**Data Ingestion**

This project aim to fetch the data from different Stock markets across the world and store the data in NoSql database like Cassandra and HBase.

**This document is not complete yet and will be updated as the project progress.**

1. **Input data file details**

We are fetching NYSE data using Yahoo Financial API and are considering the following fields as of now.

Stock Exchange

Stock Symbol

Trade Date

Open

Previous Close

Day’s low

Day’s high

P/E Ratio

52-weeks low

52-weeks high

Change & percent change

Earnings/Share

Dividend/Share

Volume

Book Value

Bid size

1. **Cluster Configuration**

Cassandra clusters are given names in order to prevent machines in one cluster from joining another that you don’t want them to be a part of.

We will name our cluster as Financial. For this we need to edit the Cassandra.yaml file. *(Refer Cassandra tutorial from the lecture)*

1. **Schema Configuration**

Keyspace – Stock Market

Column Family - Stock exchange

RowKey - Stock\_symbol

ColumnKey - Date/Month/Year

Column - Fields

* 1. **Keyspace Configuration**

We will use API to create Keyspace and Column Families and to configure them. To create a Keyspace we need to specify Keyspace name, a replica placement strategy and a replication factor.

* + 1. **Keyspace Name**

Name of our Keyspace is “StockMarket”

* + 1. **Replication factor**

As we are working with single node setup we will consider the replication factor to be 1 as of now. The replication factor can be changed using node tool.

* + 1. **Replica Placement Strategy**

The purpose of this setting is to configure the way that the node picker works. Cassandra provides three Replica Placement Strategy namely, Simple Strategy, Old Network Topology Strategy and Network Topology Strategy.

The first replica will always be in the node that claims the range in which the token falls, but the remaining replicas are placed according to the replica placement strategy used.

For our project we will use Simple Strategy as we are working with single node and once we expand our project we can consider using other appropriate strategy at that time.

* 1. **Column Family configuration**

Column family name are the names of the Stock Exchanges used. Initially we consider NYSE.

Possible Stock Exchanges we intend to use are BSE, NSE, NASDAQ, DAX and so on.

* 1. **RowKey Configuration**
     1. **Reason for RowKey selection**

According to the financial analysis project problem statement, we will need to query by Stock\_symbol and Timestamp. If we take timestamp as RowKey, then we might end up using a particular set of node more frequently (overloading particular nodes in the cluster) and some nodes may not be used at all. If we consider Stock\_symbol as the RowKey then the load will be possibly balanced.

Another option would be composite key of Stock\_symbol & Year. As of now we will consider the Stock\_Symbol as the RowKey.

Further Analysis is required and this section will be updated once we finalize the complete schema after testing the possibilities. Also the usage of ColumnKey needs more analysis.

* + 1. **Partitioners**

The purpose of partitioner is to allow you to specify how row keys should be sorted, which has a significant impact on how data will be distributed across your nodes. It also has an effect on the options available for querying ranges of rows.

Cassandra provides three different Partitioners namely, Random Partitioner (default), Order-Preserving Partitioner and Collating Order-Preserving Partitioner.

NOTE: The Partitioner can modify the on-disk SSTable representation. So if you change the partitioner type, you will have to delete the data directories.

As we are using Stock\_Symbol as the RowKey we can use Random Partitioner. This has an advantage of spreading the keys evenly across the cluster, because the distribution is random. Also our Range quires will be on the column so we can safely consider Random Partitioner.