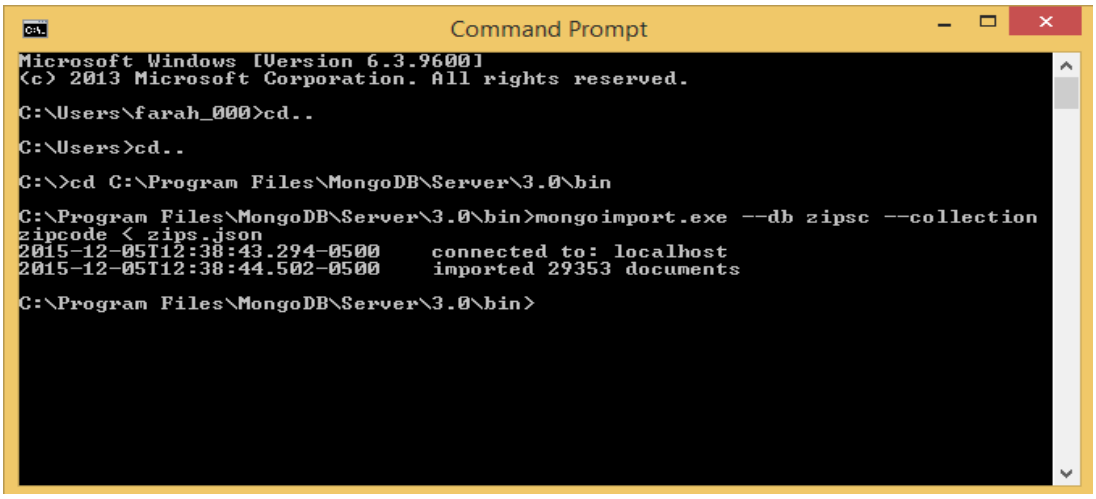
Subject: HOMEWORK #9 Advanced DataBase Name: Farah Kamw

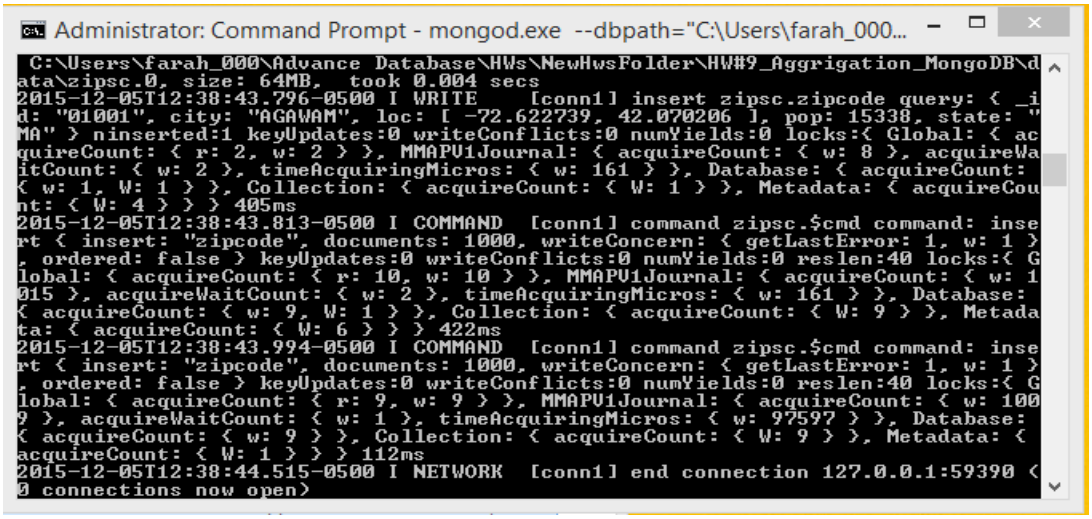
Fkamw@kent.edu Aggrigation-MongoDB

1. For aggregation, I read the tutorial on the following two links:
   https://docs.mongodb.org/manual/core/aggregation-introduction/
   https://docs.mongodb.org/manual/tutorial/aggregation-zip-code-data-set/
2. I have downloaded the zips. Json file from the following link media.mongodb.org/zips.json. Then I used mongo Import to import the zips file into mongo DB database. See the following two figures.



```
Command Prompt
Microsoft Windows [Version 6.3.9600]
(c) 2013 Microsoft Corporation. All rights reserved.

C:\Users\farah_000>cd..

C:\Users>cd..

C:\>cd C:\Program Files\MongoDB\Server\3.0\bin

C:\Program Files\MongoDB\Server\3.0\bin>mongoimport.exe --db zipsc --collection
zipcode < zips.json
2015-12-05T12:38:43.294-0500    connected to: localhost
2015-12-05T12:38:44.502-0500    imported 29353 documents

C:\Program Files\MongoDB\Server\3.0\bin>
```

