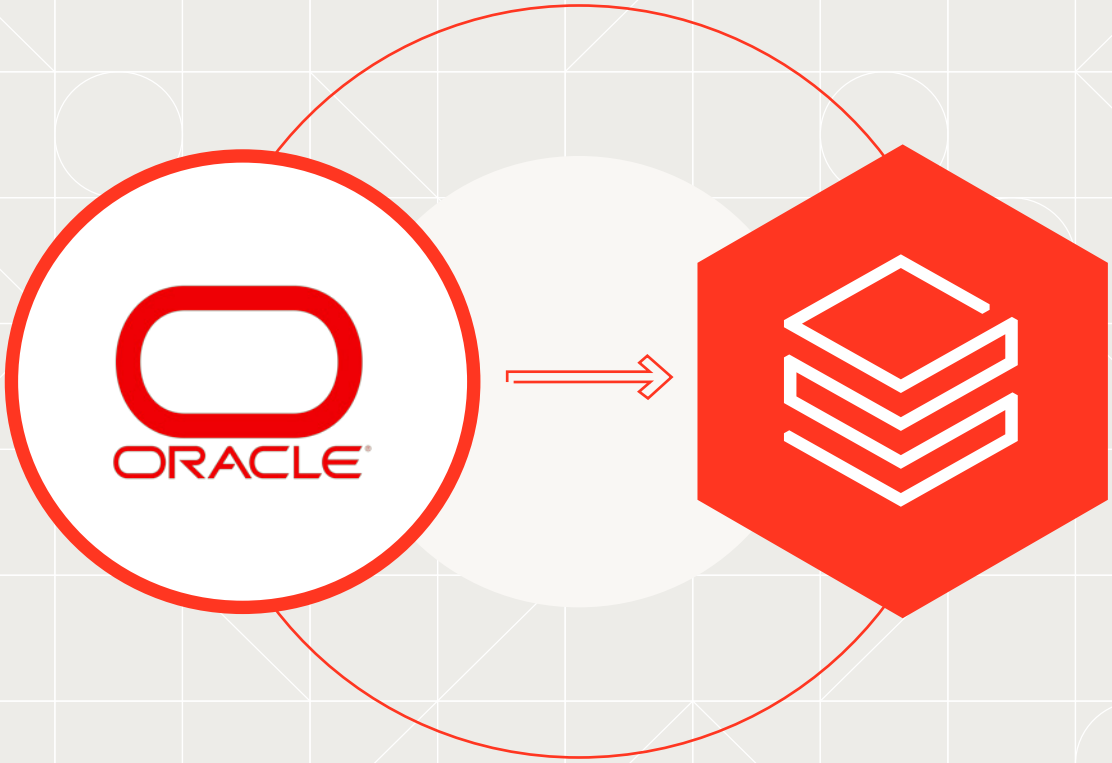
databricks

Guide

Oracle to Databricks Migration Guide



databricks

Contents

Introduction 3
About this guide 4
Migration strategy 4
Overview of the migration process 4

Phase 1: Migration discovery and assessment 6

Phase 2: Architecture design and planning 8
EDW architecture 9

Phase 3: Data warehouse migration 14
Considerations for schema and data migration 14
Phase 3.1: Schema migration 15
Phase 3.2: Data migration 18
Phase 3.3: Other database objects migration 19
Stored procedures implementation in databricks 20
Implement slowly changing dimensions 21
Phase 3.4: Data governance migration 21
Audit logging 21
Unity Catalog 22
Unity Catalog object hierarchy 23

Phase 4: Stored procedures and ETL pipelines migration 25
Orchestration migration 25
Stored procs and ETL code conversion 27
Databricks Code Converter (BladeBridge) 28
Code optimization 29

Phase 5: BI and analytics tools integration 30
Unlock advanced analytics with Databricks SQL 31
Data warehousing and BI report modernization 32

Phase 6: Migration validation 34

Need help migrating? 36

Introduction

Phase 1: Migration Discovery and Assessment

Phase 2: Architecture Design and Planning

Phase 3: Data Warehouse Migration

Phase 4: Stored Procedures and ETL Pipelines Migration

Phase 5: BI and Analytics Tools Integration

Phase 6: Migration Validation

Need Help Migrating?

Introduction

Traditional enterprise data warehouse (EDW) appliances like Oracle come with significant limitations — high costs, lack of support for unstructured data, no built-in AI, ML or real-time streaming capabilities, and scaling storage or compute that is challenging and expensive. To solve this, enterprises have different data marts, data warehouses, data lakes, ML platforms and streaming platforms that create silos because they require constant ETL processes to move data between platforms for different workloads, increasing complexity and slowing down insights.

Databricks' Data Intelligence Platform introduces a paradigm shift by eliminating the need for separate data processing systems and constant data movement. Instead of copying data between warehouses, marts, lakes and ML platforms, Databricks brings different processing engines to a single copy of data in the cloud, enabling seamless data warehousing, AI, ML and GenAI use cases on a single platform and data asset. With its lakehouse architecture, Databricks provides governance, scalability and advanced analytics while decreasing costs and operational complexity. Migrating to Databricks ensures your organization is ready for the future with a unified, efficient and intelligent data platform.

Introduction

Phase 1: Migration Discovery and Assessment

Phase 2: Architecture Design and Planning

Phase 3: Data Warehouse Migration

Phase 4: Stored Procedures and ETL Pipelines Migration

Phase 5: BI and Analytics Tools Integration

Phase 6: Migration Validation

Need Help Migrating?

ABOUT THIS GUIDE

This guide provides a detailed roadmap for migrating data warehouse workloads from Oracle to the Databricks Data Intelligence Platform. It outlines key differences between the two systems, standard data and code migration patterns and best practices to streamline the transition.

Additionally, it compiles proven methodologies, tool options and insights gained from successful migrations. This migration guide covers theoretical concepts and practical applications and is a comprehensive resource for organizations looking to leverage Databricks for enhanced performance, scalability and advanced analytics.

MIGRATION STRATEGY

Successfully migrating from Oracle to Databricks requires careful planning, strategic alignment, clear target architecture and successful objectives. Following a proven structured migration methodology is critical to achieving a seamless, effective migration to Databricks, enabling your organization to realize value and position itself for rapid future innovation.

OVERVIEW OF THE MIGRATION PROCESS

Proper planning is required to migrate data and ETL processes from legacy on-premises systems like Oracle data warehouses to cloud technologies. This migration involves transferring data and business logic from on-premises infrastructure to the cloud.

Introduction

Phase 1: Migration Discovery and Assessment

Phase 2: Architecture Design and Planning

Phase 3: Data Warehouse Migration

Phase 4: Stored Procedures and ETL Pipelines Migration

Phase 5: BI and Analytics Tools Integration

Phase 6: Migration Validation

Need Help Migrating?

Despite the substantial differences between Oracle and Databricks, there are surprising similarities that can facilitate the migration process:

- Despite its proprietary SQL dialect, Oracle adheres to ANSI SQL standards, providing compatibility with Databricks SQL syntax
- Code migration can be accomplished through code refactoring, leveraging the shared ANSI SQL compliance between the two systems
- Fundamental data warehouse concepts exhibit similarities across Oracle and Databricks, streamlining the transition process

The migration process typically consists of the following technical implementation phases:

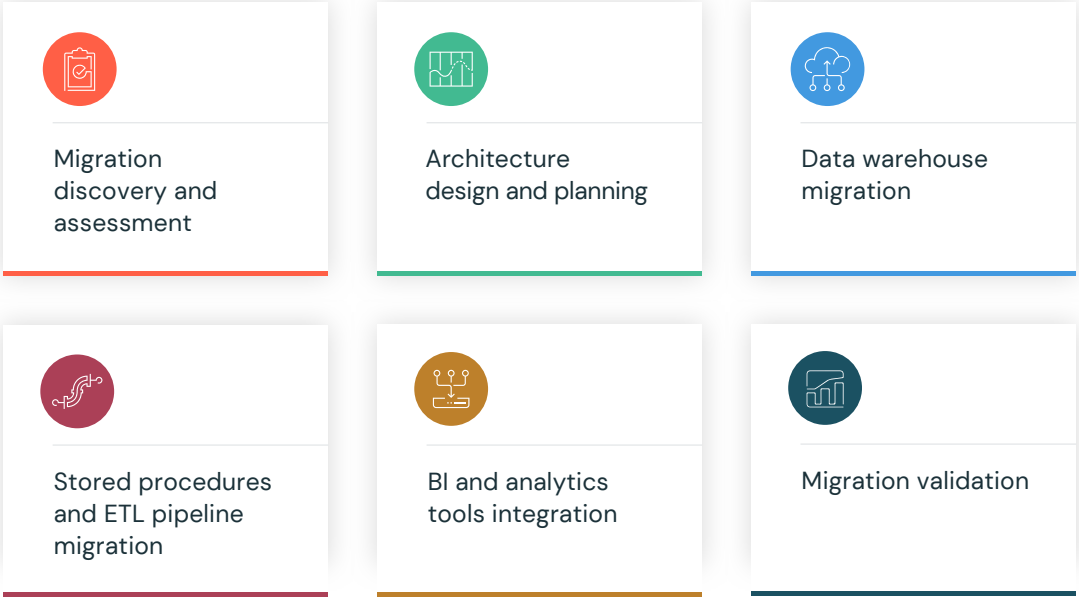


Figure 1: Technical migration approach

Introduction

Phase 1: Migration Discovery and Assessment

Phase 2: Architecture Design and Planning

Phase 3: Data Warehouse Migration

Phase 4: Stored Procedures and ETL Pipelines Migration

Phase 5: BI and Analytics Tools Integration

Phase 6: Migration Validation

Need Help Migrating?

