In [2]:

```
# First let's read Twitter ids and screen names of the 577 US congress members

congress_members = spark.read.csv("s3://us-congress-tweets/congress_members.csv", h
eader=True)
congress_members.show()
print("Number of congress members tracked:", congress_members.count())
```

```
+------------------+---------------+
|            userid|    screen_name|
+------------------+---------------+
|         776664410|  RepCartwright|
|         240363117|   RepTomMarino|
|837722935095930883| RepScottTaylor|
|        1069124515|     RepLaMalfa|
|818460870573441028|  RepTomGarrett|
|         163570705|     repcleaver|
|          19739126|      GOPLeader|
|          33563161| RepJoseSerrano|
|        2861616083|USRepGaryPalmer|
|        1074518754| SenatorBaldwin|
|         305620929|  Call_Me_Dutch|
|         381152398| RepTerriSewell|
|         834069080| RepDavidRouzer|
|         249787913|  SenatorCarper|
|         188019606|        Clyburn|
|         217543151|SenatorTimScott|
|          39249305| USRepMikeDoyle|
|          33537967|   amyklobuchar|
|         249410485|  SanfordBishop|
|          23124635|    TomColeOK04|
+------------------+---------------+
only showing top 20 rows

Number of congress members tracked: 577
```

In [3]:

```
from pyspark.sql.types import *
import pyspark.sql.functions as F
twitter_date_format="EEE MMM dd HH:mm:ss ZZZZZ yyyy"

user_schema = StructType([
    StructField('created_at',TimestampType(),True),
    StructField('followers_count',LongType(),True),
    StructField('id',LongType(),True),
    StructField('name',StringType(),True),
    StructField('screen_name',StringType(),True)
])

hashtag_schema = ArrayType(StructType([StructField('text',StringType(),True)]))
user_mentions_schema = ArrayType(StructType([StructField('id',LongType(),True),
                                             StructField('screen_name',StringType
(),True)]))
entities_schema = StructType([
    StructField('hashtags',hashtag_schema,True),
    StructField('user_mentions',user_mentions_schema,True)
    ])

retweeted_status_schema =StructType([
        StructField("id", LongType(), True),
        StructField("in_reply_to_user_id", LongType(), True),
        StructField("in_reply_to_status_id", LongType(), True),
        StructField("created_at", TimestampType(), True),
        StructField("user", user_schema)
    ])

tweet_schema =StructType([
        StructField("text", StringType(), True),
        StructField("id", LongType(), True),
        StructField("in_reply_to_user_id", LongType(), True),
        StructField("in_reply_to_status_id", LongType(), True),
        StructField("created_at", TimestampType(), True),
        StructField("user", user_schema),
        StructField("entities", entities_schema),
        StructField("retweeted_status", retweeted_status_schema)
    ])
```

In [4]:

```
tweets = spark.read.option("timestampFormat", twitter_date_format)\
                   .json('s3://us-congress-tweets/congress-sample-10k.json.gz', twe
et_schema)\
                   .withColumn('user_id',F.col('user.id'))
tweets.printSchema()
```

```
root
 |-- text: string (nullable = true)
 |-- id: long (nullable = true)
 |-- in_reply_to_user_id: long (nullable = true)
 |-- in_reply_to_status_id: long (nullable = true)
 |-- created_at: timestamp (nullable = true)
 |-- user: struct (nullable = true)
 |    |-- created_at: timestamp (nullable = true)
 |    |-- followers_count: long (nullable = true)
 |    |-- id: long (nullable = true)
 |    |-- name: string (nullable = true)
 |    |-- screen_name: string (nullable = true)
 |-- entities: struct (nullable = true)
 |    |-- hashtags: array (nullable = true)
 |    |    |-- element: struct (containsNull = true)
 |    |    |    |-- text: string (nullable = true)
 |    |-- user_mentions: array (nullable = true)
 |    |    |-- element: struct (containsNull = true)
 |    |    |    |-- id: long (nullable = true)
 |    |    |    |-- screen_name: string (nullable = true)
 |-- retweeted_status: struct (nullable = true)
 |    |-- id: long (nullable = true)
 |    |-- in_reply_to_user_id: long (nullable = true)
 |    |-- in_reply_to_status_id: long (nullable = true)
 |    |-- created_at: timestamp (nullable = true)
 |    |-- user: struct (nullable = true)
 |    |    |-- created_at: timestamp (nullable = true)
 |    |    |-- followers_count: long (nullable = true)
 |    |    |-- id: long (nullable = true)
 |    |    |-- name: string (nullable = true)
 |    |    |-- screen_name: string (nullable = true)
 |-- user_id: long (nullable = true)
```

