Data Lakehouse open source market landscape

Comparing Delta Lake, Apache Hudi, and Apache Iceberg platforms

Published February 2023

Executive Summary

Trigger

Over the past five years, a new construct designed to combine the best of the Data Warehouse and Data Lake worlds has emerged: The Data Lakehouse. The key enablers for the confluence of these worlds are new table formats overlaying atop cloud object storage that deliver the performance, governance, granular security, and ACID transaction support of data warehouses, combined with the economics of scale and the analytic flexibility of data lakes. The "action" has occurred both in the open source and proprietary data warehousing platform worlds. While AWS, Oracle, and Teradata introduced their lakehouse table constructs, predating them are several open source projects including Delta Lake, Apache Hudi, and Apache Iceberg, that are now rapidly drawing multivendor support. At first glance, these open source projects appear similar in approach. How do their technologies and market footprints compare? And what are the implications for analytics? Will this lead to a new era where differentiation occurs, not in the storage or transaction processing engine, but in the data management and analytic tier?

Our Take

The data lakehouse will not replace data lakes or purpose-built data warehouses, but in the long run they will coopt enterprise data warehouses. Data lakehouses will enable data lakes to perform and be controlled, governed, and secured like data warehouses. They will support a variety of analytic workloads including SQL *and* programmatic analytics and modeling. The trigger for data lakehouses is that enterprises are increasingly relying on data lakes, not only to perform exploratory analytics, but also to perform more business-critical functions such as the training and running of machine learning workloads. But to do so, enterprises must gain confidence in the consistency of data residing in their data lakes, and that is why ACID transaction support became the linchpin of data lakehouses. While there are lakehouse architectures built on proprietary and open source technologies, in the long run, open source will prevail because ACID support will be table stakes, not a competitive differentiator. Open source is currently dominated by three projects: Delta Lake, Apache Hudi, and Apache Iceberg. Today, there is roughly 80% functional parity between them, and over the next 12 – 18 months, most of the functional gaps should close. While each open source lakehouse project has its own unique mix of technical strengths and weaknesses, in the long run, it will be the breadth and depth of the commercial ecosystem and depth of support that will determine the winners. We expect the data lakehouse ecosystem will likely winnow down to two platforms.

Published February 2023

Why are we having this conversation?

Bridging the gap between data warehouses and data lakes

The data lakehouse is about delivering the best of both worlds: the scale and flexibility of the data lake with the SLAs, repeatability, and mature governance of the data warehouse.

When we first explored the topic in the early 2010s, we viewed the data lake as the platform for performing exploratory analytics, where schema was formed on read, and the data and questions were scoped. When it came time for a "precise" answer on which auditable decisions were made, the data would undergo an ETL process into the data warehouse, a process that hit scaling limits as data volumes mushroomed.

Today the distinctions between data warehouses and data lakes are blurring as:

- In-database machine learning, Apache Spark, and ELT support have become commonplace with cloud data warehouses;
- Data lakes, running with data in de facto standard file formats such as Parquet, powered by data transformation capabilities from Spark, Drill, Trino and other open source engines, are delivering query performance that is becoming competitive with data warehouses; and
- Cloud data warehouses and data lakes are both scaling to handling multi-PByte analytic workloads.

Bringing ACID to "Big Data"

The elevator pitch for data lakehouse is about bringing ACID transactions to the data lake. This is not about making the lakehouse into an OLTP database, but rather, to build confidence that the data in the data lake is trustworthy: it is current, consistent, and transactionally valid. That requires the Atomicity, Consistency, Isolation, and Durability of data that is essential for any system supporting business-critical decisions and processes.

Along with ACID come other key architectural requirements including:

- Support of cloud object storage, which has become the de facto data lake;
- Table-based schema to make data in cloud storage look, and behave like relational data, and deliver the performance, control, and fine-grained governance associated with relational databases;
- Support of popular file formats, with Parquet being the most common;

- Support of SQL and programmatic query, with Python becoming the most requested language.

The revenge of the SQL Nerd

While data lakehouses will support SQL and programmatic analytics and query, by imposing a relational view over polyglot data, they will be the revenge of the SQL nerd on the data lake.