Phase 1: Migration Discovery and Assessment

Conducting a migration assessment is crucial before migrating any data or workloads. This assessment enables Databricks to:

- Gain insight into data ingress and egress, ETL patterns, data volume, orchestration tools and execution frequency
- Understand the technologies involved in upstream and downstream integrations
- Assess the business criticality and value of the existing systems
- Evaluate the existing security framework and access control mechanisms
- Gather pertinent information to provide a realistic estimation of the effort required for migration
- Compare and calculate infrastructure costs
- Identify any imminent deadlines, particularly license renewal fees for the existing Oracle setup
- Document any functional or cross-functional dependencies in the migration plan

Databricks recommends automation tools, such as Oracle Profiler and the recently acquired BladeBridge Code Analyzer, to expedite gathering migration-related information during this phase.

Introduction

Phase 1: Migration Discovery and Assessment

Phase 2: Architecture Design and Planning

Phase 3: Data Warehouse Migration

Phase 4: Stored Procedures and ETL Pipelines Migration

Phase 5: BI and Analytics Tools Integration

Phase 6: Migration Validation

Need Help Migrating?

Typically, these profiling tools examine Oracle system usage via its system views and catalog tables, providing consumption and an inventory of objects and code migration complexity. They capture the types of workloads, long-running ETL queries and user access patterns. This level of analysis aids in pinpointing databases and pipelines that contribute to high operational costs and complexity, thereby supporting the prioritization process.

Our BladeBridge Code Analyzer not only classifies queries based on their complexity in "T-shirt sizes" (small, medium, large, extra large, etc.) but also assesses function compatibility of SQL scripts and stored procedures, which is vital in ensuring seamless migration.

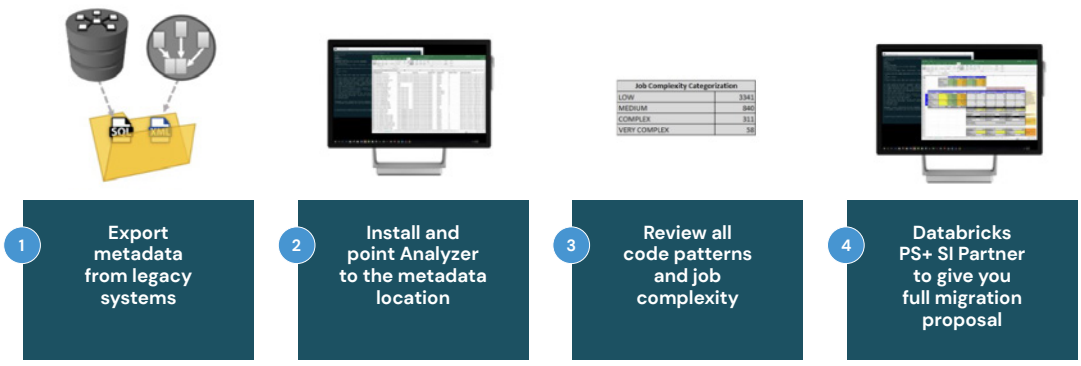


Figure 2: Running Databricks migration analyzer

Introduction

Phase 1: Migration Discovery and Assessment

Phase 2: Architecture Design and Planning

Phase 3: Data Warehouse Migration

Phase 4: Stored Procedures and ETL Pipelines Migration

Phase 5: BI and Analytics Tools Integration

Phase 6: Migration Validation

Need Help Migrating?

Phase 2: Architecture Design and Planning

Oracle and Databricks operate in markedly different ways. As an on-premises system, Oracle requires careful indexing strategies using various types of indexes:

- B*Trees of various subtypes: standard, reverse key indexes, descending indexes
- Bitmap indexes
- Function-based indexes
- Domain indexes

Data warehouses also require table-based partitioning strategies, which can be done at two different levels and will also require defining the rules of data distribution among range, list, hash, interval or reference partitioning.

By contrast, the Databricks Intelligence Platform is a distributed system by design; the data distribution depends on the configuration of a cluster and the nature of the data. For example, if data is loaded into Databricks, the data will be automatically distributed across the nodes in a cluster. By default, Spark will split the data into partitions, processing each partition by a separate task on an individual CPU. This split allows for efficient parallel processing of the data.

The Databricks distributed design facilitates horizontal scaling, enabling data distribution and computations across multiple nodes in a cluster. This capability allows Databricks to process large datasets and handle high query volumes efficiently, surpassing the capabilities of a traditional EDW system.

It is essential to consider these distinctions when migrating from Oracle to Databricks. By consciously mapping the similarities and differences between the two platforms, organizations can better understand Databricks' capabilities concerning Oracle.

Introduction

Phase 1: Migration Discovery and Assessment

Phase 2: Architecture Design and Planning

Phase 3: Data Warehouse Migration

Phase 4: Stored Procedures and ETL Pipelines Migration

Phase 5: BI and Analytics Tools Integration

Phase 6: Migration Validation

Need Help Migrating?

EDW ARCHITECTURE

In legacy EDW architectures, data from various systems is typically ingested via ETL tools or ingestion frameworks. After landing in the raw layer, the data progresses to the stage or central layer, where further cleansing and processing occur. Finally, it moves to the last layer, containing the most complex business logic.

After the final layer is prepared, use it for reporting purposes through third-party tools such as Microstrategy or BusinessObjects.

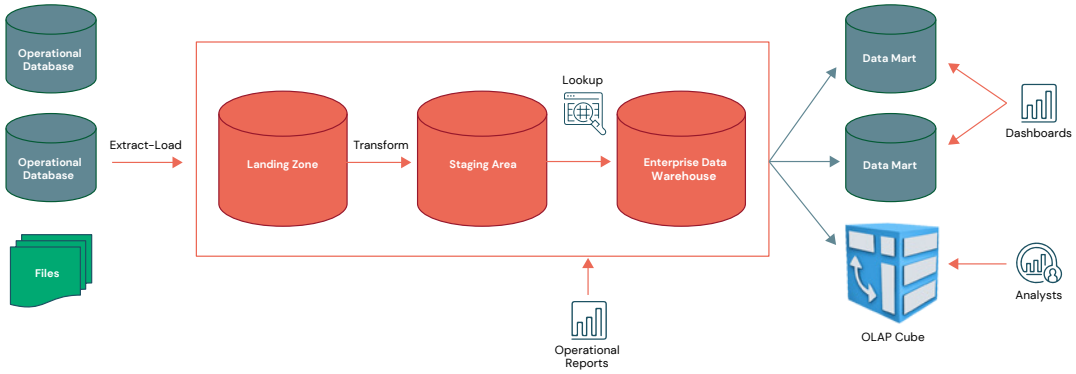


Figure 3: Typical enterprise data warehouse architecture

It is imperative to comprehensively analyze the current architecture, which involves understanding upstream and downstream integrations and the respective technologies utilized.

Following this analysis, assess the potential for modernizing each stage of the target architecture. This entails evaluating how well an organization can transition from legacy systems to modern alternatives at each stage. Key decisions on data ingest into cloud storage include evaluating features like Databricks Autoloader, Lakeflow Connect or Lakehouse Federation. Also assess ETL modernization and compatibility with partners and BI tools. A target architecture and tooling roadmap is created that guides the migration process.

Introduction

Phase 1: Migration Discovery and Assessment

Phase 2: Architecture Design and Planning

Phase 3: Data Warehouse Migration

Phase 4: Stored Procedures and ETL Pipelines Migration

Phase 5: BI and Analytics Tools Integration

Phase 6: Migration Validation

Need Help Migrating?

Below is an example of a data warehousing architecture on Databricks with various ISV partner integration options.