Figure 1

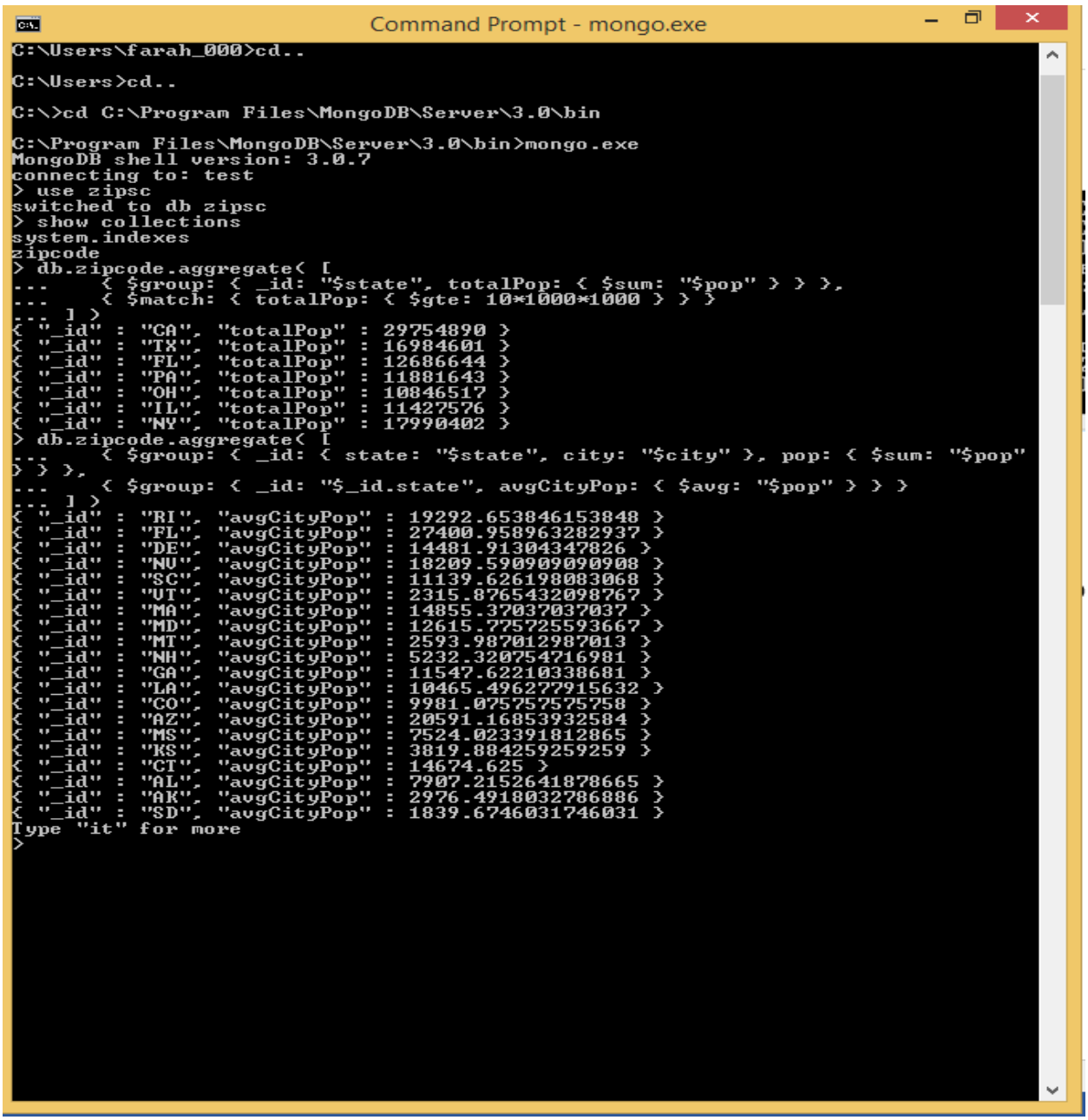


```
Administrator: Command Prompt - mongod.exe --dbpath="C:\Users\farah_000...
C:\Users\farah_000\Advance Database\HWs\NewHwsFolder\HW#9_Aggrigation_MongoDB\d
ata\zipsc.0, size: 64MB,  took 0.004 secs
2015-12-05T12:38:43.796-0500 I WRITE    [conn1] insert zipsc.zipcode query: { _i
d: "01001", city: "AGAWAM", loc: [ -72.622739, 42.070206 ], pop: 15338, state: "
MA" } ninserted:1 keyUpdates:0 writeConflicts:0 numYields:0 locks:{ Global: { ac
quireCount: { r: 2, w: 2 } }, MMAPV1Journal: { acquireCount: { w: 8 }, acquireWa
itCount: { w: 2 }, timeAcquiringMicros: { w: 161 } }, Database: { acquireCount:
{ w: 1, W: 1 } }, Collection: { acquireCount: { W: 1 } }, Metadata: { acquireCou
nt: { W: 4 } } } 405ms
2015-12-05T12:38:43.813-0500 I COMMAND  [conn1] command zipsc.$cmd command: inse
rt { insert: "zipcode", documents: 1000, writeConcern: { getLastError: 1, w: 1 }
, ordered: false } keyUpdates:0 writeConflicts:0 numYields:0 reslen:40 locks:{ G
lobal: { acquireCount: { r: 10, w: 10 } }, MMAPV1Journal: { acquireCount: { w: 1
015 }, acquireWaitCount: { w: 2 }, timeAcquiringMicros: { w: 161 } }, Database:
{ acquireCount: { w: 9, W: 1 } }, Collection: { acquireCount: { W: 9 } }, Metada
ta: { acquireCount: { W: 6 } } } 422ms
2015-12-05T12:38:43.994-0500 I COMMAND  [conn1] command zipsc.$cmd command: inse
rt { insert: "zipcode", documents: 1000, writeConcern: { getLastError: 1, w: 1 }
, ordered: false } keyUpdates:0 writeConflicts:0 numYields:0 reslen:40 locks:{ G
lobal: { acquireCount: { r: 9, w: 9 } }, MMAPV1Journal: { acquireCount: { w: 100
9 }, acquireWaitCount: { w: 1 }, timeAcquiringMicros: { w: 97597 } }, Database:
{ acquireCount: { w: 9 } }, Collection: { acquireCount: { W: 9 } }, Metadata: {
acquireCount: { W: 1 } } } 112ms
2015-12-05T12:38:44.515-0500 I NETWORK  [conn1] end connection 127.0.0.1:59390 (
0 connections now open)
```

