Proposal

# case:

The financial market is the backbone of the economy, millions of transactions occurring everyday. My proposal attempts to use a real time financial market data API to retrieve stock market data. The aim of my project is then to use the FOREX trading records of a particular currency and find patterns in a given time-frame.

# 1) Source:

The Twelve-data API is to be used to obtain financial market data of FOREX stocks. The API call will return meta and time series information in a CSV format. The data will come as a batch request with a frequency of 1min.

-Meta object consists of general information about requested currency pair

-Time series contains as array of objects: Data-time, Open, High, Low, Close

# 2) Apache Kafka: Producer

I have chosen to use Kafka for its distributed, resilient and fault tolerant architecture that is able to scale horizontally. The high performance and real time capabilities will allow for my project to store and query time-series data of the financial market data API.

-Using a batch pipeline

-Using a Kafka server to create a broker that will contain the topic.

-To begin the pipeline I will be create a Topic to store the stream of data, the topic

will be split into three partitions.

-Using a producer to write data to a topic by specifying the name and the broker created

-Acks = 1 : The producer will wait for leader acknowledgement

# 3) Apache Spark:

I will be using Spark for the processing stage of my pipeline, for high speed data analysis and transformations. I will be creating scripts in Scala in order to process to find the highest/lowest transactions in a given time-frame.

-Using data-frames to perform aggregate functions

-Potentially using the streaming data to trigger a response

e.g. When a particular stock reaches a certain value to run a script.

# 4) Apache Kafka: Consumer

-Using consumer to read data from a topic by specifying the name and the broker created

-Some potential use cases of the data will be to track old and current stock values for

a particular stock

-Another potential use case would be to trigger a script if a particular stock had gone

below/above a certain value

# 5) HBase:

I chose HBase for time series data storage because it scales.

-Linear scaling, if more storage is needed, add more nodes

-Automatic replication, data can be stored in HDFS which makes it fault tolerant

-Efficient scans, most of the time the data will be used to answer questions related to

data points between times X and Y. the implementation of scan operations

-High write throughput, the big-table design, which HBase follows, uses LSM trees to make

writes cheaper.

-HBase is column oriented and sorted key-value system

