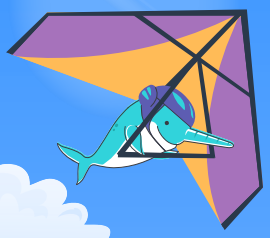
dremio

dremio.com Get Started



WHITEPAPER

Dremio: The Path to Self-Service Analytics on the Data Lake

dremio

Table of Contents

Introduction

Over the past several decades the amount of data being generated and stored has grown exponentially, and this growth has been further accelerated as new data sources (like social media and internet-connected devices) have come into existence. To keep pace with this continuing explosion in data creation and collection, the systems designed to store and derive value from this data have been forced to continually evolve through a series of paradigms. Every two to three order-of-magnitude increase in data volume and the related increase in user and application scale breaks the existing paradigm, and a new one emerges.

Data storage and query architectures have evolved from straightforward OLTP databases (where transactions and reporting were done from the same database), to relational data warehouses (offloading query and BI workloads to separate systems, fed from transactional databases), to a combination of "big data" systems such as Hadoop (in an attempt to store all the data and make sense of it later) with data warehouses to serve a subset of the data for reporting purposes.

With the rise to prominence of public clouds, with their nearly limitless cheap storage and decoupling of storage and compute, the next step in this evolutionary journey has been to combine the cloud data lake with a cloud data warehouse to provide a more-scalable version of "store everything, load the important parts into a data warehouse for analysis" model.

While offering some key advantages, this approach does not address the complexities that come with determining the subsets of data to be loaded into the warehouse, and the development and maintenance of the complex ETL pipelines required to load the data and keep it fresh. Nor does this approach do anything to address the proliferation of data copies required downstream of the data warehouse, such as BI extracts and caches required to provide the interactive query experiences that users expect.

The next logical step in the evolution of analytics and BI is the capability to directly query the vast amounts of data on the data lake, without the need to first load it into a data warehouse, and to provide sufficient query performance to the end user so that additional downstream copies, such as aggregation tables, cubes, and extracts, are no longer required.

The Dremio SQL lakehouse platform is the next step in this evolution, ushering in a new paradigm in working with data in the lake. Dremio connects BI users and analysts directly to data where it lives, eliminating complexity while providing the performance, ease of use, and security that end users and organizations require.

Here we present a roadmap of Dremio adoption, beginning with a deeper look at the data landscape as it exists today in many organizations, then walking through a stepwise journey of how many organizations today are adopting Dremio. At each step, we'll highlight the benefits gained, as well as the complexities and risks reduced as workloads are migrated from traditional systems to Dremio.

Dremio as the Path to Self-Service Analytics on the Data Lake

We begin with a review of the journey many organizations have taken to arrive at their existing traditional architectures; from there we will explore a frictionless path to introducing Dremio into the environment, progressively migrating more workloads to Dremio while maintaining coexistence and interoperability with existing systems and technologies, and finally arriving at a state where the maximum value is achieved from the Dremio platform.

While this journey will progress differently within each organization, for illustrative purposes we take three "snapshots" along a typical Dremio journey, which we'll call "Phases of Dremio Adoption."

Phase 0 – The Traditional Architecture

Let's begin by reviewing a prototypical architecture adopted by many organizations to provide self-service analytics today, illustrated below in figure 1. Here we are focused primarily on business intelligence and SQL analytics use cases; components related to streaming and data science have been greatly simplified in the diagram.

