In [1]:

```
from __future__ import print_function
import sys
from operator import add
from pyspark.sql import SparkSession
from pyspark.sql import functions as f
from pyspark.sql.functions import *
from pyspark.sql.types import StringType
from pyspark.sql.functions import udf
```

In [2]:

```
spark = SparkSession\
.builder\
.appName("Assignment_4")\
.getOrCreate()
```

In [3]:

```
df = spark\
.read\
.format("csv")\
.option("header", "true")\
.option("inferSchema", "true")\
.load("D:/Big_data/whitehouse_waves-2016_12.csv")
```

In [4]:

```
cols_to_drop = ['UIN','BDGNBR', 'ACCESS_TYPE', 'TOA', 'TOD', 'POA', 'POD', 'APPT_MADE_DATE', 'APPT_
START_DATE', \
                'APPT_END_DATE', 'APPT_CANCEL_DATE', 'Total_People', 'LAST_UPDATEDBY', 'POST',
'LASTENTRYDATE', \
                'TERMINAL_SUFFIX', 'MEETING_LOC', 'MEETING_ROOM', 'CALLER_NAME_LAST',
'CALLER_NAME_FIRST', \
                'CALLER_ROOM', 'DESCRIPTION', 'Release_Date']
```

In [5]:

```
df = df.drop(*cols_to_drop)
```

In [6]:

```
df.show(5)
```

```
+---------------+---------+-------+----------------+-----------------+
|       NAMELAST|NAMEFIRST|NAMEMID|visitee_namelast|visitee_namefirst|
+---------------+---------+-------+----------------+-----------------+
|TAJOURIBESSASSI|   HANENE|   null|        Pelofsky|             Eric|
|        bageant|    laura|      j|     Baskerville|           Steven|
|       Broemson|     Earl|      H|     Baskerville|           Steven|
|    Jackling Jr|  William|      C|     Baskerville|           Steven|
|        McCrary|  Richard|      L|     Baskerville|           Steven|
+---------------+---------+-------+----------------+-----------------+
only showing top 5 rows
```

In [7]:

```
for col in df.columns:
    df = df.withColumn(col, f.lower(f.col(col)))
```

In [8]:

```
df = df.toDF(*[c.lower() for c in df.columns])
```

In [9]:

```
df.show(3)
```

```
+---------------+---------+-------+---------------+-----------------+
|       namelast|namefirst|namemid|visitee_namelast|visitee_namefirst|
+---------------+---------+-------+---------------+-----------------+
|tajouribessassi|   hanene|   null|        pelofsky|             eric|
|        bageant|    laura|      j|     baskerville|           steven|
|       broemson|     earl|      h|     baskerville|           steven|
+---------------+---------+-------+---------------+-----------------+
only showing top 3 rows
```

In [10]:

```
original_count = df.count()
original_count
```

Out[10]:

970504

In [11]:

```
org_df = df
```

In [12]:

```
df11 = df.dropna(subset = ['namelast', 'visitee_namelast'])
```

In [15]:

```
df11.count()
```

Out[15]:

911249

In [17]:

```
df = df.filter(df.visitee_namelast.rlike('[a-z]'))
```

In [18]:

```
df.count()
```

Out[18]:

900059

In [19]:

```
new_count = df.count()
```

In [20]:

```
dropped = original_count - new_count
print(dropped)
```

70445

In [21]:

```
top_10_visitors = df.groupby(['namelast', 'namefirst', 'namemid']).count().orderBy('count',
```

```
ascending=False)
```

In [22]:

```
top_10_visitors.show(10)
```

```
+--------+---------+-------+-----+
|namelast|namefirst|namemid|count|
+--------+---------+-------+-----+
|  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|   null|  107|
|     lew| shoshana|      m|  102|
| zeitlin|   daniel|      l|   98|
+--------+---------+-------+-----+
only showing top 10 rows
```

In [23]:

```
top_10_visitees = df.groupby(['visitee_namelast', 'visitee_namefirst']).count().orderBy('count', ascending=False)
```

In [24]:

```
top_10_visitees.show(10)
```

```
+---------------+----------------+------+
|visitee_namelast|visitee_namefirst| count|
+---------------+----------------+------+
|          office|         visitors|430881|
|           waves|   visitorsoffice| 44129|
|          bryant|            ruth| 13970|
|           oneil|          olivia| 13155|
|        thompson|           jared| 11618|
|          burton|          collin|  9672|
|           megan|         matthew|  7944|
|        mayerson|           asher|  6886|
|      dessources|         kalisha|  5289|
|           evans|           karen|  2908|
+---------------+----------------+------+
only showing top 10 rows
```

In [25]:

```
list1 = ['namelast', 'namefirst', 'visitee_namelast', 'visitee_namefirst']
```

In [26]:

```
top_10_combo = df.groupby(list1).count().orderBy('count', ascending = False)
```

In [27]:

```
top_10_combo.show(10)
```

```
+--------+---------+---------------+----------------+-----+
|namelast|namefirst|visitee_namelast|visitee_namefirst|count|
+--------+---------+---------------+----------------+-----+
| kidwell|   lauren|        yudelson|            alex|  103|
|    haas|   jordan|        yudelson|            alex|   90|
|   grant|  patrick|        yudelson|            alex|   89|
|  thomas| benjamin|        yudelson|            alex|   89|
|    haro|   steven|        yudelson|            alex|   84|
```

```
|   conen|    manay|         lambrew|           jeanne|   84|
|  berner|katherine|        yudelson|             alex|   82|
|   roche|  shannon|        yudelson|             alex|   70|
|  urizar| jennifer|         johnson|            katie|   68|
|  martin|  kathryn|         lambrew|           jeanne|   61|
+--------+---------+----------------+-----------------+-----+
only showing top 10 rows
```

In [ ]: