WHITEPAPER

Data Warehouse to Data Lakehouse Migration Playbook

A Guide for Modernizing Cloud Data Warehouse to an Open Data Lakehouse with Dremio

Tony Truong, Product Marketing at Dremio **Jeremiah Morrow,** Product Marketing at Dremio

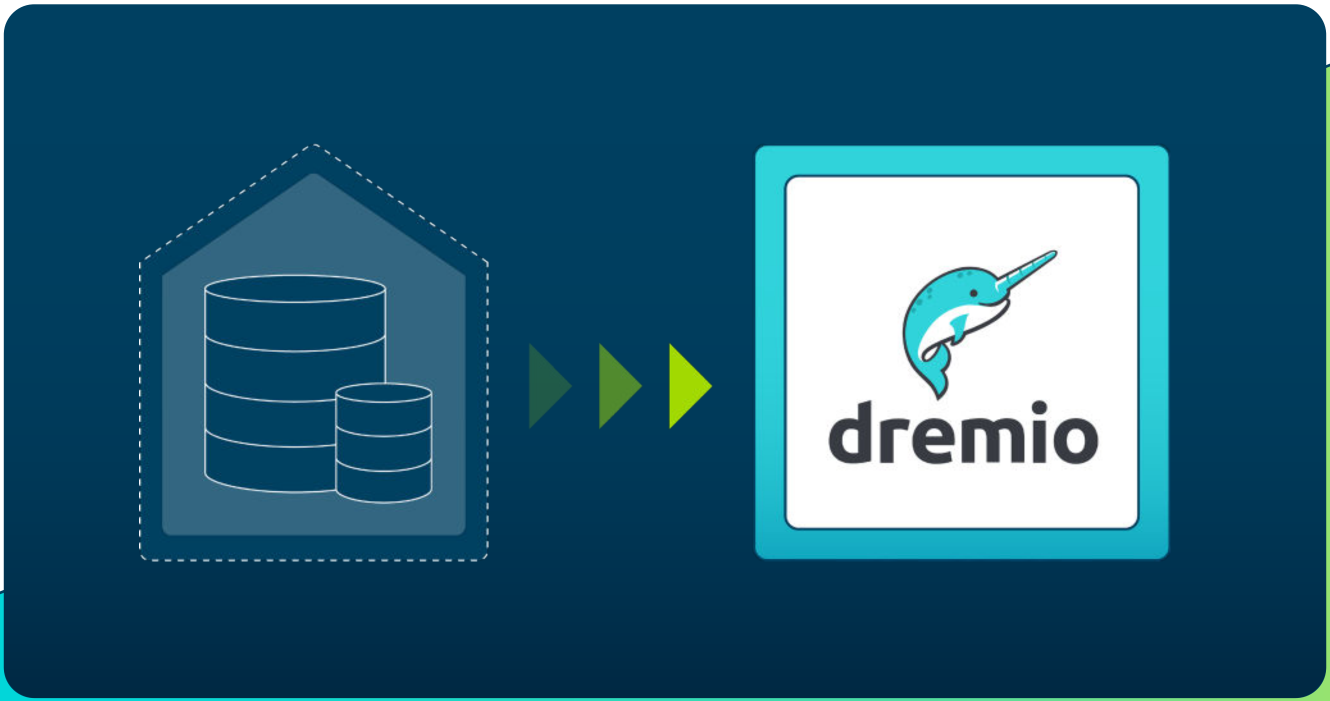


Table of Contents

Introduction 3

How We Got Here: The Evolution of Data Architectures 4

What is a Data Lakehouse? 6

Why Move to a Data Lakehouse? 7

The Data Lakehouse Architecture 8

Open Table Format 8
Data Catalog 10
Query Engine 10
Unified Semantic Layer 10

Why Dremio Cloud? 11

Unified Analytics 12
SQL Query Engine 13
Lakehouse Data Management 14

A Migration Approach 16

Phase 1 - Modernize Analytics Engine 17
Phase 2 - Unified Analytics 18
Phase 3 - Enterprise Data Lakehouse 19

Summary 21

Introduction

Data leaders are under pressure to empower their organizations by building data-driven customer experiences and improving operational efficiency with insights. To achieve this goal, they need to deliver unified self-service access to all of their data, so data consumers of all skill levels can easily leverage insights from a consistent, accurate, and up-to-date view of their data.

As a result of several decades of attempts to wrangle data volumes, organizations now struggle with siloed data across different cloud and on-premises applications and systems, and they must invest significant resources to extract, rebuild, and integrate their data to make it consumable by the business.

Modern data architectures leverage cloud object storage to manage the growing volume, variety, and velocity of data, and many organizations use a cloud data warehouse to make that data available for analytics. However, due to their reliance on proprietary formats, cloud data warehouses possess many of the same limitations in terms of performance and costs as legacy data warehouses, especially due to data movement and data copies.

A new architecture has emerged that addresses the deficiencies of the cloud data warehouse. The open data lakehouse architecture combines the flexibility and scalability of data lake storage with the analytics performance, data governance, and data management capabilities enterprises typically associate with the data warehouse, while eliminating data movement and data copies. Data remains in cloud object storage, easily accessible by the many tools most data teams and data consumers leverage.

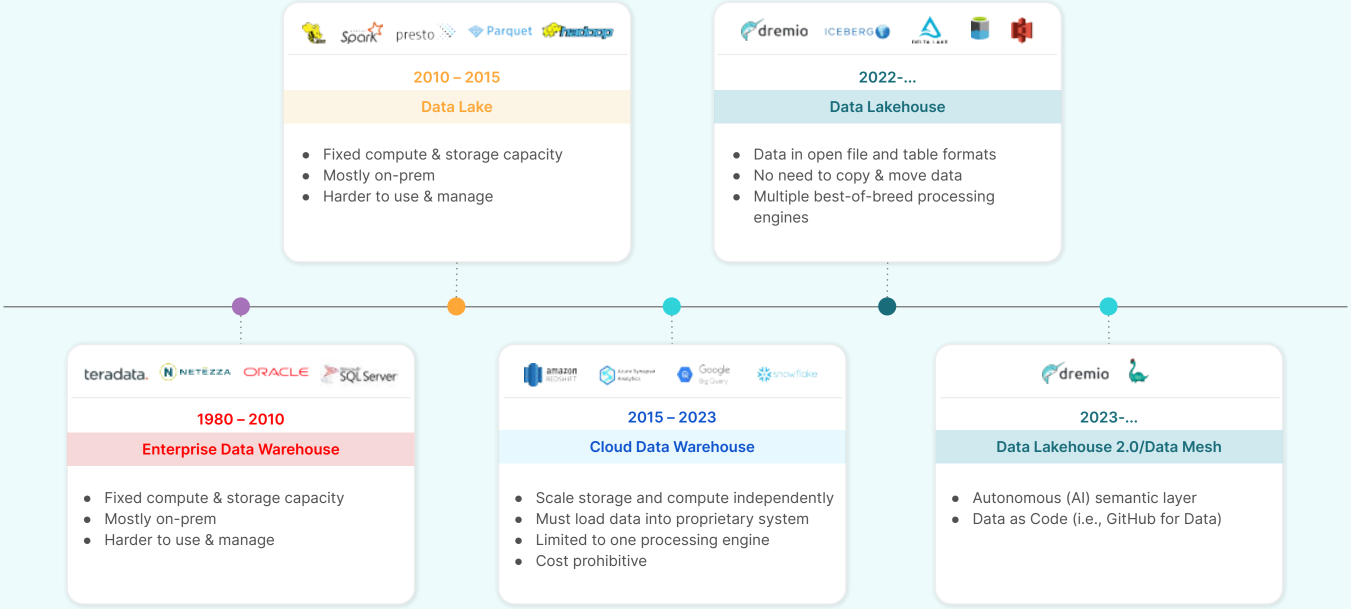
In this paper, we will discuss the evolution of data architectures, the advantages of adopting a data lakehouse strategy based on open standards, formats, and technologies, and a practical path for migration from a data warehouse to a data lakehouse.

How We Got Here: The Evolution of Data Architectures

The journey to the data lakehouse as the ideal architecture for self-service analytics for enterprises is the result of a series of technology advancements combined with the significant growth of data volumes.

The journey started in the 1980s as organizations adopted data warehouses as a central repository for storing, managing, and analyzing structured data primarily from business systems housed alongside it in the data center. Data warehouses were initially proprietary and appliance-based, and although they solved the problem at the time, they struggled to keep up with the growth in volume, variety, and velocity of data, and with demand for analytic insights closer to real-time.

