Stock Market Real-Time Data Streaming Project

Project Overview

This project is designed to build a real-time data streaming pipeline for stock market data using open-source tools and technologies. The goal is to ingest financial data from sources like Yahoo Finance and Alpha Vantage, process it in real-time using Apache Kafka and Apache Spark, store intermediate results in Minio, and load the final processed data into Snowflake for analysis.

The entire pipeline will be containerized using Docker, and Apache Airflow will be used to orchestrate the workflow.

Technical Requirements

Technologies Used

- Data Sources :
  - Yahoo Finance API
  - Alpha Vantage API
- Streaming and Processing :
  - Apache Kafka
  - Apache Spark (Batch + Streaming)
- Storage :
  - Minio (Object Storage)
  - Snowflake (Cloud Data Warehouse)
- Orchestration :
  - Apache Airflow
- Database :
  - PostgreSQL (Metadata/Intermediate Results)
- Containerization :
  - Docker

Pipeline Architecture

1. Data Ingestion

- Batch Ingestion :
    - Fetch historical stock data from Yahoo Finance and Alpha Vantage using Python scripts.
    - Store raw data in CSV format in Minio.
- Stream Ingestion :
    - Continuously fetch real-time stock data from APIs.
    - Push data to Kafka topics for real-time processing.

2. Real-Time Processing

- Kafka :
    - Set up Kafka brokers to receive real-time data streams.
    - Use Zookeeper to manage Kafka clusters and offsets.
- Spark Streaming :
    - Consume data from Kafka topics in near-real-time.
    - Perform transformations (filtering, aggregations, enrichments).
    - Write processed real-time data to Minio in Parquet format.

3. Batch Processing

- Use Spark to process historical data stored in Minio.
- Perform batch transformations (cleaning, deduplication, feature engineering).
- Store processed batch data in Minio in Parquet format.

4. Data Storage

- Minio :
    - Store raw, real-time, and processed data in CSV and Parquet formats.
- Snowflake :
    - Load final processed data from Minio into Snowflake for analysis.

5. Orchestration

- Use Apache Airflow to orchestrate the entire pipeline.
- Define DAGs to schedule and monitor tasks:
    - Data ingestion (batch and stream).
    - Real-time and batch processing.
    - Data loading into Snowflake.

6. Containerization

- Containerize all components using Docker:
    - Kafka brokers and Zookeeper.
    - Spark master and worker nodes.
    - Minio server.
    - PostgreSQL database.
    - Airflow webserver and scheduler.

Implementation Plan

Phase 1: Setup Environment

1. Install Docker and set up Kafka, Zookeeper, Spark, Minio, PostgreSQL, and Airflow.
2. Configure Kafka topics and Minio storage buckets.

Phase 2: Data Ingestion

1. Write Python scripts for batch and stream data ingestion.
2. Store raw data in Minio and push real-time data to Kafka.

Phase 3: Real-Time Processing

1. Write Spark Streaming applications to consume Kafka data.

2. Perform transformations and write processed data to Minio.

Phase 4: Batch Processing

1. Write Spark batch applications to process historical data.
2. Store processed data in Minio.

Phase 5: Data Loading

1. Use Airflow to load processed data from Minio into Snowflake.

Phase 6: Orchestration

1. Define Airflow DAGs to automate the pipeline.

Phase 7: Monitoring and Testing

1. Monitor Kafka, Spark, and Airflow.
2. Test the end-to-end pipeline with sample data.