In [5]:

```
users = spark.read.option("timestampFormat", twitter_date_format)\
                  .json('s3://us-congress-tweets/congress-sample-10k.json.gz', use
r_schema)\
                  .withColumn('user_id',F.col('id'))
users.printSchema()
users.show(10)
```

```
root
 |-- created_at: timestamp (nullable = true)
 |-- followers_count: long (nullable = true)
 |-- id: long (nullable = true)
 |-- name: string (nullable = true)
 |-- screen_name: string (nullable = true)
 |-- user_id: long (nullable = true)

+-------------------+---------------+-------------------+----+---------
--+-------------------+
|         created_at|followers_count|                 id|name|screen_na
me|            user_id|
+-------------------+---------------+-------------------+----+---------
--+-------------------+
|2018-10-03 17:30:03|           null|1047539288705880064|null|       nu
ll|1047539288705880064|
|2018-10-03 17:38:39|           null|1047541454313283585|null|       nu
ll|1047541454313283585|
|2018-10-03 17:57:59|           null|1047546320431403008|null|       nu
ll|1047546320431403008|
|2018-10-03 18:13:43|           null|1047550278332010496|null|       nu
ll|1047550278332010496|
|2018-10-03 18:28:15|           null|1047553936780144642|null|       nu
ll|1047553936780144642|
|2018-10-03 18:45:53|           null|1047558373435219968|null|       nu
ll|1047558373435219968|
|2018-10-03 19:01:05|           null|1047562198107607041|null|       nu
ll|1047562198107607041|
|2018-10-03 19:10:25|           null|1047564545177374721|null|       nu
ll|1047564545177374721|
|2018-10-03 19:40:09|           null|1047572030097428482|null|       nu
ll|1047572030097428482|
|2018-10-03 19:55:48|           null|1047575969807183874|null|       nu
ll|1047575969807183874|
+-------------------+---------------+-------------------+----+---------
--+-------------------+
only showing top 10 rows
```

In [6]:

```
users.select('user_id').distinct().count()
```

```
10000
```

In [7]:

```
tweets = spark.read.option("timestampFormat", twitter_date_format)\
                   .json('s3://us-congress-tweets/congress-sample-10k.json.gz', twe
et_schema)\
                   .withColumn('user_id',F.col('user.id'))
tweets.printSchema()
# tweets.show(1)
tweets.select('user_id').distinct().count()
```

```
root
 |-- text: string (nullable = true)
 |-- id: long (nullable = true)
 |-- in_reply_to_user_id: long (nullable = true)
 |-- in_reply_to_status_id: long (nullable = true)
 |-- created_at: timestamp (nullable = true)
 |-- user: struct (nullable = true)
 |    |-- created_at: timestamp (nullable = true)
 |    |-- followers_count: long (nullable = true)
 |    |-- id: long (nullable = true)
 |    |-- name: string (nullable = true)
 |    |-- screen_name: string (nullable = true)
 |-- entities: struct (nullable = true)
 |    |-- hashtags: array (nullable = true)
 |    |    |-- element: struct (containsNull = true)
 |    |    |    |-- text: string (nullable = true)
 |    |-- user_mentions: array (nullable = true)
 |    |    |-- element: struct (containsNull = true)
 |    |    |    |-- id: long (nullable = true)
 |    |    |    |-- screen_name: string (nullable = true)
 |-- retweeted_status: struct (nullable = true)
 |    |-- id: long (nullable = true)
 |    |-- in_reply_to_user_id: long (nullable = true)
 |    |-- in_reply_to_status_id: long (nullable = true)
 |    |-- created_at: timestamp (nullable = true)
 |    |-- user: struct (nullable = true)
 |    |    |-- created_at: timestamp (nullable = true)
 |    |    |-- followers_count: long (nullable = true)
 |    |    |-- id: long (nullable = true)
 |    |    |-- name: string (nullable = true)
 |    |    |-- screen_name: string (nullable = true)
 |-- user_id: long (nullable = true)

9735
```