For lakehouses, ACID has been implemented via relational tables, which are overlayed atop data sitting in semi-structured file formats (e.g., Parquet, ORC, JSON, CSV) residing on lower-cost, de facto standard cloud object storage. With relational tables, lakehouses can deliver a higher level of performance and granular control compared to conventional file systems. Lakehouses also facilitate schema evolution. While the lakehouse table format is an overlay, rather than a physical (binary) storage layer, once the table format is established, the data set must be read that way to ensure that ACID guarantees remain effective (e.g., that no changes are made to the underlying raw file, bypassing reading of the table structure).

The table construct at the core of data lakehouse formats is superior to file scans because each table carries the schema, statistics, and indexes that accelerate data retrieval. Data can be organized in columnar formats that are more efficient for analytic processing because they are better suited for filtering, data skipping, and compression. Tables also allow record-level mutations, meaning users can insert, update and delete individual records transactionally, just like a data warehouse. And they also support partitioning and the ability to adjust schema on immutable files. As a bonus, lakehouses should also support "time travel" where data can be queried at different points back in time; this is especially useful for providing the trail of breadcrumbs showing how a decision was made or how a model performs at different time intervals. The transaction logs of lakehouses should be queryable for specific insert, update, or delete transactions. And with data organized in tables, security and access controls can be enforced at table, column, and/or row levels.

Will Data Lakehouses be first-class analytics citizens?

A key question is whether data lakehouse tables can deliver equivalent performance compared to tables stored with proprietary binary file formats of data warehouses. Early results are encouraging, as at least one prominent SaaS provider indicates that they can deliver up to 80 – 90% of the performance compared to their existing proprietary table format. That still leaves room for improvement; for instance, many providers add their own indexing and caching to further optimize performance. Not surprisingly, as works in progress, data lakehouses still have some functional gaps to address, which we expect will be largely addressed over the next 12 – 18 months (but not all of those gaps will be addressed in open source).

Highlights include:

- **Managing cloud storage buckets.** Cloud data warehousing SaaS services handle this for you. By contrast, most early lakehouse implementations don't. Depending on your viewpoint, this can be advantage or drawback. When the SaaS provider handles this, it makes life simpler. On the other hand, some customers will prefer taking this on themselves as they can simply run the lakehouse atop the cloud storage buckets they are already paying for.
- **Multi-table transactions/joins.** The query engine or open source compute engine picks up where the lakehouse table structure leaves off. Just as data warehousing vendors differ in the capabilities of their SQL implementations, we expect that this will continue to be differentiated, proprietary functionality.
- **Time to Live (TTL).** Lakehouse tables operate as append-only systems where stale data and metadata is deleted after the transaction is committed (this is how they support data mutation). At some point, stale data needs to be pruned from the system. While Hudi automates this process, Delta Lake and Iceberg currently do not. While you could automate or orchestrate compactors with external routines (e.g., using frameworks like Airflow), we believe automated compaction should be a core capability of the lakehouse. Vendors could then differentiate in the intelligence they apply to managing or triggering schedulers that are part of the open source core.
- **Concurrency.** This is a work-in-progress with each of the lakehouses, especially with multi-cluster (distributed) implementations. Some rely on external systems for tacking locks (e./g., DynamoDB) that adds complexity to the stack. And some cannot guarantee that writes on lakehouses operating over multiple clusters will be written in proper sequence. We expect that over time each of the lakehouse projects will up their game with handling concurrency for distributed multi-cluster workloads.

Openness

That's the 64-GByte question.

While traditional data warehouses are based on a common query language (SQL) and relational schema, in actuality they are classic fit-for-purpose, proprietary systems. Although SQL is the lingua franca, each data warehouse carries its own proprietary flavor. There were similar issues with commercial Hadoop platforms supporting competing open source projects.

Will fate be any different for data lakehouses? Data lakehouses are supposed to be file- and analytic engine-agnostic. Parquet has become the de facto standard supported file format, although some lakehouses also support common formats such as CSV, ORC, or Avro. And

while the lakehouse was designed with the relational model in mind, most also support APIs for programmatic analytics, with Python typically being the go-to language.

