Financial Data Analysis on Kafka

1. Load Data from Yahoo and store time series episodes in a Kafka cluster
2. Inspect data using OpenTSx
3. Experimental setup using OpenTSx
4. Large scale experiments on KStreams and KSQL using OpenTSx-UDFs

# 1 Data Ingestion

We collect data from Yahoo! Finance using the Java-API[[1]](#footnote-1). Listing 1 illustrates the code to collect a time series bucket (TSB) from Yahoo!.

Calendar from = Calendar.*getInstance*();  
Calendar to = Calendar.*getInstance*();  
from.add(Calendar.***YEAR***, -5); *// from 5 years ago*String[] DOW30 = {**"AXP"**, **"AAPL"**, **"BA"**, **"CAT"**, **"CSCO"**, **"CVX"**, **"DD"**, **"XOM"**,  
 **"GE"**, **"GS"**, **"HD"**, **"IBM"**, **"INTC"**, **"JNJ"**, **"KO"**, **"JPM"**,  
 **"MCD"**, **"MMM"**, **"MRK"**, **"MSFT"**, **"NKE"**, **"PFE"**, **"PG"**,  
 **"TRV"**, **"UNH"**, **"UTX"**, **"VZ"**, **"V"**, **"WMT"**, **"DIS"**};  
  
Map<String, Stock> stocks2 = YahooFinance.*get*( DOW30, from, to, Interval.***DAILY*** ); *// single request*OpenTSDBConnector connector = **new** OpenTSDBConnector();  
  
Vector<TimeSeriesObject> tsos2 = **new** Vector<TimeSeriesObject>();  
**for**( String s : stocks2.keySet() ) {  
  
 Stock st = stocks2.get( s );  
 TimeSeriesObject tso = TSOMapper.*convertYahooStockToTSO*( st, s, from, to, Interval.***DAILY***, **"close"** );  
  
 tsos2.add( tso );  
  
 OpenTSDBConnector.*streamEventsToOpenTSDB*( tso, connector );  
  
}  
  
MultiChart.*open*( tsos2, **true** );

**Listing 1:** Collect stock market data using the Yahoo! API

The time series are persisted in HBase. The OpenTSDB system is used to manage the data model. It provides a convenient API and a user interface (see figure 1) for high level data inspection.

## Load Performance

Loading the data from Yahoo! took 186 seconds, but retrieving the data via OpenTSDB API took only 8 s, which is approx. 25 times faster. Plotting the time series with the OpenTSDB web API is even faster. Total time to load 5 series from OpenTSDB was less than 100 ms.

# 2 Data Inspection with OpenTSDB and OpenTSScope

We use OpenTSDB in two ways. First, we access the web user interface for quick visual inspection of individual time series. The following link renders the chart with 4 exemplary time series:

<http://cc-poc-mk-1.gce.cloudera.com:4242/#start=2012/08/18-07:38:45&end=2017/08/18-07:38:57&m=sum:CVX&o=&m=sum:AAPL&o=&m=sum:V&o=&m=sum:VZ&o=&yrange=%5B0:%5D&wxh=800x400&style=linespoint>



The second approach provides a time series bucket which can be persisted in HDFS as Parquet file or even exported as CSV or MS-Excel spreadsheet. Listing 2 shows the details of loading the persisted time series from OpenTSDB:

String[] DOW30 = {**"AXP"**, **"AAPL"**, **"BA"**, **"CAT"**, **"CSCO"**, **"CVX"**, **"DD"**, **"XOM"**,  
 **"GE"**, **"GS"**, **"HD"**, **"IBM"**, **"INTC"**, **"JNJ"**, **"KO"**, **"JPM"**,  
 **"MCD"**, **"MMM"**, **"MRK"**, **"MSFT"**, **"NKE"**, **"PFE"**, **"PG"**,  
 **"TRV"**, **"UNH"**, **"UTX"**, **"VZ"**, **"V"**, **"WMT"**, **"DIS"**};  
  
OpenTSDBConnector connector = **new** OpenTSDBConnector();  
TSBucket bucket = TSBucket.*createEmptyBucket*();  
Vector<TimeSeriesObject> tsos = OpenTSDBConnector.*loadBucketFromOpenTSDB*( DOW30, connector, bucket );  
MultiChart.*open*( tsos, **true**, **"DOW 30"**);

**Listing 2:** Retrieve stock market data from OpenTSDB using the time series bucket loader.

3. …. TBD  
4. …. TBD

1. https://github.com/sstrickx/yahoofinance-api [↑](#footnote-ref-1)