In [29]:

```
tweets.select('retweeted_status.id').distinct().count()
```

```
5733
```

In [27]:

```
```

10000

In [11]:

```
### Part 1a --> Who are the ten most mentioned users in the sample?

from pyspark.sql.functions import desc

# tweets_x.groupBy('entities.user_mentions.screen_name').count().sort(desc("coun
t")).show(11)
# tweets.groupBy('entities.user_mentions.id').count().sort(desc("count")).show(11)
tweets.groupBy('entities.user_mentions.screen_name').count().sort(desc("count")).sh
ow(11)
```

```
+---------------+-----+
|    screen_name|count|
+---------------+-----+
|             []|  469|
|[RepAdamSchiff]|  409|
|   [marcorubio]|  282|
|   [SenSchumer]|  215|
|[SpeakerPelosi]|  200|
|[ChrisMurphyCT]|  170|
|[SenGillibrand]|  112|
| [RepMattGaetz]|  111|
|     [RandPaul]|  108|
|   [CoryBooker]|   92|
| [SteveScalise]|   89|
+---------------+-----+
only showing top 11 rows
```

In [10]:

```
###### What are the top hashtags used?
# tweets.groupBy('entities.hashtags').count().sort(desc("count")).show(11)
tweets.select(F.explode("entities.hashtags.text").alias("hashtag"))\
      .groupby("hashtag").count().sort(F.desc("count")).show()
```

```
+-----------------+-----+
|          hashtag|count|
+-----------------+-----+
|        Venezuela|  102|
|    TrumpShutdown|   42|
|     MaduroRegime|   29|
|             MAGA|   20|
|           Maduro|   20|
|      NancyPelosi|   19|
|    MuellerReport|   17|
|      TrumpResign|   14|
|     BuildTheWall|   14|
|        Kavanaugh|   14|
|     ForThePeople|   14|
|     GreenNewDeal|   13|
| MyHouseMyAmerica|   12|
|        transport|   12|
|        Democrats|   10|
|          Florida|   10|
|             Cuba|   10|
|    BrowardCounty|    9|
|             EEUU|    8|
|MaduroCrimeFamily|    8|
+-----------------+-----+
only showing top 20 rows
```

In [1]:

```
#  Part 1b: Exploratory Data Analysis (Large Scale)¶
# your code here for unique users
trimmed_files = [x[0] for x in spark.read.csv("s3://us-congress-tweets/trimmed/files.txt").collect()]
tweets_all = spark.read.parquet(*trimmed_files)
tweets_all.printSchema()
```

```
Starting Spark application
```

ID	YARN Application ID	Kind	State	
0	application_1572142677721_0001	pyspark	idle	Link (http://i 80.ec2.internal:20888/proxy/application_157214267

```
SparkSession available as 'spark'.
```

```
root
 |-- text: string (nullable = true)
 |-- id: long (nullable = true)
 |-- in_reply_to_user_id: long (nullable = true)
 |-- in_reply_to_status_id: long (nullable = true)
 |-- created_at: timestamp (nullable = true)
 |-- user: struct (nullable = true)
 |    |-- created_at: timestamp (nullable = true)
 |    |-- followers_count: long (nullable = true)
 |    |-- id: long (nullable = true)
 |    |-- name: string (nullable = true)
 |    |-- screen_name: string (nullable = true)
 |-- entities: struct (nullable = true)
 |    |-- hashtags: array (nullable = true)
 |    |    |-- element: struct (containsNull = true)
 |    |    |    |-- text: string (nullable = true)
 |    |-- user_mentions: array (nullable = true)
 |    |    |-- element: struct (containsNull = true)
 |    |    |    |-- id: long (nullable = true)
 |    |    |    |-- screen_name: string (nullable = true)
 |-- retweeted_status: struct (nullable = true)
 |    |-- id: long (nullable = true)
 |    |-- in_reply_to_user_id: long (nullable = true)
 |    |-- in_reply_to_status_id: long (nullable = true)
 |    |-- created_at: timestamp (nullable = true)
 |    |-- user: struct (nullable = true)
 |    |    |-- created_at: timestamp (nullable = true)
 |    |    |-- followers_count: long (nullable = true)
 |    |    |-- id: long (nullable = true)
 |    |    |-- name: string (nullable = true)
 |    |    |-- screen_name: string (nullable = true)
```

