White Paper

Migrating from Oracle to PostgreSQL®

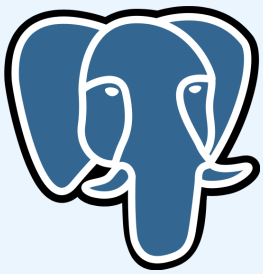
instaclustr
Now part of **Spot by NetApp**

Introduction

With unparalleled extensibility, reliability, SQL (relational) and JSON (non-relational) querying—just to name a few of its many features—it's no surprise that PostgreSQL is rapidly becoming the database platform of choice for organizations all over the globe. And being 100% open source, there are never any licensing fees to pay—ever.

Given the numerous advantages, organizations of all shapes and sizes are looking to migrate from their Oracle databases to PostgreSQL.

This white paper provides guidance on key considerations and requirements, and gives steps on how to successfully migrate database workloads from Oracle to PostgreSQL—and start reaping all the advantages that open source has to offer.



Why PostgreSQL?

- **PostgreSQL instances are not linked to financial requirements**
- **PostgreSQL foreign data wrappers are superior to federation**
- **PostgreSQL authentication hooks to nearly anything**
- **PostgreSQL is infinitely extendable**

Discover more

Click here to learn about **Instaclustr for PostgreSQL**

Features and Benefits

One major advantage of PostgreSQL is the lack of vendor lock-in and forced growth strategy. With no external financial incentives, PostgreSQL installations often utilize many more servers and services divided by business needs like traffic patterns, tuning, availability, and so on.

These may be divided by business unit, traffic pattern, scalability, tuning, availability, recovery, logical services, or any other criteria. For our purposes, we'll focus on using PostgreSQL effectively as a service delivery platform with custom tuning parameters for your specific workload.

PostgreSQL also provides superior data federation capabilities through foreign data wrappers. There are far more data system adapters in PostgreSQL compared to other databases. By combining replication with foreign data wrappers, PostgreSQL enables a highly configurable push-pull processing approach.

For authentication, PostgreSQL integrates with nearly any protocol and identity provider, both locally or remotely, as the protocol and bindings allow. This means that a 1-second authentication can no longer block a 20-millisecond query or create artificially high dwell times for service calls.

Finally, PostgreSQL provides an extension system with an unlimited API into the inner working of the database itself. You can add custom operators, data types, functions, and internals with access to any part of the system.

Requirements

You should know a few things before going down the road of an Oracle to PostgreSQL migration.

For starters, **this is an expert job**. There are coding assistance tools (e.g., Ora2Pg) that will create some basic structures for you, but these tools only create an outline around the actual code migration; there are no tools that are anywhere close to complete coding replacements. Even the tools that do provide some level of schema migration are either incomplete or leave some room for interpretation along the way.

PostgreSQL has much more limited parallel query support—a single backend handles a single request for the lifetime of the request. While there are a few parallel processes in PostgreSQL, such as sequential scanning or sorting, they do not compare to the parallel query structure of Oracle. You will not get the same per-query vertical performance curve.

Before Migrating to PostgreSQL, Consider Several Factors:

- Memory contention works differently. Expertise in tuning is required for getting approximately the same memory operation out of PostgreSQL as Oracle.
- PostgreSQL only has nested transactions when using the procedural language after version 13. Otherwise, there are no nested transactions at all.
- PostgreSQL does not require a dummy DB name or even a FROM clause. If you want to add 2 and 2 you can just SELECT 2+2.
- There are enough syntactical and grammar differences between plSQL and plpgSQL that every function/SQL conversion will have to be manually reviewed.
- There is an Oracle compatibility layer extension for PostgreSQL (orafce). This extension is about 90% complete. That is to say, not complete enough to perform an automated migration.
- PostgreSQL naming convention generally favors lowercase identifiers with underscores for word boundaries. PostgreSQL provides a way to quote identifiers for extended characters and case retention, but it is not recommended to do so.

- Reverse key, bitmap, and join indexes are not supported in PostgreSQL.
- While tablespaces exist in PostgreSQL, their functionality is far less flexible than in Oracle. They are not recommended use for simple data segregation or disk management purposes.

Migration Steps

Assessment

- Data size
- Code complexity
- Code size
- Schema complexity
- Background processes
- Application integration
- Infrastructure

Schema Migration

- Data types
- Operators
- Referential integrity
- Triggers
- Functions

Data Migration

- Switchover strategy
- ETL vs. ELT
- Data enrichment
- Conversions
- Units and precision
- Representation at rest
- Representation in the line
- Representation to the user

1-02Jeb23

Functional Testing

- Comparative testing harness vs. manual testing
- Regression management

Performance Testing

- Memory/Disk/Network/CPU scaling
- Bottleneck identification

Switchover

- High availability
- Discovery recovery

Things to Keep in Mind

A qualified database migration consultant with field experiment should conduct the assessment.