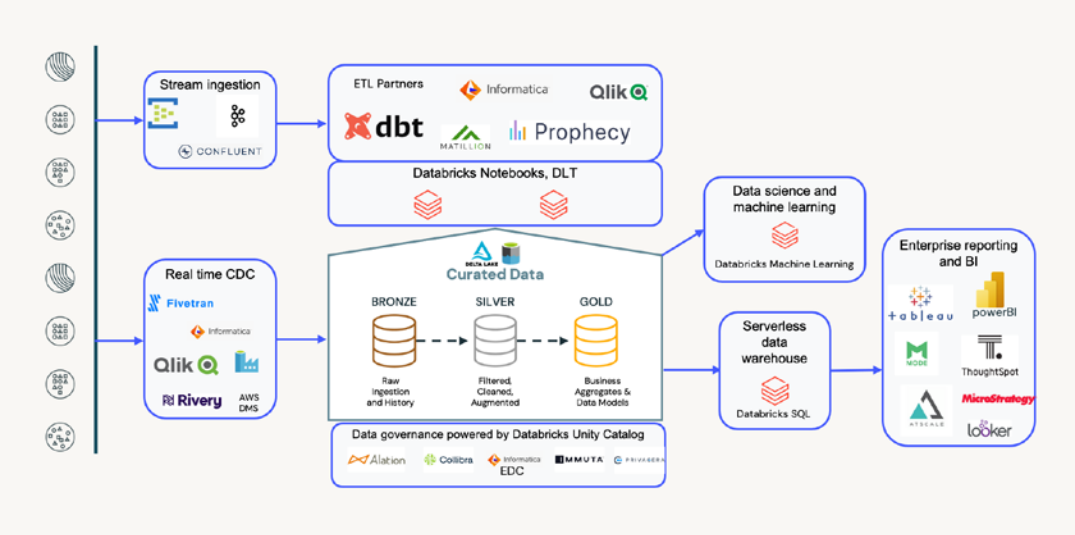


Figure 4: Modern data warehousing on Databricks

Following the architectural alignment, we will deeply investigate Oracle's current features.

ORACLE VS. DATABRICKS FEATURE MAPPING EXAMPLE

OBJECTS/ WORKLOAD	ORACLE	DATABRICKS
Compute	• Oracle on-premises compute • Oracle Exadata on-premises • Oracle Exadata Database Service (OCI) • Oracle Exadata Cloud@customer (Hybrid) • Oracle Base Database Service (OCI) • Oracle Autonomous Database service	Databricks Managed Clusters optimized for workload types with a runtime: • All-purpose for interactive/developer use • Job clusters for scheduled pipelines • SQL warehouse for BI workloads
Storage	• Physical HDD, SSD, or Virtual Disks for on-premises deployments - optionally Oracle ASM (Automatic Storage Management) as an extra management layer • Oracle ASM for Exadata deployments • Combination of Exadata storage, Object storage and local storage with different capabilities based on the underlying infrastructure for Oracle Autonomous Database deployments	Cloud storage (Amazon S3, Azure Blob Storage, Azure Data Lake Storage Gen2, Google Cloud Storage)

Introduction

Phase 1: Migration Discovery and Assessment

Phase 2: Architecture Design and Planning

Phase 3: Data Warehouse Migration

Phase 4: Stored Procedures and ETL Pipelines Migration

Phase 5: BI and Analytics Tools Integration

Phase 6: Migration Validation

Need Help Migrating?

OBJECTS/ WORKLOAD	ORACLE	DATABRICKS
Tables	Oracle Tables	Delta tables in Unity Catalog
Format	• Oracle proprietary • Read only external table capabilities with parquet, Iceberg (Cloud-based deployments only)	Delta and Iceberg Format (open source)
User Interface	• Command line: SQL*Plus, sqlCl, OCI Cli • GUI: Oracle SQL Developer, Oracle Database Creation Assistant, Enterprise Manager • API: OCI Rest API, Oracle REST Data Services (ORDS)	Databricks workspace Databricks collaborative notebooks Databricks SQL Query Editor Databricks CLI Visual Studio Code Spark Connect Databricks API Terraform
Database Objects	Tables (opt. partitioned), indexes (opt. partitioned), Views, Materialized Views, Stored Procedures, functions and packages, Java classes	Tables, Views, Materialized Views, DLT, UDFs
Metadata Catalog	Built-in system tables under the SYS schema	Unity Catalog
Data Sharing	• No native support for on-premises mode • Delta Sharing support for Autonomous Database deployments	Delta Sharing Delta Sharing Marketplace
Data Ingestion	• SQL*Loader • Oracle Data Pump (Export/import) • DBMS_CLOUD package (Cloud-based deployments) • Oracle Database Actions (For Autonomous Database deployments)	COPY INTO CONVERT TO DELTA Auto Loader DataFrameReader Integrations via Partner Connect Add data UI
Data Types	Oracle Data types	Data Types in Databricks

Introduction

Phase 1: Migration Discovery and Assessment

Phase 2: Architecture Design and Planning

Phase 3: Data Warehouse Migration

Phase 4: Stored Procedures and ETL Pipelines Migration

Phase 5: BI and Analytics Tools Integration

Phase 6: Migration Validation

Need Help Migrating?

OBJECTS/ WORKLOAD	ORACLE	DATABRICKS
Workload Management	• AWR: Automatic Workload Repository (Under Diagnostic pack license) • ASH: Active Session History (Under Diagnostic pack license) • RAC: Real Application Clusters (Extra cost license)	Cluster configuration (policies) Multi-clustering Intelligent Workload Management Intelligent Autoscaling Adaptive Routing
Security	• User privileges and roles (system, objects, schema) • Database Vault (Extra cost license) • Row-level security with Oracle Label Security (Extra cost license) • Virtual Private Database (Part of Fine-grained access control)	IAM, RBAC Database object permissions (UC) Dynamic data masking Row-level security Column level security
Storage Format	Oracle Proprietary	Delta (Parquet files with metadata) and Iceberg format
Sorting	Partitioning	Z-Ordering Liquid Clustering
Distribution Styles	Based on the partitioning strategy. This strategy can be made at two levels (partition and subpartition). Each can have a different plan (Range, hash, list, interval)	NA. Cloud Storage vs. storing on disks. As Databricks does not have the concept of data distribution in storage, it simplifies the physical design of tables
Programming Language	• SQL for all deployments • Some Python and R functions exist on cloud deployments (mainly ML)	SQL, Python, R, Scala, Java
Data Integration	External tools (Oracle Data Integrator, dbt, Matillion, Talend, Pentaho, Informatica, etc.)	DLT Databricks workflows External tools (dbt, Matillion, Prophecy, Informatica, Talend, etc.)
Orchestration	• Oracle internal scheduler • Third-party tools (Control-M, Autosys, ActiveBatch, etc.)	Databricks Workflows, Airflow

Introduction

Phase 1: Migration Discovery and Assessment

Phase 2: Architecture Design and Planning

Phase 3: Data Warehouse Migration

Phase 4: Stored Procedures and ETL Pipelines Migration

Phase 5: BI and Analytics Tools Integration

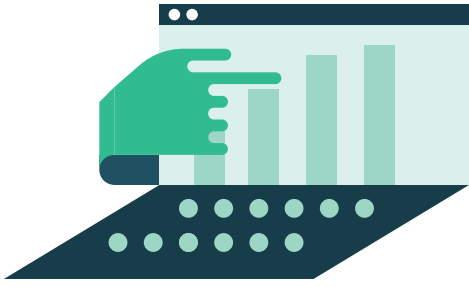
Phase 6: Migration Validation

Need Help Migrating?

OBJECTS/ WORKLOAD	ORACLE	DATABRICKS
Machine Learning	Oracle ML (proprietary) available on • On-premises deployments (SQL Only) • Cloud-based deployments (python, R, SQL)	Databricks ML (Runtime with OSS ML packages, MLflow, feature store, AutoML)
Pricing Unit	On-premises • Standard Edition (per CPU socket) • Enterprise Edition + optional options (per Oracle CPU[1]) Base Database Service • Standard, Enterprise, High Performance, Extreme Performance, BYOL (Per Oracle CPU[1]) Autonomous Databases • ECPU	Databricks units (DBUs)

The table above compares the key features of Oracle and Databricks. Undertaking a thorough comparison is essential during this stage of the migration process. This systematic process ensures a comprehensive understanding of the required transformation, facilitating a smoother transition by identifying equivalent services, functionalities and potential gaps or challenges.

Typically, by the end of this phase, we have a good handle on the scope and complexity of the migration and can come up with a more accurate migration plan and cost estimate.



1. Oracle CPU = #CPU Core x Core factor depending on the CPU Type (See here for core factors)

Introduction

Phase 1: Migration Discovery and Assessment

Phase 2: Architecture Design and Planning

Phase 3: Data Warehouse Migration

Phase 4: Stored Procedures and ETL Pipelines Migration

Phase 5: BI and Analytics Tools Integration

Phase 6: Migration Validation

Need Help Migrating?

Phase 3: Data Warehouse Migration

The Databricks environment can be set up by following the Setup and Administration Guides. Please see the Azure Databricks Administration Guide, Databricks on AWS administration introduction or Databricks administration introduction on GCP, depending on the cloud of your choice.

CONSIDERATIONS FOR SCHEMA AND DATA MIGRATION

After establishing Databricks workspaces, the initial migration phase involves migrating table schema and data, including metadata like table data definition language (DDL) scripts, views and table data.

Here are a few recommendations that can enable a smoother and less risky migration:

- Data modeling: As part of the migration, there might be a need to refactor or reproduce a similar data model in an automated and scalable fashion. Visual data modeling tools like Quest ERWIN or sqldbm are found on Databricks Partner Connect. These tools can help accelerate the development and deployment of the refactored data model in just a few clicks. Such a tool can reverse engineer an Oracle data model (table structures and views) in a way that can be implemented in Databricks easily.

