Investigation of performance impact of Delta Lake features

Karthik Radhakrishnan

***Abstract*— Traditional data lakes often face challenges such as data inconsistency, slow query performance, and the absence of transactional guarantees, leading to inefficiencies in data management and analysis, particularly with streaming and large-scale datasets.

Delta Lake, an open-source ACID table storage layer over cloud object stores, uses a transaction log that is compacted into Apache Parquet format to provide ACID properties, time travel, and significantly faster metadata operations for large tabular datasets (e.g., the ability to quickly search billions of table partitions for those relevant to a query).

In this paper, we investigate the performance impact of delta lake features compared to traditional data lake.

I. Introduction

A. *Data warehouses*

Data warehouses are a central relational repository of integrated, historical data from multiple data sources that presents a single integrated, historical view of the business with a unified schema, covering all perspectives of the enterprise.

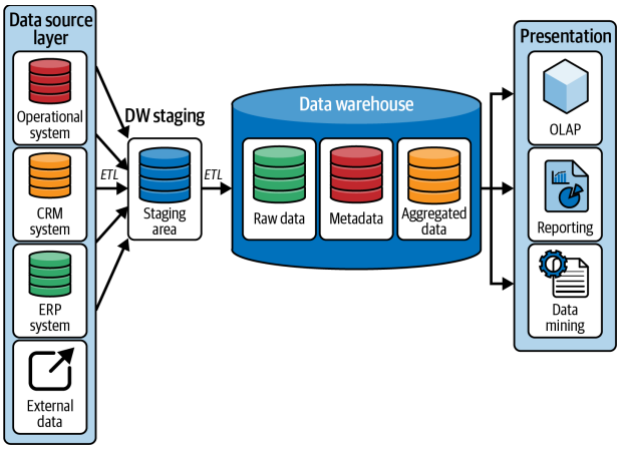


Fig. 1. Data warehouse architecture.

B. *Data warehouse – Benefits and Challenges*

Benefits

Data warehouses store large amounts of historical data, they enable historical insights, allowing users to analyze different periods and trends.

Data warehouses tend to be very reliable, based on the underlying relational database technology, which executes ACID transactions.

A data warehouse combined with business intelligence (BI) tools can generate actionable insights for marketing, finance, operations, and sales. Abbreviations and Acronyms

Challenges

Traditional data warehouse architectures struggle to facilitate exponentially increasing data volumes. They suffer from both storage and scalability issues.

Data warehouse architectures are also not a good fit to address the velocity of big data. Data warehouses do not support the types of streaming architecture required to support near-real-time data.

While data warehouses are very good at storing structured data, they are not well suited to store and query the variety of semi-structured or unstructured data.

C. *Data Lakes*

A data lake is a cost-effective central repository to store structured, semi-structured, or unstructured data at any scale, in the form of files and blobs.

Data lakes are enabled through a variety of components:

Storage

Data lakes require very large, scalable storage systems, like the ones typically offered in cloud environments. (ex: Amazon S3, Azure ADLS). The storage needs to be durable and scalable.

This paper submitted for review on 12/2/2024.

Karthik Radhakrishnan is a Master of Data Science Student at Southern Methodist University, Dallas, TX 75024 USA. (e-mail: kradhakrishnan@mail.smu.edu).

Compute

High amounts of compute power are required to process the large amounts of data stored in the storage layer. The go-to compute engine for data lakes is Apache Spark, which is an open-source unified analytics engine. Big data compute engines will leverage compute clusters. Compute clusters pool compute nodes to tackle complete data collection and processing tasks.

Format

The shape of the data on disk defines the formats. Data lakes use mostly standardized, open-source formats, such as Parquet, Avro JSON, or CSV.

Metadeta

Modern, cloud-based storage systems maintain metadata (i.e., contextual information about the data). This includes various timestamps that describe when data was written or accessed, data schemas, and a variety of tags which contains information about the usage and owner of the data.

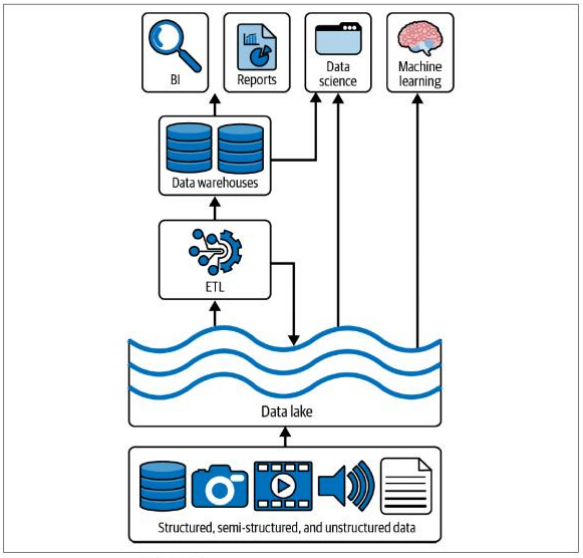


Fig. 2. Data lake architecture.

D. *Data lakes – Benefits and Challenges*

Benefits

Data lakes are deployed on mature cloud storage subsystems, allowing them to benefit from the scalability, monitoring, ease of deployment, and low storage costs. Unlike data warehouses, data lakes support all data types, including semi-structured and unstructured data, enabling workloads such as media processing.

Challenges

Traditional data lakes have poor latency query performance, so they cannot be used for interactive queries. As a result, the organization's data teams must still transform and load the data into something like a data warehouse, resulting in an extended time to value. This resulted in a data lake + warehouse architecture.

Lack of schema enforcement can result in data quality issues, allowing the pristine data lake to become a "data swamp."

Data lakes do not offer any kind of transactional guarantees. Data files can only be appended to, leading to expensive rewrites of previously written data to make a simple update.

E. *Data lakehouse*

Data lakehouse is a system that merges the flexibility, low cost, and scale of a data lake with the data management and ACID transactions of data warehouses, addressing the limitations of both.

To add these capabilities, lakehouses use an open-table format, which adds features like ACID transactions, record-level operations, indexing, and key metadata to those existing data formats. This enables data assets stored on low-cost storage systems to have the same reliability that used to be exclusive to the domain of an RDBMS. Delta Lake is an example of an open-table format that supports these types of capabilities.

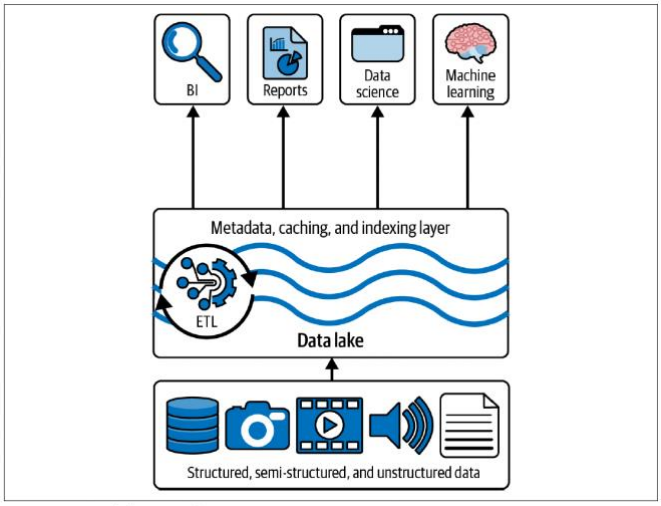


Fig. 3. Data lakehouse architecture.

The evolution from data warehouses to data lakes to a lakehouse architecture is shown in Figure 4.

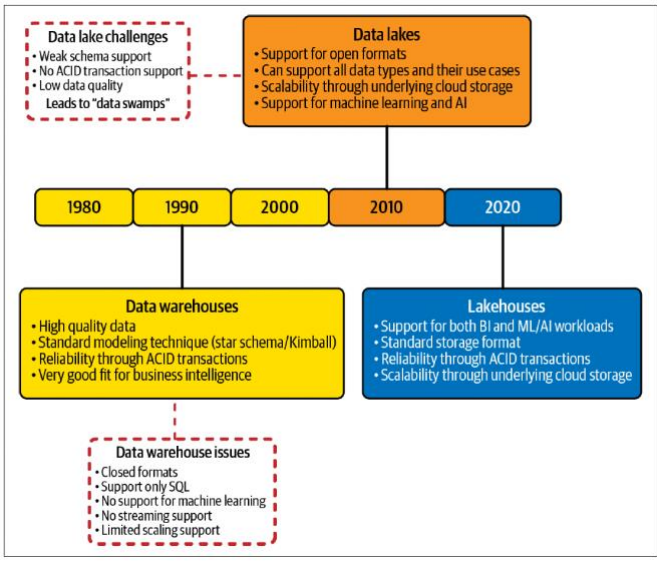


Fig. 4. Evolution of data architectures.

II. RESEARCH METHODOLOGY

A. *Benchmarking Modern data lakehouse based on delta lake open table format*

- Delta tables were established on the Databricks platform with a predefined schema tailored for a supply chain database.
- Synthetic test data was generated using the TPC-H benchmarking tool to facilitate performance evaluation.
- Data ingestion workflows were designed and implemented to load data into Delta tables. Key performance metrics, including data load time, throughput, and write latency, were captured during the data loading process.
- Complex analytical queries were executed on the Delta tables, and metrics such as query duration and bytes read were systematically measured and analyzed.

B. *Benchmarking Traditional data lake*

- The same test dataset was loaded into external tables mounted from the files stored on Amazon S3 without utilizing Delta Lake features.
- Performance metrics, including data load time, throughput, and write latency, were recorded during the process of mounting external tables from S3. These metrics were subsequently compared with those collected during data loading into Delta tables.
- Complex queries were executed on the external tables, and metrics such as query duration and bytes read were measured. These results were analyzed and compared to the query performance observed in Delta tables.

III. RESULTS

A. *Experimental Setup*

All experiments are run using Databricks runtime version 15.4 LTS that includes Apache Spark 3.5.0 and Scala 2.12. All experiments are performed with 8 workers on AWS r7gd.large instances with 16GB memory of RAM each.

B. *Load Performance*

We evaluated the effects of delta lake features in load times using TPC-H benchmarking suite.

TABLE I
COMPARISON OF LOAD TIME

Size	Record Volume	Load time (Sec)	
		Delta Lake	Data Lake
1 GB	8,661,245	112	11
10 GB	86,586,082	245	8

TABLE II
COMPARISON OF THROUGHPUT

Size	Record Volume	Throughput (MB/sec)	
		Delta Lake	Data Lake
1 GB	8,661,245	9.14	93.09
10 GB	86,586,082	41.80	1280.00

Throughput – Amount of data that can be written per second.

TABLE III
COMPARISON OF WRITE LATENCY

Size	Record Volume	Write Latency (µsec/record)	
		Delta Lake	Data Lake
1 GB	8,661,245	12.93	1.27
10 GB	86,586,082	2.83	0.0924

Write latency – Amount of time taken to write a single record.

Data load time for Delta Lake is longer than data lake mainly due to the below reliability features:

ACID Transactions: Delta Lake provides atomicity, consistency, and isolation guarantees, which require additional operations (e.g., maintaining a transaction log and checking for schema enforcement).

Metadata Overhead: Delta Lake processes and updates metadata to ensure reliability and consistency, which can add to the execution time.

File Compaction: During writes, Delta Lake may compact small files into larger ones for optimized query performance later.

This longer execution time reduces the amount of data that can be written per second in Delta take (Throughput) and results in higher write latency.

Data load time for data Lake is shorter than delta lake due to:

Minimal Metadata Management: There is no built-in mechanism to enforce schema or maintain versioning.

No ACID Guarantees: Data is directly written to the storage without any additional checks or logs.

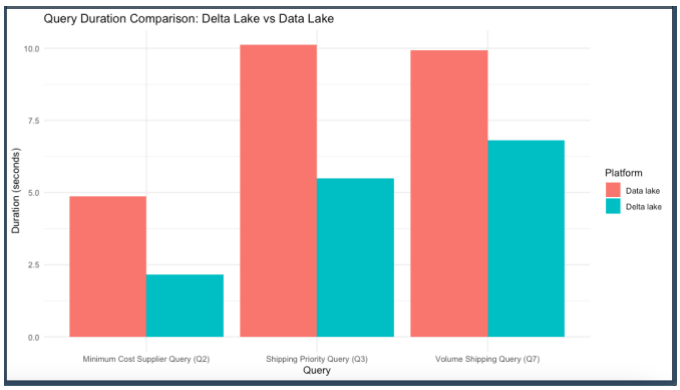
Lightweight Writing: It simply appends files to the storage.

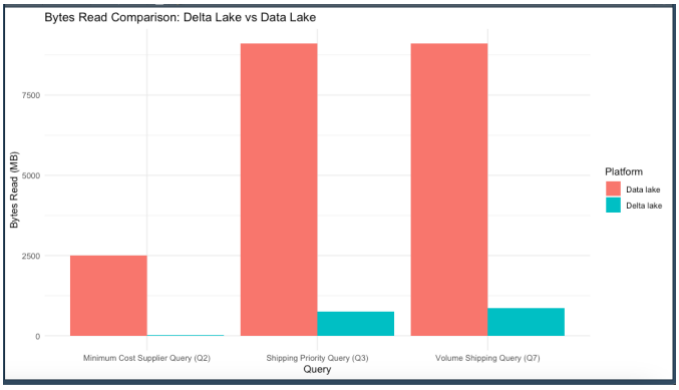
Key Takeaway

The additional processing in Delta Lake is not a drawback but a trade-off to ensure reliability, consistency, and long-term query performance. For data pipelines that need to scale and ensure high data quality, Delta Lake is worth the performance cost. However, for raw data ingestion and storage, a traditional Data Lake may suffice.

C. Query Performance

We have evaluated the effects of delta lake features in query performance using TPC-H benchmarking suite. Queries with complex computation needs are chosen for this exercise.





Key Observations

Delta Lake consistently has shorter durations compared to Data Lake for both queries, especially at larger data sizes.

Delta Lake reads significantly fewer bytes compared to Data Lake.

Causes of Performance Differences

Data Skipping

Delta Lake leverages data skipping by maintaining min/max statistics for each column in the transaction log. This allows Delta Lake to read only the files that contain relevant rows for the query, avoiding unnecessary file scans.

Data Lake lacks data skipping capabilities, so it scans entire files or even the entire dataset, resulting in significantly higher bytes read.

File Optimization (Z-Ordering)

Delta Lake uses Z-Ordering to physically group related data on disk (e.g., grouping rows with similar values for columns used in filtering). Queries with filters on such columns (e.g., supplier or shipping data) benefit from reduced I/O and faster execution.

Data Lake stores files without such physical optimization, resulting in higher I/O costs and slower execution.

Metadata Management

Delta Lake maintains metadata in the transaction log, which enables faster access to file-level statistics, schema details, and other query-relevant data.

Data Lake relies on external systems (e.g., AWS Glue or Hive) for metadata, introducing latency in query planning and execution.

IV. CONCLUSION

Delta Lake

Advantages: Delta Lake offers improved query performance and robust ACID guarantees, making it a preferred choice for production-grade data pipelines.

Limitations: Slower data loading compared to traditional data lakes due to the computational overhead of ensuring data integrity and optimization.

Use Cases: Ideal for:

- Reliable data engineering pipelines requiring ACID compliance for updates, deletes, and merges.
- Workflows that require frequent incremental data updates and schema enforcement
- Applications where high-performance querying and analytics are critical.

Data lake

Advantages: Data lakes excel at ingesting data quickly, making them ideal for environments where large-scale raw data storage is the priority.

Limitations: Query performance in data lakes is generally slower compared to advanced systems like Delta Lake due to the lack of indexing, optimization techniques, and consistency checks.

Use Cases: Suitable for organizations looking for:

- Cost-effective storage solutions.
- Scenarios where query performance or strict consistency is not critical, such as backup storage or environments focused on exploratory data analysis.

REFERENCES

Bennie Haelen and Dan Davis “Delta Lake: Up and Running, Modern Data Lakehouse architectures with Delta lake” : O’Reilly

Michael Armbrust, Ali Ghodsi, Reynold Xin and Matei Zaharia, “Lakehouse: A New Generation of Open Platforms that Unify Data Warehousing and Advanced Analytics”