Figure2

Then I have implemented the (Aggregation with the Zip Code Data Set example) on the mongo shell.

First, I implemented the aggregate query to Return States with populations above 10 Million. Then, I implemented a multistage pipeline aggregate query to return average city population by State. See the following figure.



```
Command Prompt - mongo.exe
C:\Users\farah_000>cd..

C:\Users>cd..

C:\>cd C:\Program Files\MongoDB\Server\3.0\bin

C:\Program Files\MongoDB\Server\3.0\bin>mongo.exe
MongoDB shell version: 3.0.7
connecting to: test
> use zipsc
switched to db zipsc
> show collections
system.indexes
zipcode
> db.zipcode.aggregate( [
...     { $group: { _id: "$state", totalPop: { $sum: "$pop" } } },
...     { $match: { totalPop: { $gte: 10*1000*1000 } } }
... ] )
{ "_id" : "CA", "totalPop" : 29754890 }
{ "_id" : "TX", "totalPop" : 16984601 }
{ "_id" : "FL", "totalPop" : 12686644 }
{ "_id" : "PA", "totalPop" : 11881643 }
{ "_id" : "OH", "totalPop" : 10846517 }
{ "_id" : "IL", "totalPop" : 11427576 }
{ "_id" : "NY", "totalPop" : 17990402 }
> db.zipcode.aggregate( [
...     { $group: { _id: { state: "$state", city: "$city" }, pop: { $sum: "$pop"
} } },
...     { $group: { _id: "$_id.state", avgCityPop: { $avg: "$pop" } } }
... ] )
{ "_id" : "RI", "avgCityPop" : 19292.653846153848 }
{ "_id" : "FL", "avgCityPop" : 27400.958963282937 }
{ "_id" : "DE", "avgCityPop" : 14481.91304347826 }
{ "_id" : "NV", "avgCityPop" : 18209.590909090908 }
{ "_id" : "SC", "avgCityPop" : 11139.626198083068 }
{ "_id" : "VT", "avgCityPop" : 2315.876543209876 7 }
{ "_id" : "MA", "avgCityPop" : 14855.37037037037 }
{ "_id" : "MD", "avgCityPop" : 12615.775725593667 }
{ "_id" : "MT", "avgCityPop" : 2593.987012987013 }
{ "_id" : "NH", "avgCityPop" : 5232.320754716981 }
{ "_id" : "GA", "avgCityPop" : 11547.62210338681 }
{ "_id" : "LA", "avgCityPop" : 10465.496277915632 }
{ "_id" : "CO", "avgCityPop" : 9981.075757575758 }
{ "_id" : "AZ", "avgCityPop" : 20591.16853932584 }
{ "_id" : "MS", "avgCityPop" : 7524.023391812865 }
{ "_id" : "KS", "avgCityPop" : 3819.884259259259 }
{ "_id" : "CT", "avgCityPop" : 14674.625 }
{ "_id" : "AL", "avgCityPop" : 7907.2152641878665 }
{ "_id" : "AK", "avgCityPop" : 2976.4918032786886 }
{ "_id" : "SD", "avgCityPop" : 1839.6746031746031 }
Type "it" for more
>
```

