Data Ingestion with Delta Lake

MC ID: 7124506 | Parveen KR | Parveen.R@hotmail.com

Agenda

- Introduction to Delta Lake
  - Overview of Delta Lake
  - Benefits of using Delta Lake
- Delta Lake Architecture
  - Components of Delta Lake
  - How Delta Lake works
- Comparison with Traditional Data Lakes
  - Differences between Delta Lake and traditional data lakes
  - Advantages of Delta Lake
- Creating and Loading Delta Tables
- Basic Data Transformations
- Hands-On Labs and Demos

Introduction to Databricks

Unified Data Platform

Databricks provides a cohesive environment for data processing and machine learning on Apache Spark.

Collaborative Workflows

It enables better collaboration among data engineers, scientists, and analysts by streamlining workflows.

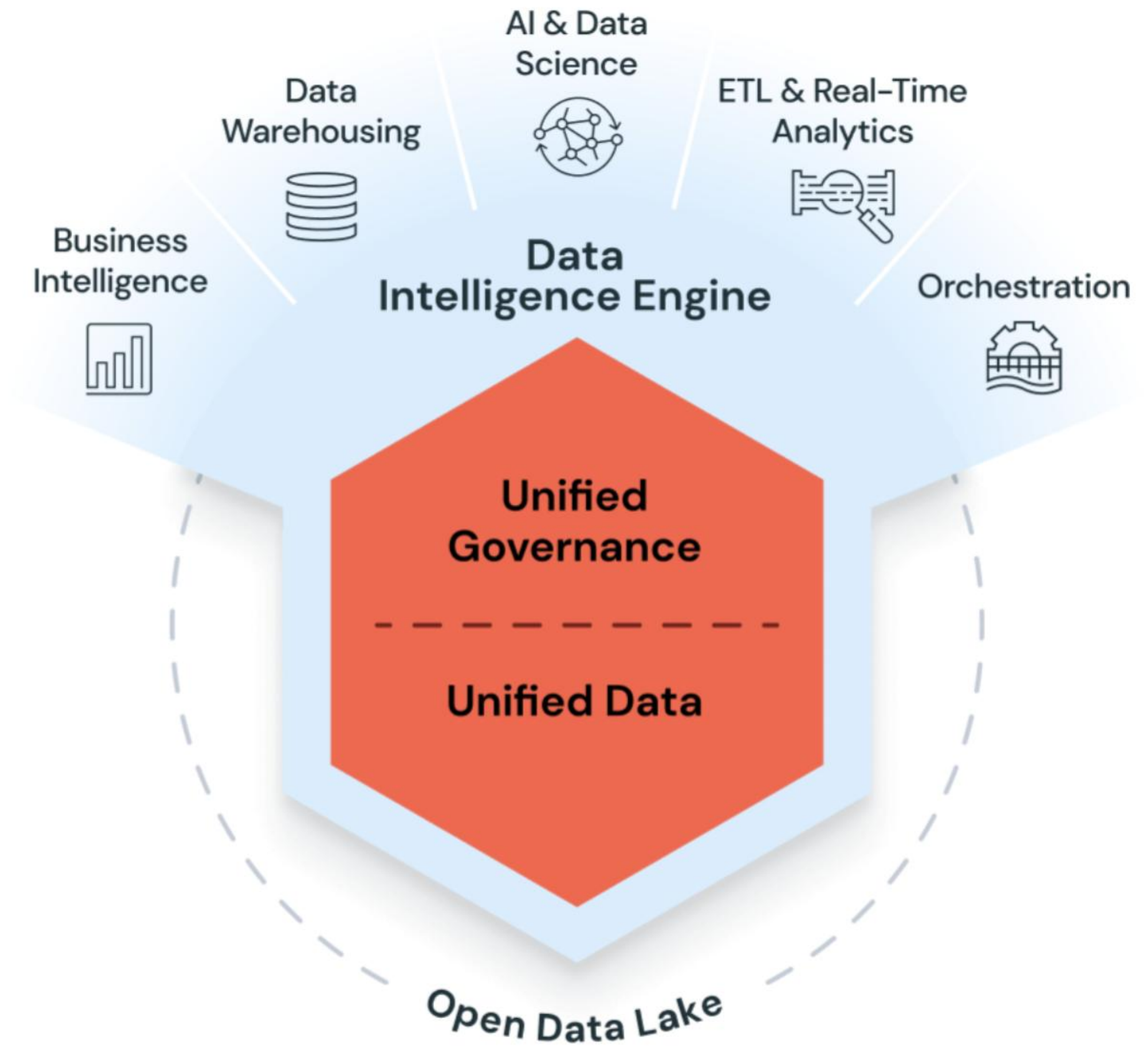
Batch and Real-Time Analytics

Databricks supports both batch and real-time analytics, making it adaptable for various data-driven applications.