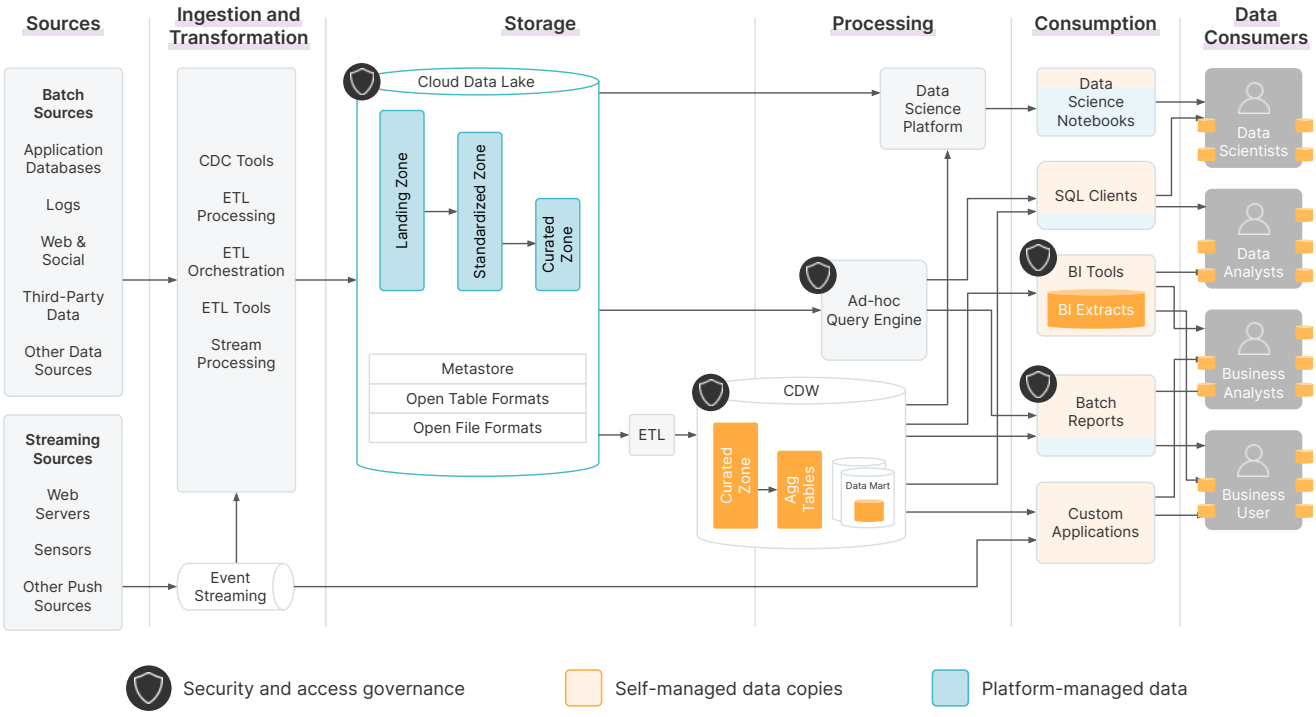


Fig. 1 Phase 0 – Traditional Architecture

In this diagram and those that follow, it's important to call out the distinction between "self-managed" data copies, which are colored in orange, versus platform-managed data copies, indicated in blue.

Self-managed copies are typically created by departments or individual users in an attempt to achieve better BI query performance, and in all cases they are derived from the canonical platform-managed datasets. While often successful in achieving query performance goals, self-managed data copies have numerous downsides; they are spread across many different systems (including end-user laptops), require additional processes and tooling to keep up to date, and can be very difficult to secure (as copies of data are usually not governed by the same access control mechanisms as platform-managed data is).

As we step through the phases of evolution toward a fully managed self-service analytics platform, note the quantity and distribution of self-managed data copies in blue versus platform-managed data in orange on the diagram.

Let's walk through how most organizations have arrived at this architecture.

The users on the right (data consumers) need to perform analysis and gain insights from the data on the far left to make business decisions and provide the business with value. In order to provide those users with data, we need to get the data from the sources into a place and form so that users can access the data in a way that enables them to ask their business questions.

1. **Ingestion and transformation** – Data from the sources generally first lands in the data lake by some combination of ETL tools, CDC tools, and streaming tools. It's then written to the landing zone in its raw form for ease of ingest as well as data retention purposes. Then, it is taken through a series of processing steps to get the data into a structure that can be used and understood: first getting the data in a standardized form so users can understand the structures and trust the data quality, then further curating it so additional users who aren't as familiar with the data can leverage it, as well as applications who need a custom view of the data. Security and governance is applied to the data in the data lake.

2. **Ad hoc query engine** – At this point, the data is in a form in the cloud data lake that all users should be able to leverage. Typically, companies provide users with a SQL engine to query the data in the data lake. However, most query engines struggle to support the majority of BI workloads for performance or ease-of-use reasons, so these query engines are generally only used for ad hoc workloads by more technical users — workloads where response times of many seconds to minutes are acceptable. In the past, the only way to address the limitations of ad hoc engines has been with self-managed data copies.

3. **Cloud data warehouse** – To address the BI workloads that traditional query engines can't support, organizations have generally had to turn to data warehouses. For these workloads, data engineering needs to spend significant time to write and operationalize ETL jobs to copy the data into a cloud data warehouse (CDW). However, CDWs are cost prohibitive, so not all of the data is loaded into the CDW — generally sacrificed are data granularity (i.e., aggregation), amount of history (e.g., six months instead of five years), and/or number of datasets. Often, the CDW isn't performant enough on the granularity of the data loaded into it for some workloads, so ELT jobs have to be written to copy and transform the data into aggregation tables. Data engineering must then manage these jobs on a perpetual basis. Duplicate security and governance rules need to be implemented and managed in the CDW.

4. **Data marts** – In addition, different business units need to view the data differently for them to make best use of it, so data engineering needs to collaborate with each business unit to write ELT/ETL jobs to copy and transform the data into data marts. Data engineering must then manage these jobs on a perpetual basis. Duplicate security and governance also need to be implemented and maintained for the data copied into these data marts.