In [3]:

```
# your code here for unique users
tweets_all.select('user.id').distinct().count()
```

```
10749403
```

In [4]:

```
# your code here for original tweets
tweets_all.select('retweeted_status.id').distinct().count()
```

```
13458029
```

In [12]:

```
# Top mentioned users code and output here
from pyspark.sql.functions import desc

# tweets_all.groupBy('entities.user_mentions.id').count().sort(desc("count")).show
(11)

tweets.select(F.col("user.screen_name").alias("user"),
              F.explode("entities.user_mentions.screen_name").alias("mention"))\
      .groupby("mention").count().sort(F.desc("count")).show()
```

```
+---------------+-----+
|        mention|count|
+---------------+-----+
|     SenSchumer|  776|
|realDonaldTrump|  751|
|  RepAdamSchiff|  739|
|     marcorubio|  695|
|  SpeakerPelosi|  505|
|    NancyPelosi|  368|
|       RandPaul|  272|
|  ChrisMurphyCT|  223|
|  SenGillibrand|  220|
|   RepMattGaetz|  220|
|     CoryBooker|  214|
|  ChuckGrassley|  206|
|      JeffFlake|  202|
|   SteveScalise|  179|
|   amyklobuchar|  176|
|      GOPLeader|  164|
| RepJerryNadler|  162|
|          POTUS|  159|
|SenKamalaHarris|  152|
| SenJeffMerkley|  141|
+---------------+-----+
only showing top 20 rows
```

In [13]:

```
# Top hashtags code and output here
# tweets_all.groupBy('entities.hashtags').count().sort(desc("count")).show(11)

tweets.select(F.explode("entities.hashtags.text").alias("hashtag"))\
      .groupby("hashtag").count().sort(F.desc("count")).show()
```

```
+-----------------+-----+
|          hashtag|count|
+-----------------+-----+
|        Venezuela|  102|
|    TrumpShutdown|   42|
|     MaduroRegime|   29|
|             MAGA|   20|
|           Maduro|   20|
|      NancyPelosi|   19|
|    MuellerReport|   17|
|     BuildTheWall|   14|
|      TrumpResign|   14|
|        Kavanaugh|   14|
|     ForThePeople|   14|
|     GreenNewDeal|   13|
| MyHouseMyAmerica|   12|
|        transport|   12|
|          Florida|   10|
|             Cuba|   10|
|        Democrats|   10|
|    BrowardCounty|    9|
|             EEUU|    8|
|MaduroCrimeFamily|    8|
+-----------------+-----+
only showing top 20 rows
```

In [2]:

```
print(1)
```

```
1
```

In [ ]:

```
'''
'''
'''
'''
'''
'''
'''
'''
'''
'''
'''
```

(10 points) Part 2: Textual Analysis (LDA)
Using the LDA algorithm provided by the Spark Machine Learning (ML) library, find o
ut the ten most important topics.
Use s3://us-congress-tweetsddcdc for this task (you can reuse tweets_all dataframe
from Part1b).
You may want to work on a small sample first but report your results on the whole d
ataset.
Hint: for better results aggregate all tweets for a user into a single document

```
'''
'''
'''
'''
'''
'''
'''
'''
'''
'''
'''
```

