Nghi Le
1843642
Big-Data Technologies

Coursework 2 Report

Task 1:

Query 1:

```
# Connect to MongoDB by opening another terminal and enter the line below:

mongod --dbpath=./'Big Data Coursework'/CW2

# Load the file in

mongoimport --db cw2 --collection cl --drop --file ~/Downloads/'Big Data Coursework'/CW2/championsleague_1.json

# Output

2019-03-11T16:40:49.476+0000	connected to: localhost
2019-03-11T16:40:49.477+0000	dropping: cw2.cl
2019-03-11T16:40:49.750+0000	imported 23285 documents
```

Query 2:

```
db.cl.aggregate([ { $match: {$and: [ {"friendsCount": { $lt: 25 } }, { "displayName": /^A/i }, {"displayName": /es$/ } ] } }, { $group:{_id:"$displayName" , displayName: { $last: "$displayName"}, followersCount: {$last: "$followersCount" }, friendsCount: {$last: "$friendsCount"} } }, { $project: {_id: 0 , "displayName": 1, "friendsCount": 1, "followersCount": 1} } ] )

# Output:

{ "displayName" : "Arizona Companies", "followersCount" : 10, "friendsCount" : 0 }
{ "displayName" : "Adejies", "followersCount" : 10, "friendsCount" : 13 }
{ "displayName" : "angie torres", "followersCount" : 33, "friendsCount" :
     23 }
```

Query 3:

```
db.cl.aggregate ([ { $match: { "friendsCount": { $gt : 100000 } } }, { $group:
     { _id:"$displayName", LastFollowers: {$last: "$followersCount" } } },
     {$group: {_id: null, TotalAvgFollowers: {$avg: "$LastFollowers"} } },
     {$project: {_id: 0, TotalAvgFollowers: 1 } } ])

# Output:

{ "TotalAvgFollowers" : 528580.125 }
```

Query 4:

```
db.cl.aggregate([ { $match: { "friendsCount": { $gt: 0 } } }, { $project: { _id:
     0, name: "$displayName",ratio: {$divide: ["$followersCount",
     "$friendsCount"] } } }, {$group: {_id:null, AverageRatios: {$avg:
     "$ratio"} } }, {$project: {_id:0, AverageRatios: 1} } ])

#Output:

{ "AverageRatios" : 156.2146935903003 }
```

Query 5:

```
db.cl.aggregate([ { $match: { friendsCount: { $gte: 1000 }, verb: "post",
     body: /Madrid/ }  }, {$group:{_id:"$displayName"} }, {$count:
     "number_of_users"} ])

# Output:

{ "number_of_users" : 124 }
```

Query 6:

```
db.cl.aggregate([{ $match: {statusesCount: {$gt: 200, $lt: 203 } } },
     {$group: {_id: "$displayName", followersCount: {$last:
     "$followersCount"} } }, {$project: {_id: 1, followersCount: 1 ,
     statusesCount: 1} } ])
```

```
#Output

{ "_id" : "Silvia Alonso", "followersCount" : 322 }
{ "_id" : "Lex van Houten", "followersCount" : 79 }
{ "_id" : "Bendita Cocina", "followersCount" : 344 }
{ "_id" : "juan pablo suazo", "followersCount" : 77 }
{ "_id" : "jUnE 6 MiNe bAbY", "followersCount" : 139 }
{ "_id" : "emanuele lombardi", "followersCount" : 7 }
{ "_id" : "Prince-Vejita", "followersCount" : 25 }
{ "_id" : "Keshav Raghav", "followersCount" : 16 }
{ "_id" : "Indra J.P. Senaen", "followersCount" : 23 }
{ "_id" : "DigitalAnniversaries", "followersCount" : 1056 }
```

Task 2:

Code:

```
from pyspark import SparkContext
from operator import add

sc = SparkContext('local','pyspark')

def age_group(age):
    if age < 10:
            return '0-10'
    elif age < 20:
            return '10-20'
    elif age < 30:
            return '20-30'
    elif age < 40:
            return '30-40'
    elif age < 50:
            return '40-50'
    elif age < 60:
            return '50-60'
    elif age < 70:
            return '60-70'
    elif age < 80:
            return '70-80'
    else:
            return '80+'

def parse_with_age_group(data):
    userid, age, gender, occupation, zip = data.split("|")
    return userid, age_group(int(age)), gender, occupation, zip, int(age)

# Create RDD of u.user file:

fs = sc.textFile("file:///home/cloudera/Downloads/CW2/u.user")

# Convert age into age groups:

data_with_age_group = fs.map(parse_with_age_group)

# Sorting the data as RDDs:

# First, we will filter the data by the age group, then we map each occupation to a value of 1. Then, we use ReduceByKey
# method to count all the entries the belongs to each group. Then we use the SortBy method to sort by the values in descending order.
# Finally we use the keys method to get only the occupations.

sorted_40_50 = data_with_age_group.filter(lambda x: ('40-50' in x)).map(lambda x: (x[3],1)).reduceByKey(lambda a,b: a + b).sortBy(lambda x: x[1], 0).keys()

sorted_50_60 = data_with_age_group.filter(lambda x: ('50-60' in x)).map(lambda x: (x[3],1)).reduceByKey(lambda a,b: a + b).sortBy(lambda x: x[1], 0).keys()

# Get the top 10 most frequent occupation of each age group

top_40_50 = sorted_40_50.take(10)
top_50_60 = sorted_50_60.take(10)

# Get the intersection of the lists

print list(set(top_40_50) & set(top_50_60))
```

Output:

```
[u'administrator', u'healthcare', u'writer', u'other', u'educator',
    u'librarian', u'programmer', u'engineer']
```

Task 3:

Query 1:

```
CREATE DATABASE log_db;

USE log-db;

CREATE TABLE logs
(user VARCHAR(20),
time VARCHAR(20),
query CHAR(255))
ROW FORMAT DELIMITED
FIELDS TERMINATED BY '\t'
STORED AS TEXTFILE;

LOAD DATA LOCAL INPATH 'query_logs.txt' INTO TABLE logs;

# Output:

Loading data to table log_db.logs
Table log_db.logs stats: [numFiles=1, totalSize=204599]
OK
Time taken: 0.944 seconds
```

Query 2:

```
SELECT user, COUNT(*) AS Visits FROM logs GROUP BY user SORT BY Visits
    DESC LIMIT 1;

# Output:

128315306CE647F6	78
```

Query 3:

```
SELECT user, COUNT(*) FROM (SELECT user, query FROM logs WHERE
    query LIKE '%business') AS table GROUP BY user;

# Output:

02E76389CBC661F7    4
0B294E3062F036C3    11
74165896F4654D30    2
Time taken: 24.963 seconds, Fetched: 3 row(s)
```

Query 4:

```
SELECT * FROM logs WHERE query LIKE '%job%';

# Output:

4077443B5801F0C3    970916182623    job openings
4077443B5801F0C3    970916182752    job openings listings
4077443B5801F0C3    970916182823    agricultural job listings
4077443B5801F0C3    970916182834    agricultural job listings
    employdogst
4077443B5801F0C3    970916182942    job listings
83607290B8BEAFC6    970916070440    jobs=hong kong
D5D8220D36969861    970916222708    job interview tips
D5D8220D36969861    970916224215    job interview tips
0E10DD8EB5EEB192    970916134219    jobs at the university of
    minnesota
567854C718273984    970916021936    part time jobs
567854C718273984    970916022012    part time jobs
567854C718273984    970916022043    part time jobs
567854C718273984    970916022117    part time jobs
567854C718273984    970916022202    part time jobs
567854C718273984    970916022313    home jobs
567854C718273984    970916022444    home jobs
```

Query 5:

```
SELECT COUNT(DISTINCT user) FROM logs WHERE query != '' AND
    (from_unixtime(UNIX_TIMESTAMP(time,'yyMMddHHmmss'),
    'HH:mm:ss') BETWEEN '21:00:00' and '22:59:59');

# Output:

67
```