5. **BI extracts** – Even with all of this effort, many BI dashboards and reports still can't get the performance they need, so data engineering and/or data SMEs need to coordinate with the people building the dashboards and create and operationalize the necessary BI extracts to provide that performance. Duplicate security and governance also needs to be applied in the BI tool.

6. **Local copies** – There are still many people who have business questions where the response time in the system is too slow for their iterative analysis and/or a single system doesn't have all of the data they need. To fulfill their needs in these situations, people copy data out of the systems and onto their laptops. Now that they have data on their laptops, all security and governance goes out the window.

Let's look at the impact of this traditional architecture on the business through the lens of a seemingly simple request — adding a column to a BI dashboard. A lot behind the scenes is required to make this happen:

1. The data consumer makes the request to the group that owns the dashboard to make the change.
2. However, this column is not in the dataset this group is getting from their data mart. So, they make a request to the team that owns the data mart.
3. The team that owns the data mart finds that it's not in the data they're getting from the data warehouse platform team, so they make a request to that group.
4. The data warehouse platform team discovers that this column is not in the data they're getting from the ETL jobs that copy the data from the data lake either.
5. So, ETL job changes are made to include the additional column from the data lake. This potentially requires having to create a duplicate table in order to not impact current usage of the table and to do a blue/green migration, so the extra column is in the data warehouse.
6. Then, the jobs that populate the data marts need to be modified in the same way to provide it with the new column.
7. Then, it's handed off to the BU team that owns the data mart.
8. Then, it's handed off to the team that owns the dashboard, who then go through the development -> QA -> UAT process before hopefully promoting it to production and fulfilling the business user's request. However, if there turns out to be an issue with the data in the column or it wasn't quite what the business user who made the request needed, the whole process needs to be repeated!

All of this inter-group coordination, modification of ETL jobs, and creation and management of data copies results in a very long time to value. For a Fortune 10 high-tech company we work with, this process, for a small change like this to a report for their CEO, took three weeks with this architecture.

This approach results in numerous challenges:

Slow turnaround time for changes

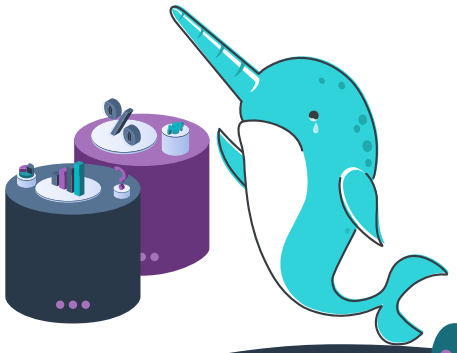
- With many report, dashboard, and application changes requiring updates to the BI extract data copy, data mart copies, data warehouse copies, and ETL processes, multiplied by the countless number of reports used throughout an organization, data engineering's backlog grows tremendously and causes long delays in fulfilling answers to business questions.
- As mentioned above, some organizations we work with report that even minor dashboard changes can take three to six weeks to address using this architecture.

Data size limitation

- The user has access to much less data than they often need, both in terms of the granularity of the data (e.g., month versus day) as well as the history of the dataset (e.g., one month versus three years).

Dashboard response time slows as data size and complexity grows

- This frustrates users and lowers the number of questions data consumers are able and/or willing to ask.



Data drift issues

- The data copies in the data warehouse, data marts, BI extracts, and local copies on laptops cause issues with data drift.
- A common issue we hear from organizations is that with this architecture and approach, two users can often come to a meeting with two different numbers to the same question.
- This is in addition to the quieter impact of when someone doesn't even know the data is out of drift, making their decisions on bad data, resulting in a decision that may not be known to be wrong for days, weeks, or months.

Regulatory compliance issues

- With the proliferation of self-managed data copies being managed by brittle ETL scripts and a kluge of different scheduling, monitoring, and management tools, it's very difficult to ensure regulatory compliance.
- In fact, with the local copies people download to their laptops, it's near impossible to remain fully compliant.

Higher security risk

- With data copies in so many different systems, the surface area of where security and governance need to be applied, maintained, and monitored is much larger.

Operational cost

- Data engineers have to create and manage the ETL pipelines, data warehouse structures, and BI extracts, increasing operational costs.

Infrastructure cost

- You have to pay extra for ETL compute resources, data warehouse compute, data warehouse data storage, data mart data storage, and BI extract resources.

BI extracts are exclusive to that BI tool

- Users who use other BI, data science, and SQL client tools in the organization aren't able to leverage the optimization, and instead have to create their own optimizations, further exacerbating data drift, regulatory compliance, and infrastructure cost problems.
- Even if the organization uses a single BI tool today, you're locked in to that dashboarding tool, prohibiting the option to allow users the freedom to choose other BI tools or migrate to another BI tool in the future.

