**Real time tweets capture using Spark – Cassandra on Windows 10**

This document explains the process of capturing real time twitter tweets using SPARK/SCALA and NoSQL Database as Cassandra.

The tweets are captured in two different formats

1. All the tweets in a given interval
2. Tweets and retweets with a hashtag (for example - @realDonaldTrump)

**Step – I – Setting up the process**

Installation of tools

1. Java 8 installation in default path and setting up of Home directory
2. Scala IDE installation and setting up the home directory
3. Spark installation and setting up the home directory
4. Winutils installation on C\winutils\bin and setting up this as home directory.
5. Downloading and copying the required Jars in C:\Spark\bin\jars directory

**Step 2 – Setting up projects in Eclipse IDE [Assuming that Cassandra is installed and Running]**

1. Open Eclipse
2. File >New >Scala Project
3. Add new package
4. Write the below code in a notepad and save as .scala. Right click on the package and import this scala file.
5. Right click Referenced Libraries > Java Build Path > Libraries > Add external Jars
6. Apply > Ok >Close
7. Create Twitter API for getting the Consumer Key, Token Id etc. Store this in a file called twitter.txt and put it under C:\SparkStreaming (our project directory).
8. Run > Run Configurations > Give Configuration Name (ex: com.xxx.xxx.ConfigurationName)
9. Apply and run the job.
10. Version used
    1. Cassandra 2.2.8
    2. Spark 2.0
    3. Scala 2.11

**Code of Capturing the Tweets and Saving to Cassandra Table**

**package** com.test.streaming

**import** org.apache.spark.\_

**import** org.apache.spark.SparkContext.\_

**import** org.apache.spark.streaming.\_

**import** org.apache.spark.streaming.twitter.\_

**import** org.apache.spark.streaming.StreamingContext.\_

**import** org.apache.log4j.Level

**import** Utilities.\_

**import** org.joda.time.DateTime.\_

**import** org.apache.spark.SparkConf

**import** org.apache.spark.streaming.{Seconds, StreamingContext}

**import** org.apache.spark.storage.StorageLevel

**import** java.util.regex.Pattern

**import** java.util.regex.Matcher

**import** com.datastax.spark.connector.\_

**import** com.datastax.spark.connector.streaming.\_

**import** StreamingContext.\_

**import** org.apache.spark.SparkContext.\_

**import** com.twitter.jsr166e;

**import** java.util.Map;

**import** java.util.Set;

**import** java.io.Serializable;

**import** java.util.concurrent.atomic.AtomicLong;

**import** java.io.Serializable;

/\*\* Simple application to listen to a stream of Tweets and print them out \*/

**object** PrintTweets3 {

/\*\* Our main function where the action happens \*/

**def** main(args: Array[*String*]) {

// Configure Twitter credentials using twitter.txt

setupTwitter()

**val** sc = **new** SparkConf()

.set("spark.cassandra.connection.host", "127.0.0.1")

.setMaster("local[\*]")

.setAppName("PrintTweets3")

.set("spark.cassandra.connection.port", "9042");

// Create the context with a 10 second batch size

**val** ssc = **new** StreamingContext(sc, Seconds(2))

**val** stream = TwitterUtils.createStream(ssc, **None**, **Nil**, storageLevel = StorageLevel.MEMORY\_ONLY\_SER\_2)

// Recompute the top hashtags every 1 second

**val** slideInterval = **new** **Duration**(2 \* 1000)

// Compute the top hashtags for the last 5 seconds

**val** windowLength = **new** **Duration**(2 \* 1000)

// Wait this many seconds before stopping the streaming job

**val** timeoutJobLength = 100 \* 1000

setupLogging()

**val** hashTags = stream.map(\_.getText)

//.filter(\_.startsWith("@realDonaldTrump"))

//.flatMap(\_.split(" "))

//val hashTags = stream.flatMap(tweet => tweet.getText.toLowerCase.split(" ").filter(tags.contains(Seq("#Trump", "#POTUS"))))

**val** tagCounts = hashTags.map((\_, 1)).reduceByKeyAndWindow((x: Int, y: Int) => x + y, windowLength, slideInterval)

**val** tagCountsAll = tagCounts.map{**case** (tag, mentions) => (tag, mentions, "ALL")}

//val tagCountsByDay = tagCounts.map{case (tag, mentions) => (tag, mentions, DateTime.now.toString("yyyyMMdd"))}

tagCountsAll.saveToCassandra("demo\_ks", "hashtags", **SomeColumns**("hashtag", "mentions", "interval"))

ssc.start()

ssc.awaitTermination()

// Create a DStream from Twitter using our streaming context

//val tweets = TwitterUtils.createStream(ssc, None)

}

}