Figure 3

In addition to that, I have implemented a complicated aggregation query to return the smallest and largest cities by population for each state. See figure 4.

```
Command Prompt - mongo.exe
{ "_id" : "AZ", "avgCityPop" : 20591.16853932584 }
{ "_id" : "MS", "avgCityPop" : 7524.023391812865 }
{ "_id" : "KS", "avgCityPop" : 3819.884259259259 }
{ "_id" : "CT", "avgCityPop" : 14674.625 }
{ "_id" : "AL", "avgCityPop" : 7907.2152641878665 }
{ "_id" : "AK", "avgCityPop" : 2976.4918032786886 }
{ "_id" : "SD", "avgCityPop" : 1839.6746031746031 }
Type "it" for more
> db.zipcode.aggregate( [
... { $group:
... {
... _id: { state: "$state", city: "$city" },
... pop: { $sum: "$pop" }
... }
... },
... { $sort: { pop: 1 } },
... { $group:
... {
... _id : "$_id.state",
... biggestCity: { $last: "$_id.city" },
... biggestPop: { $last: "$pop" },
... smallestCity: { $first: "$_id.city" },
... smallestPop: { $first: "$pop" }
... }
... }
... ] )
{ "_id" : "IN", "biggestCity" : "INDIANAPOLIS", "biggestPop" : 348868, "smallest
City" : "WESTPOINT", "smallestPop" : 145 }
{ "_id" : "RI", "biggestCity" : "CRANSTON", "biggestPop" : 176404, "smallestCity
" : "CLAYVILLE", "smallestPop" : 45 }
{ "_id" : "OH", "biggestCity" : "CLEVELAND", "biggestPop" : 536759, "smallestCit
y" : "ISLE SAINT GEORG", "smallestPop" : 38 }
{ "_id" : "MD", "biggestCity" : "BALTIMORE", "biggestPop" : 733081, "smallestCit
y" : "ANNAPOLIS JUNCTI", "smallestPop" : 32 }
{ "_id" : "NH", "biggestCity" : "MANCHESTER", "biggestPop" : 106452, "smallestCi
ty" : "WEST NOTTINGHAM", "smallestPop" : 27 }
{ "_id" : "MA", "biggestCity" : "WORCESTER", "biggestPop" : 169856, "smallestCit
y" : "BUCKLAND", "smallestPop" : 16 }
{ "_id" : "ND", "biggestCity" : "GRAND FORKS", "biggestPop" : 59527, "smallestCi
ty" : "TROTTERS", "smallestPop" : 12 }
{ "_id" : "DC", "biggestCity" : "WASHINGTON", "biggestPop" : 606879, "smallestCi
ty" : "PENTAGON", "smallestPop" : 21 }
{ "_id" : "MN", "biggestCity" : "MINNEAPOLIS", "biggestPop" : 344719, "smallestC
ity" : "JOHNSON", "smallestPop" : 12 }
{ "_id" : "UT", "biggestCity" : "SALT LAKE CITY", "biggestPop" : 186346, "smalle
stCity" : "MODENA", "smallestPop" : 9 }
{ "_id" : "OK", "biggestCity" : "TULSA", "biggestPop" : 389072, "smallestCity" :
 "SOUTHARD", "smallestPop" : 8 }
{ "_id" : "MT", "biggestCity" : "BILLINGS", "biggestPop" : 78805, "smallestCity"
 : "MOSBY", "smallestPop" : 7 }
{ "_id" : "NE", "biggestCity" : "OMAHA", "biggestPop" : 358930, "smallestCity" :
 "LAKESIDE", "smallestPop" : 5 }
{ "_id" : "WA", "biggestCity" : "SEATTLE", "biggestPop" : 520096, "smallestCity"
 : "BENGE", "smallestPop" : 2 }
{ "_id" : "TN", "biggestCity" : "MEMPHIS", "biggestPop" : 632837, "smallestCity"
 : "ALLRED", "smallestPop" : 2 }
{ "_id" : "DE", "biggestCity" : "NEWARK", "biggestPop" : 111674, "smallestCity"
: "BETHEL", "smallestPop" : 108 }
{ "_id" : "NV", "biggestCity" : "LAS VEGAS", "biggestPop" : 597557, "smallestCit
y" : "TUSCARORA", "smallestPop" : 1 }
{ "_id" : "IA", "biggestCity" : "DES MOINES", "biggestPop" : 148155, "smallestCi
ty" : "DOUDS", "smallestPop" : 15 }
{ "_id" : "NC", "biggestCity" : "CHARLOTTE", "biggestPop" : 465833, "smallestCit
y" : "GLOUCESTER", "smallestPop" : 0 }
{ "_id" : "VA", "biggestCity" : "VIRGINIA BEACH", "biggestPop" : 385080, "smalle
stCity" : "WALLOPS ISLAND", "smallestPop" : 0 }
Type "it" for more
>
```

