Available online www.ejaet.com

European Journal of Advances in Engineering and Technology, 2020,7(12):101-106

Research Article ISSN: 2394-658X

Building Resilient Big Data Pipelines with Delta Lake for Improved Data Governance

Chandrakanth Lekkala

Email ID – Chan.Lekkala@gmail.com

ABSTRACT

The rapid development of data, thereby real-time dealing with analytics, has drawn the attention of enterprises in building better, scalable big data pipelines. Nevertheless, the big data architectures of the old school, like the data lake, which is based on Apache Hadoop or the cloud object store, often technologically suffer from inconsistency, quality and governance bottlenecks. The Delta Lake is an open-source storage layer, which allows one to ACID transactions and schema enforcement and gives an easy way to batch and stream data processing with data lakes. This paper examines how Delta Lake can be used to design stable big data pipelines that provide higher trust in data, allow automatizing catalogue entries, express improvements in performance, and ease governance. It begins by describing Delta Lake's core components, such as transactions and ACID properties, and its low-latency and high throughput implementation. Also, it covers handles how to attract Spark and other analytics engines. Last, it will present some real-world use cases with Delta Lake and how you can achieve them readily. The paper will examine the architectural patterns, best practices, and the primary use cases Delta Lake applies to in enterprise big data environments for data analytics, machine learning, and compliance purposes.

Key words: big data, data lake, data lake as Apache Delta Lake, data governance, data pipeline, data pipelines, ACID transactions, schema management, Spark, cloud—Lee The statement above concerns data engineering.

INTRODUCTION

Since the age of big data is here, organizations are the collators and processors of squeezed structured, semi-structured, and unstructured data from several sources to enhance data-centred decision-making. Today's traditional data warehouse and ETL-based systems must catch up in the scope, diversity, and speed required coping with the big data. Consequently, many stores have gotten into forming data lakes - large repositories in which raw information is kept as it is [1]. Data lakes, usually established based on a distributed file system like HDFS or Amazon S3, are scalable and flexible, granting more agility and flexibility for ample data storage and analytics.

However, data lakes also introduce new challenges around data quality, consistency, and governance: However, data lakes also introduce new challenges around data quality, consistency, and governance:

A. The water in these lakes tends to be from many sources that still need cleaning, conversion, or certification of the schemas. This situation has been causing problems with the same data sets, incompatible formats, inconsistencies, and even data corruption [2].

B. Data lakes use a schema-on-read approach that, firstly, somehow raw data has been filled into, and then the schema is applied only to query afterwards. Good here could be that, on the one hand, this increases the flexibility; however, the case could be where different users or applications, on the other hand, correctly interpret the same data differently [3].

Lakes usually do not ensure referential integrity and share low performance in transactional coordination, encouraging many data batching. The primary threat to "high availability" in such systems is possible data integrity issues caused by concurrent reads or writes, and recovery from failures is both time-consuming and lossful [4].

The data lake is a real problem because it deals with an increase in data volume and variety. It is commonly used as a hard-to-manage dumping ground. This translates into finding, assessing, protecting, and managing data assets throughout their lifecycle stage, which becomes the main challenge [5].

DATA LAKE CHALLENGES

However, separating storage in the legacy systems and emerging data pipelines is one of the challenges that might occur. Delta Lake, a new storage layer, has been proposed to minimize this disruption [6]. Delta Lake refers to an open-source project that allows you to build a Lakehouse architecture, i.e., a data platform with the flexibility of scaling up the data storage space and saving your Budget.

DELTA LAKE

Delta Lake, an open-source storage layer, is ACID transactional, schemata enforcing, and time-travel-capacitive [6]. This feature uses a standard open data format—Apache Parquet—and logs stores that keep track of all the changes made to the data. Delta Lake builds a scalable, reliable foundation for the whole data pipeline and analysis applications procurement.

CRITICAL FEATURES OF DELTA LAKE INCLUDE

A. ACID Transactions

Along with ACID compliance (Atomicity, Consistency, Isolation, and Durability), Delta Lakes offers complete availability for reads, writes and deletes. It uses an optimistic concurrency mechanism whereby writers take locks on their files and detect conflict by setting the transaction log as data comparison [7]. This means that even in the case of several simultaneous read and write operations, this transaction outcome will still be serializable and atomic. ACID compliance is the basis for building a universal analytics framework that guarantees data order, prevents data distortion, and reduces recovering time during failures.

B. Schema Enforcement and Involvement

Delta Lake has the option of a schema definition during the writing process to help keep the schema intact. This data validation happens by checking that the collected data meets the specified schema to keep the mismatched data from being fed in [6]. Data evolution is enabled for arts, crafts, architecture, music, religion, etc. Doing this supports schema evolution and provides a simple mechanism to easily add/modify columns over time as the data evolves. It uses schema versioning techniques and auto-merging to perform resolving operations on schema changes without breaking previous schema versions.

C. By using the same service, batch and streaming can be unified.

Delta Lake is the meta-storage layer, but batch and streaming operations data are stored in it. A Delta table can be a source when streaming or a sink with exactly one semantics and incremental reading [8]. Such a pattern makes it easier to assemble data pipelines with tweets or other historical and current data inputs.

D. Time Travel and Data versioning are trending topics that have gained great attention from scholars across multiple fields.

The Delta data logger registers a transaction associated with changes to the data, including information necessary for making queries regarding previous versions or rolling back at a specific time [9]. The "time travel" feature helps audit, re-yield experiment results, and regain lost data after insightful deletions/changes.

E. Integration of Spark and Data Frames

Delta Lake offers native APIs as post-Spark and Spark SQL to be deeply incorporated. By reading and writing from/to Delta tables with Spark DataFrame/Dataset APIs and SQL, users need not spend extra effort and time learning new things [11]. Delta also helps to operate Spark's advanced features, such as Streaming and Machine learning pipelines.

DELTA LAKE ARCHITECTURE

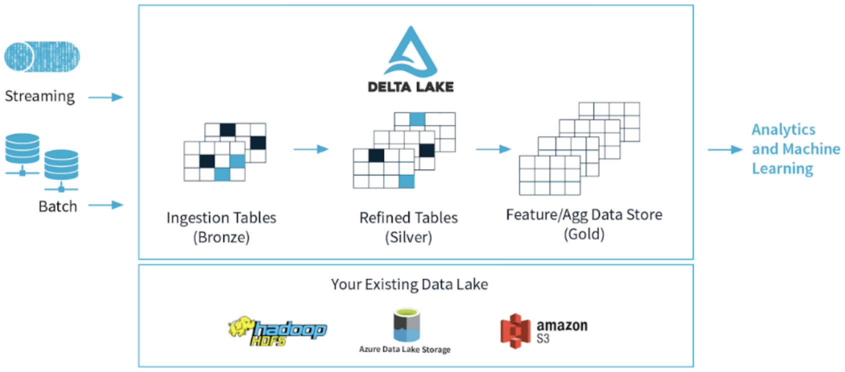

Figure 1: Delta Lake layered architecture. [12]

Fundamentally, Delta Lake has three layers: the data layer, the transactional layer, and the metadata layer. The storage layer contains the original data files (usually Parquet) and the logs that preserve append-only transactions held in a distributed file system or object storage. The execution layer is the Spark abstraction for individual reads/writes and stores the data meta information. The access layer is the function that gives us the Delta APIs that allow all the apps and services outside of Delta to connect to the system [12].

In short, Delta Lake makes turbos out of data lakes, adding chops like transactional, reliability, and management features usually found in traditional data warehouses. The Delta engine optimized for massively scalable data lakes combines the ACID properties of data pipelines with the openness of the data lakes. It enables building data pipelines that are not degraded by failures and have high data quality scores.

INTEGRATION WITH APACHE SPARK

An essential advantage of using Delta Lake is that, as integrated software, it works smoothly with Apache Spark, a standard for processing big data and is widely used. Delta Lake improves Spark on the data manipulation component with the transaction and zero-copy and on improved metadata handling and performance [10].

A. Spark SQL and Data Frame APIs combine structured data from tables and accomplish computations using more than one data source.

The idea behind Delta Lake here is that users can select familiar SQL queries and Data Frame APIs as read methods for writing purposes. A delta table can be created as a non-persistent or view into view in SQL by registering it as a temporary or global view, respectively, which makes it accessible [11]. Being Data Frame API-exposed, these APIs present the opportunity to conduct various operations such as Insertions, Deletions, and Merges.

B. Spark Structured Streaming

Structured streaming in Spark can send and receive data at Delta Lake. Due to the resulting join, streaming and batch queries on the same dataset can be executed in or close to real-time [13]. Delta was designed for easy integration with data centres. As opposed to cumbersome batch processing, it can track which data is new to carry on incremental processing downstream. It acts as the sink for streaming data and supports idempotent writes and exactly-once semantics, thus providing a way for fault-tolerant and proven reliability.

C. Performance Optimizations

Delta Lake introduces several performance optimizations that speed up Spark workloads: Data skipping the primary reason for the transformation is to preserve the organization's professionalism while caring for customers. This ensures speed in the data retrieval process (irrelevant files' reading skipping), thus boosting performance.

Z-Ordering: Delta tabular sorting can help shard the data in multiple columns by physically co-locating the data with identical values. The fact that this index requires only writing costs for updates and running costs for reads [14] means it is much faster. Indexes Based on Memory Need Info Caching: Delta eliminates the repeated ones that will come up on cloud indexing of data (object on the store), which is the list of files. Thus, data that should be frequently accessed during run time can also be copied into memory or on SSDs [15].

D. Other Integrations

Beyond Spark, Delta Lake integrates with various tools and frameworks in the extensive data ecosystem: Beyond Spark, Delta Lake integrates with various tools and frameworks in the extensive data ecosystem:

Presto and Athena: Delta tables can be queried by the query engines of both Presto and Amazon Athena, empowering analysts' capabilities to make nothing but bursting ad hoc analytics [16].

Airflow and Data bricks: Workflow managers like Apache Airflow can coordinate data Delta pipelines. Data bricks, trained behind the scenes at Delta, offer a single platform for creating and developing Delta Lake maintenance pipelines [17].

Hive Metastore: Delta can i. uses the Hive megastore as an external catalog for seating table metadata, which allows for interoperability with Hive [18].

BI Tools: Delta data lakes also have table links that languages such as Tableau, PowerBI, and Looker can comprehend, which makes it possible for self-service analytics engines [19].

DELTA LAKE INTEGRATIONS

Delta Lake does that by offering Spark integration and broadening the access spectrum: Delta Lake makes a wide range of tools and use cases, from end-to-end data pipelines to advanced analytics and machine learning applications, accessible.

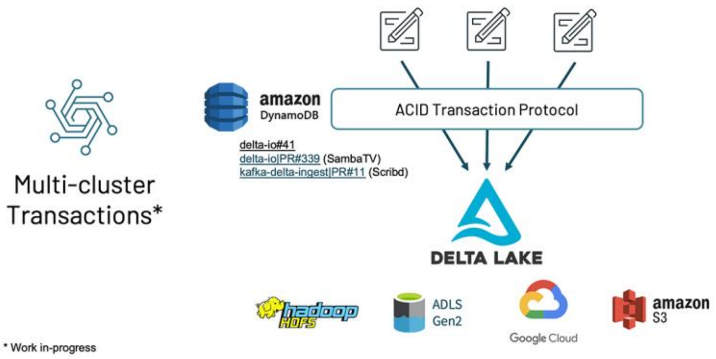
Multi-cluster Transactions*
amazon DynamoDB
delta-io#41
delta-io|PR#339 (SambaTV)
kafka-delta-ingest|PR#11 (Scribd)
ACID Transaction Protocol
DELTA LAKE
hadoop HDFS
ADLS Gen2
Google Cloud
amazon S3
* Work in-progress

Figure 2: Delta Lake integration with Spark and other tools. [20]

REAL-WORLD USE CASES

Many companies in different industries have already integrated Delta Lake to build resilient and robust Spark-backed data platforms. Some prominent use cases include: Some prominent use cases include:

A. Real-Time Recommendation Systems

Edmunds, known as a provider of all your automotive needs, taps into the betting power of Delta Lake for instant car recommendations [21]. The streaming data is retrieved by the cluster and stored in the Delta tables, which are eventually joined with the batch feature data. The integrated Delta tables have an underlying data warehouse model with data structures; this makes them the feature store for training machine-learning models, which predict user preferences. For Delta, batch and streaming data are processed simultaneously, making the recommendation pipeline more straightforward and newer.

B. Fraud Detection

PayPal combined the usage of Delta Lake and made a fraud detection platform that is done through real-time processing of petabytes of transactions [22]. Streaming data is deposited in delta tables and treated with reference data and algorithmic analysis, which detect unusual patterns algorithmically. ACID compliance by the Delta streams and schema validation have a data quality guarantee. Its modified operation of time travel capability allows it to retrain when new historical data arrives, exhibiting new fraud patterns.

C. Regulatory Reporting

The financial software company Senses, which was evolving its previously traditional EDW to a Delta Lake architecture, did it to get better future times of changes and lower costs for such freedom [23]. Data is consumed from source systems into Delta tables on Amazon S3 via Kafka and Spark streaming, leveraging Spark SQL and table partitioning. The data collection is done, followed via curation and submission to the finance teams for reporting to come out with the required reporting to comply with regulatory and management requirements. Enhanced features like support for ACID transactions, audit logs, and data retention have simplified procedures with compliance workflows.

D. IoT Analytics

GE Aviation created the Lambda Lake pipeline to handle petabytes of engine sensor data for predictive maintenance on board [14]. Engine data means being brought together in Delta tables, where the manufacturing, MRO, and other systems store process information. Data Scientists dig into the data-shrewdness and employ Jupyter notebooks to develop engine failures-predictive models. Delta's dlp and time travel mechanism allows users to modify existing models and test their performance. This highlights how Delta Lake supports data reliability, simplifies architecture, etc., and enables innovative applications across various domains, including real-time analytics and machine learning.

RECOMMENDATIONS AND FUTURE LINES OF ACTION OR SUGGESTIONS

Here are some best practices to consider when building Delta Lake pipelines: Here are some best practices to consider when building Delta Lake pipelines:

Use the Delta transaction log for metadata: Maintain data manifests and track data flow metadata populated by sources, transformations, and data quality metrics in the Delta log. This must be transparent from edge to edge and ease debugging and auditing.

IMPLEMENT DATA QUALITY CHECKS

[1]. Implement Delta Validation Schema along with Constraint Checking to guarantee data quality.
[2]. Use our AI to write for you about any assignment or topic.
[3]. Include the custom check constraints spotted off the domain-specific abnormalities [15].

Optimize storage layout: Control file sizes, apply Z order to prevent query-related issues, and use data skipping for faster query performance. Splice small files to the others and assemble them to form larger files to avoid the impact of metadata overhead [14].

A. Secure and govern data:

Securing and governing data is crucial in Delta Lake pipelines. Apply table and row-level control using Spark SQL permissions and Apache Ranger policies on views. Rest lambda's Delta allows for complicated data retention and compliance [23]. Implement role-based access control to ensure that only authorized users can access sensitive data. Use encryption techniques to protect data at rest and in transit. Regularly audit data access logs to detect any suspicious activities. Establish data governance policies and procedures to maintain data quality, consistency, and integrity across the organization. Ensure compliance with relevant data protection regulations, such as GDPR or HIPAA, depending on the industry and geographic location. Leverage data lineage and provenance features to track data movement and transformations throughout the pipeline. Implement data masking and anonymization techniques to protect sensitive information while still allowing for data analysis and reporting.

B. Monitor and manage pipelines:

This is instrumental in developing popular tools like Apache Spark and Airflow, which create production-grade CI/CD workflows for data pipelines. Alerts of Data Quality problems and performance lags should be implemented [17].

As for future research directions, here are some areas that can be explored; Polyglot support: Study the implementations of a native language library for non-JVM languages like Python and R to make them accessible to data science dependencies.

Unstructured data: Investigate ways of turning Delta Lake into a universal data store for text, image, and video data, maybe creating links with supplementary tools such as Apache Tika.

Governance automation: Build mechanisms that simplify tasks for aligning data in Delta tables with database standards and descriptions. Implement machine learning in data lifecycle management.

Query federation: Dissect the gravity that Delta Lake's expression of querying in various clusters, clouds, and zones could hold, possibly by interlinking with federated query engines for Presto, for Example.

Block chain integration: Explore the diversifications of block chain technologies with Delta Lake to get more data pipelines for diverse private sectors [19].

CONCLUSION

Various businesses have long adopted data lake setups on Hadoop and cloud object stores such as AWS and GCP, but significant problems are associated with data reliability and governance. Delta Lake solves these loopholes by providing a transactional layer that sits above Data Lakes. Through ACID characteristics, schema verification, time travel, etc., Delta Lake allows the building of pipelines as with ACID properties; data can guarantee integrity and consistency, which is critical in the process. Integration with Spark and other tools allows for formulating a unified programming environment that simplifies the ones involving batch, streaming, and machine learning workloads.

Actual-time uses have shown that Delta Lake is robustly adapted by different industries, for instance, real-time analyses, fraud detection, regulatory compliance, and AI. Enacting the suggestions earlier illustrated creates credibility, which could lead to the progress of future research about Delta Lake. Today, the world generates an unprecedented amount of data, and utilizing this data is becoming one of the critical factors for business prosperity; thus, having resilient big data platforms is essential. Delta Lake demonstrates its potential to fill the gap and contribute to uncovering the next step in digital transformation through data-driven development.

REFERENCES

[1]. Khine, P.P. and Wang, Z.S., 10/2018. Data lake: a new ideology in big data era. In ITM web of conferences (Vol. 17, p. 03025). EDP Sciences.

[2]. Gorelik, A., 07/2019. The enterprise big data lake: Delivering the promise of big data and data science. O'Reilly Media.

[3]. Quix, C., Hai, R. and Vatov, I., 12/2016. Metadata extraction and management in data lakes with GEMMS. Complex Systems Informatics and Modeling Quarterly, (9), pp.67-83.

[4]. Gaffoor, Z., Pietersen, K., Jovanovic, N., Bagula, A. and Kanyerere, T., 10/2020. Big data analytics and its role to support groundwater management in the southern African development community. Water, 12(10), p.2796.

[5]. O'Leary, D.E., 09/2014. Embedding AI and crowdsourcing in the big data lake. IEEE Intelligent Systems, 29(5), pp.70-73.

[6]. Armbrust, M., Das, T., Sun, L., Yavuz, B., Zhu, S., Murthy, M., Torres, J., van Hovell, H., Ionescu, A., Łuszczak, A. and Świtakowski, M., 08/2020. Delta lake: high-performance ACID table storage over cloud object stores. Proceedings of the VLDB Endowment, 13(12), pp.3411-3424.

[7]. Beheshti, A., Benatallah, B., Nouri, R., Chhieng, V.M., Xiong, H. and Zhao, X., 03/2017. Coredb: a data lake service. In Proceedings of the 2017 ACM on Conference on Information and Knowledge Management (pp. 2451-2454).

[8]. Zhou, N., Hu, X., Wang, L. and Zhao, J., 12/2017. eZlake: a unified data lake service based on semantic enrichment. In Asia-Pacific Web (APWeb) and Web-Age Information Management (WAIM) Joint Conference on Web and Big Data (pp. 87-102). Springer, Cham.

[9]. Strengholt, P., 11/2020. Data Management at scale. " O'Reilly Media, Inc.".

[10]. Armbrust, M., Das, T., Torres, J., Yavuz, B., Zhu, S., Xin, R., Ghodsi, A., Stoica, I. and Zaharia, M., 05/2018. Structured streaming: A declarative api for real-time applications in apache spark. In Proceedings of the 2018 International Conference on Management of Data (pp. 601-613).

[11]. Armbrust, M., Xin, R.S., Lian, C., Huai, Y., Liu, D., Bradley, J.K., Meng, X., Kaftan, T., Franklin, M.J., Ghodsi, A. and Zaharia, M., 05/2015. Spark sql: Relational data processing in spark. In Proceedings of the 2015 ACM SIGMOD international conference on management of data (pp. 1383-1394).

[12]. Databricks Delta Project. https://delta.io/

[13]. Armbrust, M., Das, T., Davidson, A., Ghodsi, A., Or, A., Rosen, J., Samvaliyev, N., Stoica, I., Woodford, K., Xin, R. and Zaharia, M., 05/2018. Structured streaming: A declarative API for real-time applications in Apache Spark. In Proceedings of the 2018 International Conference on Management of Data (pp. 601-613).

[14]. Taherizadeh, S. and Stankovski, V., 02/2019. Dynamic multi-level auto-scaling rules for containerized applications. The Computer Journal, 62(2), pp.174-197.

[15]. Databricks Documentation. https://docs.databricks.com/delta/optimizations/delta-cache.html

[16]. Hai, R., Geisler, S. and Quix, C., 06/2016. Constance: An intelligent data lake system. In Proceedings of the 2016 international conference on management of data (pp. 2097-2100).

[17]. Almeida, P. and Bernardino, J., 10/2015. A comprehensive overview of open source big data platforms and frameworks. Int. J. Big Data, 2, pp.1-19.

[18]. Ruan, W., Chen, Y. and Forouraghi, B., 06/2019. On Development of Data Science and Machine Learning Applications in Databricks. In World Congress on Services (pp. 78-91). Cham: Springer International Publishing.

[19]. Costa, R.C., 07/2020. Implementation of a big data cloud-based catalog using open data (Master's thesis, Universidade de Évora).

[20]. Fang, H., 05/2015. Managing data lakes in big data era: What's a data lake and why has it became popular in data management ecosystem. In 2015 IEEE International Conference on Cyber Technology in Automation, Control, and Intelligent Systems (CYBER) (pp. 820-824). IEEE.

[21]. Russom, P., 11/2017. Data lakes: Purposes, practices, patterns, and platforms. TDWI Best Practices Report, pp.1-35.

[22]. Mathis, C., 06/2017. Data lakes. Datenbank-Spektrum, 17(3), pp.289-293.

[23]. Lwakatare, L.E., Raj, A., Bosch, J., Olsson, H.H. and Crnkovic, I., 05/2019. A taxonomy of software engineering challenges for machine learning systems: An empirical investigation. In Agile Processes in Software Engineering and Extreme Programming: 20th International Conference, XP 2019, Montréal, QC, Canada, May 21–25, 2019, Proceedings 20 (pp. 227-243). Springer International Publishing.