HANOI UNIVERSITY OF
SCIENCE AND TECHNOLOGY

calable Lakehouse for Financial Analy

Big Data Storage and Processing Capstone Project

Trinh Hoang Anh
Bui Khac Chien
Bach Nhat Minh
Tran Quang Minh
Vo Ta Quang Nhat

Supervised by: Dr. Tran Viet Trung

Contents

e Challenge: Taming Financial Data

l markets generate data at extreme volume, velocity, and variety, creating
nt engineering challenges that demand a unified architecture.

eneous
s

g real-time streams
quotes) with
historical datasets
ces, GDELT global

Dual Analytical Demands

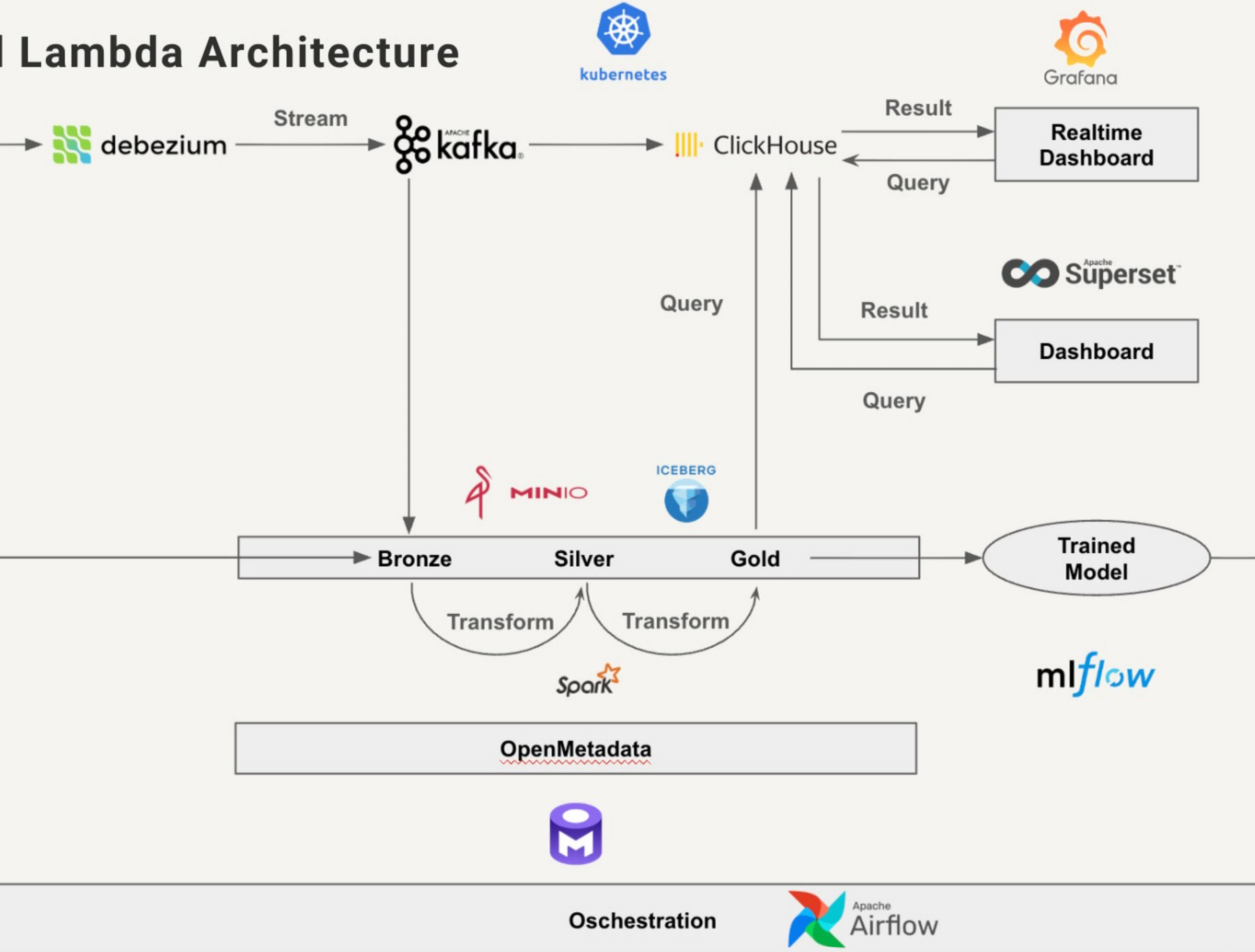
Supporting low-latency dashboards for real-time monitoring **and** deep, historical analysis for ML models.