PostgreSQL has equivalent functions, data types, operators, and strategies for every Oracle workload. However, the strategies can vary greatly for how to specifically handle the situation. A direct translation is possible in trivial cases, but as the complexity increases, the underpinnings demand custom answers to each architectural challenge.

As a general rule, the latest available version of PostgreSQL should be the target. It is very likely that a couple of minor releases of PostgreSQL will occur during the development and implementation phase of the migration. The latest available minor version of PostgreSQL is recommended corresponding to the major version that was available at the beginning of the effort.

- A sample dataset for functional testing is mandatory during the development phase, along with a basic regression testing framework. This will mitigate nasty surprises in the field. Ideally, the outputs of the PostgreSQL functions and processes should identically match the Oracle output at some point in the process.

Important Questions to Ask

Here are some of the key questions to consider during the assessment:

- The total size of the database in bytes and number of tables
- Number of custom functions and approximate line counts
- Oracle version
- Oracle installed features (forms, etc.)
- Application language
- Current backup strategy vs. intended backup strategy
- High availability architecture (including network infrastructure)
- Disaster recovery strategy

Challenges to Plan Ahead For

Migrating from Oracle to PostgreSQL can present some unique challenges that organizations should anticipate and develop mitigation plans for. Being proactive about these potential pitfalls can help ensure a smooth and successful migration.

Feature Compatibility

A key challenge when switching database platforms is ensuring compatibility with features used in the original system. Oracle and PostgreSQL have significant overlap in capabilities but also differences in their implementation of features like JSON support, spatial data, and partitioning.

To avoid unwanted surprises, comprehensively evaluate which Oracle features are utilized and identify any gaps in PostgreSQL's offerings.

User Training

As Oracle and PostgreSQL are based on different paradigms and architectures, developers and DBAs will need training to work effectively in PostgreSQL. Concepts like the handling of schemas, the SQL dialect, and performance tuning vary between the databases.

Plan ahead for adequate training time when transitioning teams to PostgreSQL. Alternatively, consider using a managed service such as Instaclustr for PostgreSQL to offload the day-to-day complexities of managing PostgreSQL and allowing you to focus your resources on other areas of innovation.

Stored Procedures and Functions

Applications that make extensive use of stored procedures and functions written in PL/SQL will need code migration to a PostgreSQL procedural language such as PL/pgSQL.

Ensure that any complex logic is rewritten in a PostgreSQL-compatible language and test it thoroughly before go-live.

Performance Tuning

Because database products differ in their architectures, query processing, caching and more, performance tuning is often required when migrating data platforms.

Analyze existing queries and database performance and optimize tables, queries, and configurations based on PostgreSQL best practices. Allocate time post-migration to continue performance testing and tuning.

Triggers

Oracle and PostgreSQL implement database triggers differently. Key differences include when triggers fire relative to the operation, and how modified rows are accessed within trigger logic. Review triggers in use and convert logic to be compatible with PostgreSQL's approach. Test triggers thoroughly in pre-migration testing to avoid issues.

By anticipating and proactively addressing these common migration challenges, organizations can minimize disruptions and ensure a successful transition from Oracle to PostgreSQL. Partnering with an experienced open source data platform provider can further mitigate risks and streamline the process.

Oracle RAC (Real Applications Clusters)

Oracle RAC is an Oracle add-on tool that enables horizontal scalability to maximize database availability and reduce application impact during outages. A single Oracle database runs across multiple servers, allowing for connection failover and safe replay of changes during databases outages.

There are currently no auto balancing tools for PostgreSQL. In this case, some re-architecting is required to prevent overwhelming a single PostgreSQL instance.

Ready to get started migrating from Oracle to PostgreSQL, or interested to learn more?

Get in touch with our team of experts to discuss your particular use case!

About Instaclustr

Instaclustr helps organizations deliver applications at scale through its managed platform for open source technologies such as Apache Cassandra®, Apache Kafka®, Redis™, OpenSearch®, PostgreSQL®, and Cadence®.

Instaclustr combines a complete data infrastructure environment with hands-on technology expertise to ensure ongoing performance and optimization. By removing the infrastructure complexity, we enable companies to focus internal development and operational resources on building cutting edge customer-facing applications at lower cost. Instaclustr customers include some of the largest and most innovative Fortune 500 companies.