data-loss tolerance
- <1 hour maintenance windows
- Regulatory compliance requirements
- Performance SLAs must be maintained

How EDB solves this

- ❖70%–80% automatic PL/SQL compatibility
- ❖Oracle packages run natively (DBMS_ , UTL_)
- ❖Near-zero downtime with Replication Server
- ❖Performance parity or better for most workloads

2.2. Automated Conversion Strategies: What Works, What Needs Attention

Your Oracle code running on EDB

```
CREATE PACKAGE BODY payroll_pkg AS
  PROCEDURE calculate_salary(emp_id NUMBER) IS
    v_salary NUMBER(10,2);
    v_bonus  VARCHAR2(100);
  BEGIN
    SELECT salary INTO v_salary FROM employees WHERE id = emp_id;
    DBMS_OUTPUT.PUT_LINE('Salary: ' || v_salary);
    UTL_FILE.PUT_LINE(file_handle, 'Processed');
    COMMIT;
  EXCEPTION
    WHEN NO_DATA_FOUND THEN
      DBMS_JOB.SUBMIT(job_id, 'retry_process');
  END;
END payroll_pkg;
```

Status: Runs UNCHANGED on EDB Postgres

What works automatically (80%)

- PL/SQL packages, procedures, functions
- Oracle packages (DBMS_ , UTL_)
- Exception handling with Oracle codes
- Cursors, loops, conditional logic
- Data types (VARCHAR2, NUMBER, DATE)

What needs attention (20%)

- RAC dependencies → Single instance design
- Flashback queries → Temporal tables
- Advanced queuing → Message queue tools
- Materialized view fast refresh → Standard refresh

2.3. EDB Migration Factory Toolkit: Integrated Tool Suite

One platform, complete migration

EDB Migration Factory Toolkit

Phase	Tools	Purpose
Assess	Migration Portal	Cloud-based complexity scoring
Convert	Migration Toolkit + SPL Check	80% automatic schema/code conversion
Migrate	Replication Server	CDC for zero-downtime migration
Validate	LiveCompare	Continuous data verification
Operate	PEM + Failover Manager	Monitoring and HA management

Why integration matters

- **Without EDB:** 10+ separate tools, 6 months of integration, compatibility issues
- **With EDB:** Single platform, pre-integrated, proven compatibility

Key advantages

- **All tools designed to work together**
- **Single-vendor support**
- **Unified licensing model**
- **40% faster project delivery**

From assessment to production — one toolkit does it all

2.4. Critical Gotchas and Solutions: Oracle and Cloud Migration Pitfalls

Oracle → EDB: The hidden surprises

Gotcha	Impact	EDB Solution
Empty strings = NULL	Application logic breaks	*edb_redwood_strings = true*
DUAL table queries	Syntax errors	DUAL supported natively
Sequence.CURRVAL	Session not found	Use *LASTVAL() or RETURNING*
Global temp tables	Data visible across sessions	EDB session-based temps Extension: *pgtt*
DATE includes time	Time stamp mismatches	Oracle-compatible *DATE* type

Key insight: Most “gotchas” have built-in **EDB solutions**—just enable them.

Test these scenarios first; save weeks of debugging later.

2.5. EDB Migration Framework ACTIONS:
Architectural Compatibility and Assessment

Architectural Compatibility

- **Oracle SGA/PGA/RAC** → EDB simpler cluster model + read replicas
- **Oracle background processes** → PostgreSQL process-per-connection (use PgBouncer connection pooler)
- **Data Guard/Flashback** → EDB streaming replication + Failover Manager

EDB Oracle Compatibility

- 85% PL/SQL syntax supported (packages, triggers, functions)
- **Native Oracle packages**: DBMS_OUTPUT, UTL_FILE, DBMS_JOB
- **Compatible types:** VARCHAR2, NUMBER, DATE + CONNECT BY queries

Gaps Requiring Redesign and need requirement investigation

- RAC, Fast Refresh MVs, and Flashback

Application Changes

- **Drivers**: OCI/JDBC → EDB drivers
- **Connections**: TNS → host:port; Shared servers → PgBouncer
- **PL/SQL:** 70-80% runs unchanged with minor adjustments

Assessment

EDB:

-
- Migration Portal (web-based)
- Migration Toolkit, and Professional Services custom tools

Community/Third-party:

- ora2pg (SHOW_REPORT)
- AWS SCT

Manual using Oracle Catalogs: Identify licensed features, top SQL, dependencies & workload patterns

- DBA_FEATURE_USAGE_STATISTICS
- V$SQL
- DBA_DEPENDENCIES
- AWR

Know the gaps before you leap – assessment prevents migration creep

2.5. EDB Migration Framework ACTIONS:
Conversion

Automated Conversion (80-85% coverage)

- **EDB Migration Portal:** Web-based, generates downloadable DDL scripts
- **EDB Migration Toolkit:** Tables, indexes, constraints, views, PL/SQL objects
- **Preserves Oracle-compatible mode** (VARCHAR2, NUMBER, DATE)

