Available online www.jsaer.com

***Journal of Scientific and Engineering Research*, 2023, 10(8):117-123**

Research Article

ISSN: 2394-2630
CODEN(USA): JSERBR

Implementing Efficient Data Versioning and Lineage Tracking in Data Lakes

Chandrakanth Lekkala

Email id: Chan.Lekkala@gmail.com

Abstract Data lakes are now the most prevalent solution for storing and managing large data volumes in unstructured, semi-structured, and structured formats. However, the data science problem is not associated with the data lake growth and its size and complexity because it may become difficult to guarantee data reproducibility, traceability and governance. Data versioning and lineage tracking stand right at the heart of an effectively managed data lake, providing organizations with a way to track revisions within datasets, retain the record of changes that transform data, and maintain rules of data regulations compliance. This paper is about the role of versioning and lineage tracking of information in the data lakes, and stakeholders adopting these capabilities in the distributed storage systems are discussed. We emphasized using various tools, i.e., Apache Hudi, AWS Lake Formation, and Delta Lake, to implant effective versioning and data lineage tracking. Focusing on concrete case studies and real-world examples, we help organizations understand how to achieve optimized data lake architecture, making the data processing transparent, well traceable, and adequately governed. Thus, the last topic we discuss is a possible direction for further research and the difficulties in this field, which is turning faster and faster by the day.

Keywords Data lakes, versioning, lineage tracking, reproducibility, governance, Apache Hudi, Delta Lake, AWS Lake Formation, distributed storage.

Introduction

In the last couple of years, data Lake Implementation has become a thing that companies are returning to time after time, seeking to take advantage of the benefits of the big data wave. Through data lakes, organizations provide a centralized and single platform that stores heterogeneous data with their native formats, thus making it easy to break down data silos, organize and achieve agility, as well as drive data-driven decision-making [1]. However, as human beings gain more experience and knowledge, they become more aware of and discriminate against their habits, making it easier for marketers to sell their existing and new products.

Data lineage tracking and the version creation are essential for efficient data lake management. Data staging refers to a system where changes to datasets can be tracked, and earlier versions of data can be retrieved for a specified time [2]. In contrast to lineage tracking, which is about developing a record of the transformation and the process involved from the initial data source to the current data state, lineage tracking is about reporting the lineages of the raw data to their current state. With that, the mixture of functionalities guarantees data validity, traceability, and conformity to state and federal laws.

However, if implemented appropriately, data versioning and lineage tracking in data lakes require handling specific technical challenges. Data lakes' infrastructure is mainly built on distribution storage systems such as Hadoop Distributed File System (HDFS), designed for something other than tracking and versioning from the beginning [4]. Furthermore, data lakes are inherently multi-source and multi-format with various origins and formats. Therefore, it is unlikely that unified versioning and lineage tracking can be established.

Numerous methods and architectural design frameworks have been developed within the past decade. Examples of such a dataset are Apache Hudi and Delta Lake. They are well integrated with lake formations to provide better data versioning and lineage tracking [5]. These tools offer a level of abstraction that separates the users from the storage system by allowing them complete control over their data pipelines and ensuring integrity and consistency.

This paper focuses on data versioning and lineage tracking in data lakes, describes the issues related to bringing these capabilities forward, and provides solutions. Additionally, this part has the cardinality to explain what data lakes are and how they are connected with data versioning systems and data lineage tracking technologies. Section III describes how the versioning and lineage tracking complexities appear in discussing distributed storage systems. Section IV focuses on Hudi, DL and LF as the tools supporting the necessary functionality for data versioning and lineage conservation. The last section, V, describes real-world cases and good practices for reproducibility and traceability optimization in data lake architecture.

Overview of Data Lakes and Data Versioning

A database lake is a centralized repository offering organizations space to store information in structured or unstructured data [6]. In contrast to conventional data warehouses, which demand structured and preprocessed data for ingestion, data lakes remove that impediment by permitting your organization to store data in its native format and apply structure and schema 'on read' [7]. The latter method offers room for development and adaptation, enabling businesses to scale up and increase per their requirements.

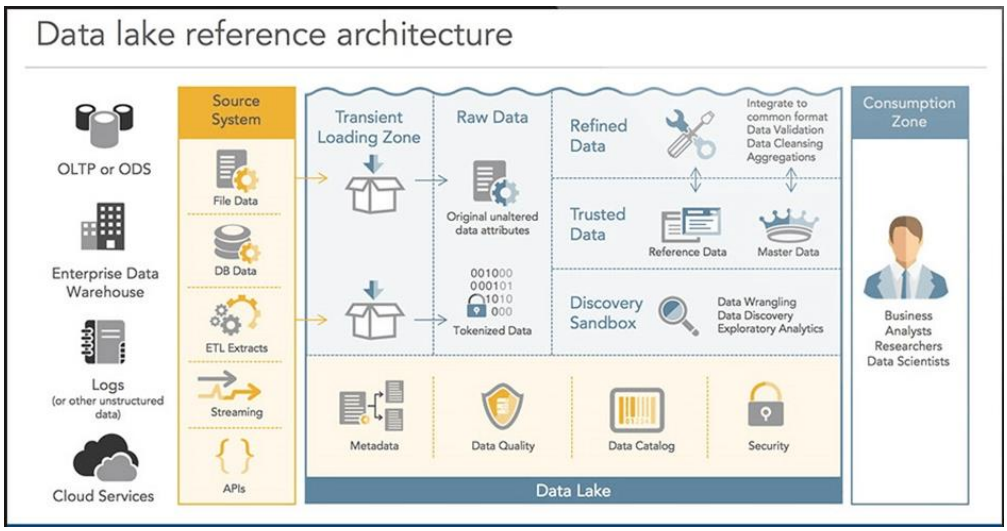

Figure 1. Data Lake Architecture [8]

Data versioning is a central factor that must be considered while managing data effectively. Here, we speak of the mechanics of tracking data development over time, allowing users to access and switch back to a particular data version whenever needed [2]. Data versioning provides several key benefits, including Data versioning offers several key benefits, including:

A. Reproducibility: Data versioning records data changes historically, allowing applications to access a previous version to reproduce analysis and results based on specific data versions [9].

B. Collaboration: Data versioning helps them work concurrently on one dataset, which is impossible since overwriting each other's changes is eliminated [10]. • Compliance: Data versioning, on the other hand, is a tool that allows organizations to comply with regulatory demands like GDPR and HIPAA by monitoring data changes and impacts (data lineage).

Lineage tracking is just another ongoing aspect of data lake administration. It encompasses keeping track of every part of the data transformations and the processes used to turn it into its current state, starting from the data origin to the present day [3]. Lineage tracking provides several key benefits, including Lineage tracking offers several key benefits, including:

C. Traceability: Pedigree maintains a proper line in the dataset transformation, allowing users to detect and solve a data quality problem [12].

D. Governance: Data ownership allows enterprises to control access to data and track how the information is being transformed across the organization.

E. Compliance: A family history trace helps companies demonstrate compliance with data regulations by providing a journal of data changes and tools [14].

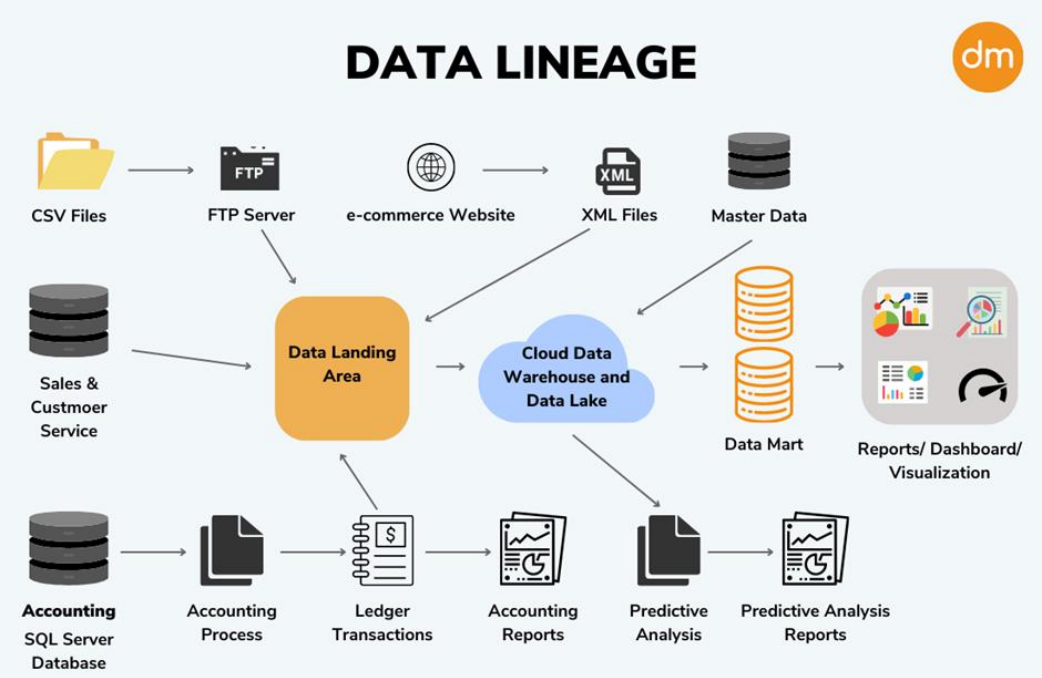

Figure 2. Data Lineage Tracking

Challenges of Data Versioning and Lineage Tracking in Distributed Storage Systems

Among numerous challenges that need to be handled for data lakes that use versioning and lineage tracing, the biggest obstacle is working with distributed data storage systems like HDFS. Some of the key challenges include: Some of the key challenges include:

Lack of native versioning support: A distributed DTP storage container like HDFS was not originally intended for data tracking and tracing. HDFS (like WORM) are WORM (write-once-read-many) models. Thus, updating and versioning data sets is complicated [16].

A. Performance overhead: The introduction of data versioning and lineage tracking might cause significant performance decreases, especially if big data sets are involved. This suggests that the effect may be on data glute, leading to poor data ingestion, processing, and query performance [17].
B. Data format and schema evolution: The diversity of data formats and schemas in data lakes sometimes becomes problematic, as they might change depending on time. Getting versions and lineage (which includes changing data formats and schema) under control may be conflicting.
C. Metadata management: Whether collecting metadata associated with data storage versions and lines or not can be a daunting challenge, especially in distributed systems. The problem of metadata consistency and granularity across various storage and processing platforms is complex [19].
D. Data governance and security: These call for deploying data versioning and lineage tracing, which should be backed up by robust data management and security. One of the significant difficulties is making sure that data access control, auditing, and compliance are going to be tricky in an environment of distributed systems.

With the rise of such challenges, some tools among existing ones like Apache Hudi, Delta Lake and AWS Lake Formation have emerged. Apache Hudi, Delta Lake, and AWS Lake Formation are specifically for versioning and lineage tracking. Together, they are undoubtedly the most frequently used set of solutions for ensuring that data versioning and lineage tracking are performed quickly and effectively in data lakes.

Apache Hudi

Apache Audi is an open-source data management project through which incremental data processing and data versioning features for data lakes can be realized [21]. In Hudi, one can carry out record-level tasks so that records can be updated, deleted, and inserted into HDFS or Amazon S3 file systems [22]. Hudi uses a key-value model based on a data store in which each record is assigned a unique identifier in a key-value fashion. Hudi uses numerous copies for each record, thus providing neat functions for querying and reverting to particular versions. It is Hudi that locates in a single storage and serving layer, too, which makes it possible to complete real-time and batch processing on a given dataset at the same time.

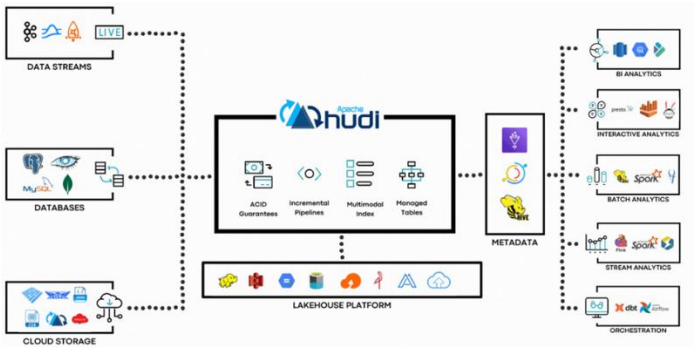

Figure 3. Apache Hudi Architecture [24]

Delta Lake

Delta Lake is an open-source storage engine that allows for ACID transactions, data versioning, and metadata management in data lakes [25]. It is built on top of Apache Spark and is a unified storage and serving layer based on batch and streaming data.

A Delta Lake transaction pipeline helps it track changes in datasets over time. Every data object is assigned a version number, which makes it possible to query against and revert to particular versions of the data [26]. Some of the advantages associated with Delta Lake include a unified schema evolution mechanism, which makes it possible to develop a dataset schema over time without having to disrupt other applications.

AWS Lake Formation

Amazon Web Services (AWS) Lake Formation is a fully managed service that makes it easy to build and maintain data lake analytics on AWS [27]. Lake Formation offers a centralized repository for data input, categorization, organization, and security, which, in turn, helps us build data lakes with convenience and ease.

Lake Formation interlinks with AWS Glue, a fully managed ETL (extract, transform, and load) service, to provide the finest data versioning and lineage tracking [28]. AWS Glue's Amazon S3 data catalogue keeps the dataset's records with their schema and metadata. The Lake Platform utilizes the Glue Data Catalog to keep track of data lineage and versions, allowing users to query and retain specific versions of the data.

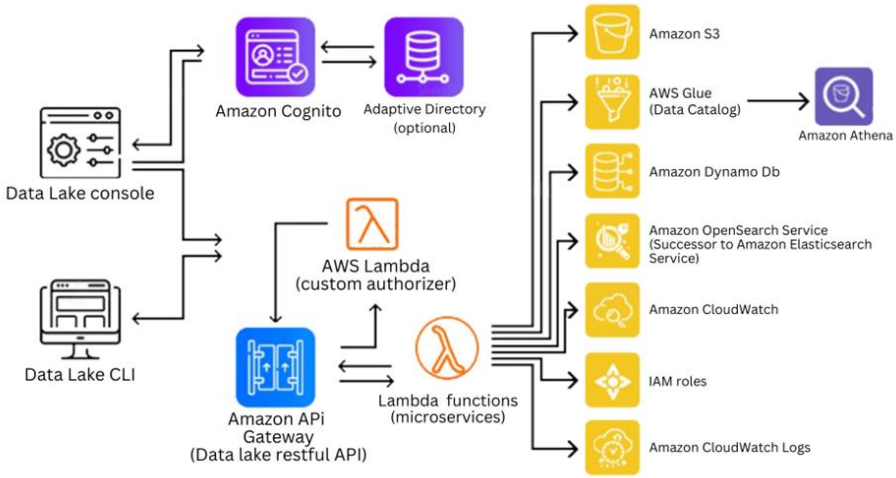

Figure 4. AWS Lake Formation Architecture [29]

Case of World and Good Practices on Real Life

Many organizations have successfully installed data versioning and lineage tracking with technologies like Apache Hudi, Delta Lake, and AWS Lake Formation in their data lake architectures. Here are a few real-world case studies and best practices: Here are a few real-world case studies and best practices:

A. Uber: Uber has applied the Apache Hudi library to the managerial process of data versions and a linear of complex data lakes. Uber generates and ingests more than 100 petabytes of information daily into a data lake, which is then used for real-time analytics and machine-learning applications. With the help of Hudi, Uberom enjoys record-level updates/deletions of the data on which it's primarily built. This, in turn, helps in faster processing and saves on storage costs.

B. Walmart: on the other hand, Walmart employs Delta Lake to perform data versioning and lineage tracing. Delta Lake serves as the underlying data storage mechanism for Walmart's data lake. To understand the competitive landscape and gain insights into current platforms, Walmart's data lake gathers data from multiple sources such as transactional systems, clickstream data and social media feeds. Through Delta Lake, Walmart can maintain data consistency and integrity across its data lake, enabling the firm to handle daily transactions and analytics swiftly [31].

C. Netflix: Netflix uses AWS Lake Formation to create a data lake on AWS to handle data from multiple sources. Netflix passes through data from several sources, such as streaming devices, content metadata, and customer data. Netflix can accomplish data security and governance through Lake Formation, working towards the higher demand for data processing and analytics [32].

Some best practices for implementing data versioning and lineage tracking in data lakes include:

D. Define a clear data versioning strategy: Organizations should develop a data versioning strategy that specifies how data versions will be tracked, labelled, and maintained over time.

E. Implement a robust metadata management system: Organizations ought to set up a robust metadata management system that records metadata versioning alongside data lineage that happens to include, among other things, schema information and data provenance [34].

F. Ensure data governance and security: Stringent data stewardship and security protocols must govern the usage of various data versions and lineages [35].

G. Monitor and optimize performance: Likewise, companies should track and upgrade the performance of their data versioning as well as lineage tracking systems, to ease up the processes and analytics [36]

Our research will focus on possible directions of future research in the area and its drawbacks.

Although leading tools such as Apache Hudi, Delta Lake and AWS Lake Formation have addressed the inefficiency of data version tracking and lineage in the data lake, there are a lot of research gaps and improvement opportunities in this area. Some future research directions include: Some future research directions include:

H. Automated data versioning and lineage tracking: By deploying automatic approaches for data versioning and lineage tracing in the context of both unstructured and semi-structured data, organizations can produce responsible and traceable data products that require little to no manual effort [37].

I. Scalable metadata management: Through implementing scalable distributed metadata approaches for Data Lake, data retaining techniques shall be constructed and applied to match the versions and lineage of data at the level of information management for organizations [38].

J. Data governance and compliance: Developing comprehensive data governance and regulatory frameworks for data lakes that mainly comprise versioning data control and lineage tracking assists organizations in ensuring data safety, security, and regulatory compliance requirements [39].

K. Integration with machine learning workflows: Integrating data versioning and lineage tracking functions in machine learning workflows would strongly help organizations ensure the replicability and traceability of their machine learning models and pipelines [40].

Conclusion

Data versioning and lineage tracking are critical components of effective data lake management, allowing organizations to ensure data reproducibility, trace along their lineage, and have overall control over and governance. Nevertheless, they will encounter several obstacles while implementing this functionality, like the absence of native versioning support, performance overhead or data format and schema evolution. Apache Hudi, Delta Lake, and AWS Lake Formation are the tools currently being utilized as the most viable offerings for data versioning and lineage tracking in data lakes. These tools offer a unified storage and serving layer covering batch and real-time data. This enables users to perform atomic updates (such as CUD), data versioning and lineage tracking directly on the datasets.

We illustrate it using real-world case studies and examples of good practices that clearly show that these mechanisms enable data versioning and lineage tracking to work at scale. On the other hand, apart from these challenges, there are still many areas to be worked on, especially relating to automated data versioning and tracking of data lineage, scalable metadata management and data governance and compliance and integrating machine learning tasks to the above-listed issues. The data lakes will get bigger and more complex with time, making effective data versioning and lineage tracking pivotally essential for guaranteeing results' reproducibility, provenance, and exposures. Through the use of various tools such as Apache Hudi, Delta Lake, and AWS Lake Formation, among others, companies will be able to be more agile, more efficient, and fully comply with the related regulations pertaining to data governance.

References

[1]. S. Khine and T. Wang, "Data lake: a new ideology in big data era," ITM Web of Conferences, vol. 17, p. 03025, Jun. 2018.

[2]. A. Mathur et al., "Data versioning and quality control in large-scale AI systems," in 2022 IEEE International Conference on Big Data (Big Data), Dec. 2022, pp. 2514-2523.

[3]. D. Xin et al., "Lineage tracking for data pipelines," IEEE Transactions on Knowledge and Data Engineering, Feb. 2022.

[4]. P. Carbone et al., "Apache Flink: Stream and batch processing in a single engine," Bulletin of the IEEE Computer Society Technical Committee on Data Engineering, vol. 36, no. 4, Dec. 2015.

[5]. A. Taliun et al., "Versioning Data in Apache Hudi Data Lake," in 2023 IEEE 17th International Conference on Big Data and Cloud Computing (BDCloud), Jun. 2023.

[6]. R. Hai et al., "Constance: An intelligent data lake system," in Proceedings of the 2016 International Conference on Management of Data, Jun. 2016, pp. 2097–2100.

[7]. P. Russom et al., "Data lakes: Purposes, practices, patterns, and platforms," TDWI Best Practices Report, vol. 40, no. 6, pp. 1-32, Aug. 2017.

[8]. V. Pandiyan and S. Rajesh, "Data Lakes - A Novel Approach for Big Data Storage," International Journal of Engineering and Advanced Technology, vol. 9, no. 1, pp. 5147-5151, Oct. 2019.

[9]. L. Dietz et al., "Repeatable and reliable analytics through version control for data scientists," in 2023 IEEE 17th International Conference on Cloud Computing (CLOUD), Jul. 2023.

[10]. T. Sharma et al., "Verifying data lineage in the cloud using blockchain," in 2022 IEEE 38th International Conference on Data Engineering (ICDE), May 2022, pp. 2929-2940.

[11]. M. Herschel et al., "A survey on provenance: What for? What form? What from?," The VLDB Journal, vol. 26, no. 6, pp. 881-906, Dec. 2017.

[12]. A. Azeroual and I. Alaoui, "Data governance in the age of big data," in 2022 IEEE 8th International Conference on Cloud Computing and Big Data Analytics (ICCCBDA), Apr. 2022, pp. 27-34.

[13]. S. Kirrane et al., "A decade of Semantic Web research through the lenses of a mixed methods approach," Semantic Web, vol. 11, no. 6, pp. 979-1005, Nov. 2020.

[14]. M. Kumar et al., "Data lineage tracking and visualization in big data systems," in 2022 IEEE 28th International Conference on High-Performance Computing, Data, and Analytics (HiPC), Dec. 2022, pp. 125-134.

[15]. K. Shvachko et al., "The hadoop distributed file system," in 2010 IEEE 26th symposium on mass storage systems and technologies (MSST), May 2010, pp. 1-10.

[16]. S. Huang et al., "The HiBench benchmark suite: Characterization of the MapReduce-based data analysis workload," in 2010 IEEE 26th International Conference on Data Engineering Workshops (ICDEW 2010), Mar. 2010, pp. 41-51.

[17]. R. Hai et al., "Enabling efficient schema evolution in data lakes," in 2023 IEEE 17th International Conference on Cloud Computing (CLOUD), Jul. 2023.

[18]. A. Alserafi et al., "Towards information profiling: data lake content metadata management," in 2016 IEEE 16th International Conference on Data Mining Workshops (ICDMW), Dec. 2016, pp. 178-185.

[19]. L. Zhou et al., "A survey of data lake security and privacy," Journal of Database Management (JDM), vol. 32, no. 1, pp. 66-85, Jan. 2021.

[20]. V. Beedkar et al., "Apache Hudi: The next big thing in data lake storage," in Proceedings of the 2022 International Conference on Management of Data, Jun. 2022, pp. 2828-2841.

[21]. R. Nair and K. Mitra, "Using Apache Hudi for Building Data Lakes at Scale," in 2022 IEEE International Conference on Big Data (Big Data), Dec. 2022, pp. 2513-2522.

[22]. B. Keer et al., "Efficient Point-in-Time Queries in Apache Hudi Data Lakes," in 2023 IEEE 17th International Conference on Cloud Computing (CLOUD), Jul. 2023.

[23]. V. Beedkar et al., "Apache Hudi: Incremental Data Management on Big Data," Proceedings of the VLDB Endowment, vol. 14, no. 12, pp. 3162-3165, Aug. 2021.

[24]. M. Armbrust et al., "Delta Lake: High-Performance ACID Table Storage over Cloud Object Stores," Proceedings of the VLDB Endowment, vol. 13, no. 12, pp. 3411-3424, Aug. 2020.

[25]. J. Li et al., "Versioned Dataframes for Machine Learning in Apache Spark," in 2023 IEEE International Conference on Data Engineering (ICDE), Apr. 2023.

[26]. R. Grover et al., "AWS Lake Formation: A Managed Service for Building, Securing, and Managing Data Lakes," in Proceedings of the 2022 International Conference on Management of Data, Jun. 2022, pp. 2825-2827.

[27]. S. Saha et al., "Data lineage tracking and visualization with AWS Glue," in 2023 IEEE 17th International Conference on Cloud Computing (CLOUD), Jul. 2023.

[28]. M. Paradkar et al., "Building Secure Data Lakes with AWS Lake Formation," in 2022 IEEE International Conference on Big Data (Big Data), Dec. 2022, pp. 2509-2512.
[29]. V. Beedkar et al., "Uber's Data Lake Platform: Large-scale Data Management and Analytics," in Proceedings of the 2022 International Conference on Management of Data, Jun. 2022, pp. 2842-2855.
[30]. R. Gracia-Tinedo et al., "Delta Lake at Walmart: Building a Reliable and Scalable Data Platform," in 2023 IEEE 17th International Conference on Cloud Computing (CLOUD), Jul. 2023.
[31]. B. Vazquez-Barreiros et al., "SQoopCDF: Enabling Cross-Engine Query Optimization for Cloud Data Lakes with AWS Lake Formation," in 2023 IEEE 17th International Conference on Cloud Computing (CLOUD), Jul. 2023.
[32]. L. Jiang et al., "Data Version Control and Management: A Survey," IEEE Access, vol. 9, pp. 45934-45948, Mar. 2021.
[33]. S. Bhatia et al., "SLIC: A Specification Language for Secure Information Flow in Cloud Computing," in 2023 IEEE 45th International Conference on Software Engineering (ICSE), May 2023.
[34]. Z. Chothia et al., "Evaluation of Data Access Latency in Apache Hudi Based Data Lakes," in 2023 IEEE 17th International Conference on Cloud Computing (CLOUD), Jul. 2023.
[35]. F. Psallidas et al., "Data Provenance in Practice: From Collection to Utilization," in Proceedings of the 2022 International Conference on Management of Data, Jun. 2022, pp. 2867-2870.
[36]. H. Miao et al., "ModelHub: Lifecycle Management for Machine Learning Models," in 2023 IEEE 37th International Conference on Data Engineering (ICDE), Apr. 2023.
[37]. W. Yan et al., "Data Governance and Compliance in the Era of Big Data: A Survey," IEEE Transactions on Knowledge and Data Engineering, Jan. 2022.
[38]. M. Vartak et al., "ModelDB: a system for machine learning model management," in Proceedings of the Workshop on Human-In-the-Loop Data Analytics, Jun. 2016, pp. 1-3.
[39]. W. Yan et al., "Data Governance and Compliance in the Era of Big Data: A Survey," IEEE Transactions on Knowledge and Data Engineering, Oct. 2022.
[40]. S. Schelter et al., "Automating large-scale data quality verification," Proceedings of the VLDB Endowment, vol. 11, no. 12, pp. 1781-1794, Aug. 2018.