Decipher Delta Lake

By Nisha Kumari and Divya Dua

ThoughtWorks®

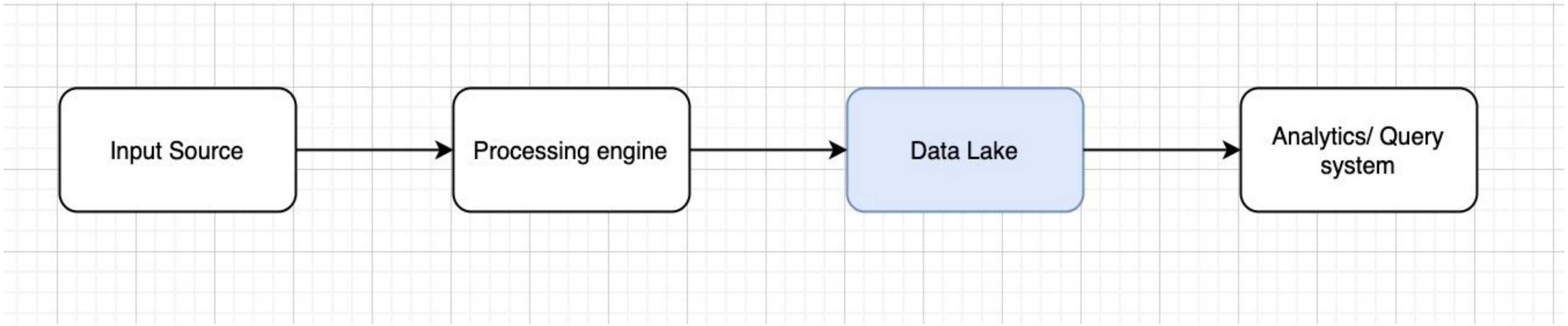
Agenda

- Intro
    - Data lake
    - Challenges & Mitigation
    - Possible solution
    - Delta Lake
    - Delta Lake architecture

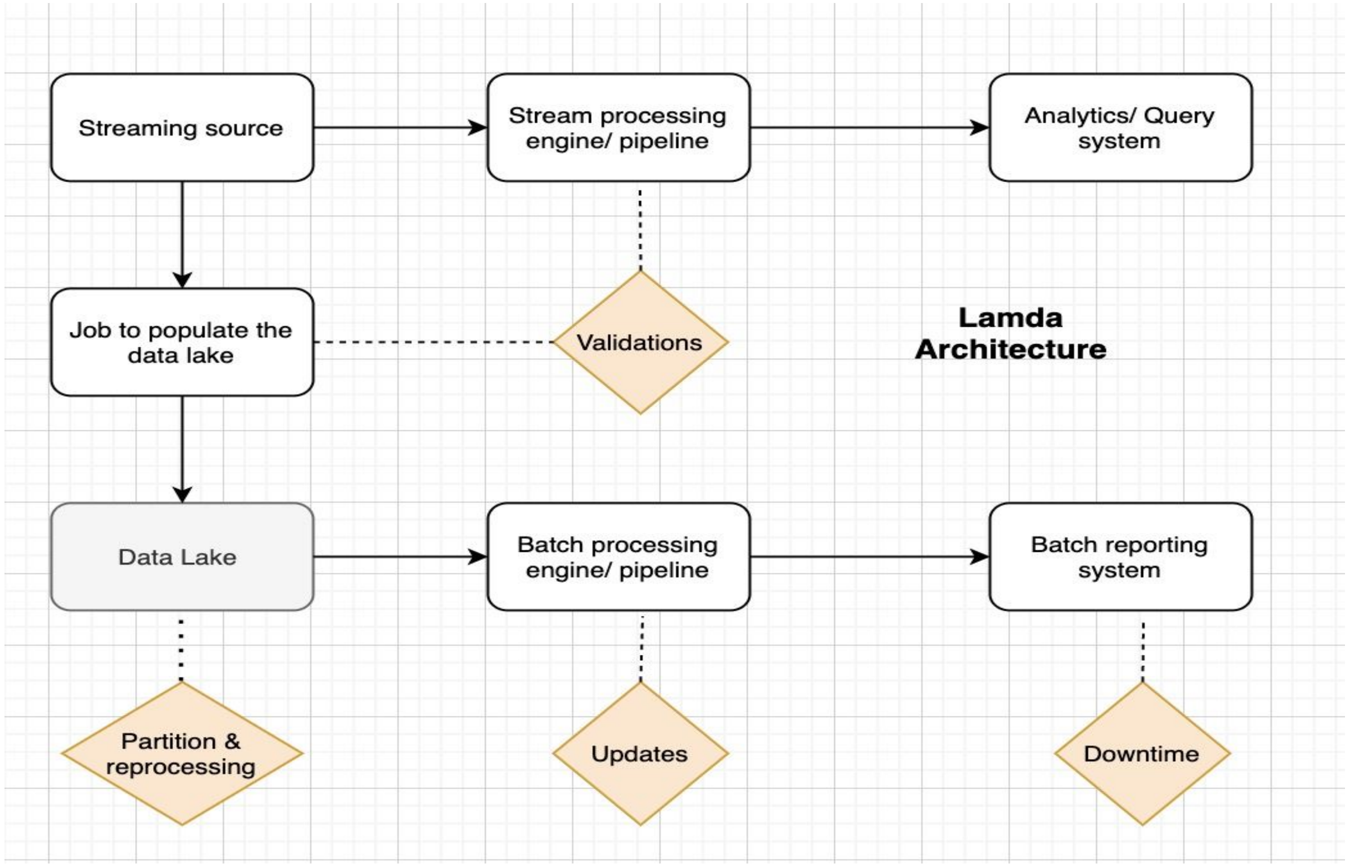
- Demo
    - Setup
    - ACID
    - Schema evolution
    - Data versioning