Data Analytics - A History



1980 – 2010
Enterprise Data Warehouse

- Fixed compute & storage capacity
- Mostly on-prem
- Harder to use & manage

2010 – 2015
Data Lake

- Fixed compute & storage capacity
- Mostly on-prem
- Harder to use & manage

2015 – 2023
Cloud Data Warehouse

- Scale storage and compute independently
- Must load data into proprietary system
- Limited to one processing engine
- Cost prohibitive

2022-...
Data Lakehouse

- Data in open file and table formats
- No need to copy & move data
- Multiple best-of-breed processing engines

2023-...
Data Lakehouse 2.0/Data Mesh

- Autonomous (AI) semantic layer
- Data as Code (i.e., GitHub for Data)

In the mid-2000s, organizations began to adopt data lakes, first with Hadoop Distributed File Storage (HDFS) and then with cloud and on-premises object storage, to meet the need for cheap and efficient storage for growing data volumes. Even though a robust ecosystem of tools, including open source projects like Apache Hive and Apache Spark and commercial query engines, emerged to meet the demand for data warehouse functionality on data lake storage, data lakes never replaced the data warehouse for high-performance Business Intelligence (BI) and reporting. Instead, most organizations leveraged the data warehouse for data science projects.

Today, most customer and operational data lands first in a data lake, and cloud object storage is the ideal destination for its flexibility, scalability, and high availability. As a result, cloud data warehouses have emerged as a platform that delivers enterprise-grade BI and reporting on cloud object storage. However, the cloud data warehouse possesses many of the same limitations as legacy data warehouses.

Proprietary Formats & Vendor Lock-In:

In order to achieve the necessary query performance, cloud data warehouses rely on moving data into proprietary formats. As a result, data requires additional movement and transformation to make it consumable, and that data is not accessible by other tools and technologies.

Data Movement:

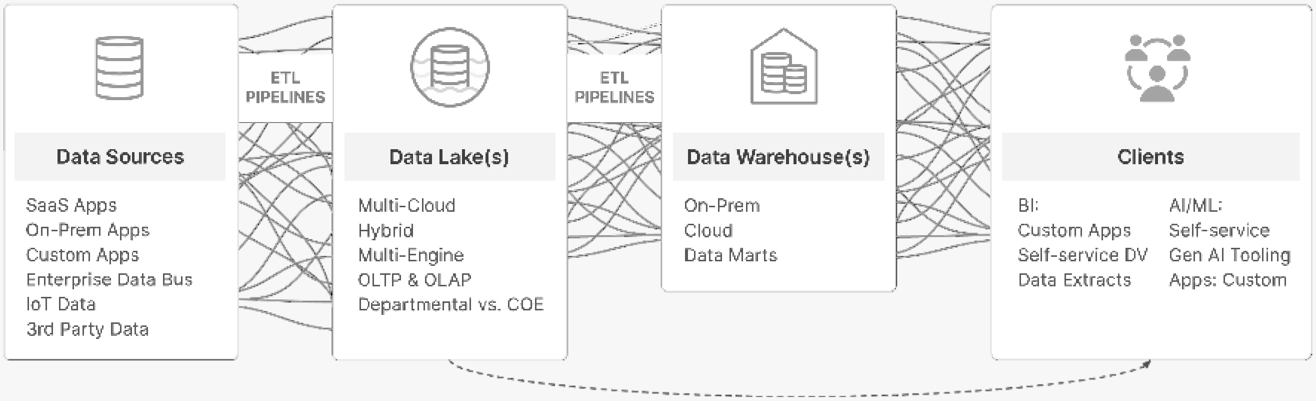
To get data into proprietary formats, data teams must build and maintain Extract, Transform, & Load (ETL) pipelines. These ETL pipelines are often manual and ad-hoc, and data teams must continue to manage them over time.

Data Copies:

Within data warehouse and BI environments, the creation of data copies such as BI cubes and extracts is often essential for optimizing performance or enabling self-service analytics. While these copies enhance query responsiveness and cater to specific reporting needs, they introduce complexities in data management. Each data copy represents another asset that teams must upkeep to synchronize with the source data, adding to the maintenance overhead faced by data teams.

The resulting architecture for most organizations looks something like this:

Data Lifecycle Remains Complex, Brittle and Expensive



Common enterprise data warehouse architecture

Data lifecycle and management remains complex, especially for large organizations
Duplicative copies, ‘expert’ ETL, “dark data”, governance complexity, not self service

Organizations maintaining this architecture face several issues:

It’s complex:

The data integration process starts by placing raw data into a data lake for storage and curation before copying it into the data warehouse. These pipelines and copies proliferate as the data lake becomes the first and primary storage destination for more customer and operational data, and data consumers experience bottlenecks for data access and performance. Additionally, organizations managing multiple data copies are at risk of inconsistent data, rising costs, broken pipelines, and more. The problem with this process is that it creates more ETL processes to manage. Engineers spend the majority of their time on pager ticket responsibilities, managing and fixing ETL pipelines, all while handling competing data requests from the business.

It’s inefficient:

Data consumers face bottlenecks for data access and query performance, both of which require intervention from the data team. The architecture is not self-service, and data consumers, especially non-technical analysts, can wait weeks or even months to get access to the data they need.

It’s expensive:

The proliferation of analytical tools and ETL processes can lead to an increase in the total cost of ownership. Different teams require their view of data with their preferred tool. It becomes difficult to self-service and report against distributed datasets with data sprawled across multiple platforms. Siloed data integration processes build reliances on OLAP cubes and data extracts. By the time central IT updates the data, business requirements may have changed, and the data is stale and can no longer be consumed.

The data warehouse, and its reliance on proprietary formats, is the primary culprit in these architectural inefficiencies. As companies consider re-platforming to reduce the rising cost of cloud data warehouses, they are encountering the same challenge they had migrating from legacy on-premises solutions: it is much more difficult to get data out than in.

Fortunately, a new architecture has emerged that delivers data warehouse performance and functionality while preserving an organization’s control over its data with open formats and technologies. That architecture is the open data lakehouse.

What is a Data Lakehouse?

The data lakehouse is an architecture that takes advantage of the scalability and flexibility of data lake storage while providing full data warehouse functionality, performance, and concurrency.
It eliminates complex, brittle ETL pipelines and proliferating data copies, which simplifies data management and dramatically reduces costs.

The data lakehouse is an approach to data management that combines the advantages of data lakes and data warehouses, addressing the limitations of traditional cloud data warehouses and data lakes, while providing a more unified, scalable, and cost-effective solution for analytics. Open table formats such as Apache Iceberg make it possible to leverage multiple tools and satisfy many use cases on your data, while still achieving high performance.

Why Move to a Data Lakehouse?

Here are some of the benefits of moving to a data lakehouse.

Eliminating data silos:

A data lakehouse consolidates all structured and unstructured data into one central repository. Maintaining a single copy of the data ensures consistency and accuracy across all analytic processes.

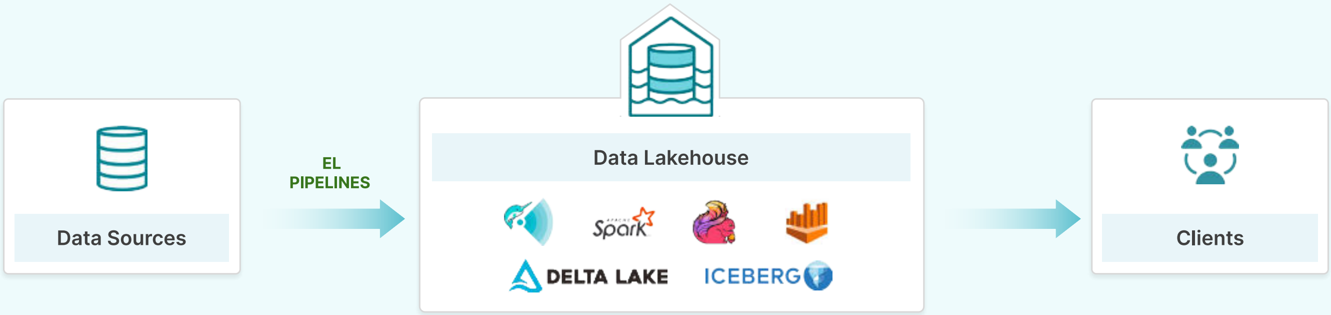
Improving the quality of data:

Open table formats in general, and Apache Iceberg in particular, supports schema evolution, so data schemas can be modified without disrupting existing data pipelines. This flexibility ensures that data consumers can work with evolving data structures without encountering compatibility issues, enabling seamless collaboration even as data schemas change over time.

Increased efficiency and productivity:

With a single copy of your data available on the data lake, the complexities of making and maintaining multiple data copies and ETL pipelines are no longer present. Data teams can spend their time on more high-value work, instead of responding to data access requests and performance issues.

Enterprises are Moving to a Lakehouse to Simplify



ETL to ELT

- Reduce complex transform pipelines in Java / Scala / Python (e.g., Spark)
- Move to SQL-based Transforms (DBT)
- Full transform lifecycle lives in the lake

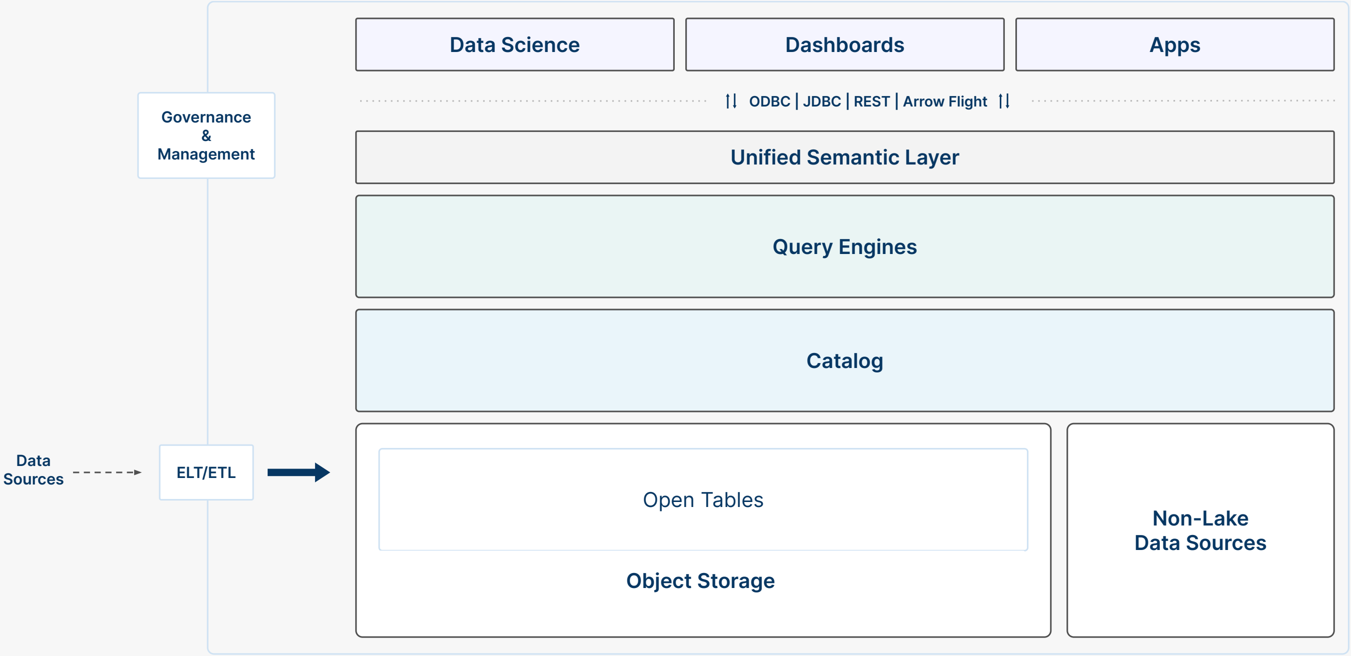
Lakehouse Advantages

- Open data and table formats
- Storage / compute separated, elastic SQL engine
- No Copy Architecture
- Full ACID Transactions, Time-Travel, Schema / Partitioning Evolution
- Compelling Economics

The Data Lakehouse Architecture

Organizations have been using query engines for read-only workloads on top of data lake storage for at least a decade now. To bridge the gap between a data lake and a data lakehouse, data teams need a technology stack that enables read and write workloads with ACID guarantees, simplifies data management and data access, and ensures high performance queries on massive volumes of data. The following is a review of the components of an open data lakehouse.

Core components of an enterprise data lakehouse



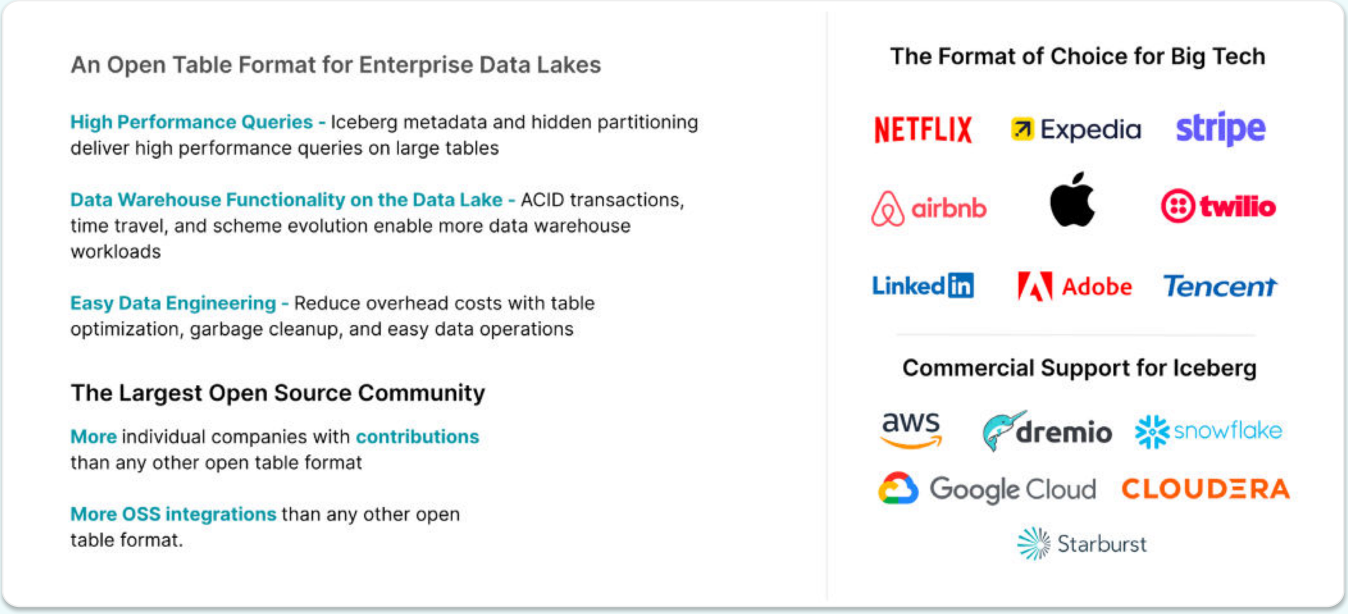
Open Table Format

Data lakes were initially designed as a cost-effective and flexible storage tier for data but they lacked the capabilities necessary to provide a way for users to analyze their data like the data warehouse did. Open file formats like Apache Parquet provide compression and performance benefits over traditional data formats like CSV and JSON for read-only workloads.

Open table formats such as Apache Iceberg, build on the advantages of file formats by bringing data warehouse functionality to the data lake. It offers full ACID guarantees that improve the consistency of the data, which is important when multiple engines are accessing the same datasets. Like tables in a traditional data warehouse, open table formats support important capabilities like schema evolution, partition evolution, time travel, version rollback, and table optimization.

Customers have several open table formats to choose from. Apache Iceberg is an open table format that originated with Netflix and is purpose-built for large enterprise data volumes. It features the broadest commercial and Open Source Software (OSS) ecosystem, the broadest community of contributors, and has the most momentum in terms of enterprise adoption.

Apache Iceberg



An Open Table Format for Enterprise Data Lakes

High Performance Queries - Iceberg metadata and hidden partitioning deliver high performance queries on large tables

Data Warehouse Functionality on the Data Lake - ACID transactions, time travel, and scheme evolution enable more data warehouse workloads

Easy Data Engineering - Reduce overhead costs with table optimization, garbage cleanup, and easy data operations

The Largest Open Source Community

More individual companies with **contributions** than any other open table format

More OSS integrations than any other open table format.

The Format of Choice for Big Tech

Commercial Support for Iceberg

Here are some of the capabilities that Iceberg delivers to data teams:

Benefits	What This Means
Schema Evolution	Add, drop, update, or rename column commands with no side effects or inconsistency.
Partition Evolution	Facilitates the modification of partition layouts in a table, such as data volume or query pattern changes without needing to rewrite the entire table.
Time Travel	Allows users to query any previous versions of the table to examine and compare data or reproduce results using previous queries.
Transactional Consistency	Helps users avoid partial or uncommitted changes by tracking atomic transactions with atomicity, consistency, isolation, and durability (ACID) properties.
Version Rollback	Corrects any discovered problems quickly by resetting tables to a known good state
Table Optimization	Optimize query performance to maximize the speed and efficiency with which data is retrieved.

Apache Iceberg delivers high-performance analytics, data warehouse functionality on data lake storage, and easy data operations.

Learn more about Apache Iceberg

Data Catalog

One of the advantages of the open data lakehouse is the ability to manage a single copy of the data that is accessible by multiple tools and engines, and the secret to enabling easy data access to a consistent view of the data is the data lakehouse catalog.

A data lakehouse catalog is the metadata layer and provides information about the tables in a data lake. The most commonly used catalog, which evolved out of a desire to manage HDFS data lakes, is Hive Metastore. Although it is a more mature offering, its architecture does not support large enterprise data volumes, and most Hive Metastore users experience significant performance challenges.

Cloud-managed offerings such as Dremio Cloud's lakehouse management service and AWS Glue provide a more modern approach compared to Hive Metastore. They can efficiently monitor tables and simplify queries by abstracting file paths, ensuring uniformity in data interpretation among readers and simplifying data management.

Query Engine

With an open data lakehouse architecture, customers get more flexibility to work with the data. Unlike a data warehouse, data stored in Iceberg tables is open and is accessible by multiple execution engines, such as Dremio, Spark, and Flink. These engines can operate directly on the same datasets with ACID guarantees, so data teams can use the best engine for each workload.

Unified Semantic Layer

Historically, copying data from the data warehouse into materialized views, data marts, and BI extracts was a common solution for providing performant access to the data for business users. However, this approach is not self-service, and it requires significant effort from the data team to build and manage this architecture, and it's expensive to maintain multiple physical copies of the data.

The open data lakehouse leverages a unified semantic layer that sits between the data store and consumption tools and provides business context to technical and non-technical consumers, so data analysts can work with the data without needing to understand the underlying physical data structure.

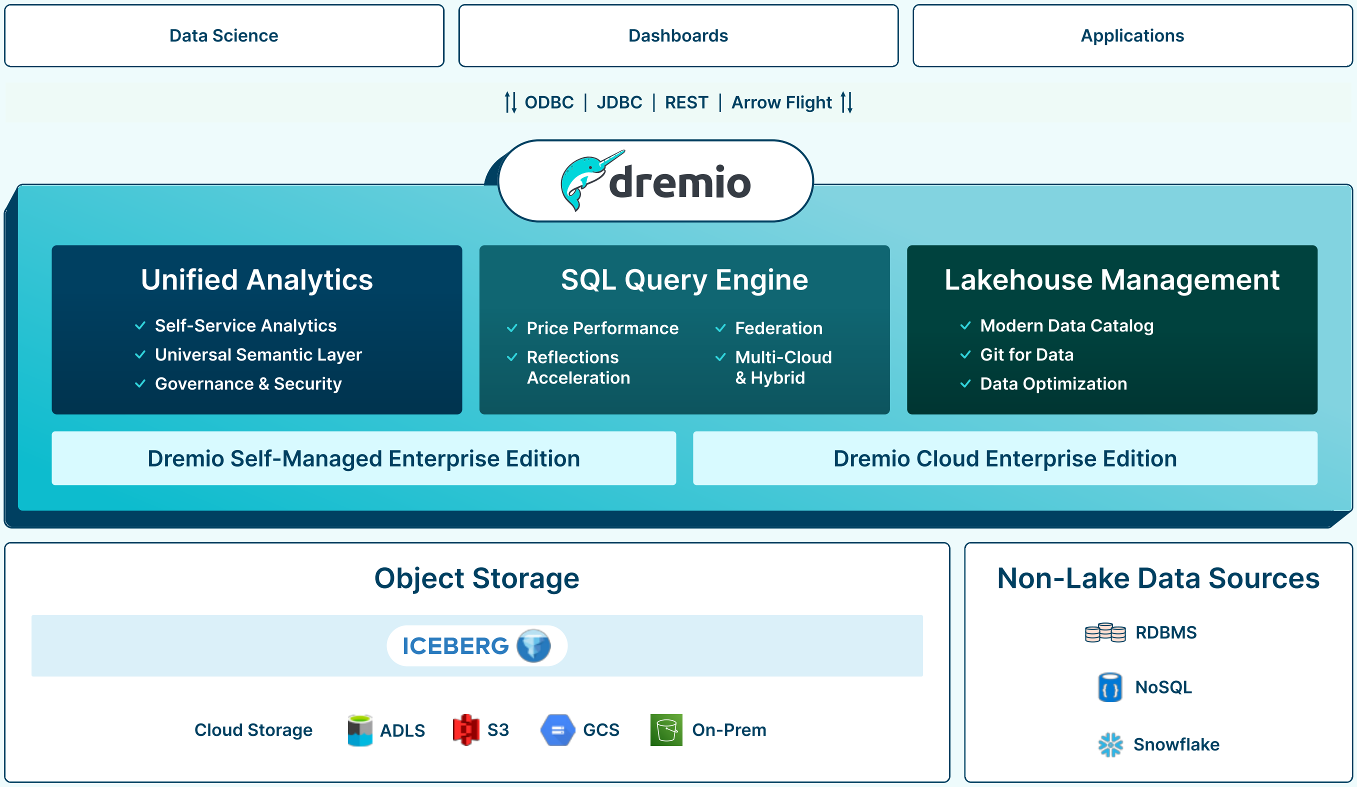
The semantic layer makes it easy to find and use all of an organization's data without creating physical copies.

Now that we have reviewed the components of an open data lakehouse architecture, we will discuss the capabilities Dremio Cloud provides across the technology stack.

Why Dremio Cloud?

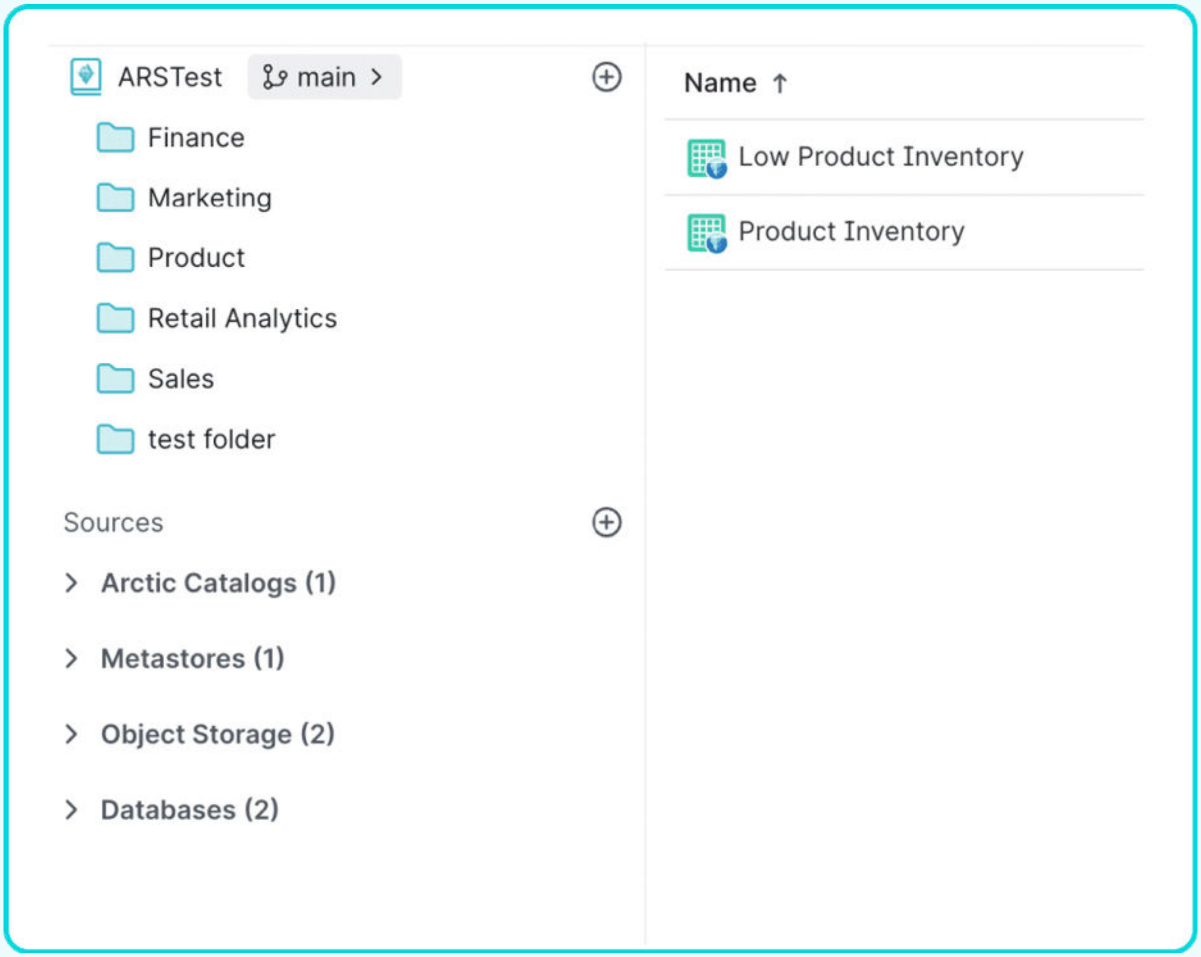
Dremio Cloud is a unified analytics platform that makes it easy to transition to an open data lakehouse architecture. Organizations use Dremio Cloud to deliver faster access to all of their data without complex ETL pipelines or data copies, to simplify data management, and enable high-performance analytics, all at a fraction of the cost of traditional and cloud data warehouse solutions. Dremio Cloud offers three services that enable an open data lakehouse architecture: unified analytics, a SQL query engine, and lakehouse data management.