A couple things won't change. Higher-level metadata (beyond the basic statistics collected by each lakehouse table format) is likely to stay proprietary, with interoperability being a function of each lakehouse provider's partner ecosystem. The same goes with the flavor of SQL; it will still continue to be tied to the analytic query engine. That leads to the next question: while the table format is open, can the customer swap out analytic engines (and with it, the flavor of SQL)? For instance, can Iceberg users readily swap out Cloudera's analytic engine for Snowflake or vice versa? Or could they be used side-by-side, where Dremio implementations of Iceberg read the latest updates made by a Snowflake user?

Snowflake's response provides a case in point. The company states that it is implementing the open source specification and won't fork the file or table format. Neither will Snowflake implement its proprietary micropartition architecture on Iceberg, which should make data transparent to third parties supporting Iceberg like Starburst Data, Dremio, or Cloudera, and all the third party tools that work with those platforms. But as noted, there will need to be some way for metadata to synchronize. We hope that the open source projects standardize on a mechanism.

Snowflake's strategy could set a precedent for third parties not to impose proprietary extensions that make their data in Iceberg unreadable to other analytic engines or data management tools. The table format is *not* where Snowflake or others will differentiate. The analytic/query engine is where support for operational management (e.g., data ingestion and lifecycle management); specific analytic libraries or functions (e.g., for specific vertical industries); and general ease of use kick in. If any third party were to "optimize" down at the table level with an "API-compatible" implementation of Iceberg (or any of the other lakehouse formats), it would have to deliver performance of at least an order of magnitude or two greater to offset the risk of jeopardizing third party support. For the moment, this question is purely theoretical, but never say never.

High-level functional comparison

Delta Lake, Hudi, and Iceberg each meet the core requirements for lakehouses, including supporting ACID transactions, time travel, granular access control, multifunction (SQL and programmatic) analytics, and delivering superior performance compared to data lakes. There is roughly 80 – 90% functional parity between the lakehouse table formats.

There are a few differences. For instance, some are more efficient at querying change data feeds while others have more flexible indexing options. Furthermore, while all of the lakehouse table formats support ACID transactions (the core reason for their existence), each has different approaches to transaction logging. Likewise, the lakehouses handle schema

changes quite differently, with Hudi and Iceberg supporting deleting or modifying columns, while Delta Lake bypasses deprecated columns. We cover these differences in detail in the companion deep dive report.

Delta Lake, Hudi, and Iceberg should achieve functional equivalence within the next 12 – 18 months. The pace of technology development, degree of community involvement, and latent demand for lakehouses should ensure rapid development to address the gaps.

Market ecosystems

Cut to the chase. Ecosystem, not technology, will determine which data lakehouse platforms prevail for several reasons, of which first and foremost, is this:

True differentiation, and value, to enterprises will be in the analytics engine, not the table format.

The commercial landscape

Ostensibly, the three lakehouse projects were developed by creators at individual firms: Delta Lake, augmented the Databricks Spark Unified Analytics Platform; Hudi, where the creator, Vinoth Chandar of Uber, went on to found Onehouse; and Iceberg, originated by Ryan Blue of Netflix, who went on to found Tabular. A current listing of commercial support (as of February 2023) is provided in the Appendix.

It is still early days for development of the ecosystem, but some patterns are unmistakable. Look for vendors that support read *and* write (see Table 1 in the Appendix). Delta Lake leverages the Databricks partner ecosystem; Iceberg has drawn a handful of analytics data platforms; while Hudi has developed a loyal community of early adopters. But unlike Delta Lake or Iceberg, Hudi has so far failed to develop meaningful commercial support beyond the hyperscalers (who are also planning support for Delta Lake and Iceberg as well) and the usual cross-platform tools supplier long tail. Hyperscaler support should be treated as marketing feelers, rather than "wins" for Hudi; ultimately, they will support the formats with the most customer traction. Hudi needs a marquee supporter with a critical mass third party ecosystem. For now, Hudi is starting with a clear disadvantage.