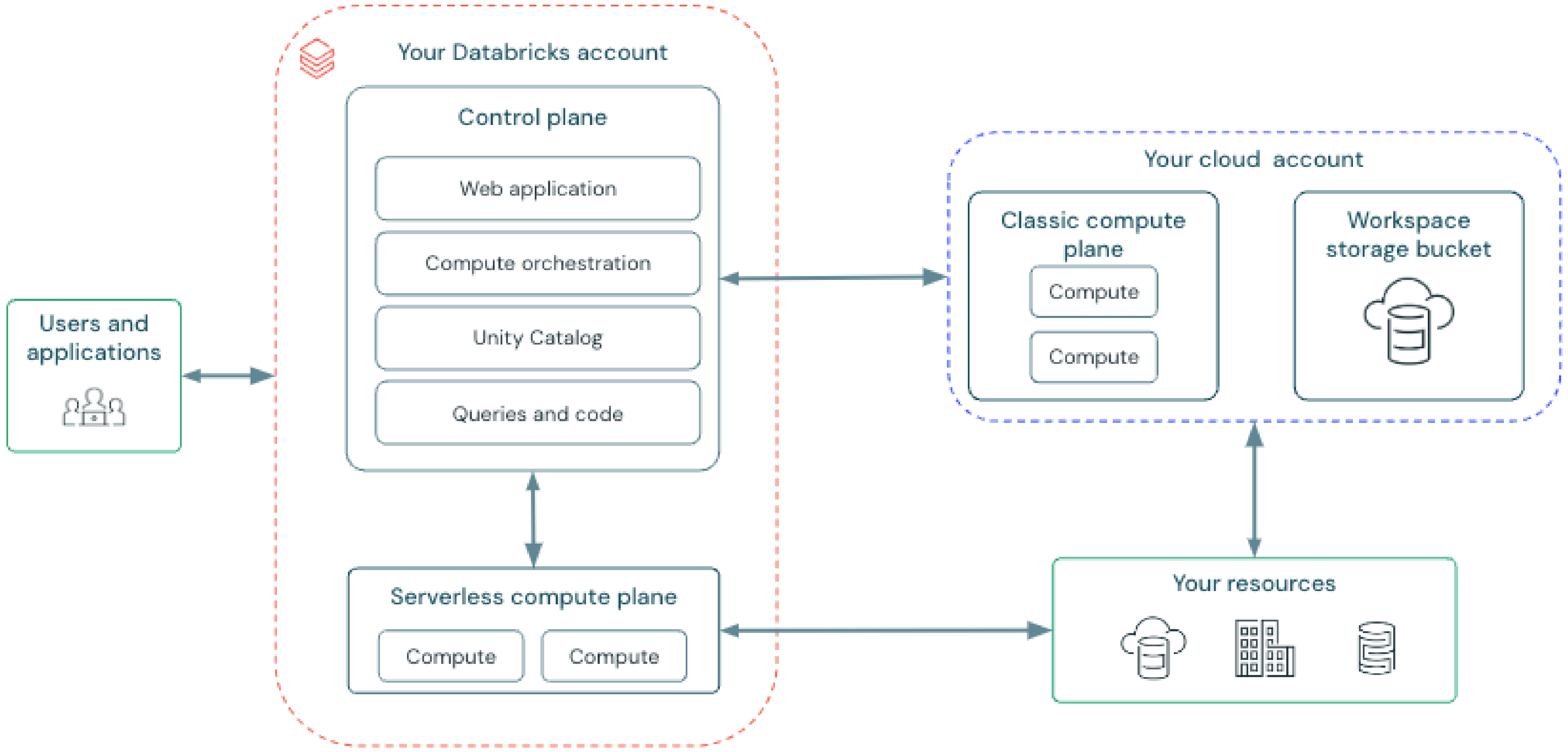
Integration with Tools

It integrates seamlessly with various data sources and tools to enhance data operations.