Given these disadvantages, why have variations of this same architecture been adopted by so many organizations today? The reason is simple: Given the tools and technologies available, this approach has been the only way to meet business needs for high-performing analytics, and to address the scale of modern data and the requirements for interactive query performance across broad user populations.

Fortunately, Dremio bypasses the limitations of the traditional tools, resulting in greatly improved time to insight, data democratization, and increased business value obtained from your data.

With Dremio, migration is easy. Dremio's ability to access data where it lives — in multiple formats, sources, and catalogs — allows for a smooth adoption curve that is frictionless to implement, drives value right away, and scales as you grow over time.

Dremio as a Solution – The SQL Lakehouse Platform

Before jumping into the journey, let's briefly cover why Dremio avoids the limitations of traditional tooling. Dremio is a SQL lakehouse platform that enables high-performing business intelligence (BI) and analytics directly on data lake storage. Dremio's performance capabilities and semantic layer provide sub-second response times and self-service analytics directly on data lake storage, without proprietary data warehouses or the creation of cubes, aggregation tables, and BI extracts. Dremio's SQL lakehouse platform simplifies data engineering by eliminating the need to copy and move data to numerous systems to obtain performance, while providing flexibility and control for data architects and data engineers, and self-service for data consumers.

Some of the key benefits of Dremio include:

- **Centralized, consistent data semantics**, providing a consistent view of data across all users and tools
- **Significant reduction in data copies**, helping to reducing the overall cost and complexity involved in building the data platform
- **Lightning-fast analytics**, with the Dremio engine powered by Apache Arrow, which accelerates dashboard queries up to ~100x (achieving sub-second response times), while providing significant savings on compute
- **Elastic, multi-engine architecture** provides the ability to scale infinitely while isolating workloads from each other
- **Significant reduction of infrastructure and compute costs**, through highly efficient vectorized processing for efficient CPU utilization during query processing, and automatic start/stop of compute resources
- **Centralized security and governance**, made stronger through the elimination of self-managed copies of data

To learn more about how Dremio delivers on this promise, we recommend further reading here.

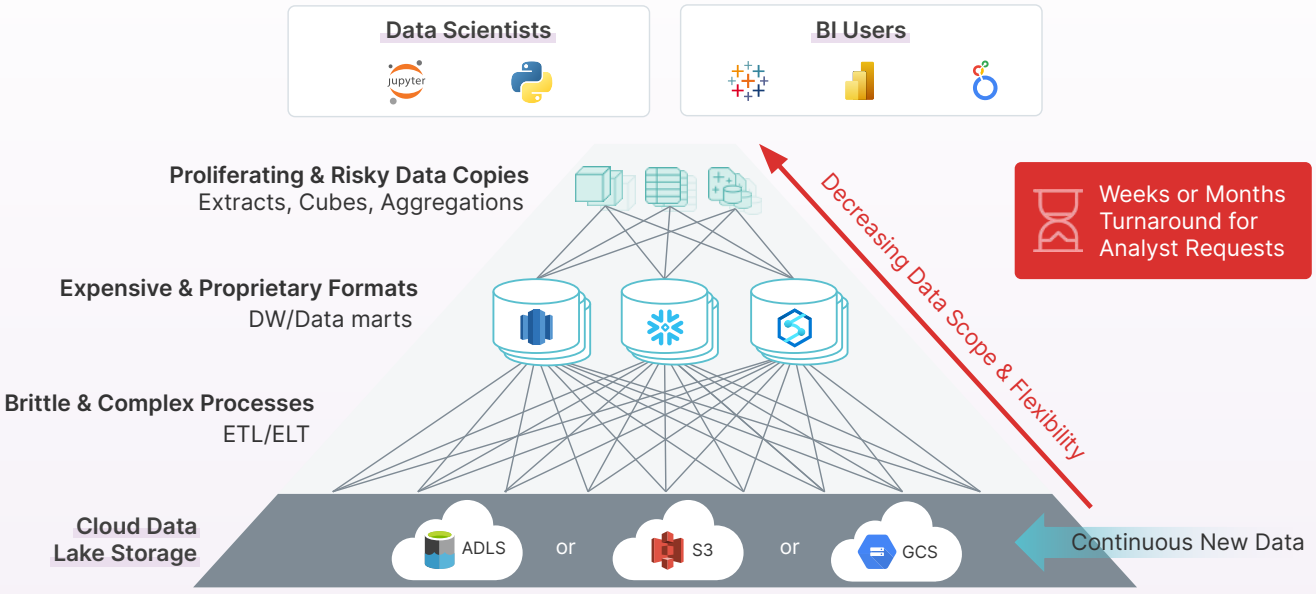


Fig. 1.1 A simplified view of the traditional architecture

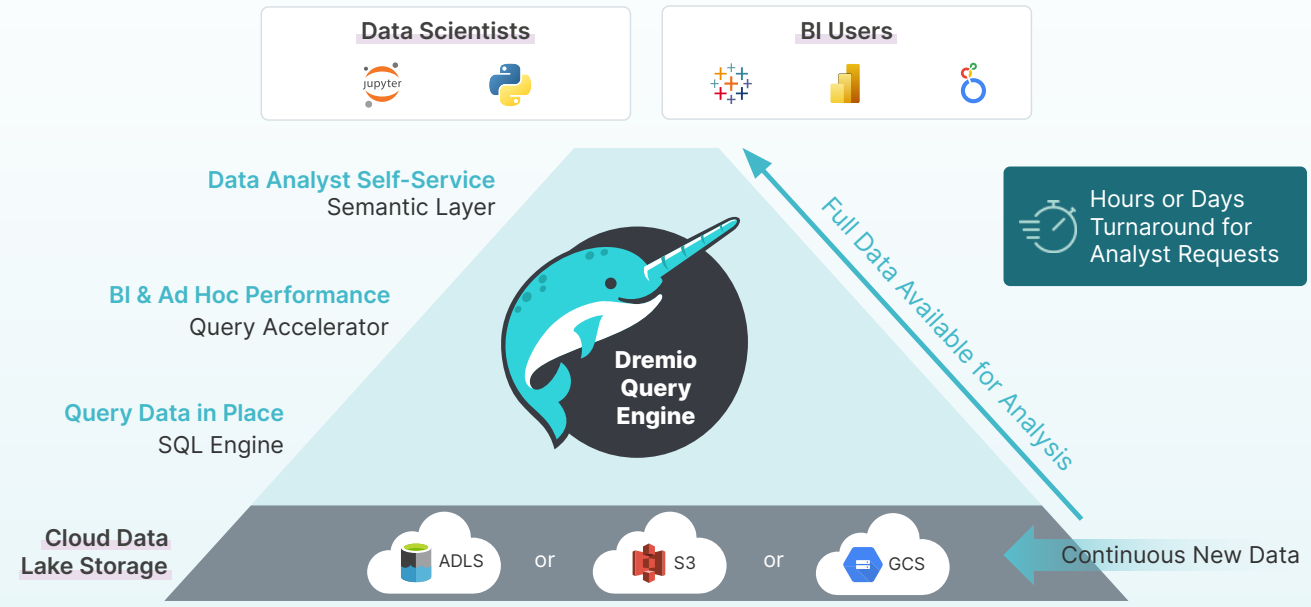


Fig. 1.2 Dremio as the solution

Phase 1 – Introduction of Dremio and Quick Wins

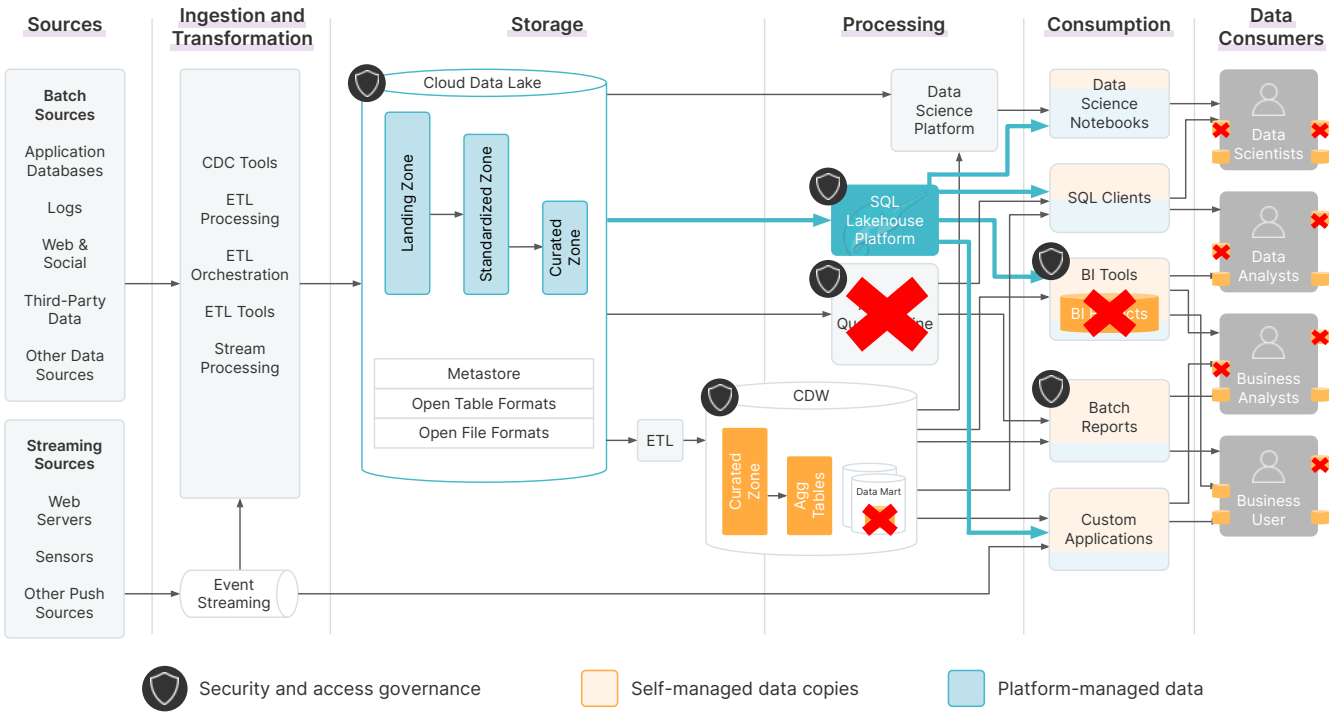
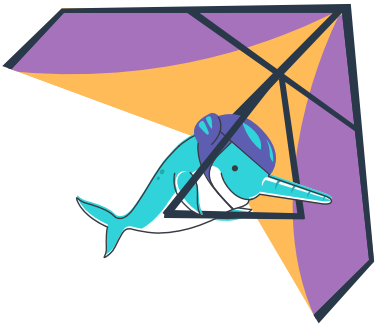