Drilling down on the makeup of the commercial landscapes of Delta Lake and Iceberg, currently they resemble that of the mobile device ecosystem: iOS (Delta Lake) vs. Android (Iceberg). Like iOS, Delta Lake was created and long dominated by a single vendor (Databricks), while Iceberg has not had the same single vendor dominance.

The dynamics are changing. The bulk of the content of Delta Lake has come from Databricks, as it chose to incubate and steer the project to production readiness before fully unleashing it to the open source community. However, with Databricks open sourcing the remainder of the Delta Lake platform last summer under an Apache license, the project is beginning to evolve

toward a community model with distributed responsibilities. Yet, the Delta Lake project site does not currently list who is in charge of the project; transparency is necessary for Databricks to affirm that Delta Lake is becoming a community project.

Technology providers supporting Delta Lake have come from the Databricks partner program because, until recently, Delta Lake was largely a Databricks product. Consequently, aside from Azure Synapse Analytics, the bulk of support comes from tools rather than data platform providers; data platforms that are listed as Databricks partners function as sources of data and are not currently using Delta Lake (or the classic Databricks Unified Analytics Platform) for their analytics tier. We expect that the Delta Lake ecosystem will add more data platform providers in the future; Oracle, which supports Delta Sharing on its MySQL Lakehouse service would be a likely candidate.

Conversely, the Iceberg ecosystem is less developed, but more diverse, as there are several data platforms that use it as lakehouse target (e.g., Celerdata, Cloudera, Dremio, Google, Snowflake, Starburst Data, and Tabular). The list will grow.

Another way to look at the potential market reach of the lakehouse platforms is by comparing the ecosystems of prominent backers, with cloud analytics rivals Databricks and Snowflake being prime examples. These partner ecosystems provide an illustration of the *potential reach* of Delta Lake and Iceberg through the existing technology partner programs of Databricks and Snowflake, respectively.

Does this mean that, by extension, these partner ecosystems are ready on Day 1 to support Delta Lake or Iceberg because Databricks and Snowflake, respectively, offer native support? The answer is, it depends. It should be straightforward for BI vendors with standard SQL connectors to visualize from lakehouse tables that look like any relational source. On the other hand, tools that transform, manage, or secure data may require adaptations to support lakehouse table structures.

Takeaways

Still early days

The landscape is in early stages of development. It is not yet game over as the enterprise market is still in early awareness stage for data lakehouses. In the analytics world, currently, data mesh is sucking up much of the oxygen. Here's some anecdotal evidence: during 2022, our LinkedIn posts on data mesh garnered roughly 10x the number of responses as lakehouses.

The early stage of awareness of data lakehouses explains why many of the household enterprise technology names are not yet fully active. IBM and Microsoft at this point are involved through partners: Cloudera and Databricks, respectively. SAP has not yet chimed in

on its lakehouse strategy, which would logically extend SAP HANA Data Warehouse relational data lake. AWS, Oracle, and Teradata have introduced initial offerings. In summary, both Oracle and Teradata are relying proprietary data lakehouse table formats, while AWS and Google are in the early days of embracing open source, as detailed below. We expect that enterprise vendors will get more fully engaged in 2023.

Commercial implementations are immature

In the long run, vendors will differentiate based on the control planes that they deliver atop data lakehouses, and on the connectivity or integration to adjacent/related services. But as these are still early days, many early commercial offerings remain works in progress. For instance, the ability to read *and* write to data lakehouses will ultimately become table stakes; in the short run, some commercial implementations lack basic write capabilities.

The market will consolidate to two lakehouses

Markets demand competition, not fragmentation. Our bet is that when the dust settles, there will be two major lakehouse ecosystems left standing. It will look a lot like the mobile world where the landscape has settled between Android and iOS.