In [2]:

```
trimmed_files = [x[0] for x in spark.read.csv("s3://us-congress-tweets/trimmed/files.txt").collect()]
tweets_all = spark.read.parquet(*trimmed_files)
tweets_all.printSchema()
```

```
root
 |-- text: string (nullable = true)
 |-- id: long (nullable = true)
 |-- in_reply_to_user_id: long (nullable = true)
 |-- in_reply_to_status_id: long (nullable = true)
 |-- created_at: timestamp (nullable = true)
 |-- user: struct (nullable = true)
 |    |-- created_at: timestamp (nullable = true)
 |    |-- followers_count: long (nullable = true)
 |    |-- id: long (nullable = true)
 |    |-- name: string (nullable = true)
 |    |-- screen_name: string (nullable = true)
 |-- entities: struct (nullable = true)
 |    |-- hashtags: array (nullable = true)
 |    |    |-- element: struct (containsNull = true)
 |    |    |    |-- text: string (nullable = true)
 |    |-- user_mentions: array (nullable = true)
 |    |    |-- element: struct (containsNull = true)
 |    |    |    |-- id: long (nullable = true)
 |    |    |    |-- screen_name: string (nullable = true)
 |-- retweeted_status: struct (nullable = true)
 |    |-- id: long (nullable = true)
 |    |-- in_reply_to_user_id: long (nullable = true)
 |    |-- in_reply_to_status_id: long (nullable = true)
 |    |-- created_at: timestamp (nullable = true)
 |    |-- user: struct (nullable = true)
 |    |    |-- created_at: timestamp (nullable = true)
 |    |    |-- followers_count: long (nullable = true)
 |    |    |-- id: long (nullable = true)
 |    |    |-- name: string (nullable = true)
 |    |    |-- screen_name: string (nullable = true)
```

In [3]:

```
tweets_sample = tweets_all.limit(1000)
tweets_sample.count()
```

```
1000
```

In [4]:

```
tweets_sample.printschema
```

```
'DataFrame' object has no attribute 'printschema'
Traceback (most recent call last):
  File "/usr/lib/spark/python/lib/pyspark.zip/pyspark/sql/dataframe.p
y", line 1301, in __getattr__
    "'%s' object has no attribute '%s'" % (self.__class__.__name__, nam
e))
AttributeError: 'DataFrame' object has no attribute 'printschema'
```

In [46]:

```

```

```
10000
```

In [57]:

```
tweets_sample.printSchema()
```

```
root
 |-- text: string (nullable = true)
 |-- id: long (nullable = true)
 |-- in_reply_to_user_id: long (nullable = true)
 |-- in_reply_to_status_id: long (nullable = true)
 |-- created_at: timestamp (nullable = true)
 |-- user: struct (nullable = true)
 |    |-- created_at: timestamp (nullable = true)
 |    |-- followers_count: long (nullable = true)
 |    |-- id: long (nullable = true)
 |    |-- name: string (nullable = true)
 |    |-- screen_name: string (nullable = true)
 |-- entities: struct (nullable = true)
 |    |-- hashtags: array (nullable = true)
 |    |    |-- element: struct (containsNull = true)
 |    |    |    |-- text: string (nullable = true)
 |    |-- user_mentions: array (nullable = true)
 |    |    |-- element: struct (containsNull = true)
 |    |    |    |-- id: long (nullable = true)
 |    |    |    |-- screen_name: string (nullable = true)
 |-- retweeted_status: struct (nullable = true)
 |    |-- id: long (nullable = true)
 |    |-- in_reply_to_user_id: long (nullable = true)
 |    |-- in_reply_to_status_id: long (nullable = true)
 |    |-- created_at: timestamp (nullable = true)
 |    |-- user: struct (nullable = true)
 |    |    |-- created_at: timestamp (nullable = true)
 |    |    |-- followers_count: long (nullable = true)
 |    |    |-- id: long (nullable = true)
 |    |    |-- name: string (nullable = true)
 |    |    |-- screen_name: string (nullable = true)
```

In [45]:

```
# tweets_sample.select("text").show(1)
from pyspark.sql.functions import desc
import pyspark.sql.functions as F

test_agg = tweets_sample.groupby('user.id').agg(F.concat_ws(", ", F.collect_list('t
ext')).alias('features'))

test_agg.show(2)
```

```
+--------+--------------------+
|      id|            features|
+--------+--------------------+
| 1408631|RT @AdyBarkan: Pl...|
|10196912|@SethAbramson @Ch...|
+--------+--------------------+
only showing top 2 rows
```

In [51]:

```
test_agg.show(2)
```

```
+------+--------------------+
|    id|            features|
+------+--------------------+
|640893|RT @DrewHolden360...|
|818186|RT @TheDemCoaliti...|
+------+--------------------+
only showing top 2 rows
```

In [53]:

```
test_agg.count()
test_agg.write.csv("s3://aws-logs-358879944178-us-east-1/tweets_sample_test_agg_102
5_x_features",header=True)
```