Manual Conversion Required

- **Index-organized tables** → Regular tables
- **Virtual columns** → Generated columns/views
- **External tables** → Foreign Data Wrappers
- **RAC-specific objects** → Remove/redesign

Key Conversion Challenges

- **SQL:** MODEL → CTEs, PIVOT/UNPIVOT → crosstab/CASE, MERGE → minor adjustments
- **PL/SQL:** Remove EDITIONABLE, adjust PRAGMA, bulk operations syntax
- **Storage:** Bitmap → B-tree/GIN, fast refresh MVs not supported, result cache → app-level

Conversion Tools

EDB:

-
- Migration Portal (web-based)
- Migration Toolkit, and
  SPL Check (validates converted code)

Community/Third-party:

- ora2pg

Validation

- Run *spl_check_function()* for PL/SQL validation
- Compare object counts Oracle vs EDB
- Test procedures with production-like data
- Document remaining incompatibilities for manual fix

80% automated, 20% expertise – EDB compatibility does the heavy lifting

2.5. EDB Migration Framework ACTIONS:
Tooling and Planning

EDB Migration Suite

- **Migration Portal:** Cloud-based assessment & DDL conversion (no infrastructure needed)
- **Migration Toolkit:** Schema + data migration with Oracle compatibility
- **SPL Check:** PL/SQL code validation post-conversion
- LiveCompare: Data validation between Oracle & EDB

Supplementary Tools

- **ora2pg:** Complexity assessment & custom migrations
- **Replication Server**: Near-zero downtime via CDC
- **pgBackRest/Barman:** Backup (replaces RMAN)
- **PgBouncer:** Connection pooling (replaces shared servers)

Planning Phases

- **Discovery:** Migration Portal assessment, AWR analysis, RAC dependencies, sizing
- **Preparation:** Schema conversion, test environment, Replication Server config, team training
- **Execution:** Phased migration (non-critical → read-heavy → core), LiveCompare validation

Complexity Multipliers & Timeline

- **Heavy PL/SQL:** +2x time | RAC conversion: +3x | Oracle-specific features: +2x
- **Simple migrations**: 4-6 weeks | Complex systems: 3-4 months

Risk Mitigation

- Run SPL Check early to identify code issues Test LiveCompare
- with actual data volumes Plan for unsupported features
- (Flashback, Advanced Queuing) Maintain Oracle 30 days for
- rollback

Success Criteria

- Proceed if PL/SQL compatibility >70% and performance meets SLAs
- Effort split: 70% schema/code conversion, 30% data migration

Right tools + right plan = predictable outcomes

2.5. EDB Migration Framework ACTIONS:
Implementation (Data & Migration)

Offline migration (simple and reliable)

- Stop app → Migrateschema → Transferdata → Validate → Start app
- Tools: Migration Toolkit or pg_dump/restore
- Best for: Dev/test environments, weekend maintenance windows

Hybrid approach (balanced)

- Initial load (weekend)→ CDCsync (weekdays) → Brief cutover
- Tools: MTK for bulk + Replication Server for changes
- Best for: Most production systems

Online migration (mission critical)

- Setup CDC → Continuoussync → Validate → Zero-downtime cutover
- Tools: EDB Replication Server or AWS DMS
- Best for: 24x7 operations, multi-TB databases

Choose your strategy based on requirements

Database Size	Downtime Window	Strategy	Tools
< 100GB	2–4 hours	Offline migration	EDB MTK, pg_dump
100GB–1TB	< 1 hour	Hybrid approach	EDB MTK + EDB Replication Server
> 1TB	Near-zero	Online CDC	EDB Replication Server/DMS

Key decision factor: Acceptable downtime drives strategy, not just size

Right strategy = successful migration

2.5. EDB Migration Framework ACTIONS:
Operation (monitoring, automation, observability)

Monitoring Transition

- AWR/ASH → pg_stat_statements, edb_wait_states
- OEM → Postgres Enterprise Manager (PEM)
- V$SYSSTAT → pg_stat_database; Alert logs → pgBadger
- New concepts: vacuum activity, bloat monitoring

Automation Migration

- DBMS_SCHEDULER → DBMS_JOB / pg_cron
- Data Guard broker → EDB Failover Manager
- RMAN → pgBackRest / Barman
- Shared servers → PgBouncer

Aler ting

- PL/SQL errors, package failures, replication lag
- Connection exhaustion, vacuum freeze warnings

Integration

- PEM REST API → existing monitoring tools
- Export to Prometheus/Grafana for unified dashboard

Different platform, same visibility – transition tools, not practices

2.5. EDB Migration Framework ACTIONS:
Normalization and Validation

Data Validation

- **LiveCompare** for automated row counts and data comparison
- Exclude Oracle-specific columns (ROWID, ORA_ROWSCN)
- Enable **edb_redwood_strings** = true for empty string = NULL

Schema Validation

- **Compare object counts:** Oracle DBA_OBJECTS vs EDB catalog
- Verify constraints, indexes, sequences match

Code Validation

- **SPL Check:** spl_check_function() on all procedures
- Test DBMS_*, UTL_* packages and exception handling

Acceptance Targets

- **Data:** 100% verified by LiveCompare
- **Code:** All PL/SQL executing without errors
- **Performance:** Within 20% of Oracle baselines
- **Issues:** Zero critical discrepancies

Trust but verify: Validation ensures migration success, not hope.

2.5. EDB Migration Framework ACTIONS:
Switch and Support

Go-Live Preparation

- **LiveCompare** for automated row counts and data comparison
- Exclude Oracle-specific columns (ROWID, ORA_ROWSCN)
- Enable **edb_redwood_strings** = true for empty string = NULL

Cutover Execution

- **Update connections:** TNS → EDB, jdbc:oracle:thin → jdbc:edb://
- Final Replication Server sync or Migration Toolkit delta
- **LiveCompare** validation → Switch apps → Keep Oracle for rollback
-

Stabilization (First 72 Hours)

- Monitor PL/SQL errors, connection pools, vacuum activity
- Watch for NULL/empty string edge cases
- Train DBAs: PEM navigation, pgBackRest vs RMA**N**

Success Criteria

- 14 days stable operation without critical issues
- 30 days Oracle read-only as safety net
- Performance meeting/exceeding Oracle baselines

Rehearse like it's live, switch with confidence, support to success.

2.6. Ensuring Success: Optimization and Verification

The 3 pillars of migration excellence

- Performanceoptimization:Transform bottlenecks into breakthroughs with parallel processing, intelligent indexing, and EDB-optimized query patterns that deliver sub-second response times
- Data integrity assurance: Zero-tolerance validation using LiveCompare's real-time verification, cryptographic checksums, and automated reconciliation ensuring 100% data fidelity
- Application continuity: Seamless transition with EDB SPL Check validating PL/SQL compatibility, preserving transaction semantics, and maintaining business-critical SLAs

Success metrics that matter

- 99.999% data accuracy guaranteed
- <20% performance variance from baseline
- Zero business disruption during cutover
- 100% application functionality preserved

Your safety net: LiveCompare + SPL check = Migration confidence

3. Breaking Cloud Lock-in

3.1. Breaking Cloud Lock-in: From RDS/Aurora to Freedom

Cloud PostgreSQL limitations vs. EDB freedom

Cloud Constraints	EDB Solution
No super-user access	Full administrative control
Limited extensions (~80)	200+ extensions available
Single vendor lock-in	Deploy anywhere: Multi-cloud/on-prem
Forced maintenance windows	Maintenance on your schedule
Proprietary backup formats	Standard PostgreSQL tools

Strategic migration: RDS/Aurora PostgreSQL → EDB PostgreSQL

The beauty: 100% PostgreSQL compatible

- Nocodechanges required
- Same SQL, same drivers
- Extensions work identically
- Zero application refactoring

Migration approach

1. Setup: EDB Postgres target environment
2. Sync: Logical replication from RDS/Aurora
3. Validate: LiveCompare ensures data integrity
4. Cutover: Update connection endpoints
5. Freedom: Fullcontrol achieved

Strategic advantages

- Deployanywhere: AWStoday, Azure tomorrow, on-prem next year
- No vendor pricing surprises: Predictable costs
- Full feature access: Super user, all extensions, custom builds
- Your rules: Backup, maintenance, upgrades on your terms

Same PostgreSQL—your infrastructure, your control.

3.2. Cloud-Native to Portable Architecture

Architectural transformation

From managed services constraints

AWS RDS/Aurora → EDB Postgres (pn-premises/IaaS)

- Proprietary APIs → PostgreSQL native protocol
- Black-box operations → Full OS/kernel access
- Limited parameters → Complete postgresql.conf control
- Vendor-specific HA → Patroni/repmgr/EDB Failover Manager

Technical comparison matrix

Capability	AWS RDS/Aurora	EDB Portable Architecture	Advantage
Parameter control	~200 modifiable	300+ full access	Complete tuning
Extension support	40–60 extensions	Unlimited	No restrictions
Replication lag	Opaque metrics	Byte-level control	Predictable RPO
Cost predictability	Variable IOPS/storage	Fixed infrastructure	60% TCO reduction

Portable architecture stack

- Infrastructure layer
  - Deployment targets: VMware vSphere | OpenStack | Bare Metal | AWS EC2 | Azure VMs
  - Storage flexibility: SAN/NAS | Local NVMe | Software-defined (Ceph/GlusterFS)
  - Network architecture: Self-managed VPC | Direct Connect | MPLS

- Database architecture components
  - High availability: Streaming replication + automatic failover
  - Load balancing: HAProxy/PgBouncer/Pgpool-II
  - Backup strategies: Barman/pgBackRest to any S3-compatible storage
  - Monitoring: Prometheus + Grafana + pg_stat_statements

Migration path: RDS Snapshot → S3 → pg_restore → EDB Postgres → Production

4. Q&A and Next Steps

Thank You

Please complete the survey for EDB Days Track 4 now.

EDB POSTGRES AI

JPMorganChase