Why will the lakehouse landscape consolidate to two, rather three (or more) platforms? What about the argument that there are three major hyperscalers, so therefore, why shouldn't the lakehouse landscape support at least that many? Here's why:

- **High capital investment hurdles.** It takes billions of dollars of capital to establish a global network of data centers and regions, and of necessity, hyperscalers must exercise care when choosing where to locate physical presence. They will have varying presence across different geos, which will drive customer decisions, especially where data sovereignty requirements are involved.. Two providers alone won't adequately be able to guarantee critical mass presence in every region, overnight.
- **Diverse requirements for products and services.** For instance, while each hyperscaler offers different machine learning services, each has unique strengths and weaknesses. Enterprises may choose Google for its call center AI services, while opting for Amazon SageMaker for its lifecycle management and/or Azure for test driving Chat-GPT.
- **Operational risk.** This favors having more hyperscalers on which to spread the risk. Not surprisingly, the norm for most enterprises is multi-cloud strategy, either by formal policy or inertia.

The bottom line is that there is far more room for a major third player in the hyperscaling world than there would be for a niche technology such as a data lakehouse table format, where the stakes are far lower.

Published February 2023

From the vendor side, there are significant savings to be had for only facing the choice to support two systems, rather than three, four, or more. For customers, there is less risk to standardizing on a single lakehouse table structure than it would be for binding their livelihoods with a single cloud provider, which effectively should narrow the market.

Today, Delta Lake and Iceberg have the clear momentum and are clearly the early favorites to be the lakehouses left standing. For Hudi, the challenge is building a commercial ecosystem beyond the long tail, with pressing need to line up a major data platform heavyweight. The good news is that, as noted above, none of the usual suspects (e.g., IBM, Oracle, SAP) have yet planted their lakehouse stakes. It ain't over till it's over.

Open source will prevail

Table structure has long been table stakes, not the differentiator, among data warehouses, and that won't change with data lakehouses. The table format is simply a means to an end. Instead, differentiation will be in the analytics engine; the power of their SQL language; and the breadth of the commercial/technology ecosystem (e.g., will the lakehouse table platform have support from the data integration, BI, and analytic tools that the organization uses).

As noted above, technology providers, from data platforms to hyperscalers and integration/analytic tool providers are hopping the open source bandwagon to the point where it is impossible to list them all in a single sentence. Several providers are still relying, or hedging their bets, on proprietary lakehouse table formats, including:

- AWS introduced Governed Tables as part of a governance service and process framework for managing and securing data lakes. However, AWS subsequently ramped up support for open source lakehouse formats with Athena, EMR, and Glue, along with more limited (read-only) external table support in Redshift Spectrum; we believe that open source is AWS's primary future direction.
- Oracle developed a lakehouse table format, but it is currently limited to its MySQL Heatwave service and has not yet graduated to the flagship.
- Teradata also just announced its first foray with VantageCloud Lake.

We believe that proprietary formats are just a transitional stage in the maturation of the market for the following reasons:

- For enterprises, the issue is getting access to the tools that they use. They don't want lock-in at the table format because (1) it could constrain their selection, and (2) is not a differentiator or value-add for them. Open source reduces the chances for lock-in.
- For vendors, it is about the analytic engine and their partner ecosystem; it is not worth fighting battles that won't differentiate their platforms.

For those reasons, we believe that open source will win out over proprietary formats for data lakehouses.

Data lakehouses will coopt the enterprise data warehouse

Data Lakehouses will eventually coopt the enterprise data warehouse because it provides many of the same capabilities for multifunction analytics.

Most enterprise data warehouses have been adding data lake-like support of polyglot data (typically through user-defined or other custom functions); non-SQL analytics; and extensions to cloud storage through external tables. The difference with the lakehouse is that, as long as the data is in a supported file format (e.g., Parquet), data sitting in cloud storage will be treated as first-class citizens with the same ACID guarantees, governance and control, and nearly equitable performance compared to data sitting in established proprietary table structures. That isn't possible with traditional federated query approaches where Parquet files are treated as external tables. The pace of technology development should overcome many of the gaps that could manifest with capabilities such as in-database machine learning and custom functions. As noted below, there will be a moving bar at the high end.

Data Lakehouses will not replace data lakes or purpose-built data warehouses or data marts.