In [50]:

```
test_agg.printSchema()
```

```
root
 |-- id: long (nullable = true)
 |-- features: string (nullable = false)
```

In [8]:

```
test_agg.count()
```

```
993
```

In [63]:

```
test_agg_x = spark.read.csv("s3://aws-logs-358879944178-us-east-1/tweets_sample_tes
t_agg_1025_x_features", header=True)

test_agg_x.count()

test_agg_x.printSchema()

test_agg_x.show(50)
```

```
root
 |-- id: string (nullable = true)
 |-- features: string (nullable = true)

+--------------------+--------------------+
|                  id|            features|
+--------------------+--------------------+
|             8362562|@seungminkim @Dea...|
|             8406492|RT @MoveOn: #MeTo...|
|             9233842|RT @JeffBezos: Th...|
|             9715012|Cecelia Corey fro...|
|            10565952|RT @PPact: .@sena...|
|Your duty is to them| not partisan pol...|
|            12980002|RT @RBrownlowe1: ...|
|            14062467|RT @SenWarren: La...|
|            14167654|RT @OTOOLEFAN: Th...|
|            14352930|@JeffFlake - Wome...|
|            15050912|@SenSanders Are y...|
|            15254388|RT @CheriJacobus:...|
|            15932264|@TalbertSwan @Lin...|
|            15985211|RT @senatemajldr:...|
|            16117237|RT @DougJones: So...|
|            16376378|@gordy_shanks @Pa...|
|            16390848|RT @SenWarren: La...|
|            16479240|@senatemajldr @Ca...|
|            16504310|@Rep_Hunter When ...|
|            16538186|.@SenatorCollins ...|
|            16565972|RT @SteveSchmidtS...|
|            16592306|@LindseyGrahamSC ...|
|            17223391|@LindseyGrahamSC ...|
|            17224548|RT @DerekCressman...|
|            17230367|@d_e_mol @Gordon_...|
|            17278992|RT @Fight4Change2...|
| Glad you stood up. |                null|
|Now continue the ...|                null|
|            17493647|RT @howeasyweforg...|
|WHAT THE HELL KIN...|                null|
|            17495913|@TalbertSwan @Lin...|
|            17553587|@kerrijacobi @Sen...|
|     He totally lied|       several times|
|\"I don't care if...|                null|
|            17645505|RT @Lawrence: He ...|
|            17839038|@DrDenaGrayson @J...|
|            17890990|RT @SenWhitehouse...|
|            18063584|RT @JeffMerkley: ...|
|            18176907|RT @TomCottonAR: ...|
|            18244428|RT @charlescwcook...|
|            18923484|Senator @LisaMurk...|
|            19088152|RT @RepAdamSchiff...|
|            19519125|@SenDeanHeller - ...|
|            19790638|RT @beth2_k3a: So...|
|            20176416|RT @TheDemCoaliti...|
|            20319989|@SenSanders Buy B...|
|            20643785|RT @kim: @SteveSc...|
|            20667925|@FoxNews @SenJohn...|
|            20799185|RT @SteveSchmidtS...|
|            20952040|@matthewamiller @...|
```

```
+--------------------+--------------------+
only showing top 50 rows
```

In [60]:

```
test_agg_x.printSchema()
```

```
root
 |-- id: string (nullable = true)
 |-- features: string (nullable = true)
```

In [ ]:

```
# Remove the stop words
splitRDD_no_stop_words = splitRDD_no_stop.map(lambda w: (____, ____))
```