Figure 4

3. I have implemented the same things on python using PyMongo.py. See figure 5 for the python code.


```
from pymongo import MongoClient
from bson.son import SON
db = MongoClient().zipsc
pipeline = [ { "$group": { "_id": "$state", "totalPop": { "$sum": "$pop" } } },{ "$match": { "totalPop": { "$gte": 10*1000*1000 } } }]
x=list(db.zipcode.aggregate(pipeline))
print x
pipeline2 = [{ "$group": { "_id": { "state": "$state", "city": "$city" }, "pop": { "$sum": "$pop" } } },{ "$group": { "_id": "$_id.state", "avgCityPop": { "$avg": "$pop" } } }]
x2=list(db.zipcode.aggregate(pipeline2))
print x2
pipeline3 = [
   { "$group":
      {
        "_id": { "state": "$state", "city": "$city" },
        "pop": { "$sum": "$pop" }
      }
   },
   { "$sort": { "pop": 1 } },
   { "$group":
      {
        "_id" : "$_id.state",
        "biggestCity":  { "$last": "$_id.city" },
        "biggestPop":   { "$last": "$pop" },
        "smallestCity": { "$first": "$_id.city" },
        "smallestPop":  { "$first": "$pop" }
      }
   }]
x3=list(db.zipcode.aggregate(pipeline3))
print x3
```

Figure 5

The following figures 6 and 7 show the results of implementation of the three aggregation queries in python shell.

```
Python 2.7.9 Shell
File  Edit  Shell  Debug  Options  Windows  Help
Python 2.7.9 (default, Dec 10 2014, 12:24:55) [MSC v.1500 32 bit (Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>> from pymongo import MongoClient
>>> from bson.son import SON

>>> db = MongoClient().zipsc
>>> pipeline = [ { "$group": { "_id": "$state", "totalPop": { "$sum": "$pop" } } },{ "$match": { "totalPop": { "$gte": 10*1000*1000 } } }]

>>> x=list(db.zipcode.aggregate(pipeline))

>>> print x
[{u'_id': u'CA', u'totalPop': 29754890}, {u'_id': u'TX', u'totalPop': 16984601}, {u'_id': u'FL', u'totalPop': 12686644}, {u'_id': u'PA', u'totalPop': 11881643}, {u'_id': u'OH', u'totalPop': 108465
17}, {u'_id': u'IL', u'totalPop': 11427576}, {u'_id': u'NY', u'totalPop': 17990402}]
>>> pipeline2 = [{ "$group": { "_id": { "state": "$state", "city": "$city" }, "pop": { "$sum": "$pop" } } },{ "$group": { "_id": "$_id.state", "avgCityPop": { "$avg": "$pop" } } }]

>>> x2=list(db.zipcode.aggregate(pipeline2))
>>> print x2
[{u'_id': u'RI', u'avgCityPop': 19292.653846153846}, {u'_id': u'VT', u'avgCityPop': 2315.8765432098767}, {u'_id': u'CT', u'avgCityPop': 14674.625}, {u'_id': u'AL', u'avgCityPop': 7907.215264187866
5}, {u'_id': u'ND', u'avgCityPop': 1645.0309278350514}, {u'_id': u'OK', u'avgCityPop': 6155.743639921722}, {u'_id': u'MD', u'avgCityPop': 12615.775725593667}, {u'_id': u'MA', u'avgCityPop': 14855.
37037037037}, {u'_id': u'UT', u'avgCityPop': 9518.508287292818}, {u'_id': u'MT', u'avgCityPop': 2593.987012987013}, {u'_id': u'NH', u'avgCityPop': 5232.320754716981}, {u'_id': u'VA', u'avgCityPop'
: 8526.177931034483}, {u'_id': u'NY', u'avgCityPop': 13131.680291970803}, {u'_id': u'TN', u'avgCityPop': 9656.350495049504}, {u'_id': u'AZ', u'avgCityPop': 20591.16853932584}, {u'_id': u'KS', u'av
gCityPop': 3819.884259259259}, {u'_id': u'MS', u'avgCityPop': 7524.023391812865}, {u'_id': u'CO', u'avgCityPop': 9981.075757575758}, {u'_id': u'LA', u'avgCityPop': 10465.496277915632}, {u'_id': u'
IA', u'avgCityPop': 3123.0821147356583}, {u'_id': u'NC', u'avgCityPop': 10622.815705128205}, {u'_id': u'FL', u'avgCityPop': 27400.958963282937}, {u'_id': u'DE', u'avgCityPop': 14481.91304347826},
{u'_id': u'NV', u'avgCityPop': 18209.590909090908}, {u'_id': u'SC', u'avgCityPop': 11139.626198083068}, {u'_id': u'PA', u'avgCityPop': 8679.067202337472}, {u'_id': u'ID', u'avgCityPop': 4320.81115
8798283}, {u'_id': u'WI', u'avgCityPop': 7323.00748502994}, {u'_id': u'KY', u'avgCityPop': 4767.164721141375}, {u'_id': u'AK', u'avgCityPop': 2976.4918032786886}, {u'_id': u'SD', u'avgCityPop': 18
39.6746031746031}, {u'_id': u'HI', u'avgCityPop': 15831.842857142858}, {u'_id': u'MO', u'avgCityPop': 5672.195338512764}, {u'_id': u'TX', u'avgCityPop': 13775.02108678021}, {u'_id': u'GA', u'avgCi
tyPop': 11547.62210338681}, {u'_id': u'ME', u'avgCityPop': 3006.4901960784314}, {u'_id': u'IN', u'avgCityPop': 9271.130434782608}, {u'_id': u'WV', u'avgCityPop': 2771.4775888717154}, {u'_id': u'AR
', u'avgCityPop': 4175.355239786856}, {u'_id': u'NJ', u'avgCityPop': 15775.89387755102}, {u'_id': u'IL', u'avgCityPop': 9954.334494773519}, {u'_id': u'WY', u'avgCityPop': 3384.5373134328356}, {u'_
id': u'MI', u'avgCityPop': 12087.512353706112}, {u'_id': u'OH', u'avgCityPop': 12700.839578454332}, {u'_id': u'NE', u'avgCityPop': 3034.882692307692}, {u'_id': u'NM', u'avgCityPop': 5872.360465116
279}, {u'_id': u'OR', u'avgCityPop': 8262.561046511628}, {u'_id': u'WA', u'avgCityPop': 12258.670025188916}, {u'_id': u'DC', u'avgCityPop': 303450.0}, {u'_id': u'MN', u'avgCityPop': 5372.213759213
76}, {u'_id': u'CA', u'avgCityPop': 27756.42723880597}]
```

