
README for Stock Data Streaming with Kafka, Spark, and ClickHouse

Overview

This repository contains two Python scripts that work together to create a stock data streaming pipeline using Kafka, Spark, and ClickHouse. The `fetch_yfinanceV1.py` script fetches stock data and publishes it to a Kafka topic, while the `spark_kafka_consumerV2.py` script consumes this data from Kafka using Spark Streaming, processes it, and stores it in a ClickHouse database. Together, these programs demonstrate a real-time data ingestion and processing pipeline for financial data.

Scripts

1. `fetch_yfinanceV1.py`

Description: This script fetches live stock data using the `yfinance` library, formats it, and sends it to a Kafka topic. It is designed to simulate a real-time feed of stock data, which can then be consumed by downstream applications.

- **Data Source**: Uses Yahoo Finance API to retrieve stock data (Open, High, Low, Close, Volume) for a specified stock symbol at one-minute intervals.
- **Data Publishing**: Publishes the fetched stock data to a Kafka topic as JSON-formatted messages.
- **Usage**:
  1. Start a Kafka broker on the default port (`localhost:9092`).
  2. Run the script with `python fetch_yfinanceV1.py`. The script will continuously fetch stock data and publish it to Kafka.

Example Data:

- `symbol` : The stock symbol (e.g., AAPL for Apple).
- `Datetime` : Timestamp of the data.
- `Open` , `High` , `Low` , `Close` : Stock prices.
- `Volume` : Volume of stocks traded.

2. `spark_kafka_consumerV2.py`

Description: This script consumes stock data from a Kafka topic using Spark Structured Streaming, applies a schema to the incoming data, and processes it in real time. The processed data is then stored in a ClickHouse database for analysis.

- **Kafka Integration**: Reads data from a specified Kafka topic, where each message represents stock data in JSON format.
- **Schema Definition**: Defines a schema that includes fields for stock price data (symbol, datetime, open, high, low, close, volume, dividends, stock splits).
- **Data Processing and Storage**:
  - Processes the streaming data in Spark, making it ready for analysis.
  - Writes the processed data to a ClickHouse database using the ClickHouse JDBC driver, allowing for efficient storage and querying.
- **Usage**:
  1. Ensure a running Spark cluster, Kafka broker, and accessible ClickHouse database.
  2. Run the script with `spark-submit spark_kafka_consumerV2.py`. The script will continuously consume, process, and store the stock data from Kafka into ClickHouse.

Example Output:

- Real-time streaming DataFrame with columns for stock symbol, datetime, prices, volume, and dividends, stored in a ClickHouse table for further analysis.

Prerequisites

- **Kafka**: Ensure Kafka is running on `localhost:9092`.
- **Spark**: Spark should be installed, and `spark-submit` should be available in the environment.
- **ClickHouse**: Ensure ClickHouse is running, and the JDBC driver is accessible for Spark.
- **Python Packages**: Install required packages:

```
pip install yfinance kafka-python pyspark
```

Notes

- Adjust the stock symbol in `fetch_yfinanceV1.py` to fetch data for different stocks.
- Modify the Kafka topic in both scripts as needed to prevent topic conflicts in a multi-topic Kafka setup.
- Ensure that the ClickHouse database and table structure are prepared to receive the processed data from Spark.