Introduction

Phase 1: Migration Discovery and Assessment

Phase 2: Architecture Design and Planning

Phase 3: Data Warehouse Migration

Phase 4: Stored Procedures and ETL Pipelines Migration

Phase 5: BI and Analytics Tools Integration

Phase 6: Migration Validation

Need Help Migrating?

- When migrating DDLs, verifying the provenance of the data schema (e.g., source data) is essential. Consider an instance where one of the Oracle tables is presented with a data type that is proprietary to Oracle (Ex. BFILE, RAW, etc.). That exact data type might not find a precise equivalent for such data types in Databricks, so replacement data types are needed. It is also the case for NUMBER conversion to Databricks DECIMAL, where precision and scale don't use the same limit values. That type of conversion will require specific attention to avoid automatic roundings and data loss (especially when the business requires high-precision decimals, e.g., FSI and HLS).
- We recommend using the Databricks medallion architecture to call out landing zones in Bronze, central repository or data domains in Silver and presentation layers in Gold for logical data organization in the Lakehouse architecture. For more details on modeling approaches and design patterns, refer to Data Warehouse Modeling Techniques.

PHASE 3.1: SCHEMA MIGRATION

Before offloading tables to Databricks, it's essential to establish the schema of the tables within the Databricks environment. These can be beneficial if an organization possesses DDL scripts from its existing system. With minor adjustments, mainly to accommodate the data types used in Databricks, these scripts can be utilized to create corresponding table schemas in Databricks. Reusing existing scripts can streamline the preparation process, thus fostering efficiency and maintaining schema consistency during the migration.

Introduction

Phase 1: Migration Discovery and Assessment

Phase 2: Architecture Design and Planning

Phase 3: Data Warehouse Migration

Phase 4: Stored Procedures and ETL Pipelines Migration

Phase 5: BI and Analytics Tools Integration

Phase 6: Migration Validation

Need Help Migrating?

Oracle DDLs can also be extracted from the SYS schema by querying the dictionary views like CDB_TABLES, DBA_TABLES, CDB_VIEWS, DBA_VIEWS etc. or use the supplied PL/SQL package DBMS_METADATA to extract the table DDL and dependent objects:

```
SQL> declare
       ddl CLOB;
     begin
       DBMS_METADATA.SET_TRANSFORM_PARAM(DBMS_METADATA.SESSION_TRANSFORM,'SEGMENT_ATTRIBUTES',
false);
       ddl := DBMS_METADATA.GET_DDL('TABLE','FACT_TRANSACTION');

       DBMS_OUTPUT.PUT_LINE(ddl);
     END;
     /

  CREATE TABLE "LAURENT"."FACT_TRANSACTION"
   (     "TRANSACTION_ID" NUMBER(19,0),
         "ACCOUNT_ID" NUMBER(10,0),
         "CUSTOMER_ID" NUMBER(19,0),
         "BRANCH_ID" NUMBER(10,0),
         "TRANSACTION_DATE" TIMESTAMP (6),
         "TRANSACTION_TYPE" VARCHAR2(50),
         "TRANSACTION_AMOUNT" NUMBER(15,2),
         "TRANSACTION_DESCRIPTION" VARCHAR2(255),
         "IS_FOREIGN_TRANSACTION" NUMBER(1,0),
         "CURRENCY" VARCHAR2(3),
          PRIMARY KEY ("TRANSACTION_ID")
  USING INDEX  ENABLE,
          CONSTRAINT "FK_TRANS_ACCOUNT" FOREIGN KEY ("ACCOUNT_ID")
           REFERENCES "LAURENT"."DIM_ACCOUNT" ("ACCOUNT_ID") ENABLE,
          CONSTRAINT "FK_TRANS_CUSTOMER" FOREIGN KEY ("CUSTOMER_ID")
           REFERENCES "LAURENT"."DIM_CUSTOMER" ("CUSTOMER_ID") ENABLE,
          CONSTRAINT "FK_TRANS_BRANCH" FOREIGN KEY ("BRANCH_ID")
           REFERENCES "LAURENT"."DIM_BRANCH" ("BRANCH_ID") ENABLE
   )
  PARTITION BY RANGE ("TRANSACTION_DATE") INTERVAL (NUMTOYMINTERVAL(1, 'MONTH'))
 (PARTITION "P_INITIAL"  VALUES LESS THAN (TIMESTAMP' 2010-01-01 00:00:00') )

PL/SQL procedure successfully completed.
```

Introduction

Phase 1: Migration Discovery and Assessment

Phase 2: Architecture Design and Planning

Phase 3: Data Warehouse Migration

Phase 4: Stored Procedures and ETL Pipelines Migration

Phase 5: BI and Analytics Tools Integration

Phase 6: Migration Validation

Need Help Migrating?

Once the DDLs are extracted, converting them to comply with Databricks is essential. Below are some of the example scenarios to be considered:

1. Several keywords in the Oracle CREATE TABLE statement are no longer valid in Databricks, for example, TABLESPACE and the related segment attributes (e.g., PCTSPACE) or PARTITION clauses.

2. Databricks supports IDENTITY columns only on bigint columns.

3. Table options such as distributions and indexes are not applicable in Delta tables.

4. Caution must be exercised when converting Oracle partitioning strategies to partitions in Delta tables. Overpartitioning can lead to unnecessary overhead and minor file problems, ultimately compromising performance in the Lakehouse architecture. Delta's default partition size is 1TB, and Z-order indexes or liquid clustering, completed with predictive optimization, simplify the design in Databricks.

5. Additional Delta table properties can be specified via the TBLPROPERTIES clause, e.g., delta.targetFileSize, delta.tuneFileSizesForRewrites, delta.columnMapping.mode and others.

Introduction

Phase 1: Migration Discovery and Assessment

Phase 2: Architecture Design and Planning

Phase 3: Data Warehouse Migration

Phase 4: Stored Procedures and ETL Pipelines Migration

Phase 5: BI and Analytics Tools Integration

Phase 6: Migration Validation

Need Help Migrating?

PHASE 3.2: DATA MIGRATION

Transferring legacy on-premises data to a cloud storage location for seamless consumption in Databricks can be a demanding task, but there are a few viable options:

1. **Oracle DBMS_CLOUD package** + AzCopy: Leveraging the DBMS_CLOUD package, we can export data to a Unix jumpbox and subsequently utilize Microsoft's AzCopy utility to store it in a cloud storage account, formatted in a Databricks-compatible manner (e.g., Parquet, CSV).

   Note: *For on-premises deployment (19.9 onwards), you can install DBMS_CLOUD by following the Oracle support note: How To Setup And Use DBMS_CLOUD Package (Doc ID 2748362.1)*

2. **Oracle SQL Developer** can offload table content into multiple formats, notably CSV and XML, which can be transferred to an object storage bucket (S3, ADLS or GCS) and then processed by Databricks.

3. **Microsoft Azure Data Factory** (ADF): Another approach involves using Azure ADF to extract data directly from Oracle using an Oracle JDBC/ODBC connector and then store it in a Blob.

4. You can use DMS (Data Migration Service) on AWS to connect to your Oracle Database and export data to S3.

5. **Ingestion ISV Partners:** Databricks Ingestion ISV partners like Qlik or Fivetran can replicate data from Oracle to the Databricks Delta table for historical and CDC data.

Introduction

Phase 1: Migration Discovery and Assessment

Phase 2: Architecture Design and Planning

Phase 3: Data Warehouse Migration

Phase 4: Stored Procedures and ETL Pipelines Migration

Phase 5: BI and Analytics Tools Integration

Phase 6: Migration Validation

Need Help Migrating?

PHASE 3.3: OTHER DATABASE OBJECTS MIGRATION

Other Database Objects such as Views, Stored procedures, Macros, Functions and SQL*Loader Utilities can also be easily migrated to Databricks via our automated code conversion processes. Please review this helpful cheat sheet packed with essential tips and tricks to help you start on Databricks using SQL programming in no time. Some key pointers while converting Oracle-specific SQL objects:

- **Materialized Views** — Materialized views in Oracle are precomputed database objects that store query results physically for faster access. They improve performance for expensive operations by allowing queries to fetch pre-aggregated or pre-joined data. Materialized views can be refreshed periodically or on-demand to stay in sync with the underlying source data. Databricks' Materialized Views can easily replace this functionality by persisting the query results as a delta table and leveraging DLT to refresh the content automatically.
- **Stored Procedures** — Oracle stored procedures (PLSQL) can be easily migrated to Databricks using the newly supported SQL scripting that provides stored procedure functionality to Databricks. Databricks Notebooks and Workflows can also modularize and orchestrate the stored procedure steps. For more details, see the blog post on Converting Stored Procedures to Databricks.
- **Oracle Proprietary Utilities** — Oracle proprietary utilities, such as SQL*Loader, SQL*plus or sqlcl Scripts, need to be rewritten using Databricks-compliant SQL or PySpark. Our Migration Automation tooling (acquired via BladeBridge) makes this simple.

Introduction

Phase 1: Migration Discovery and Assessment

Phase 2: Architecture Design and Planning

Phase 3: Data Warehouse Migration

Phase 4: Stored Procedures and ETL Pipelines Migration

Phase 5: BI and Analytics Tools Integration

Phase 6: Migration Validation

Need Help Migrating?

Stored Procedures Implementation in Databricks

Note that with the newly released Databricks SQL scripting support, you can now quickly deploy or convert powerful procedural logic within Databricks. Databricks SQL scripting supports compound statement blocks (with BEGIN....END). Within the Databricks SQL scripting procedures, we can declare local variables user-defined functions, use condition handlers for catching exceptions and use flow control statements such as FOR loops over query results, conditional logic such as IF and CASE and means to break out loops such as LEAVE and ITERATE. These features make stored procedures migration to Databricks even easier.

For example, if we take the following basic example, we can easily convert this logic into SQL scripting in Databricks.

Oracle PL/SQL Anonymous block

```
DECLARE
  result VARCHAR2(50);
  avgSalary NUMBER;
  CURSOR c is select avg(salary) as value
from hr.employees;
BEGIN
  OPEN c;
  FETCH c INTO avgSalary;
  IF (avgSalary < 5000) THEN
    result := 'POOR';
  ELSIF (avgSalary >= 5000 AND avgSalary
< 7500) THEN
    result := 'AVERAGE';
  ELSE
    result := 'RICH';
  END IF;
  DBMS_OUTPUT.PUT_LINE(result);
END;
```

Databricks SQL Scripting

```
BEGIN
DECLARE x INT;
DECLARE avgSalary STRING;
SET x = (SELECT avg(salary) FROM shared.
laurentleturgez_sandbox.employees);

IF (x < 5000) then
  SET avgSalary = 'POOR';
ELSEIF (x >= 5000 AND x < 7500) then
  SET avgSalary = 'AVERAGE';
ELSE
  SET avgSalary = 'RICH';
END IF;
values(avgSalary);
END;
```

Introduction

Phase 1: Migration Discovery and Assessment

Phase 2: Architecture Design and Planning

Phase 3: Data Warehouse Migration

Phase 4: Stored Procedures and ETL Pipelines Migration

Phase 5: BI and Analytics Tools Integration

Phase 6: Migration Validation

Need Help Migrating?

Implement Slowly Changing Dimensions

Slowly Changing Dimensions (SCDs) are crucial in data warehousing. They manage historical changes to dimensional data, including updates, inserts or deletions, to maintain accurate historical data. DLT pipelines make implementing SCD pipelines very easy, with simple SCD keywords like STORED AS SCD TYPE 1 and STORED AS SCD TYPE 2 to decide whether to overwrite the history or store history as a type 2 dimension.

Here are a few resources that can assist in achieving a successful transition:

- How to implement SCDs when you have duplicates - Part 1
- How to implement SCDs when you have duplicates - Part 2: DLT
- APPLY CHANGES API: Simplify change data capture in DLT

PHASE 3.4: DATA GOVERNANCE MIGRATION

When discussing Data Security Migration, it's essential to consider both authentication and authorization. When planning the migration of Oracle security objects, it's critical to understand the differences between the two platforms and accurately map Oracle's security capabilities to Databricks Data Intelligence platform's security capabilities.

Audit Logging

Oracle offers row-level and column-level security, data masking, integration with LDAP, Entra ID (formerly Active Directory) and audit logging. In Databricks, most user-related security features are managed through Unity Catalog. Audit logging is also available in system tables. For more information, refer to the Audit log system.

Introduction

Phase 1: Migration Discovery and Assessment

Phase 2: Architecture Design and Planning

Phase 3: Data Warehouse Migration

Phase 4: Stored Procedures and ETL Pipelines Migration

Phase 5: BI and Analytics Tools Integration

Phase 6: Migration Validation

Need Help Migrating?

Unity Catalog

We recommend leveraging Databricks Unity Catalog, which offers a unified governance layer for data and AI within the Databricks Data Intelligence Platform. With Unity Catalog, organizations can seamlessly govern their structured and unstructured data, machine learning models, notebooks, dashboards and files on any cloud or platform.

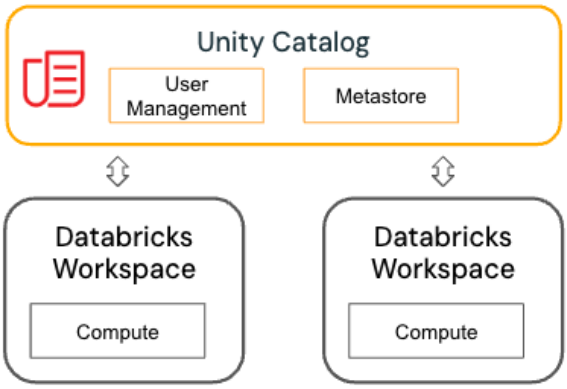


Figure 5: Unity Catalog works across clouds or platforms

Key features of Unity Catalog include:

- **Define once, secure everywhere:** Unity Catalog offers a single place to administer data access policies that apply across all workspaces.
- **Standards-compliant security model:** Unity Catalog's security model is based on standard ANSI SQL and allows administrators to grant permissions in their existing data lake using familiar syntax at the level of catalogs, databases (also called schemas), tables and views.
- **Built-in auditing and lineage:** Unity Catalog automatically captures user-level audit logs that record access to your data. Unity Catalog also captures lineage data that tracks how data assets are created and used across all languages.
- **Data discovery:** Unity Catalog lets you tag and document data assets and provides a search interface to help consumers find data.
- **System tables:** Unity Catalog lets you easily access and query your account's operational data, including audit logs, billable usage and lineage.

Introduction

Phase 1: Migration Discovery and Assessment

Phase 2: Architecture Design and Planning

Phase 3: Data Warehouse Migration

Phase 4: Stored Procedures and ETL Pipelines Migration

Phase 5: BI and Analytics Tools Integration

Phase 6: Migration Validation

Need Help Migrating?

Unity Catalog Object Hierarchy

Unity Catalog consists of a hierarchy of securable objects. The following illustrates a top flow of primary objects:

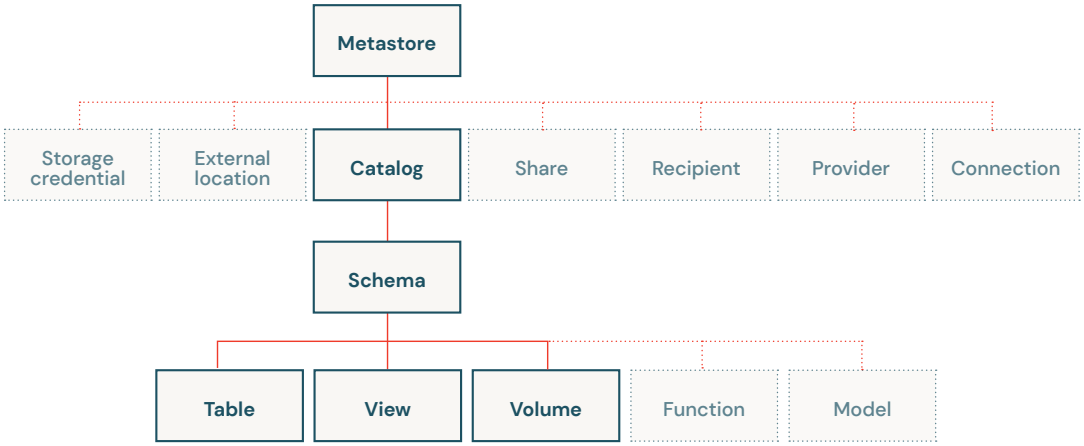


Figure 6: Unity Catalog object hierarchy

From a migration perspective from Oracle to Databricks, pluggable databases can be mapped to Databricks' catalog, Oracle schema can be mapped to Databricks' schema or database and objects are located within this schema/database.

When migrating from Oracle, you can use supplied PL/SQL packages to extract the system and object privileges for a user or a role.

Introduction

Phase 1: Migration Discovery and Assessment

Phase 2: Architecture Design and Planning

Phase 3: Data Warehouse Migration

Phase 4: Stored Procedures and ETL Pipelines Migration

Phase 5: BI and Analytics Tools Integration

Phase 6: Migration Validation

Need Help Migrating?

Using the following pattern will help you convert the governance policy on your Oracle objects to Databricks:

```
-       user/role system privileges
set verify off
set serveroutput on
set linesize 5000 long 99999
DECLARE
  output CLOB;
  username VARCHAR2(120);
BEGIN
  username := upper('&user');
  output := DBMS_METADATA.GET_GRANTED_DDL('SYSTEM_GRANT',username);
  DBMS_OUTPUT.PUT_LINE(output);
END;
/

Enter value for user: laurent

  GRANT UNLIMITED TABLESPACE TO "LAURENT"
  GRANT SELECT ANY TABLE TO "LAURENT"
  GRANT SELECT ANY DICTIONARY TO "LAURENT"
PL/SQL procedure successfully completed.


-       user/role privileges on the object
set verify off
set serveroutput on
set linesize 5000 long 99999

DECLARE
   output CLOB;
   objname VARCHAR2(120);
   schema VARCHAR2(30);
BEGIN
   schema:= upper('&schema');
   objname        := upper('&objectName');
   output := DBMS_METADATA.GET_DEPENDENT_DDL('OBJECT_GRANT',objname,schema);
   DBMS_OUTPUT.PUT_LINE(output);
END;
/

Enter value for schema: hr
Enter value for objectname: employees

  GRANT SELECT ON "HR"."EMPLOYEES" TO "LAURENT"

PL/SQL procedure successfully completed.
```

Introduction

Phase 1: Migration Discovery and Assessment

Phase 2: Architecture Design and Planning

Phase 3: Data Warehouse Migration

Phase 4: Stored Procedures and ETL Pipelines Migration

Phase 5: BI and Analytics Tools Integration

Phase 6: Migration Validation

Need Help Migrating?

Phase 4: Stored Procedures and ETL Pipelines Migration

Data orchestration migration, stored procedure migration and ETL migration are the key elements of the migration process, and Databricks Automated Code converters can help here.

ORCHESTRATION MIGRATION

ETL orchestration involves coordinating and scheduling end-to-end pipelines, including data ingestion, integration and result generation. Oracle typically manages this orchestration using third-party tools like Control-M or Autosys. When migrating these workflows, there are usually several options available to replicate this orchestration functionality:

1. Use Databricks Workflows to orchestrate the migrated pipelines — Databricks Workflows support tasks like Python scripts, Notebooks, DBT transformations and SQL tasks. The customer needs to provide job sequences and schedules as a prerequisite for converting them into Databricks workflows.

2. DLT provides a standard framework for building batch and streaming use cases. It also includes critical data engineering features such as automatic data testing, deep pipeline monitoring and recovery. It also has out-of-the-box functionality for Slow Change Dimension (SCD) Type 1 and Type 2 tables.

Introduction

Phase 1: Migration Discovery and Assessment

Phase 2: Architecture Design and Planning

Phase 3: Data Warehouse Migration

Phase 4: Stored Procedures and ETL Pipelines Migration

Phase 5: BI and Analytics Tools Integration

Phase 6: Migration Validation

Need Help Migrating?

3 | It is also possible to use external tools like Apache Airflow. Considering how tightly coupled Databricks Workflows is with the Databricks Intelligence Platform, it is recommended that Databricks Workflows be used for better integration, simplicity and lineage.

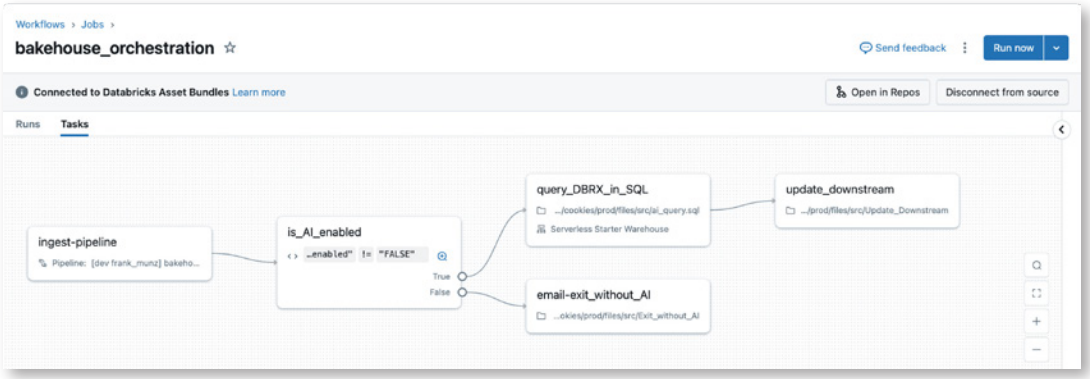


Figure 7: Databricks Workflows

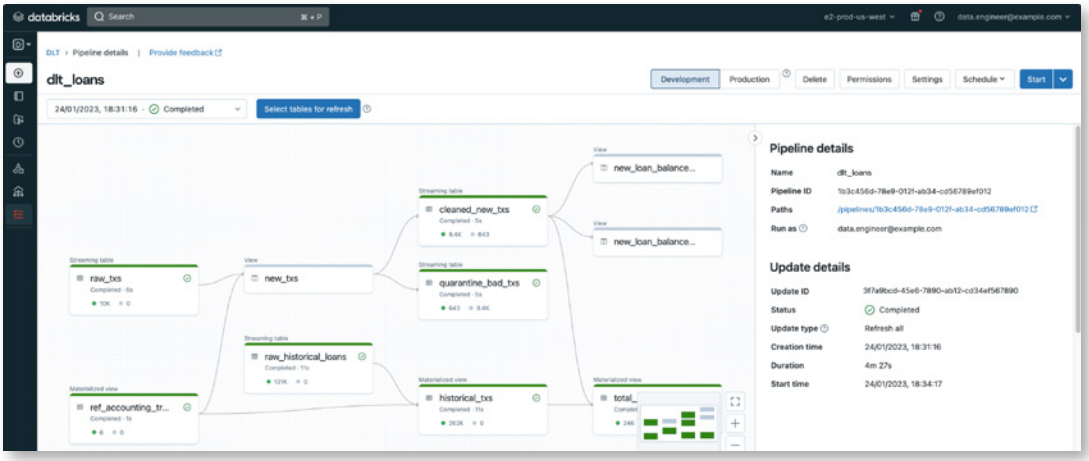


Figure 8: DLT pipelines

Introduction

Phase 1: Migration Discovery and Assessment

Phase 2: Architecture Design and Planning

Phase 3: Data Warehouse Migration

Phase 4: Stored Procedures and ETL Pipelines Migration

Phase 5: BI and Analytics Tools Integration

Phase 6: Migration Validation

Need Help Migrating?

STORED PROCS AND ETL CODE CONVERSION

Migrating from Oracle SQL and PLSQL to Databricks SQL requires identifying and replacing any incompatible/proprietary Oracle SQL functions or syntax. Databricks has mature code converters and migration tooling to make this process smoother and highly automated.

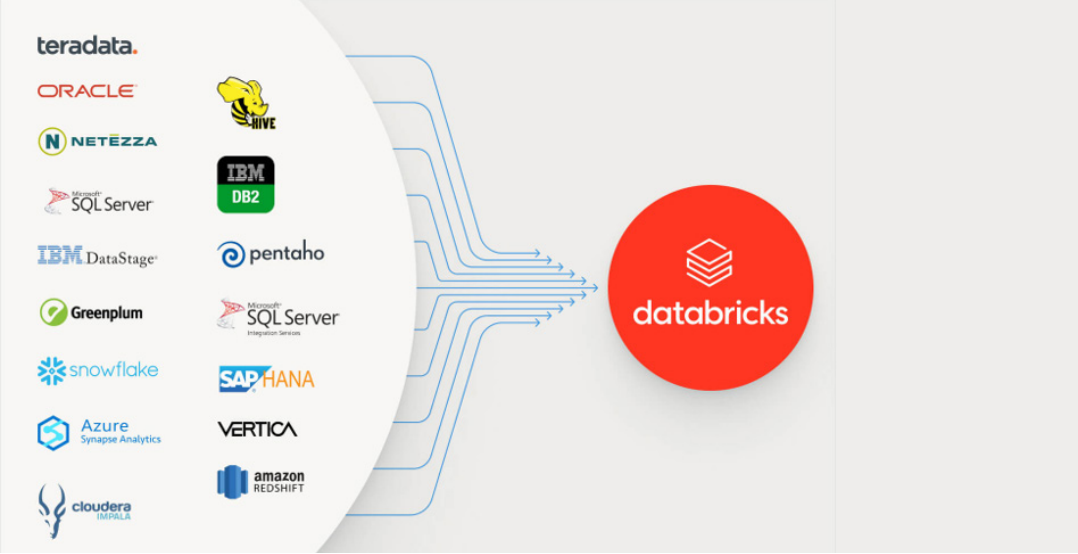


Figure 9: Databricks migration tooling simplifies legacy EDW migrations

Introduction

Phase 1: Migration Discovery and Assessment

Phase 2: Architecture Design and Planning

Phase 3: Data Warehouse Migration

Phase 4: Stored Procedures and ETL Pipelines Migration

Phase 5: BI and Analytics Tools Integration

Phase 6: Migration Validation

Need Help Migrating?

Databricks Code Converter (BladeBridge)

Databricks Code Converter (acquired from BladeBridge) offers automated tooling to modernize and convert Oracle code to Databricks.

- **Automated conversion:** Databricks Converter can automatically convert SQL workloads, significantly speeding up and de-risking migration projects.
- **Broad support:** It supports a wide range of legacy EDW and ETL platform syntax and can convert legacy code to Databricks.
- **Broad adoption by services firms:** Most system integrator partners have deep expertise and access to our converters.
- **Cost and time-effective:** Databricks Converter reduces the cost and time required for a migration project by automating the process.
- **Decreases complexity:** The tool reduces the complexity of the migration process by providing a systematic approach to conversion.

Databricks Code Converter supports schema conversion (tables and views), SQL queries (select statements, expressions, functions, user-defined functions, etc.), stored procedures and data loading utilities such as SQL*Loader. The conversion configuration is externalized, meaning users can extend conversion rules during migration projects to handle new code pattern sets to achieve a more significant percentage of automation. A migration proposal with automated converter tooling can be created for your organization via Databricks Professional Services or certified Migration Brickbuilder SI Partners. Databricks Code Converter tooling requires Databricks professional services or a Databricks SI Partner agreement.

Please review this short demonstration of the conversion tool.

Introduction

Phase 1: Migration Discovery and Assessment

Phase 2: Architecture Design and Planning

Phase 3: Data Warehouse Migration

Phase 4: Stored Procedures and ETL Pipelines Migration

Phase 5: BI and Analytics Tools Integration

Phase 6: Migration Validation

Need Help Migrating?

CODE OPTIMIZATION

Many queries must be refactored and optimized during the migration process. Easy techniques like automated liquid clustering and predictive optimization make performance tuning almost an automated process in Databricks. Predictive optimization uses techniques like:

1. Compaction - which optimizes file sizes.
2. Liquid clustering - which incrementally clusters incoming data, enabling optimal data layout and efficient data skipping.
3. Running vacuum - which reduces costs by deleting unneeded files from storage.
4. Automatic updating of statistics - running the ANALYZE TABLE COMPUTE STATISTIC command on the required columns for best performance.

Figure 10: Automatic liquid clustering

Introduction

Phase 1: Migration Discovery and Assessment

Phase 2: Architecture Design and Planning

Phase 3: Data Warehouse Migration

Phase 4: Stored Procedures and ETL Pipelines Migration

Phase 5: BI and Analytics Tools Integration

Phase 6: Migration Validation

Need Help Migrating?

Phase 5: BI and Analytics Tools Integration

UNLOCK ADVANCED ANALYTICS WITH DATABRICKS SQL

With the data and ETL migration done and security and governance setup via Unity Catalog, unleash AI and BI use cases by spinning up Databricks SQL.

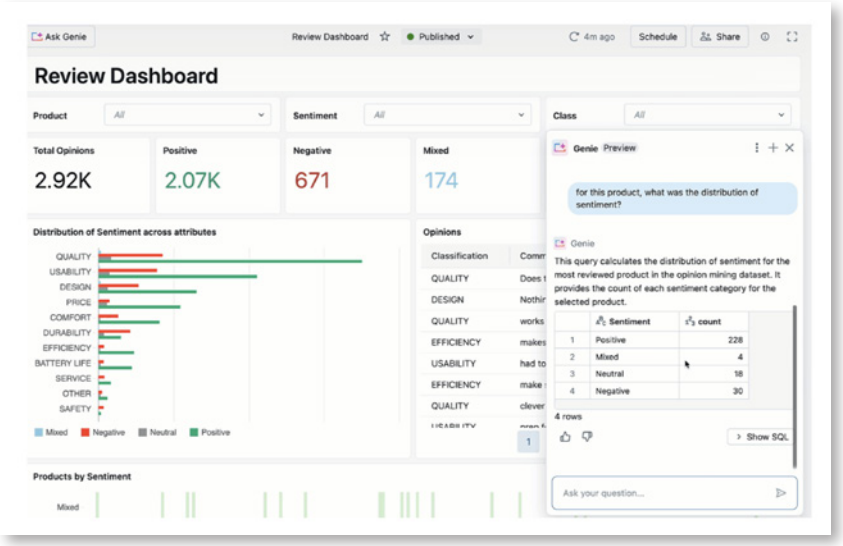


Figure 11: Databricks SQL

Databricks SQL is a serverless data warehousing solution that integrates seamlessly with the Databricks Data Intelligence Platform, offering a unified environment for data, analytics and AI workloads. Key features include:

- **Serverless Architecture:** Provides instant and elastic compute resources, eliminating the need for manual infrastructure management and ensuring rapid scalability to handle varying workloads efficiently.
- **AI-Driven Performance:** Utilizes AI-powered optimizations, such as the Photon query engine, Predictive IO and Intelligent Workload Management, to enhance query execution speed and resource efficiency.

Introduction

Phase 1: Migration Discovery and Assessment

Phase 2: Architecture Design and Planning

Phase 3: Data Warehouse Migration

Phase 4: Stored Procedures and ETL Pipelines Migration

Phase 5: BI and Analytics Tools Integration

Phase 6: Migration Validation

Need Help Migrating?

- **Unified Governance with Unity Catalog:** This solution offers centralized governance and security features, including data discovery, auditing and fine-grained access controls, ensuring compliance and data integrity across the organization.
- **Comprehensive SQL Functionality:** Supports standard ANSI SQL, materialized views, primary and foreign key constraints and advanced data types like 'Variant' for semi-structured data, enabling efficient and flexible data modeling and querying capabilities.
- **Seamless Integration with BI tools:** Integrates with various business intelligence tools like Tableau, Power BI, Thoughtspot, Mode and more. It also supports Lakehouse Federation to legacy EDWs like Oracle and others. This integration allows users to query and govern siloed data systems as an extension of the lakehouse, enhancing data accessibility and collaboration.

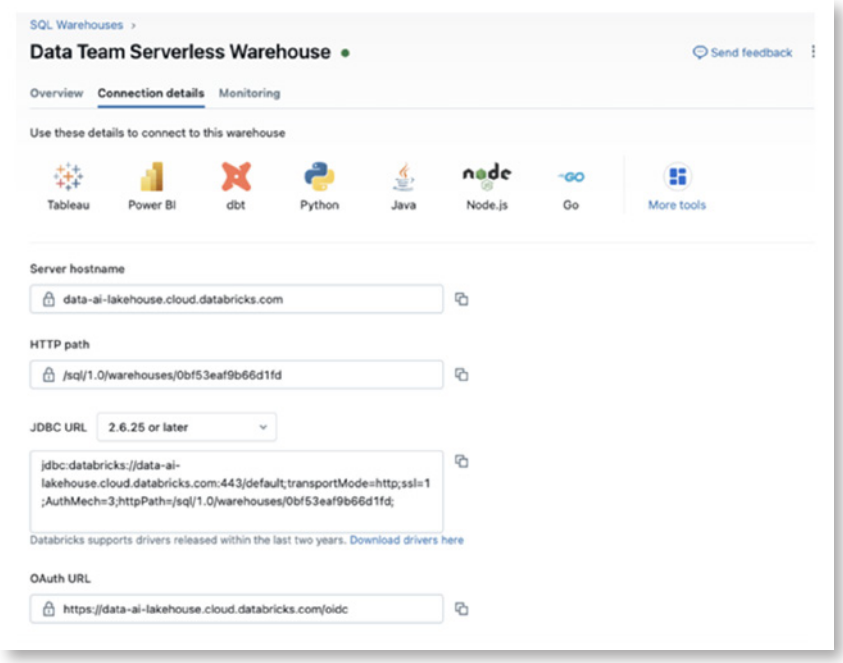


Figure 12: Databricks SQL BI integrations

These features empower organizations to perform high-performance analytics, streamline data workflows and derive actionable insights from their data with reduced operational complexity.

Introduction

Phase 1: Migration Discovery and Assessment

Phase 2: Architecture Design and Planning

Phase 3: Data Warehouse Migration

Phase 4: Stored Procedures and ETL Pipelines Migration

Phase 5: BI and Analytics Tools Integration

Phase 6: Migration Validation

Need Help Migrating?

DATA WAREHOUSING AND BI REPORT MODERNIZATION

Once ingestion and transformation pipelines are migrated to the Databricks Data Intelligence Platform, it is critical to ensure the business continuity of downstream applications and data consumers. Databricks Data Intelligence Platform has validated large-scale BI integrations with many popular BI tools like Tableau, Power BI, Qlik, ThoughtSpot, Sigma, Looker and more.

When repointing BI workloads to Databricks, one common step after data migration is to test sample reports, rename existing tables' data sources or table names and point them to the new ones.

Typically, if the schema of the tables and views post-migration hasn't changed, repointing is a straightforward exercise in handling switching databases on the BI dashboard tool. If the schema of the tables has changed, you will need to modify the tables/views in the lakehouse to match the expected schema of the report/dashboard and publish them as a new data source for the reports.

Many customers take this opportunity to optimize their BI models and semantic layers to align with business needs.

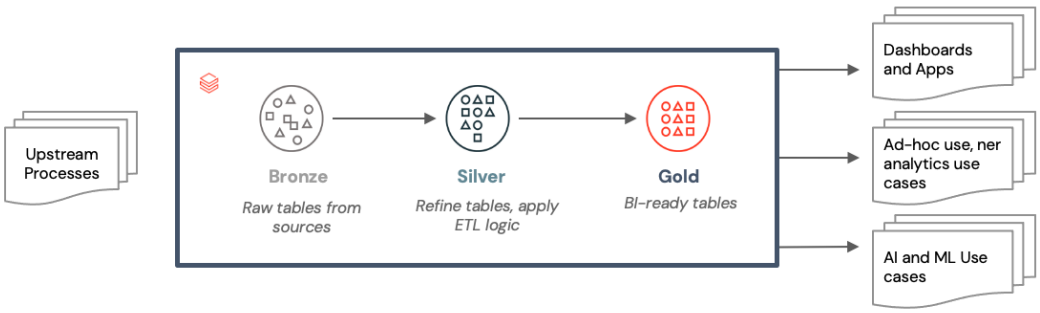


Figure 13: Future-state architecture

Introduction

Phase 1: Migration Discovery and Assessment

Phase 2: Architecture Design and Planning

Phase 3: Data Warehouse Migration

Phase 4: Stored Procedures and ETL Pipelines Migration

Phase 5: BI and Analytics Tools Integration

Phase 6: Migration Validation

Need Help Migrating?

Another advantage of using Databricks for BI is that organizations get a conversational user interface to chat with their data using Databricks' Genie AI/BI Interface, which opens up data for analytics and Q&A even for non-SQL users.

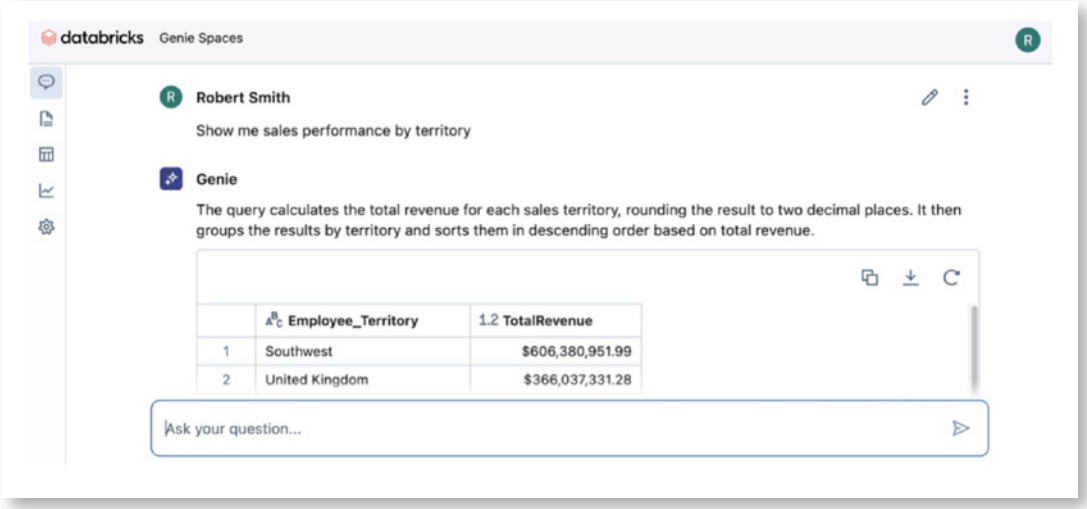


Figure 14: Databricks Genie

Introduction

Phase 1: Migration Discovery and Assessment

Phase 2: Architecture Design and Planning

Phase 3: Data Warehouse Migration

Phase 4: Stored Procedures and ETL Pipelines Migration

Phase 5: BI and Analytics Tools Integration

Phase 6: Migration Validation

Need Help Migrating?

Phase 6: Migration Validation

The primary validation method for a data pipeline is the resulting dataset itself. Databricks recommends establishing an automated testing framework that can be applied to any pipeline. Typically, this involves using a testing framework with a script capable of automatically comparing values in both platforms.

Databricks recommends the following checks at a minimum:

- Check to see if a table exists
- Check the counts of rows and columns across the tables
- Calculate various aggregates over columns and compare, for example:
  - SUM, MIN, MAX, AVG of numeric columns
  - MIN, MAX for string and date/time columns
  - COUNT(*), COUNT(NULL), COUNT(DISTINCT) for all columns

Run the pipelines in parallel for a specific period and review the comparison results to ensure the data is ingested and transformed into the proper context. It is advisable to initiate validation with the most critical tables, which often drive the results or calculations of tables in the gold layer. This includes control tables, lookup tables and other essential datasets.

Introduction

Phase 1: Migration Discovery and Assessment

Phase 2: Architecture Design and Planning

Phase 3: Data Warehouse Migration

Phase 4: Stored Procedures and ETL Pipelines Migration

Phase 5: BI and Analytics Tools Integration

Phase 6: Migration Validation

Need Help Migrating?

A robust data validation requires the following components:

- **Snapshot(s):** Data to work with, including a *pre-* and *post-version* for each script (ideal) and job being migrated.
- **Table comparison code:** A standardized way to compare the result table to determine whether the test is successful. The tables can be compared based on:
  - schema checks
  - row count checks
  - row-by-row checks
- Identifying the primary key combination from the customer is essential to check counts and row-by-row comparisons.

Databricks has tooling such as Remorph Reconcile to streamline the reconciliation process between source data and target data residing on Databricks and other source platforms.

For more advanced table data and schema comparison, tools like Datacompy can be used.

Introduction

Phase 1: Migration Discovery and Assessment

Phase 2: Architecture Design and Planning

Phase 3: Data Warehouse Migration

Phase 4: Stored Procedures and ETL Pipelines Migration

Phase 5: BI and Analytics Tools Integration

Phase 6: Migration Validation

Need Help Migrating?

Need Help Migrating?

Regardless of size and complexity, the Databricks Professional Services team and an ecosystem of certified migration services partners and ISV partners offer different levels of support (advisory/assurance, staff augmentation, scoped implementation) to accelerate your migration and ensure successful implementation.

When engaging with our experts, you can expect:

- **Discovery and Profiling:** Our team starts by clearly understanding migration drivers and identifying challenges within the existing Oracle deployment. We conduct collaborative discussions with key stakeholders, leveraging automated profiling tools to analyze legacy workloads. This is used to determine drivers of business value and total cost of ownership (TCO) savings achievable with Databricks.
- **Assessment:** Using automated tooling, we perform an analysis of existing code complexity and architecture. This assessment helps estimate migration effort and costs, refine migration scope and determine which parts of the legacy environment require modernization or can be retired.
- **Migration Strategy and Design:** Our architects will work with your team to finalize the target Databricks architecture, detailed migration plan and technical approaches for the migration phases outlined in this guide. We will help select appropriate migration patterns, tools and delivery partners and collaborate with our certified SI partners to develop a comprehensive Statement of Work (SOW).

Introduction

Phase 1: Migration Discovery and Assessment

Phase 2: Architecture Design and Planning

Phase 3: Data Warehouse Migration

Phase 4: Stored Procedures and ETL Pipelines Migration

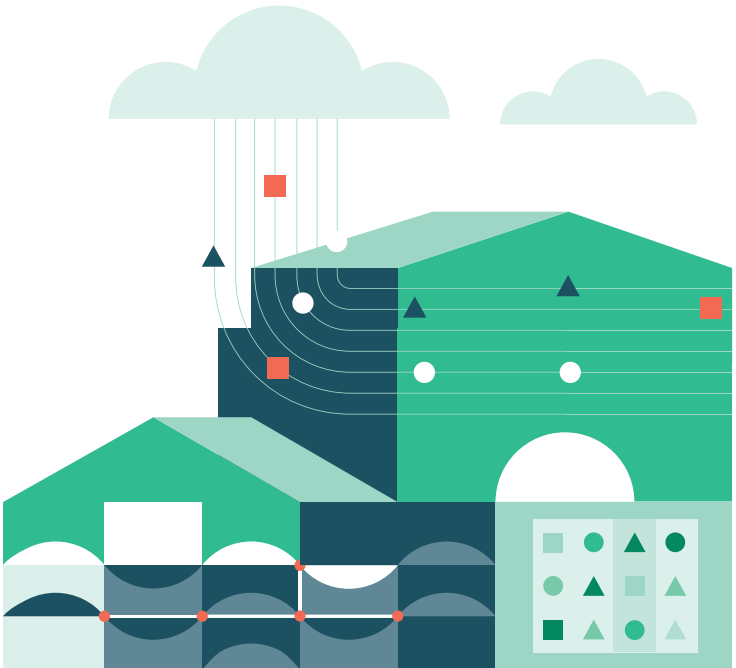
Phase 5: BI and Analytics Tools Integration

Phase 6: Migration Validation

Need Help Migrating?

- **Execute and Scale:** We and our certified partners deliver on our comprehensive migration plan and then work with your team to facilitate knowledge sharing and collaboration and scale successful practices across the organization. Our experts can help you set up a Databricks Center of Excellence (CoE) to capture and disseminate lessons learned and drive standardization and best practices as you expand to new use cases.

Contact your Databricks representative or use this form for more information. Our specialists can help you every step of the way!



About Databricks

Databricks is the data and AI company. More than 10,000 organizations worldwide — including Block, Comcast, Condé Nast, Rivian, Shell and over 60% of the Fortune 500 — rely on the Databricks Data Intelligence Platform to take control of their data and put it to work with AI. Databricks is headquartered in San Francisco, with offices around the globe, and was founded by the original creators of Lakehouse, Apache Spark™, Delta Lake and MLflow.

To learn more, follow Databricks on LinkedIn, X and Facebook.

Start your free trial