Figure 6

```
Python 2.7.9 Shell
File  Edit  Shell  Debug  Options  Windows  Help
>>> pipeline3 = [
   { "$group":
      {
        "_id": { "state": "$state", "city": "$city" },
        "pop": { "$sum": "$pop" }
      }
   },
   { "$sort": { "pop": 1 } },
   { "$group":
      {
        "_id" : "$_id.state",
        "biggestCity":  { "$last": "$_id.city" },
        "biggestPop":   { "$last": "$pop" },
        "smallestCity": { "$first": "$_id.city" },
        "smallestPop":  { "$first": "$pop" }
      }
   }]
>>> x3=list(db.zipcode.aggregate(pipeline3))
>>> print x3
[{u'biggestPop': 348868, u'_id': u'IN', u'biggestCity': u'INDIANAPOLIS', u'smallestCity': u'WESTPOINT', u'smallestPop': 145}, {u'biggestPop': 176404, u'_id': u'RI', u'biggestCity': u'CRANSTON', u'
smallestCity': u'CLAYVILLE', u'smallestPop': 45}, {u'biggestPop': 536759, u'_id': u'OH', u'biggestCity': u'CLEVELAND', u'smallestCity': u'ISLE SAINT GEORG', u'smallestPop': 38}, {u'biggestPop': 73
3081, u'_id': u'MD', u'biggestCity': u'BALTIMORE', u'smallestCity': u'ANNAPOLIS JUNCTI', u'smallestPop': 32}, {u'biggestPop': 106452, u'_id': u'NH', u'biggestCity': u'MANCHESTER', u'smallestCity':
u'WEST NOTTINGHAM', u'smallestPop': 27}, {u'biggestPop': 169856, u'_id': u'MA', u'biggestCity': u'WORCESTER', u'smallestCity': u'BUCKLAND', u'smallestPop': 16}, {u'biggestPop': 606879, u'_id': u'D
C', u'biggestCity': u'WASHINGTON', u'smallestCity': u'PENTAGON', u'smallestPop': 21}, {u'biggestPop': 344719, u'_id': u'MN', u'biggestCity': u'MINNEAPOLIS', u'smallestCity': u'JOHNSON', u'smallest
Pop': 12}, {u'biggestPop': 59527, u'_id': u'ND', u'biggestCity': u'GRAND FORKS', u'smallestCity': u'TROTTERS', u'smallestPop': 12}, {u'biggestPop': 186346, u'_id': u'UT', u'biggestCity': u'SALT LA
KE CITY', u'smallestCity': u'MODENA', u'smallestPop': 9}, {u'biggestPop': 389072, u'_id': u'OK', u'biggestCity': u'TULSA', u'smallestCity': u'SOUTHARD', u'smallestPop': 8}, {u'biggestPop': 78805,
u'_id': u'MT', u'biggestCity': u'BILLINGS', u'smallestCity': u'HOMESTEAD', u'smallestPop': 7}, {u'biggestPop': 358930, u'_id': u'NE', u'biggestCity': u'OMAHA', u'smallestCity': u'LAKESIDE', u'smal
lestPop': 5}, {u'biggestPop': 632837, u'_id': u'TN', u'biggestCity': u'MEMPHIS', u'smallestCity': u'ALLRED', u'smallestPop': 2}, {u'biggestPop': 520096, u'_id': u'WA', u'biggestCity': u'SEATTLE',
u'smallestCity': u'BENGE', u'smallestPop': 2}, {u'biggestPop': 111674, u'_id': u'DE', u'biggestCity': u'NEWARK', u'smallestCity': u'BETHEL', u'smallestPop': 108}, {u'biggestPop': 597557, u'_id': u
'NV', u'biggestCity': u'LAS VEGAS', u'smallestCity': u'TUSCARORA', u'smallestPop': 1}, {u'biggestPop': 1610956, u'_id': u'PA', u'biggestCity': u'PHILADELPHIA', u'smallestCity': u'HAMILTON', u'smal
lestPop': 0}, {u'biggestPop': 39127, u'_id': u'VT', u'biggestCity': u'BURLINGTON', u'smallestCity': u'UNIV OF VERMONT', u'smallestPop': 0}, {u'biggestPop': 148155, u'_id': u'IA', u'biggestCity': u
'DES MOINES', u'smallestCity': u'DOUDS', u'smallestPop': 15}, {u'biggestPop': 465833, u'_id': u'NC', u'biggestCity': u'CHARLOTTE', u'smallestCity': u'GLOUCESTER', u'smallestPop': 0}, {u'biggestPop
': 385080, u'_id': u'VA', u'biggestCity': u'VIRGINIA BEACH', u'smallestCity': u'WALLOPS ISLAND', u'smallestPop': 0}, {u'biggestPop': 63268, u'_id': u'ME', u'biggestCity': u'PORTLAND', u'smallestCi
ty': u'BUSTINS ISLAND', u'smallestPop': 0}, {u'biggestPop': 75343, u'_id': u'WV', u'biggestCity': u'HUNTINGTON', u'smallestCity': u'MOUNT CARBON', u'smallestPop': 0}, {u'biggestPop': 609591, u'_id
': u'GA', u'biggestCity': u'ATLANTA', u'smallestCity': u'FORT STEWART', u'smallestPop': 0}, {u'biggestPop': 269521, u'_id': u'SC', u'biggestCity': u'COLUMBIA', u'smallestCity': u'QUINBY', u'smalle
stPop': 0}, {u'biggestPop': 496937, u'_id': u'LA', u'biggestCity': u'NEW ORLEANS', u'smallestCity': u'FORDOCHE', u'smallestPop': 0}, {u'biggestPop': 102046, u'_id': u'SD', u'biggestCity': u'SIOUX
FALLS', u'smallestCity': u'ZEONA', u'smallestPop': 8}, {u'biggestPop': 183987, u'_id': u'AK', u'biggestCity': u'ANCHORAGE', u'smallestCity': u'SELAWIK', u'smallestPop': 0}, {u'biggestPop': 2452177
, u'_id': u'IL', u'biggestCity': u'CHICAGO', u'smallestCity': u'ANCONA', u'smallestPop': 38}, {u'biggestPop': 275572, u'_id': u'NJ', u'biggestCity': u'NEWARK', u'smallestCity': u'IMLAYSTOWN', u'sm
allestPop': 17}, {u'biggestPop': 192895, u'_id': u'AR', u'biggestCity': u'LITTLE ROCK', u'smallestCity': u'TOMATO', u'smallestPop': 0}, {u'biggestPop': 288058, u'_id': u'KY', u'biggestCity': u'LOU
ISVILLE', u'smallestCity': u'BIG LAUREL', u'smallestPop': 0}, {u'biggestPop': 165522, u'_id': u'ID', u'biggestCity': u'BOISE', u'smallestCity': u'KEUTERVILLE', u'smallestPop': 0}, {u'biggestPop':
597324, u'_id': u'WI', u'biggestCity': u'MILWAUKEE', u'smallestCity': u'CLAM LAKE', u'smallestPop': 2}, {u'biggestPop': 397802, u'_id': u'MO', u'biggestCity': u'SAINT LOUIS', u'smallestCity': u'BE
NDAVIS', u'smallestPop': 44}, {u'biggestPop': 2095918, u'_id': u'TX', u'biggestCity': u'HOUSTON', u'smallestCity': u'ECLETO', u'smallestPop': 0}, {u'biggestPop': 396643, u'_id': u'HI', u'biggestCi
ty': u'HONOLULU', u'smallestCity': u'NINOLE', u'smallestPop': 0}, {u'biggestPop': 449584, u'_id': u'NM', u'biggestCity': u'ALBUQUERQUE', u'smallestCity': u'OIL CENTER', u'smallestPop': 0}, {u'bigg
estPop': 518543, u'_id': u'OR', u'biggestCity': u'PORTLAND', u'smallestCity': u'KENT', u'smallestPop': 0}, {u'biggestPop': 204788, u'_id': u'MS', u'biggestCity': u'JACKSON', u'smallestCity': u'CHU
NKY', u'smallestPop': 79}, {u'biggestPop': 890853, u'_id': u'AZ', u'biggestCity': u'PHOENIX', u'smallestCity': u'HUALAPAI', u'smallestPop': 2}, {u'biggestPop': 451182, u'_id': u'CO', u'biggestCity
': u'DENVER', u'smallestCity': u'CHEYENNE MTN AFB', u'smallestPop': 0}, {u'biggestPop': 295115, u'_id': u'KS', u'biggestCity': u'WICHITA', u'smallestCity': u'ARNOLD', u'smallestPop': 0}, {u'bigges
tPop': 825232, u'_id': u'FL', u'biggestCity': u'MIAMI', u'smallestCity': u'CECIL FIELD NAS', u'smallestPop': 0}, {u'biggestPop': 2300504, u'_id': u'NY', u'biggestCity': u'BROOKLYN', u'smallestCity
': u'RAQUETTE LAKE', u'smallestPop': 0}, {u'biggestPop': 70185, u'_id': u'WY', u'biggestCity': u'CHEYENNE', u'smallestCity': u'LOST SPRINGS', u'smallestPop': 6}, {u'biggestPop': 963243, u'_id': u'
MI', u'biggestCity': u'DETROIT', u'smallestCity': u'LELAND', u'smallestPop': 0}, {u'biggestPop': 141638, u'_id': u'CT', u'biggestCity': u'BRIDGEPORT', u'smallestCity': u'EAST KILLINGLY', u'smalles
```

Figure 7

4. I have added a function to the To Do application (app.py) to count number of completed and not done tasks. See figures 8, 9, and 10.

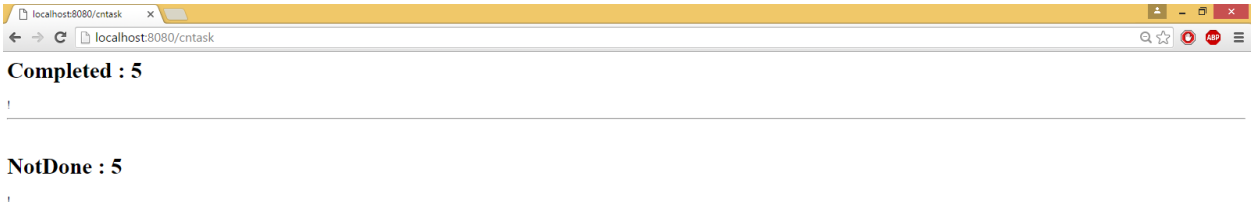


Figure 8

```
def information():
    result={};
    pipeline =  [
      { "$match": { "status": { "$eq": 1 } } },{ "$group": { "_id": "null", "Completed": { "$sum": 1 } } }

   ]
    x=list(db.tasktable.aggregate(pipeline))
    print "Completed: ",x[0]["Completed"]
    result["Completed"]=x[0]["Completed"]
    pipeline2 =  [
      { "$match": { "status": { "$eq": 0 } } },{ "$group": { "_id": "null", "NotDone": { "$sum": 1 } } }

   ]
    x1=list(db.tasktable.aggregate(pipeline2))
    print "Not Done : ",x1[0]["NotDone"]
    result["NotDone"]=x1[0]["NotDone"]
    return result;
```

Figure 9

```
@route('/cntask')
def cntask():
    r=information();
    completed=r["Completed"]
    NotDone=r["NotDone"]
    return template('<h1>Completed : {{c}}</h1> !<br/><hr/><br/><h1>NotDone : {{NotDone}}</h1> !',c=completed, NotDone=NotDone)

run(host='localhost', port=8080)
```

Figure 10

5. For the map reduce I have read and run the examples that you give us in the big data folder in GitHub repository.