In [8]:

```
from __future__ import print_function
import sys
from pyspark.sql import SparkSession
from pyspark import SparkConf
from pyspark import SparkContext


# create a SparkSession object
spark = SparkSession\
   .builder\
   .appName("White_house")\
   .getOrCreate()


#  sys.argv[0] is the name of the script.
#  sys.argv[1] is the first parameter
input_path = "D:/Big_data/whitehouse_waves-2016_12.csv"
print("input_path: {}".format(input_path))
```

```
input_path: D:/Big_data/whitehouse_waves-2016_12.csv
```

In [9]:

```
records = spark.sparkContext.textFile(input_path)
tokens = records.map(lambda x: x.lower())
tokens = tokens.map(lambda x: x.split(','))

tokens1 = tokens.map(lambda x: (x[0], x[1],x[2], x[19], x[20]))
header = tokens1.first()

tokens2 = tokens1.filter(lambda x: x != header)

filtered = tokens2.filter(lambda x: x[3] != "")
filtered = filtered.filter(lambda x: x[0] != "")

dropped = tokens2.count() - filtered.count()

print("Dropped records: ", dropped)
```

```
Dropped records:  59255
```

In [10]:

```
visitors = filtered.map(lambda x: ((x[0], x[1], x[2]), 1))
visitors = visitors.reduceByKey(lambda x,y: x+y)
top_10_visitors = visitors.takeOrdered(10, key = lambda x: -x[1])

print("Top 10 visitors are: ", top_10_visitors)
```

```
Top 10 visitors are:  [(('thomas', 'benjamin', 'l'), 185), (('berner', 'katherine', 'k'), 176),
(('haas', 'jordan', 'm'), 152), (('grant', 'patrick', 'c'), 151), (('kidwell', 'lauren', 'k'),
145), (('haro', 'steven', 'm'), 140), (('garza', 'steven', 'a'), 127), (('strait', 'elan', ''),
107), (('lew', 'shoshana', 'm'), 102), (('zeitlin', 'daniel', 'l'), 98)]
```

In [11]:

```
visitee = filtered.map(lambda x: ((x[3], x[4]), 1))
visitee = visitee.reduceByKey(lambda x,y: x+y)
top_10_visitee = visitee.takeOrdered(10, key = lambda x: -x[1])

print("Top 10 visitees are: ", top_10_visitee)
```

```
Top 10 visitees are:  [(('office', 'visitors'), 430881), (('waves', 'visitorsoffice'), 44129),
(('bryant', 'ruth'), 13970), (('oneil', 'olivia'), 13155), (('thompson', 'jared'), 11618), (('/',
'potus'), 10900), (('burton', 'collin'), 9672), (('megan', 'matthew'), 7944), (('mayerson',
'asher'), 6886), (('dessources', 'kalisha'), 5289)]
```

In [12]:

```
combo = filtered.map(lambda x: ((x[0], x[1], x[3], x[4]), 1))
combo = combo.reduceByKey(lambda x,y: x+y)
top_10_combo = combo.takeOrdered(10, key = lambda x: -x[1])

print("Top 10 visitors-visitees are: ", top_10_combo)
```

```
Top 10 visitors-visitees are:  [(('kidwell', 'lauren', 'yudelson', 'alex'), 103), (('haas',
'jordan', 'yudelson', 'alex'), 90), (('grant', 'patrick', 'yudelson', 'alex'), 89), (('thomas', 'b
enjamin', 'yudelson', 'alex'), 89), (('cohen', 'mandy', 'lambrew', 'jeanne'), 84), (('haro', 'stev
en', 'yudelson', 'alex'), 84), (('berner', 'katherine', 'yudelson', 'alex'), 82), (('roche', 'shan
non', 'yudelson', 'alex'), 70), (('urizar', 'jennifer', 'johnson', 'katie'), 68), (('martin', 'kat
hryn', 'lambrew', 'jeanne'), 61)]
```

In [ ]:

```

```