Dremio Cloud Architecture



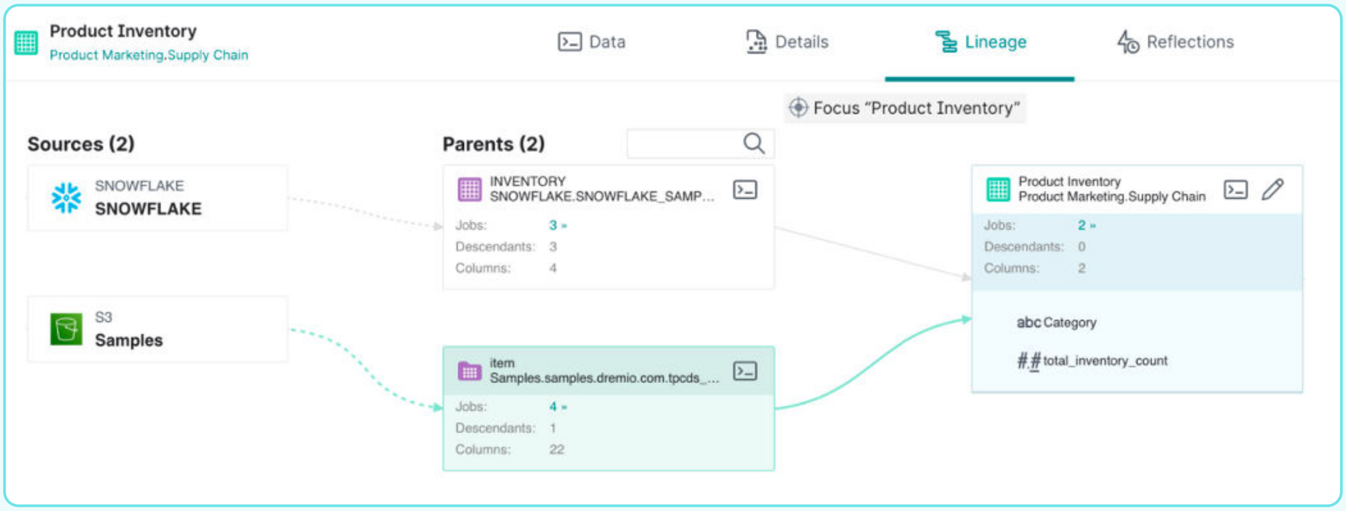
Unified Analytics

Semantic Layer: Dremio's universal semantic layer provides broad self-service access to all of an organization's data for analytics while centralizing data governance and security. Through data federation, users can search, construct, and distribute virtual data marts across multiple data sources without duplicating their data.



Dremio Cloud's universal semantic layer

Centralized data governance: Discover and understand all of the data in the cloud and on-premises with minimal data engineering overhead using data lineage. Empower data consumers to discover, access, and leverage data products and business context on their own with self-service capabilities like search, tags, and wikis that enable domain owners to provide business context to data products.



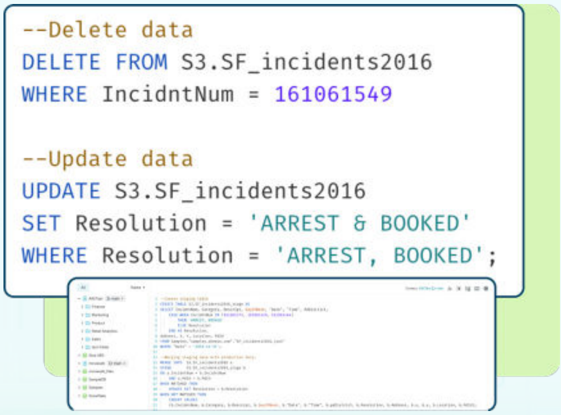
Discover and understand data with full business context

SQL Query Engine

High-Performance Query Engine: Dremio features an in-memory distributed SQL query engine based on Apache Arrow that is purpose-built for high-performance BI, interactive queries, and ad hoc exploration. Dremio also 's SQL lakehouse engine supports critical data warehouse operations like DML, DDL, schema and partition evolution, time travel, and more.

Price-Performance: Dremio's SQL query engine optimizes price-performance for every query. The multi-engine architecture enables organizations to isolate various use cases for performance predictability, and autoscale dynamically based on query workloads.

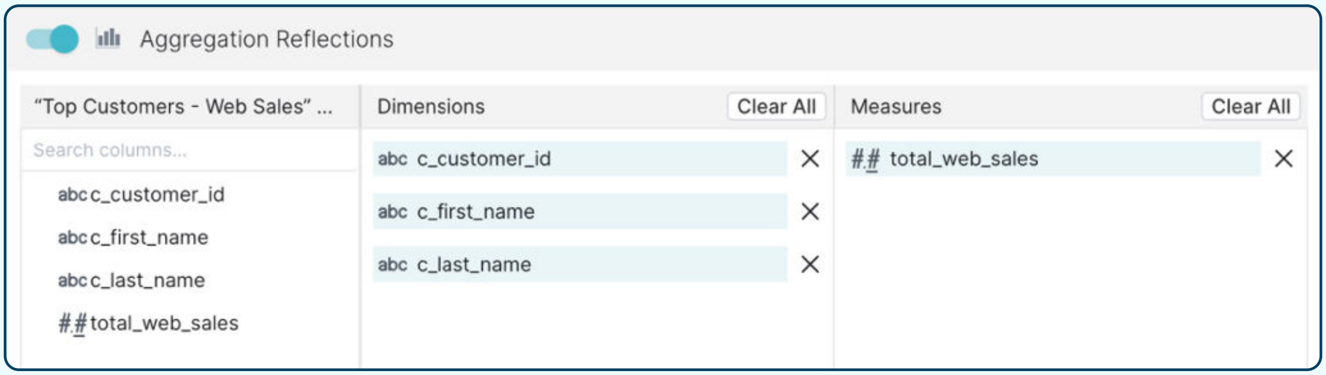
Query Acceleration with Reflections: Dremio uses **Reflections** to optimize queries across all data stores, accelerating performance and reducing the Total Cost of Ownership. Unlike materialized views, a single reflection can apply to many different use cases, so data engineers don't need to create new reflections. For example, if a user creates a reflection on a dataset that joins a fact table with multiple dimensions tables, Dremio can accelerate queries that include any subset of these joins. Reflections provide high performance without requiring multiple copies of the data.



```
--Delete data
DELETE FROM S3.SF_incidents2016
WHERE IncidntNum = 161061549

--Update data
UPDATE S3.SF_incidents2016
SET Resolution = 'ARREST & BOOKED'
WHERE Resolution = 'ARREST, BOOKED';
```

Dremio Cloud supports full lakehouse DML and DDL capabilities



Aggregation Reflections

"Top Customers - Web Sales" ...

Search columns...

abc c_customer_id
abc c_first_name
abc c_last_name
total_web_sales

Dimensions — Clear All

abc c_customer_id ✕
abc c_first_name ✕
abc c_last_name ✕

Measures — Clear All

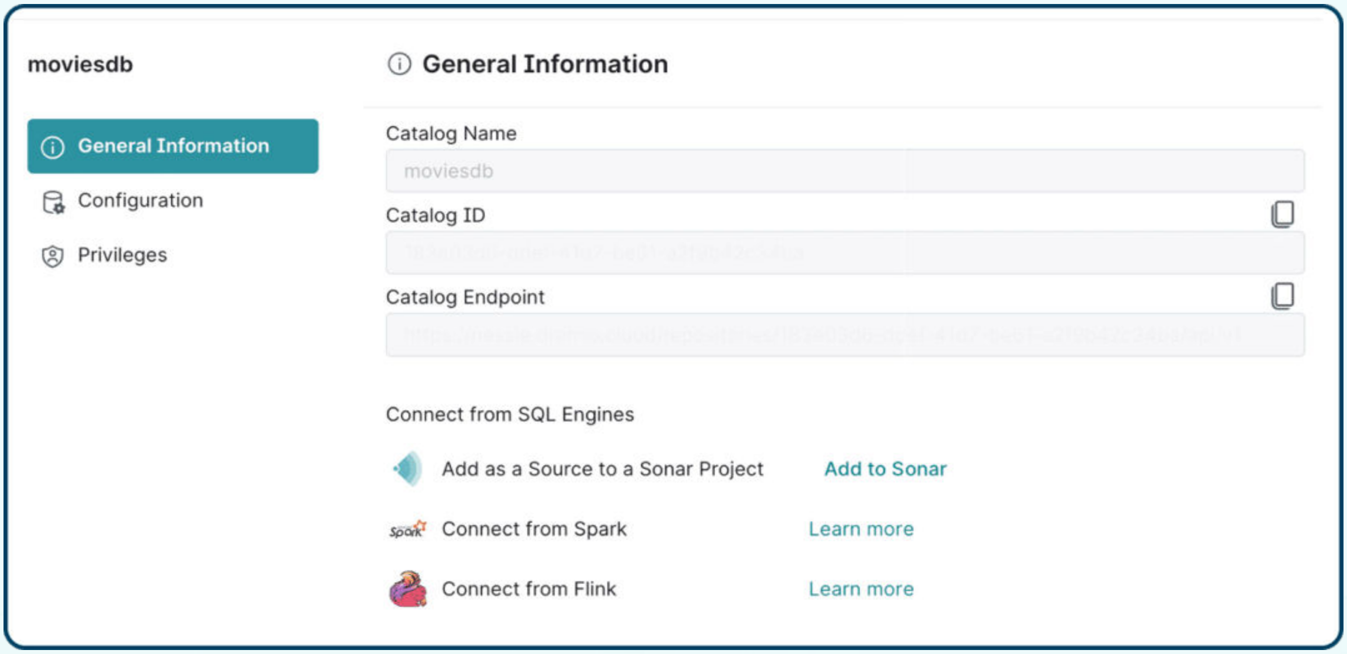
total_web_sales ✕

Accelerate workloads with Reflections

Lakehouse Data Management

Dremio Cloud provides an Iceberg-native lakehouse data management service that automatically optimizes Iceberg tables for high performance and storage utilization and builds on Iceberg's time travel capabilities to enable Git-inspired data versioning.

Lakehouse Catalog: Dremio Cloud offers a native Iceberg catalog. In addition to supporting BI workloads, engines like Spark, Flink, and Dremio's SQL engine can operate directly on Iceberg tables using commands such as INSERT, UPDATE, DELETE, AND MERGE.



moviesdb

General Information
Configuration
Privileges

General Information

Catalog Name: moviesdb

Catalog ID: [redacted]

Catalog Endpoint: [redacted]

Connect from SQL Engines

Add as a Source to a Sonar Project — Add to Sonar

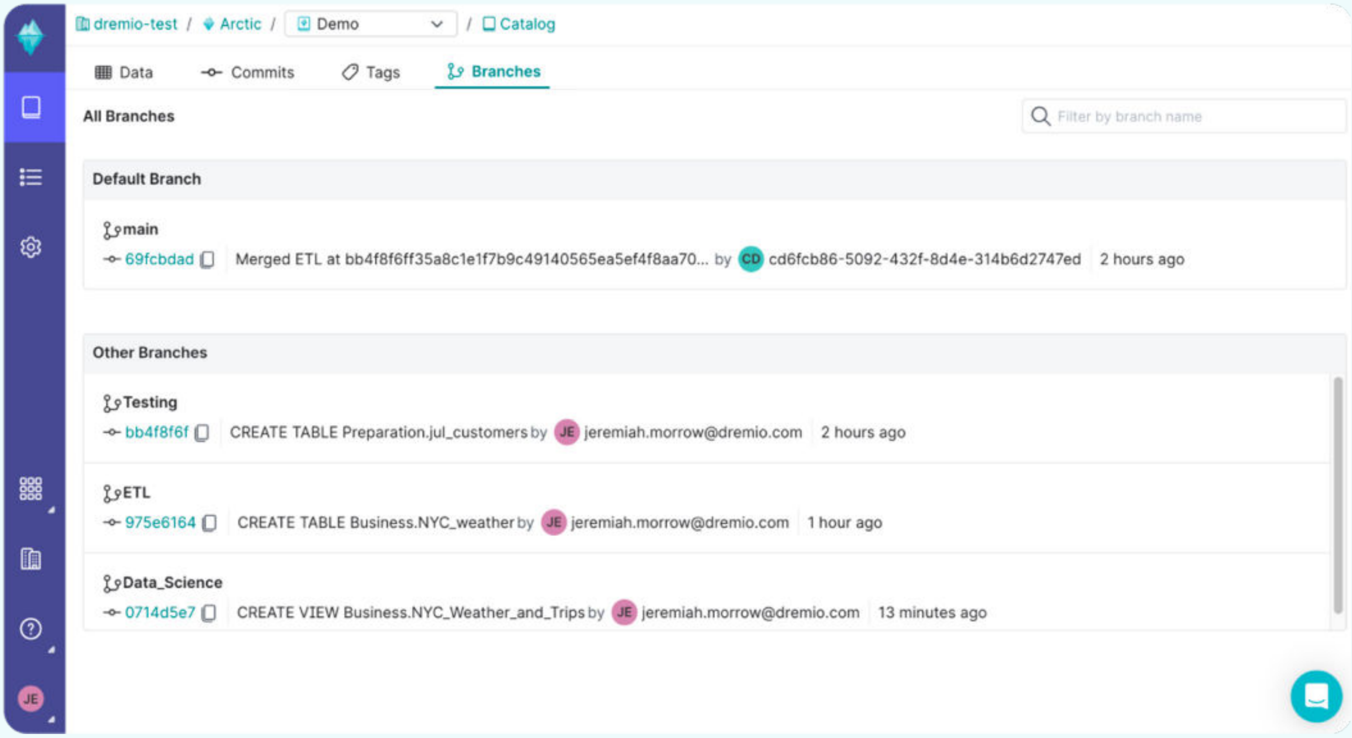
Connect from Spark — Learn more

Connect from Flink — Learn more

Open Iceberg data catalog

Automatic Optimization: Dremio automatically optimizes Iceberg tables with features like compaction, which rewrites small files into larger ones to improve performance, and garbage collection, which removes unused files to improve storage utilization. These features eliminate necessary but tedious data lakehouse management tasks.

Git for Data: Git for Data uses Iceberg's time travel capabilities to introduce catalog-level versioning. Using metadata pointers, users can create a branch, or a zero-copy clone, of their data catalog, so they can ingest data, test changes, and even experiment on production data, all in isolation, without impacting production users. Data teams can expose changes to data atomically through a merge, making it easier than ever to deliver a consistent and accurate view of the data.



All Branches

Default Branch

main
69fcbdad | Merged ETL at bb4f8f6ff35a8c1e1f7b9c49140565ea5ef4f8aa70... by CD cd6fcb86-5092-432f-8d4e-314b6d2747ed | 2 hours ago

Other Branches

Testing
bb4f8f6f | CREATE TABLE Preparation.jul_customers by JE jeremiah.morrow@dremio.com | 2 hours ago

ETL
975e6164 | CREATE TABLE Business.NYC_weather by JE jeremiah.morrow@dremio.com | 1 hour ago

Data_Science
0714d5e7 | CREATE VIEW Business.NYC_Weather_and_Trips by JE jeremiah.morrow@dremio.com | 13 minutes ago

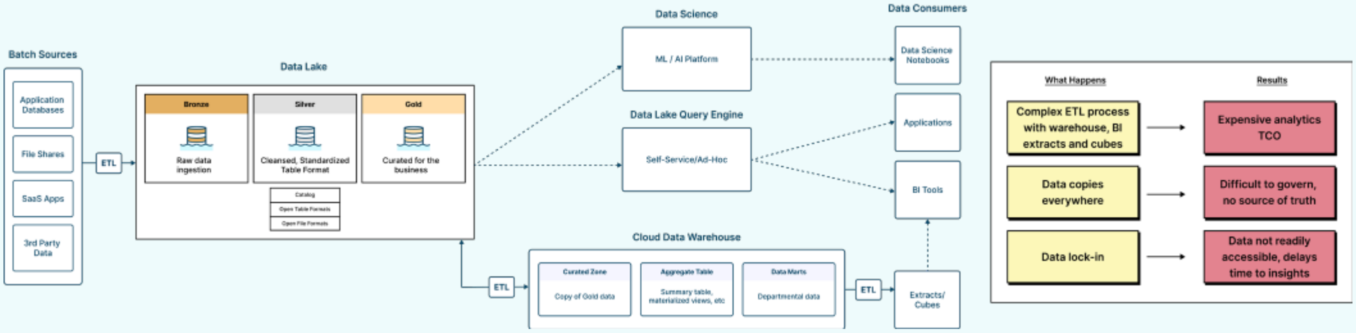
Data versioning and branching with Git for Data

A Migration Approach

While each organization's journey will be different, these are the broad phases most customers follow. The following graphic details a common cloud data warehouse architecture pattern for enterprise organizations:

Cloud Data Warehouse Architecture

Common two-tiered data architecture



Traditional Cloud Data Warehouse Architecture

Data teams load data from various sources into the data lake by ETL, change data capture (CDC), and streaming tools, in its raw form. Then, they perform a series of processing steps to prepare it for use by data consumers, including standardizing it and structuring it so end users can understand it and trust its quality. Broad consumption by a wide range of users requires even further curation.

Finally, the data is in a format and structure that all users should be able to leverage. However, the work is not done: traditional SQL query engines on data lake storage are suitable for non-interactive workloads, such as transformations. As a result, in order to provide access to the data for BI and reporting, data teams typically copy the data into a data warehouse.

Still, more work is required to ensure high performance. Data teams copy the data into aggregation tables. For certain use cases and end users, they need to create BI cubes and extracts, materialized views, and data marts. Each new copy is an asset the data team needs to manage, and each new data copy depends on the previous one, creating complex, brittle pipelines, overhead, and risk.

Phase 1 - Modernize Analytics Engine

A phased, workload-by-workload approach to migration shows immediate value and builds confidence in the project. Dremio enables customers to instantly connect to existing data sources, including data lakes and data warehouses, to start joining and querying data. Identify long-running queries or queries that require joins across multiple disparate sources, and connect to those sources with Dremio.

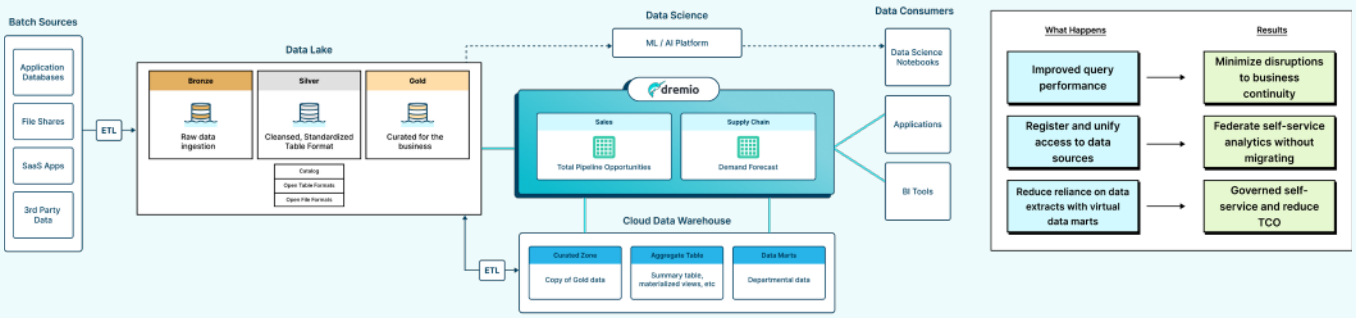
Using Dremio's semantic layer, register and unify access to data sources, and deliver self-service analytics. This makes it possible to define semantics such as KPIs and business logic consistent across all downstream apps, analytical tools, and users. From here, data teams can use Reflections to automatically accelerate and precompute data aggregates across data sources. Now, instead of copying data into BI extracts, Dremio accelerates queries regardless of where the data lives.

Repeat this process, and gradually reduce movement of data into a data warehouse. Dremio can now be rolled out to data consumers for direct, governed data access. These users now have the option to connect and use Dremio from their BI tools and SQL clients, especially for their ad-hoc analytic needs.

Learn how to modernize your BI workloads to Dremio's semantic layer

Phase 1: Modernize Query Engine

Federate and self-service data access across cloud data warehouse and data lake using Dremio's semantic layer



Phase 1 starts with modernizing the analytics query engine

Phase 2 - Unified Analytics

This phase involves offloading workloads completely from the cloud data warehouse onto the data lake and standardizing on Apache Iceberg. This can be a one-time load or an incremental batch load. Here are a few options.

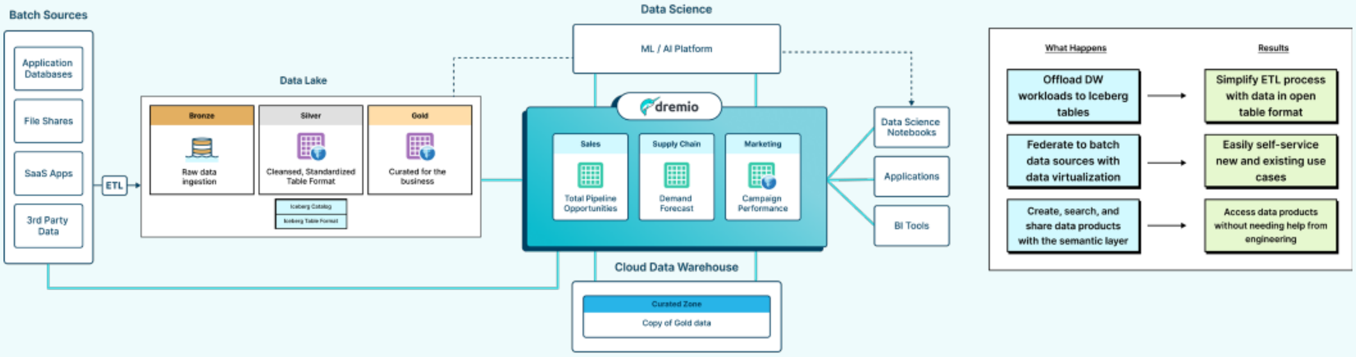
Export: The best way to extract data from the data warehouse is to use its native Export command if it is available. This option is generally multi-threaded and more optimized, as opposed to using an ETL tool. However, customers are limited by the capabilities of this Export command. For example, while it may be able to export to Parquet, it may not have the option to export to Iceberg in an Iceberg metastore. Data teams may still require an intermediary process after the extract, such as an ETL tool.

Modern ETL Tool: The second option is to use an ETL tool such as Apache Spark, one of Dremio's integration partners like Airbyte or Fivetran, or other vendors that specialize in ETL. While they may not be as fast as the native export option, these tools provide much more flexibility to make changes to the data while it is being extracted and to land them as Iceberg tables in an Iceberg metastore.

Use Dremio: Dremio offers COPY INTO, MERGE INTO, and INSERT INTO as options for loading data in a variety of formats into Iceberg tables. This option is especially good for bulk loads. Many customers have already standardized on Parquet, an open file format that delivers superior performance and compression over other file formats. Now, they can build on those benefits by easily moving from Parquet to Iceberg with Dremio.

Phase 2: Unified Analytics with Data Mesh

Deliver self-service data products with a no-copy data architecture



Building a unified analytics platform for data mesh

In this phase, customers also often adopt **data mesh with Dremio**. With data virtualization, organizations can use Dremio to easily provide self-service for new and existing use cases without copying data. Some organizations have additional regulations, and data must stay in the source system (on-prem/cloud relational databases, data lakes, etc). Regardless of where the data is, teams use Dremio's semantic layer to build, search, and share virtual data products without needing help from the engineering team.

How Amazon's Supply Chain Analytics team reduced the cost of their analytics architecture with an open data lakehouse on Dremio

Phase 3 - Enterprise Data Lakehouse

In this phase, organizations standardize on an enterprise data lakehouse based on Iceberg. Dremio's approach to the data lakehouse gives data teams a simpler way to deliver production data with no ETL pipelines and no additional physical copies of the data. This workflow leverages three important features of the Dremio open data lakehouse architecture:

1. GIT FOR DATA: SOFTWARE DEVELOPMENT PRINCIPLES FOR MANAGING DATA PRODUCTS

Iceberg uses metadata pointers to store a series of snapshots, which represent a view of the data lake at certain points in time. Dremio's Lakehouse Management Service leverages those snapshots to provide Git for Data functionality, which enables data teams to apply software development principles to the building and management of data products.

To make changes to a data product, data teams can use branching to create a zero-copy clone of their data catalog. They can create a branch for ingestion workloads, for atomic multi-table transactions, or for data science and experimentation, all in isolation from the production branch so data consumers are not impacted. After thoroughly testing any changes, they can expose those changes by merging the branch, or they can drop the branch if a merge is not necessary.

ETL branches serve as the primary method for landing new data in the data lakehouse. Create a branch, ingest new data with COPY INTO, MERGE INTO, or INSERT INTO, test the new data, and then merge changes into the production branch.

2. VIRTUAL DATASETS: LAST-MILE TRANSFORMATIONS AND JOINS OF PHYSICAL DATASETS

Dremio uses Virtual Datasets (VDSs) to enable both technical and non-technical data consumers to join data from disparate sources, including catalogs, relational databases, and other data lakes, without moving or copying the data. Users can also add columns like calculated fields to tables without changing the underlying physical dataset. They can save these views, and even share them with other users without creating a physical copy. VDSs are simply saved SQL statements.

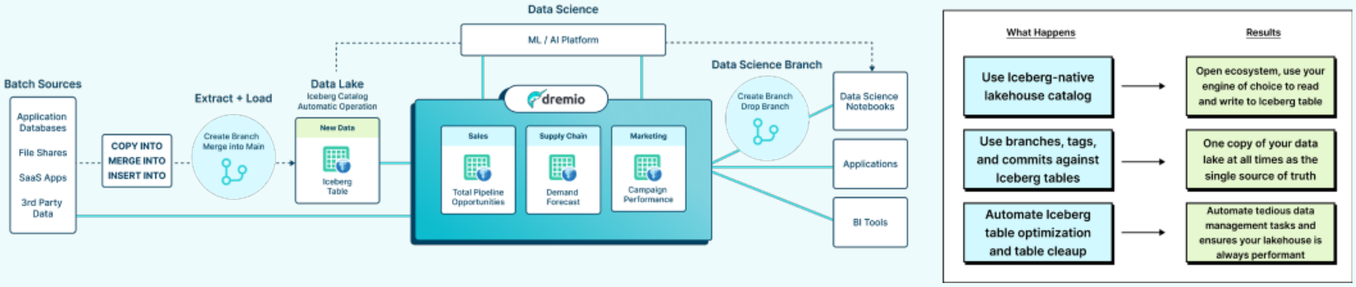
3. REFLECTIONS: HIGH PERFORMANCE FOR EVERY QUERY

Reflections are similar to materialized views, and ensure that queries read a table in an optimized way. Reflections ensure high performance even on complex queries with large volumes of data. Reflection Recommender enables even non-technical users to take advantage of Reflections by suggesting specific views, so the consumer does not need to know how to optimize a query.

Together, these features enable data teams to simplify data pipelines, ingest data one time in Iceberg tables, and deliver curated data products as Virtual Datasets powered by Reflections, with no additional transformations or data copies required.

Phase 3: Enterprise Data Lakehouse

Land data once into Iceberg table and simplify data lakehouse management with zero-ETL



What Happens	Results
Use Iceberg-native lakehouse catalog	Open ecosystem, use your engine of choice to read and write to Iceberg table
Use branches, tags, and commits against Iceberg tables	One copy of your data lake at all times as the single source of truth
Automate Iceberg table optimization and table cleaup	Automate tedious data management tasks and ensures your lakehouse is always performant

Zero-ETL enterprise data lakehouse

The Dremio Open Data Lakehouse: Zero-Copy, Zero-ETL Data Products

Let's take a look at an example zero-ETL, zero-copy ingestion workflow using the Dremio open data lakehouse:

STEP 1: CREATE A BRANCH OF YOUR LAKEHOUSE CATALOG

Use the CREATE BRANCH feature to create a zero-copy clone of your lakehouse catalog. You can name it "ETL" for this ingestion workload.

ETL branches serve as the primary method for landing new data in the data lakehouse. Create a branch, ingest new data with COPY INTO, MERGE INTO, or INSERT INTO, test the new data, and then merge changes into the production branch.

STEP 2: INGEST DATA

Use COPY INTO, MERGE INTO, or INSERT INTO to move data from Parquet, CSV, or JSON format into an existing Iceberg table in your ETL branch. Alternatively, if it is a new table, you can use CREATE TABLE first to make a new table.

STEP 3: OPTIMIZE AND TEST YOUR TABLES

Dremio Cloud automatically runs VACUUM, which removes unused manifest files, manifest lists, and data files, on an automatic schedule in the background. Dremio Cloud runs OPTIMIZE TABLE, which rewrites smaller files into larger files, on a schedule the user sets. VACUUM and OPTIMIZE TABLE keep queries performant while reducing storage utilization, while reducing the manual work required to maintain tables.

Data teams can test their data for quality and even run views and dashboards against the data in the branch to ensure the data has updated properly and nothing has broken. Branching replicates all tables and views within a catalog.

STEP 4: MERGE THE BRANCH

Once the team is confident in their changes, they can merge the ETL branch into the main branch. Changes are made atomically, so if users have a view of the data that depends on several tables where the data has changed, they will never see a partial or incomplete view of the data.

As a result of this workflow, the **data team ingested new data into the data lakehouse in isolation from their production view**, they optimized and tested the new data, and then they exposed those changes to their data consumers. They maintained only a single physical copy of the data, and data consumers always saw a consistent and accurate view of the data.

Learn how Maersk is building the next-generation data platform for unified analytics with Dremio - while on a budget.

Summary

The open data lakehouse is an architectural pattern designed for modern data volumes that land first in cloud data lakes. It simplifies data pipelines, reduces the time to insight, and dramatically reduces the Total Cost of Ownership of your data architecture. This playbook is a practical guide for modernizing your data warehouse to an open data lakehouse architecture on Dremio Cloud. The three phases offer a seamless transition from cloud data warehouses to an open data lakehouse.

- **Phase 1:** The first phase starts with modernizing the query engine for self-service analytics, without migrating any data. Organizations will reduce reliance on BI extracts and OLAP cubes while simplifying their ETL processes.
- **Phase 2:** Data is offloaded to Apache Iceberg and organizations start to adopt an enterprise data mesh for unified analytics, regardless of where the data resides.
- **Phase 3:** Standardization of open table format (Iceberg) and using Dremio Cloud for a zero-ETL data lakehouse architecture. The Iceberg data catalog and Git-for-Data capabilities creates an open data architecture, which makes data engineering easy and automates lakehouse data management operations.

Dremio Cloud is free to try. Sign up and start accessing, joining, and querying data from cloud data lakes and other data sources today. **www.dremio.com/get-started**

For more information on Dremio's approach to unified data access, please see the following resources:

- **Semantic Layer Best Practices**
- **Modernize BI workloads to Dremio's semantic layer**
- **Modernizing legacy Hadoop data lakes to the data lakehouse**
- **Delivering unified data access with data mesh**

About Dremio

Dremio is the easy and open data lakehouse, providing self-service analytics with data warehouse functionality and data lake flexibility across all of your data. Use Dremio's lightning-fast SQL query service and any other processing engine on the same data. Dremio increases agility with a revolutionary data-as-code approach that enables Git-like data experimentation, version control, and governance. In addition, Dremio eliminates data silos by enabling queries across data lakes, databases, and data warehouses, and by simplifying ingestion into the lakehouse. Dremio's fully managed service helps organizations get started with analytics in minutes, and automatically optimizes data for every workload. As the original creator of Apache Arrow and committed to Arrow and Iceberg's community-driven standards, Dremio is on a mission to reinvent SQL for data lakes and meet customers where they are on their lakehouse journey.

Hundreds of global enterprises like JPMorgan Chase, Microsoft, Regeneron, Maersk, and Allianz Global Investors use Dremio to deliver self-service analytics on the data lakehouse. Founded in 2015, Dremio is headquartered in Santa Clara. CNBC recognized Dremio as a **Top Startup for the Enterprise** and Deloitte named Dremio to its **2022 Technology Fast 500**. To learn more, follow the company on **GitHub**, **LinkedIn**, **Twitter**, and **Facebook**, or visit **www.dremio.com**.