**Malini Mittal**

**W205**

**Assignment 3**

**Task 2 – Key/Value (AWS S3):**

**Data Cleaning:**

Each tweet that gets read in will be cleaned to store only the required fields:

for tweet in tweets:

d = {}

d['created\_at'] = parser.parse(tweet['created\_at']).isoformat()

d['userhashtags'] = [h[u'text'] for h in tweet['entities']['hashtags']]

d['userscreenname'] = tweet['user']['screen\_name'].encode('utf-8')

d['username'] = tweet['user']['name'].encode('utf-8')

d[‘tweet\_id’] = tweet[‘id’]

# … any other operation … #

**Data Organization:**

The data will be organized to store each tweet as a single value. To be able to answer the questions about the user with maximum tweets and the number of tweets produced in a particular hour, the following organization can be used:

s3://ConferenceTweetData/dates/<datetime>/<user\_screen\_name>/<tweet\_id>.json

For the question regarding the top hashtags used, the following organization will be used:

s3://ConferenceTweetData/hashtags/<hashtag>/<tweet\_id>

**Storage:**

To store information in this model, my python script will first create the main bucket. Then it will load each file, iterate through the tweets, create the two keys as follows:

1. key1 = ‘’.join(‘dates/’, d['created\_at'], ‘/’, d['userscreenname'], ‘/’, str(d[‘tweet\_id’]), ‘.json’)

The tweet will be uploaded onto S3 as a json file on this key.

1. The number of keys generated for hashtags will depend on the number of hashtags in the tweet:

for ht in d['userhashtags':

hashkey = ’’.join(‘hashtags/’, ht, str(d[‘tweet\_id’]))

…

For each of these keys, set the contents from string by converting the tweet\_id into a string.

**Retrieval:**

# Create an S3 connection using boto, and then get the bucket:

conn = S3Connection()

bucket = conn.get\_bucket(‘ConferenceTweetData’)

keylist = bucket.list(‘hashtags’)

1. Who tweeted the most during the conference?

def getUserDictionary():

datesiter = bucket.list(“dates/”, “/”)

user\_dict = {}

foreach date in dateiter:

dt = date.name.strip(‘/’).split(‘/’)[-1]

useriter = bucket.list(date.name, ‘/’)

for user in useriter:

username = user. name.strip(‘/’).split(‘/’)[-1]

if username not in user\_dict:

user\_dict[username] = 0

tweetiter = bucket.list(user.name)

numtweets = 0

for tweet in tweetiter:

numtweets += 1

user\_dict[username] += numtweets

return user\_dict

udict = getUserDictionary()

sortedlist = sorted(udict, key=udict.\_\_getitem\_\_, reverse=True)[0:5]

print sortedlist

1. What were the top ten hashtags used?

def getHashTagDictionary():

hashtagsiter = bucket.list(“hashtags/”, “/”)

hashtag\_dict = {}

for hashtag in hashtagiter:

name = hashtag.name

tweetlist = bucket.list(name)

count = 0

for tweet in tweetlist:

count += 1

hashtag\_dict[name.strip(‘/’).split(‘/’)[-1]] = count

return hashtag\_dict

htdict = getHashTagDictionary()

sortedlist = sorted(htdict, key=htdict.\_\_getitem\_\_, reverse=True)[0:10]

print sortedlist

1. How many tweets were produced each hour?

def getTweetCount(timestart, timeend):

datesiter = bucket.list(“dates/”, “/”)

numtweets = 0

foreach date in dateiter:

dt = date.name.strip(‘/’).split(‘/’)[-1]

if (dt >= starttime and dt < endtime):

useriter = bucket.list(date.name, ‘/’)

for user in useriter:

tweetiter = bucket.list(user.name)

for tweet in tweetiter:

numtweets += 1

return numtweets

date1 = datetime.datetime.strptime("2015-02-14T08:00:00", "%Y-%m-%dT%H:%M:%S")

date2 = date1 + datetime.timedelta(days=1)

for day in (date1, date2):

for hour in range(0,7):

starttime = day + datetime.timedelta(hours=hour)

endtime = day + datetime.timedelta(hours=hour+1)

numtweets = getNumTweets(tweets, starttime.isoformat(), endtime.isoformat())

print ("Number of tweets on {0} between {1} and {2} = {3}").format(starttime.date(), starttime.time(), endtime.time(), numtweets)