- Performance tuning

Data Lake


Input Source
Processing engine
Data Lake
Analytics/ Query system

Challenges & Mitigations



1. Data velocity
2. Historical queries
3. Messy Data
4. Job failures
5. Updates

Possible solution

A storage solution that can provide transactional guarantees.

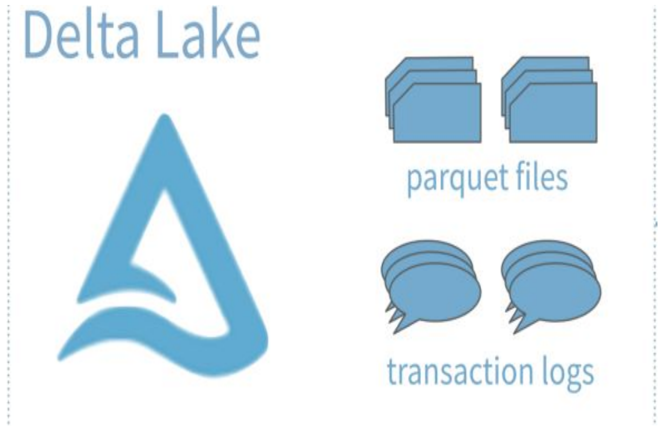
1. **Atomic visibility:** There must be a way for a file to be visible in its entirety or not visible at all.
2. **Mutual exclusion:** Only one writer must be able to create (or rename) a file at the final destination.
3. **Consistent listing:** Once a file has been written in a directory, all future listings for that directory must return that file.

Typical Storage systems do not necessarily provide this. This is where the **delta lake** comes in play.

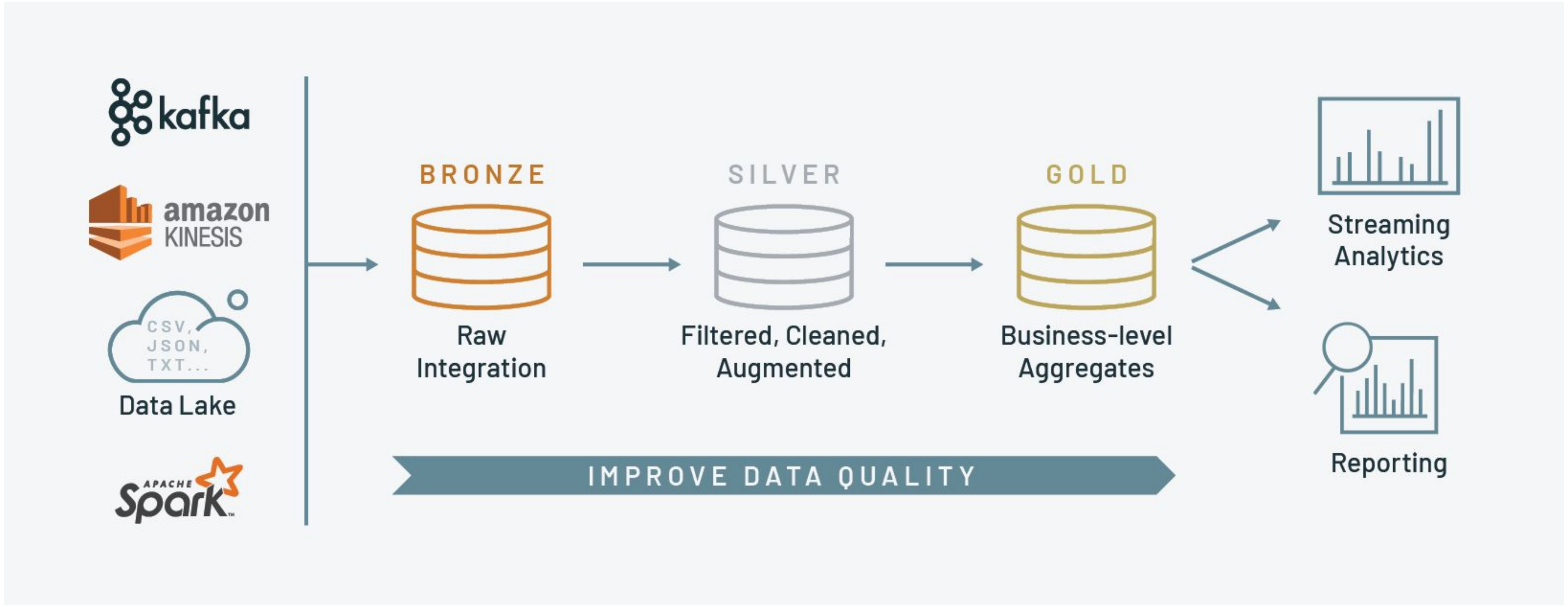
*Delta Lake transactional operations typically go through the **LogStore API** instead of accessing the storage system directly. (called Delta Transaction Log protocol).*

Delta Lake

1. Storage Layer
2. Open source
3. Based on Spark APIs
   a. Stream and batch
   b. First class support for upserts and deletes.
4. Uses parquet as underlying filesystem.
5. Uses Delta Transaction Log Protocol. Logs
   a. committed before any modification.
   b. maintain data change.
   c. maintain & enforce schema.
   d. Allow time travel.
6. Scalable metadata handling & checkpointing.



Delta Lake Architecture



https://databricks.com/blog/2020/06/02/monitor-your-databricks-workspace-with-audit-logs.html

Demo

1. Setup
2. ACID Transactions
3. Data Versioning
4. Schema evolution

Performance tuning

1. On schema aspect
   a. Uses Spark
   b. Checkpointing
2. On data aspect
   a. Compaction (# optimize)
   b. Delete old files (# vacuum)
   c. Data skipping (runs when applicable)
   d. Colocating predicates (Zorder)

References

- https://docs.delta.io/latest/delta-batch.html#language-scala - Demo
- https://docs.microsoft.com/en-us/azure/databricks/delta/
- https://akashrehan.wordpress.com/2019/07/11/anatomy-of-databricks-delta-lake/
- https://blog.knoldus.com/databricks-delta-architecture/
- https://github.com/delta-io/delta

Thank You

Divya Dua
Data Engineer
divya.dua@thoughtworks.com

Nisha Kumari
Data Engineer
nishak@thoughtworks.com

ThoughtWorks®