Data Integrity &
Governance

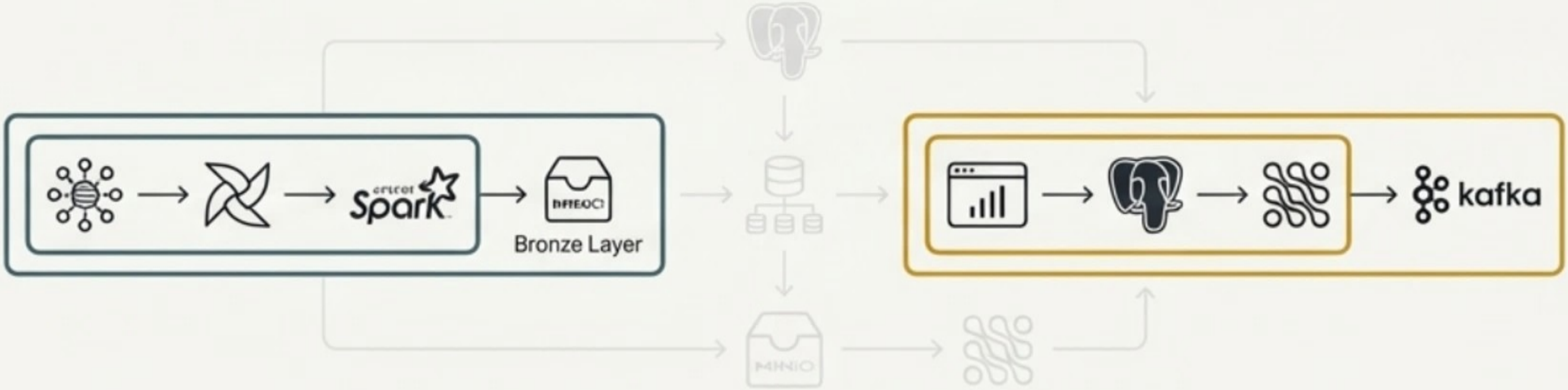
Ensuring consistenc
quality, and traceab
across a massive, e
dataset from raw ing
aggregated insights

Mission: To design and implement a scalable Big Data Lakehouse capabl
g, processing, and analyzing financial data in both real-time and batch mod

Lambda Architecture



stion: A Hybrid, Purpose-Built Strategy



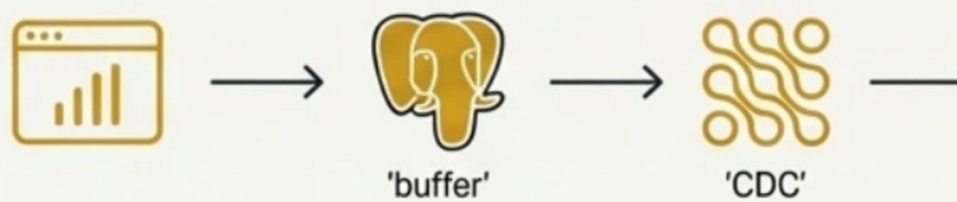
er (High Throughput)

Stooq.

ct-to-Lake' approach bypasses relational databases to
hput and avoid I/O bottlenecks for bulk historical data.

Speed Layer (Low Latency & Integr

Source: Finnhub API.



Principle: Uses PostgreSQL as a transactional buffer for
deduplication. Debezium's Change Data Capture on the
creates a real-time, event-driven stream without impact

cessing: The Medallion Lakehouse on Spark & Iceberg

w)

raw data from sources (GDELT, Stooq) stored as-is. Partitioned by date for efficient writes. The
uth for reprocessing.

ed & Enriched)

idated, structured, and deduplicated data. Key transformations include schema enforcement,
on (UTC to EST), and linking news events to specific stock tickers.

gated)

el aggregates optimized for analytics and ML. Examples: daily sentiment scores, news volume, a
icker.

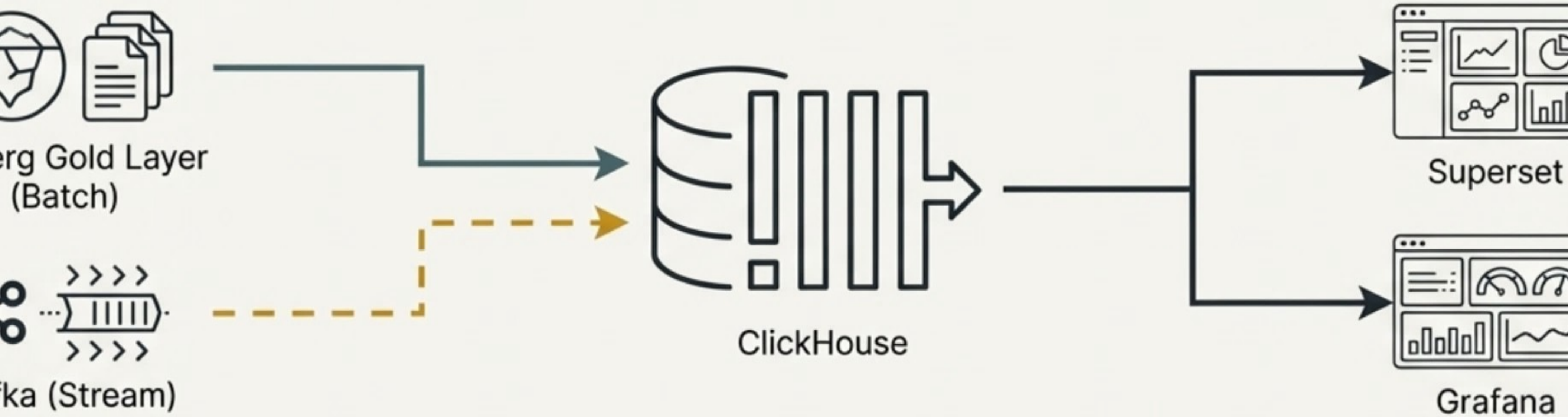
---

er - Apache Iceberg

IO, Iceberg provides ACID transactions, schema evolution, and time-travel directly on the data lake, combi
iability with data lake scalability.

rving: Unifying Batch and Stream with ClickHouse

se provides a single, high-performance OLAP access point for both deep hi
low-latency real-time streams.



Strategies:

Data: Utilizes the Iceberg Table Engine for **zero-copy queries** directly on Parquet files in M
gated data is also loaded into `MergeTree` tables for maximum dashboard performance.

n Data: Ingests high-throughput streams from Kafka into optimized `MergeTree` tables des
eries queries.

Sub-second query performance for interactive dashboards and immediate visibility of mar

Stack: A Containerized & Cloud-Native Foundation

e system is built on a containerized, cloud-agnostic infrastructure, enabling
ible and scalable deployments.

ent & Orchestration

Processing & Storage

Machine Lea

low & Metadata

Streaming & Database

Core Principles: Infrastructure as Code (IaC) • Scalability • Fault Tolerance

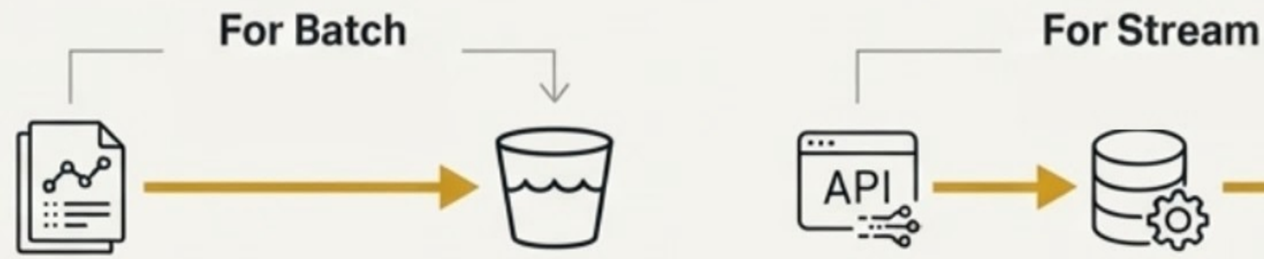
able Lesson #1: Hybrid Ingestion is a Necessity
Choice

Problem
Clash Grotesk Medium

A “one-size-fits-all” ingestion approach created severe bottlenecks.

Routing massive batch files (GDELT) through PostgreSQL caused I/O overhead, while writing API streams directly to the lake risked data duplication.

Solution
Clash Grotesk Medium



Direct-to-Lake

Spark writes directly to MinIO, bypassing transactional overhead for maximum throughput.

Transactional Bufferi

API -> Postgres -> CDC, u
database for its strengths
integrity and deduplicatio
streaming.

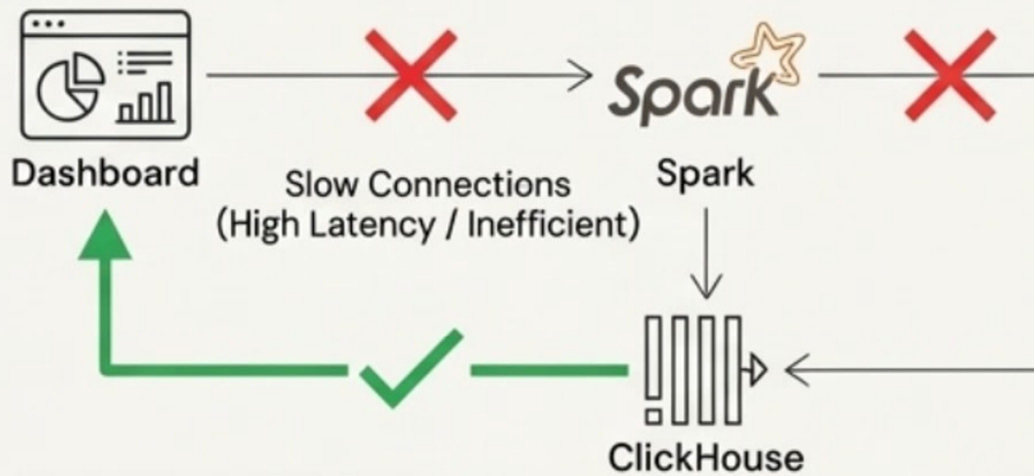
keaway

a database in the middle unless you have a transactional reason. Use raw object storage for bulk throu

able Lesson #2: Separate Processing from Serv
w-Latency BI

ashboards were sluggish and timed out.
onnecting Superset directly to Spark was
low (high latency), and using PostgreSQL for
arge-scale aggregations was inefficient (row-
riented).

Solution



Dedicated OLAP Serving Layer

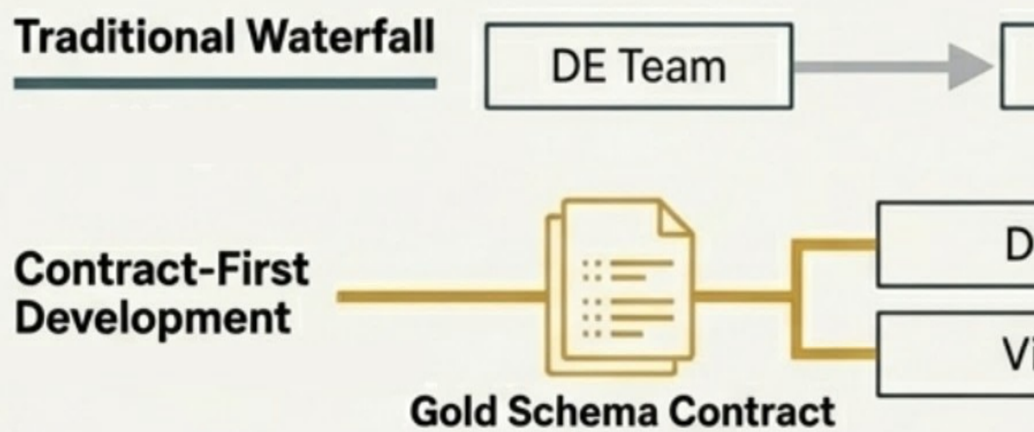
Implemented **ClickHouse** as a dedicated **OLAP servin**
columnar storage and vectorized query execution are p
for the fast aggregation queries required by BI dashboa
sub-second responses.

eaway

r throughput (ETL processing). ClickHouse is for latency (interactive serving). Use the right engi
ern. Satoshi Regular

able Lesson #3: Data Schemas as Contracts En el Work

Solution



he visualization team was blocked, waiting or the data engineering pipeline to be ompleted. This forced a slow, sequential waterfall" process.

Adopted **Contract-First Development**. The teams agree "Gold" layer data schema (column names, types, format visualization team built dashboards using mock data ma contract, while the DE team built the pipeline to produc required zero changes to the dashboard logic.

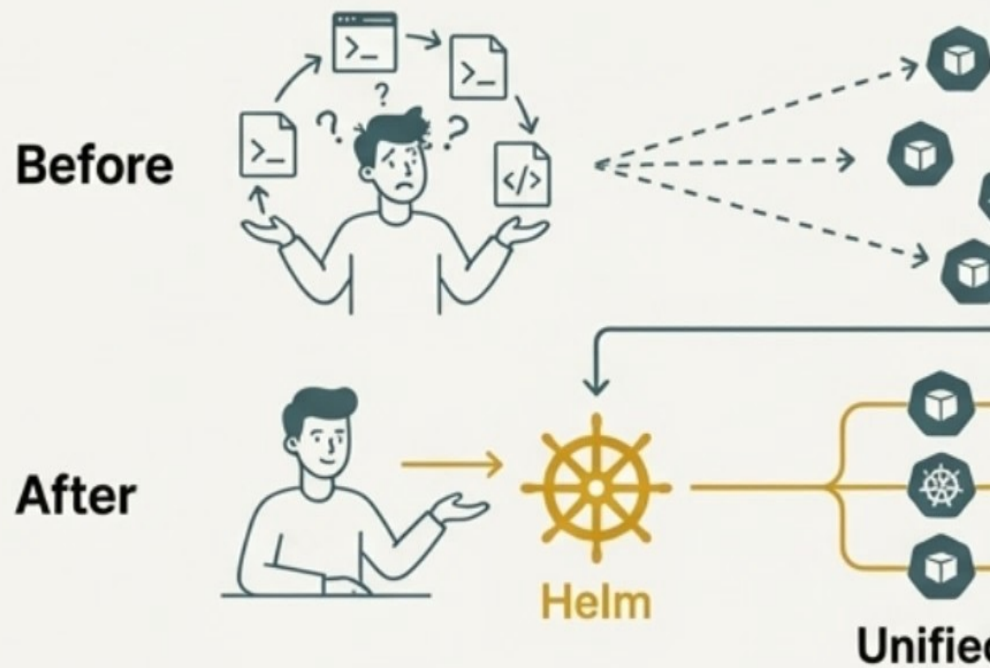
eaway

ned data contract decouples teams, is the foundation for agile development in data projects, and egration risk.

ble Lesson #4: Robust Deployment Requires Infrastruc
e

Managing 15+ Kubernetes resources with manual `kubectl apply` commands and shell scripts was brittle, error-prone, and lacked versioning.

Solution



Migrated all Kubernetes YAML manifests in
unified **Helm chart**. This templated the ent
application, centralizing configuration in a
`values.yaml` file and enabling versionec
deployments and rollbacks with a single cc

keaway

s infrastructure as versioned code (IaC),
configuration drift and making complex
nts manageable, resilient, and repeatable.

nclusion: An End-to-End
ncial Data Platform

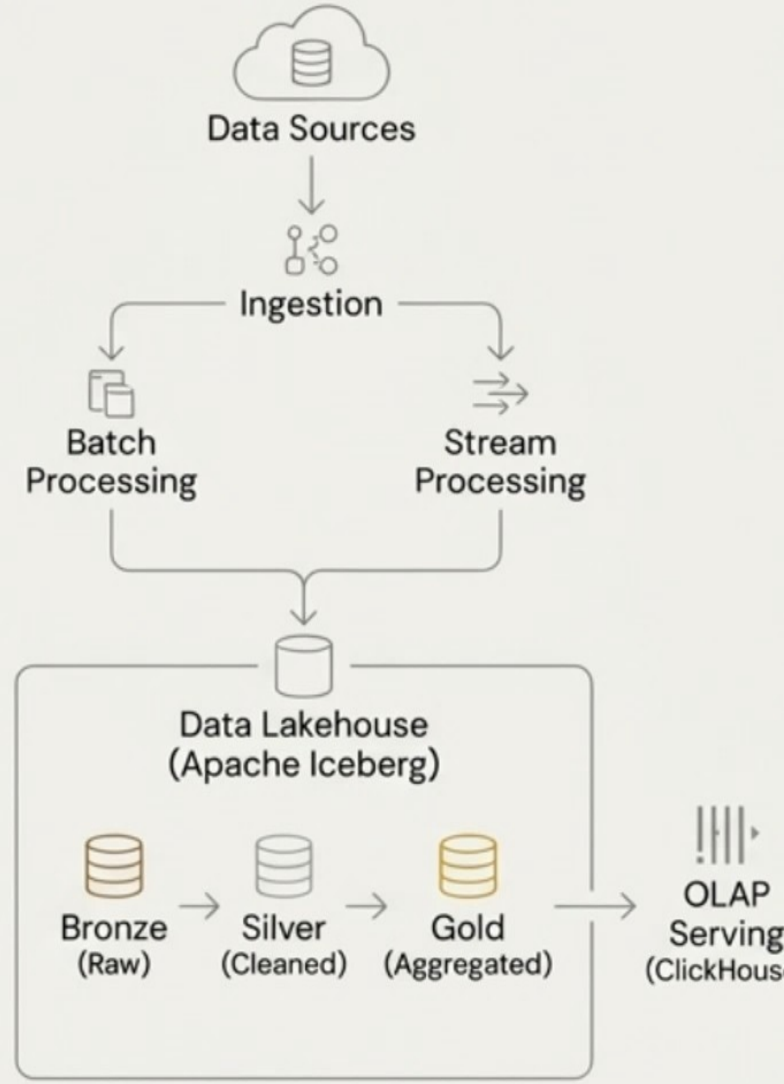
ssfully designed and implemented a scalable, nd Big Data Lakehouse that addresses the enges of complex financial analysis.

d Architecture: Integrated batch and stream ssing via a Lambda pattern.

Quality & Reliability: Ensured curated, reliable e data with the Medallion model on Apache g.

Performance Analytics: Delivered low-latency oards through a dedicated OLAP serving layer House).

t & Reproducible System: Built on a modern, nerized infrastructure managed with ructure as Code (Helm).



Simplified Architecture Overview