In [ ]:

```

```

In [64]:

```
from pyspark.ml.clustering import LDA

from pyspark.sql import SparkSession

dataset = spark.read.format("libsvm").load(test_agg_x)

# dateset = test_agg_x

# Trains a LDA model.
lda = LDA(k=10, maxIter=10)
model = lda.fit(dataset)

ll = model.logLikelihood(dataset)
lp = model.logPerplexity(dataset)

# Describe topics.
topics = model.describeTopics(3)

topics.show(truncate=False)

# Shows the result
transformed = model.transform(dataset)
transformed.show(truncate=False)
# $example off$

spark.stop()
```

```
'DataFrame' object has no attribute '_get_object_id'
Traceback (most recent call last):
  File "/usr/lib/spark/python/lib/pyspark.zip/pyspark/sql/readwriter.p
y", line 170, in load
    return self._df(self._jreader.load(self._spark._sc._jvm.PythonUtil
s.toSeq(path)))
  File "/usr/lib/spark/python/lib/py4j-0.10.7-src.zip/py4j/java_gatewa
y.py", line 1248, in __call__
    args_command, temp_args = self._build_args(*args)
  File "/usr/lib/spark/python/lib/py4j-0.10.7-src.zip/py4j/java_gatewa
y.py", line 1212, in _build_args
    (new_args, temp_args) = self._get_args(args)
  File "/usr/lib/spark/python/lib/py4j-0.10.7-src.zip/py4j/java_gatewa
y.py", line 1199, in _get_args
    temp_arg = converter.convert(arg, self.gateway_client)
  File "/usr/lib/spark/python/lib/py4j-0.10.7-src.zip/py4j/java_collect
ions.py", line 501, in convert
    java_list.add(element)
  File "/usr/lib/spark/python/lib/py4j-0.10.7-src.zip/py4j/java_gatewa
y.py", line 1248, in __call__
    args_command, temp_args = self._build_args(*args)
  File "/usr/lib/spark/python/lib/py4j-0.10.7-src.zip/py4j/java_gatewa
y.py", line 1218, in _build_args
    [get_command_part(arg, self.pool) for arg in new_args])
  File "/usr/lib/spark/python/lib/py4j-0.10.7-src.zip/py4j/java_gatewa
y.py", line 1218, in <listcomp>
    [get_command_part(arg, self.pool) for arg in new_args])
  File "/usr/lib/spark/python/lib/py4j-0.10.7-src.zip/py4j/protocol.p
y", line 298, in get_command_part
    command_part = REFERENCE_TYPE + parameter._get_object_id()
  File "/usr/lib/spark/python/lib/pyspark.zip/pyspark/sql/dataframe.p
y", line 1301, in __getattr__
    "'%s' object has no attribute '%s'" % (self.__class__.__name__, nam
e))
AttributeError: 'DataFrame' object has no attribute '_get_object_id'
```

In [29]:

```

```

```
name '__file__' is not defined
Traceback (most recent call last):
  File "<stdin>", line 32, in config
NameError: name '__file__' is not defined
```

In [ ]:

```
'''
'''
'''
'''
'''
'''
'''
'''
'''
'''
'''

(10 points) Part 3a: MapRedce
In this task, design a MapReduce program in python that reads all the original twee
ts (no retweets) in the sample tweets (congress-sample-10k.json.gz) and if a tweet
 is a reply to another tweet then output a record of the form <src_id, src_user, ds
t_id, dst_user>.

Create a small cluster (2 or 3 nodes) as per the AWS Guide and then ssh to your clu
ster and use Hadoop streaming to execute your mapreduce program.

Note: the Hadoop streaming jar file can be found at /usr/lib/hadoop-mapreduce/hadoo
p-streaming.jar
```

In [ ]:

```

```

In [ ]:

```

```

In [ ]:

```
'''
'''
'''
'''
'''
'''
'''
'''
'''
'''
(5 points) Part 3b: Going Large-Scale with MapReduce¶
Rerun the same MapReduce job above but on the whole dataset (s3://us-congress-tweet
s/raw/*.snappy). All the files under s3://us-congress-tweets/raw can be read from t
he following file:

s3://us-congress-tweets/raw/files.txt

Use shell scripting to parse this file and prepare the input to your MapReduce job
 as comma seperated string of all the files. (e.g. your input should be like this s
3://us-congress-tweets/raw/part-00000.snappy,s3://us-congress-tweets/raw/part-0000
1.snappy,s3://us-congress-tweets/raw/part-00002.snappy,...)

Inspecting the job logs, how many files did the job operate on? how many input spli
ts were there?
'''
'''
'''
'''
'''
'''
'''
'''
'''
'''
'''
```

In [ ]:

In [ ]:

In [ ]:

In [1]:

```
Starting Spark application
```

ID	YARN Application ID	Kind	State	
0	application_1572499199263_0001	pyspark	idle	Link (http://100.ec2.internal:20888/proxy/application_15724991

```
SparkSession available as 'spark'.
```

2

In [ ]: