[6]. Armbrust M, Das T, Sun L, Yavuz B, Zhu S, Murthy M, Torres J, van Hovell H, Ionescu A, Łuszczak A, Świtakowski M. Delta lake: high-performance ACID table storage over cloud object stores. Proceedings of the VLDB Endowment. 2020 Aug 1;13(12):3411-24.

The problem being addressed?

The widespread adoption of cloud computing led various organizations to use cheap storage services like S3 across all domains, which triggered the need to track and version data accrued over the cloud. Since then, the concept of a data lake was proposed. A data lake is a centralized repository on cloud object stores that is used to store structured, semi-structured, and unstructured data. Regardless of various innovative architectures, however there was huge need to manage data in place over object stores. Therefore, the authors Zaharia et al. studied the problem, of managing data over object stores by enforcing ACID (Atomicity, Consistency, Isolation, Durability) properties, which eliminates the need of having multiple layers in data architecture.

Limitations of Existing work?

The authors stated that current data pipeline architectures contain separate layers for data lake and data warehouse to manage and report data. A Data warehouse is a centralized system containing facts and dimension tables, and is highly used to generate business insights. This not only increases the level of isolation but also requires huge computing resources to build data warehouses. In this process, the data lake is used as a storage component that consists of multiple files. Therefore, the authors mention that having multiple layers not only increases the computing cost, but also increases the network cost to move the data from one layer to another. Overall they propose a new architectural pattern that can achieve the functionality of both data warehousing and data lake embedded inside a single layer.

Summary of Proposed Strategy?

The authors proposed a system named delta lake that uses a novel Lakehouse architecture. It combines the functionalities of both the data lake and data warehouse by leveraging the benefits of the MVCC (Multi Version Concurrency Control) protocol and ACID properties on cloud object stores. Multi Version Concurrency Control is a technique used in database systems to manage concurrent access to data. The existing big data systems such as Spark store the table using the parquet format as a bunch of objects, without maintaining strong consistency. Parquet is a specialized big data based vectorized columnar format that is optimized for big data processing.

Every table on a delta lake has 3 underlying files - a log file, and a data file and a checkpoint file.

A log file is annotated with numerical values and is stored in a json format, with every key value entry in the file representing addition or remove operation on a particular table. The remove operation does not immediately remove the data instead it flags the record and removes it after particular time threshold. Also, it is used to manage the metadata of tables.

The table columns are partitioned and organized in the parquet format, whereas the checkpoint file which is also stored in Parquet is periodically compressed for optimized performance. In the same way the access protocols are designed to achieve serializable transactions over object stores despite the eventual consistency of cloud object stores.

The reading of tables follows sequence of steps:

a) When a read operation is initiated, the query engine pulls the last checkpoint ID from the tables log directory,

b) Later, from the above-obtained checkpoint ID the engine tries to perform a LIST operation and reconstructs the table state from most recent checkpoint.

c) Subsequently, it identifies all the add records in data objects excluding remove label records,

d) Finally it understands the objects that are relevant for read query and will parallelly query object store across a cluster using a compute engine.

The write operations are performed to eliminate any data corruption errors that happen via parallel writes. Also, the authors discussed that deltalake write transaction rate varies depending on implementation of cloud object stores, and explains if a higher rate is required, then having custom log store might alleviate the problem. As the system uses MVCC mechanism, it fetches the most recent version of data, and provides a time travel feature, to query back data in time.

Summary of Methodology and Result of Evaluation?

The authors evaluated the performance of Delta Lake by conducting various tests on the following parameters: (a) measuring the impact of multiple objects/ partitions and (b) read and write performance of tables. For parameter (a), the authors used a file with 33 million records and partitioned data into 1000 to 1 Million partitions. Delta Lake outperforms all the existing engines, with minimal processing latency when compared to standard Presto (a query engine for object stores and RDBMS engines) and Hive (a query engine for Hadoop and object stores). Similarly, for write performance, huge tables with statistics collection do not add any significant overhead to the ingestion and are similar to other engines

Any Research Directions Provided?

Overall, the authors provided various research directions, which include novel query processing methodologies, over cloud objects stores. Also, they mentioned that delta lake does not support secondary indexes and has performance issues in streaming data for sub-millisecond workloads. Therefore this could be an interesting problem to look into for implementing and optimizing the performance of deltalake.