Data scientists conducting model development or exploratory analytics may find the schema-on-design table structure of lakehouses limiting their ability to explore diverse data types and sources. Although the lakehouse format is a software-defined overlay that will not physically change the underlying data, if data scientists make changes to the data without going through the table overlay, it will undermine the ACID guarantees that are essential to lakehouses. So data scientists will likely prefer working in less encumbered data lakes.

As for data warehouses, at each end of the spectrum there will be exceptions. The lakehouse will be overkill for data marts, especially as they are enhanced with capabilities such as AutoML. It will also be overkill single-purpose workloads such as operational query and reporting. At the other end of the spectrum, open source may not yet be robust enough to handle workloads requiring extreme table joins, high concurrency, and sophisticated workload management. But we also expect that open source will steadily up its game, just as relational databases did back in the early days, when they were still considered underpowered compared to legacy hierarchical or networked data stores.

Cloud data warehouses with support for polyglot data types, Python, and AutoML capabilities will be the gateway drugs for data lakehouses.

With capabilities such as support of JSON data and in-database AutoML getting baked into cloud data mart-type offerings targeting business analysts and “citizen data scientists,” there are many cloud data warehousing offerings that are treading lightly into the capabilities associated with lakehouses. They support analyzing JSON data, in-database programmatic

analytics (via UDFs), and/or in-database AutoML that lower the barriers to using machine learning for predictive or prescriptive analytics. We could foresee a transition pathway where,

as these services gain traction, that customers grow more ambitious with data and evolve or gravitate toward lakehouse services designed expressly for ease of use and targeted at the long tail of the enterprise market.

Appendix

Table 1. Tool/technology support

Tool/technology	Delta Lake	Apache Hudi	Apache Iceberg
Ahana	Read	Read	Read + Write
Alibaba Cloud		Read + Write	
Amazon Athena	Read	Read	Read + Write
Amazon EMR*	Read + Write	Read + Write	Read + Write
Amazon Glue*	Read + Write	Read + Write	Read + Write
Amazon Redshift*	Read	Read	
Apache Beam	Write		
Apache Flink	Read + Write	Read + Write	Read + Write
Apache Hive	Read	Read + Write	Read + Write
Apache Impala		Read	Read + Partial Write support
Apache Spark	Read + Write	Read + Write	Read + Write
Azure Synapse	Read + Write		
Celerdata			Read + Write
Clickhouse	Read	Read	
Cloudera Data Platform			Read + Write
Confluent		Read + Write	
Databricks	Read + Write	Read + Write	Read + Write
Databricks SQL	Read + Write		
dbt		Read + Write	Read + Write
Debezium	Write	Write	Write
Dremio Sonar	Read	Read	Read + Write

Tool/technology	Delta Lake	Apache Hudi	Apache Iceberg
Google BigQuery*	Read	Read	Read
Microsoft Azure Synapse Analytics	Read + Write	Read	Read
Microsoft HDInsight	Read + Write	Read + Write	
Onehouse		Read + Write	
Presto	Read	Read	Read + Write
Snowflake (native)			Read + Write
Starburst Enterprise	Read + Write	Read	Read + Write
Tabular			Read + Write
Trino	Read + Write	Read	Read + Write
Vertica	Read	Read	

Sources: Apache Hudi community, Apache Iceberg community, Databricks, Dremio, Onehouse
*Note: Support roadmap.

Author

Tony Baer, Principal, dbInsight

tony@dbinsight.io

LinkedIn https://www.linkedin.com/in/onstrategies/

About dbInsight

dbInsight LLC® provides an independent view on the database and analytics technology ecosystem. dbInsight publishes independent research, and from our research, distills insights to help data and analytics technology providers understand their competitive positioning and sharpen their message.

Tony Baer, the founder and principal of dbInsight, is a recognized industry expert on data-driven transformation. *Onalytica* named him as a Top Cloud Influencer for 2022 for the fourth straight year. *Analytics Insight* named him one of the 2019 Top 100 Artificial Intelligence and Big Data Influencers. His combined expertise in both legacy database technologies and emerging cloud and analytics technologies shapes how technology providers go to market in an industry undergoing significant transformation.

Published February 2023