Fig. 2 Phase 1 – Introduction of Dremio



Getting started with Dremio is a simple process which doesn't require changes in the existing environment — simply install Dremio and connect it to your data lake. There's no requirement to decommission components or systems in your existing environment to start getting value from Dremio.

The initial step is to identify a set of use cases you want to migrate first for quick wins. This is an iterative process that repeats, first on a smaller number of use cases, then gradually more and more use cases and workloads. There's also usually an assessment of a broader set of use cases and workloads, which are prioritized according to the business impact of the downsides of its current implementation and the effort required to migrate.

In parallel, Dremio is rolled out to users for direct access. These users now have the option to connect and use Dremio from their BI tools and SQL clients for their ad hoc needs. Once users experience the improved ease of use and performance, they quickly switch to only using Dremio for ad hoc access to data. Once this happens, you're able to retire your old ad hoc query engine, saving costs and complexity.

Many interactive workloads that used to run in the CDW can be migrated to run directly on Dremio, which results in reducing the overall footprint of the CDW. Many organizations choose to start with BI workloads and a few of the CDW workloads initially, then evaluate the results of the migration to ensure Dremio is providing the value they expect before migrating even more workloads.

Because Dremio delivers interactivity and different views of the data to different business units virtually instead of having to rely on self-managed data copies, data mart workloads are migrated to Dremio, eliminating the need for self-managed data copies and the downsides that result from them.

More low-hanging fruit workloads are dashboards that currently require BI extracts. Data teams can eliminate BI extracts as these dashboards now provide the needed performance via live queries on the data lake because of Dremio's numerous performance capabilities.

Now that users have interactive access to all data in the environment, users' incentive to download data to their laptops is significantly reduced. The remaining local data copies are usually a result of people who are slow to change habits over time or offline reports that haven't been migrated yet.

This is also generally when data scientists begin to leverage Dremio for core datasets and standard KPI definitions in their work. They're able to use standard definitions to enrich their machine learning models, and still access them when their queries have large result sets, for instance when going from initial model development to more extensive testing at scale, and leveraging Apache Arrow Flight for very high speed data transfer.

In addition to migrating existing workloads, when new workloads and use cases arise, they are implemented directly on Dremio.

Phase 2 – Further Reducing Data Copy Proliferation

Once the initial migrations are complete, the organization's architecture looks something like this:

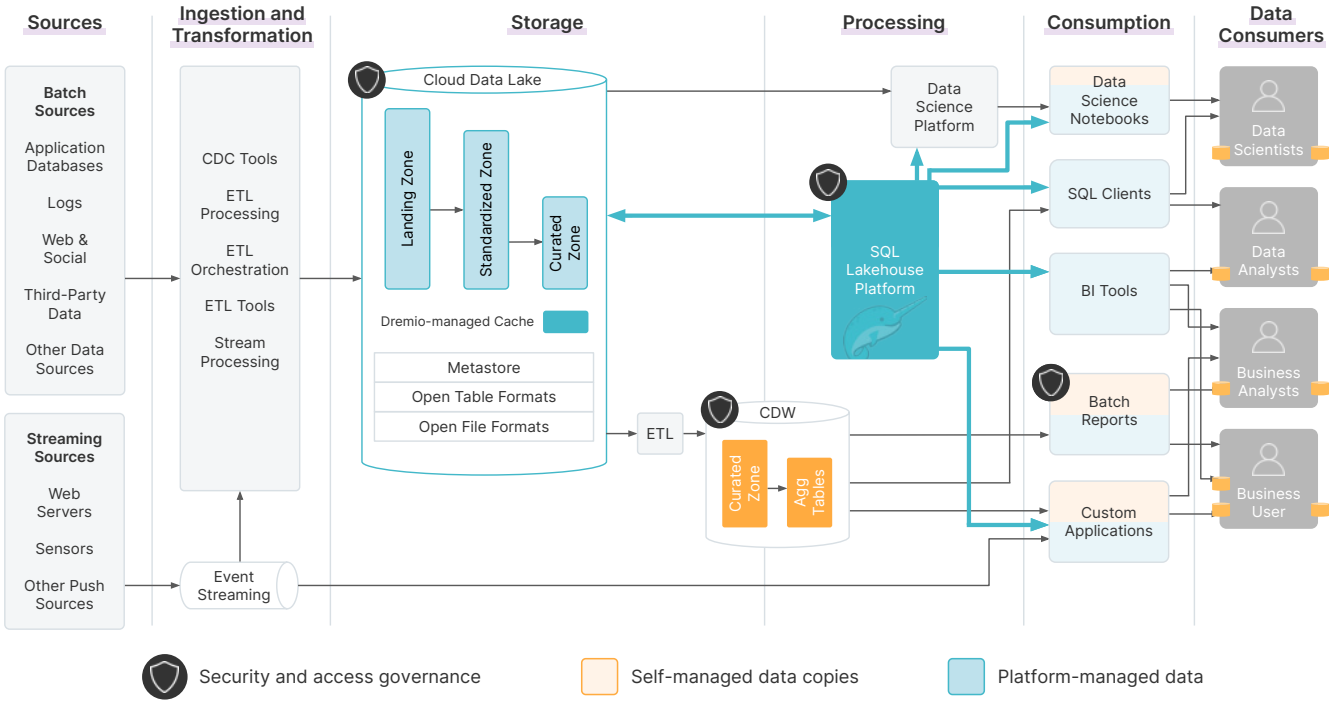


Fig. 3 Phase 2 – Further reducing data copy proliferation

As the organization continues its journey and experiences the value of Dremio, more workloads are migrated to run directly on the data lake via Dremio. Migrating workloads off of the CDW helps to reduce data copies, ETL and ELT jobs, and the footprint of the CDW, thereby also reducing the associated costs.

In order to provide seamless interactivity at scale, Dremio leverages data reflections, which exhibit some of the best properties of both materialized views and indexes, and are completely managed internally by the Dremio platform. Requiring no administrative burden, and completely transparent to the end user, data reflections allow users to interact with their data in a data model that makes the most sense to them and still achieve interactive response times for any workload at any scale. You can see the introduction of this Dremio-managed cache of Apache Parquet files in the "Cloud Data Lake" area of the foregoing diagram.

The order-of-magnitude increase in query performance provided by Dremio eliminates the need for many of the local copies of data made previously to achieve good query performance (i.e., BI extracts), and mitigates the downsides of maintaining and securing these copies.

Batch reports that send users emails, PDFs, Excel files, etc., are often employed due to a lack of cost-efficient interactivity for live reporting. With Dremio efficiently providing interactivity directly on the data lake, the need for these batch reports is reduced; these canned reports can be made interactive providing users of the report the ability to further drill down into the report and better data freshness.

Aggregation tables, another form of self-managed data copies, are used as a result of being unable to provide interactivity in the CDW on the lower granularity data, whether for pure performance or cost-efficiency reasons. These are created and managed by ELT scripts which must be written, scheduled, operated, monitored, and maintained as things change upstream and downstream. Dremio removes the need for self-managed aggregation tables via raw interactivity and data reflections.

Phase 3 – Reaping the Rewards of a Modern SQL Lakehouse

This is the architecture that most of our users and customers currently employ for self-service analytics. With this architecture, they're able to democratize interactive analytics directly against their data lakes, while avoiding the downsides of the traditional architecture.