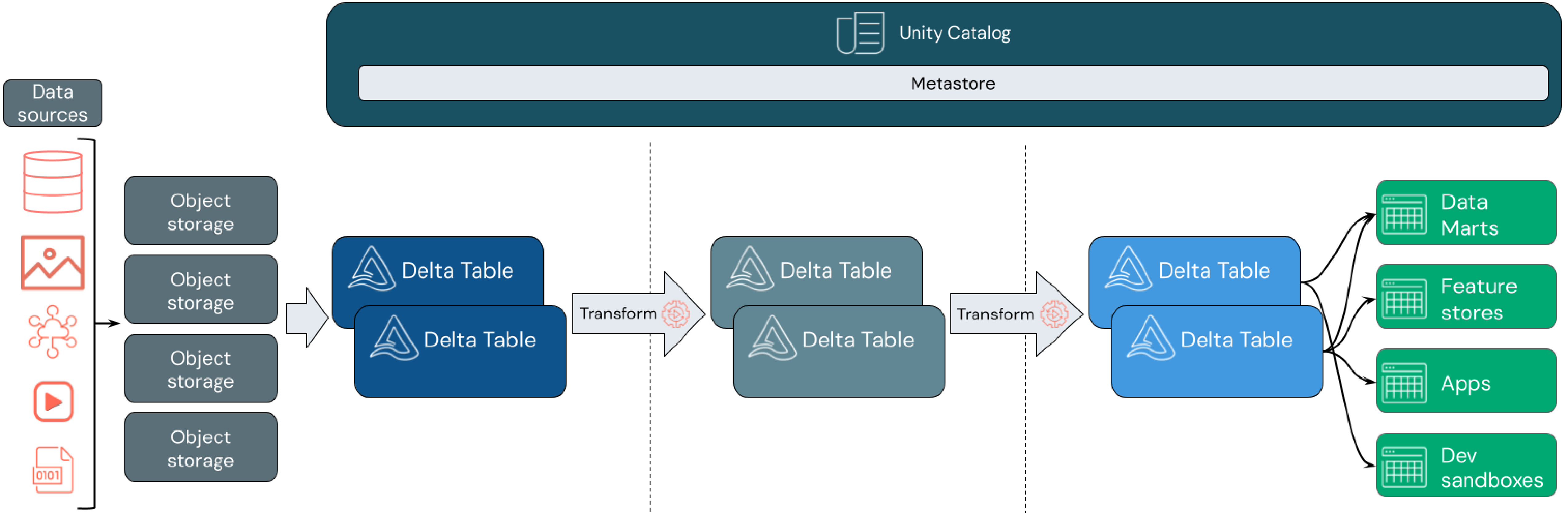
Databricks


AI & Data Science
Data Warehousing
ETL & Real-Time Analytics
Business Intelligence
Data Intelligence Engine
Orchestration
Unified Governance
Unified Data
Open Data Lake

Databricks High Level Architecture



Lakehouse


Unity Catalog
Metastore
Data sources
Object storage
Object storage
Object storage
Object storage
Delta Table
Delta Table
Transform
Delta Table
Delta Table
Transform
Delta Table
Delta Table
Data Marts
Feature stores
Apps
Dev sandboxes

What is Delta Lake?

Open-source storage layer	Designed for data lakes
ACID transactions	Enhances Apache Spark Optimizes big data workloads
Schema enforcement and evolution	Ensures data integrity Adapts to changes
Scalable and reliable	Supports large-scale data Ensures high performance

Delta Lake Architecture Overview

Foundation of Delta Lake Architecture	Built on Apache Spark and Parquet files
Transaction Log	Tracks all changes using .json and checkpoint files
Time Travel	Enables data versioning
Data Processing	Supports both batch and streaming data processing

Key Differences

Data Consistency and Schema Enforcement	Traditional data lakes often struggle with these issues Delta Lake solves them with ACID transactions
Time Travel	Allows easy rollback of data changes Facilitates auditing of data changes

Comparison Table

Feature	Traditional Data Lakes	Delta Lake
Data Consistency	Limited (eventual)	ACID transactions
Schema Enforcement	Manual & error-prone	Built-in schema enforcement
Data Versioning	Usually not available	Built-in time travel
Performance Optimizations	Limited	Z-ordering, compaction

Loading Data into Delta Tables

Ingest data from various file formats

CSV files
JSON files
Parquet files

Batch ingestion method

Use write.format("delta").save() for batch ingestion

Streaming ingestion

Overview provided
Detailed coverage later

Creating Delta Tables

- Syntax to create Delta tables from existing data
  - Steps to convert existing data into Delta format
  - Commands and examples for implementation
- Managed vs external Delta tables
  - Differences between managed and external tables
  - Advantages and disadvantages of each type
- Creating tables using SQL and PySpark
  - SQL commands for table creation
  - PySpark methods for table creation

DataFrame Transformations

- Select, filter, add columns, drop columns
  - Using PySpark DataFrame API
  - Example: Add a total price column (Quantity * UnitPrice)

SQL Transformations

Query Delta tables using Spark SQL	Utilize Spark SQL to interact with Delta tables
Simple SELECT, WHERE, and aggregate queries	Perform basic SQL operations Filter data using WHERE clause Aggregate data for
Example: Total sales per customer	Calculate total sales for each customer Demonstrate practical application of SQL queries

Hands-On Labs and Demos

- Create Delta Table from provided sales CSV
    - Import sales data from CSV file
    - Initialize Delta Table
- Load sales data into Delta Table
    - Insert data into Delta Table
    - Verify data loading
- Perform simple DataFrame transformations
    - Apply basic transformations
    - Filter and aggregate data
- Run SQL queries on Delta Table
    - Execute SQL commands
    - Analyze query results