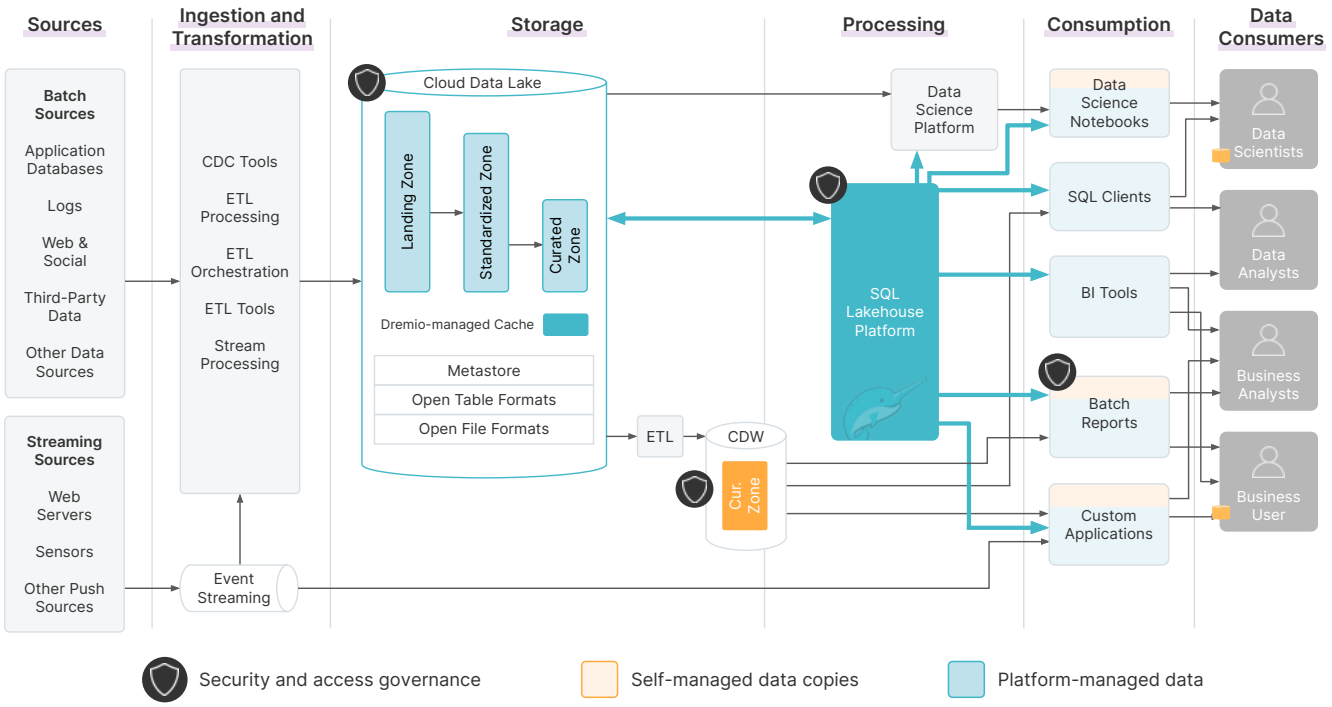


Fig. 4 Phase 3 – Reaping the rewards of a modern SQL lakehouse

In this architecture, users and customers approach new use cases with a data-lake-first mindset, only resorting to the more traditional CDW approach for a small subset of tasks for which the data warehouse is ideally suited (typically smaller datasets which are rapidly mutating).

Many Dremio customers follow this architecture today, and are reaping the benefits of greatly improved self-service analytics with better time to insight and user experience, enterprise-grade security and governance, and data democratization.

Let's revisit the earlier example of adding a column to a dashboard, with Dremio as the SQL layer powering BI in the organization:

1. The business user makes the request of the team that owns the dashboard.
2. However, now this team has direct but governed access to the broader and more granular data in the data lake. This team now has access to the standardized zone which has the additional column they need to add, so they can be self-sufficient. But, let's assume they don't, and still need to request the column from the data lake platform team.
3. Because the additional column is in the lake, but no further than the landing zone, the platform team is able to make the lightweight change to the SQL logic of the virtual datasets in Dremio that the dashboard team does have access to, providing them with the new column. The data team also easily adds it to the reflection that optimizes the dashboard.

4. After these two changes quickly proceed through the promotion process to ensure data quality, the dashboard team then adds the new column to the dashboard.

Just like that, the business user now has the additional column they need in the dashboard, and can now take it into account when making decisions.

This process generally takes our customers anywhere from an hour to at most a day or two, in contrast to their previous architecture which required in three to six weeks.

Conclusion

Dremio enables organizations to achieve the best capabilities of both the data warehouse and data lake, in a single platform. The adoption of Dremio's SQL lakehouse platform greatly mitigates the issues related to data copy proliferation, such as data drift and regulatory compliance. Organizations of all sizes — three of the Fortune 5, many of the Fortune Global 2000, and many smaller organizations — currently leverage Dremio to provide self-service analytics directly against data lakes while ensuring enterprise-grade security and governance.

Get Started with Dremio Cloud today.

ABOUT DREMIO

Dremio is a SQL Lakehouse Platform company enabling organizations to leverage open data architectures. Dremio's SQL Lakehouse Platform simplifies data engineering and eliminates the need to copy and move data to proprietary data warehouses or create cubes, aggregation tables and BI extracts, providing flexibility and control for data architects and data engineers, and self-service for data consumers. Dremio Cloud, a frictionless, infinitely scalable service, enables high performance SQL workloads directly on cloud storage, eliminating the cost and complexity of copying and moving data. Dremio Cloud reimagines the traditional data lake by combining the best of traditional data warehouses and data lakes into a SQL lakehouse, while removing the limitations of traditional data warehouses resulting from closed data architectures.