Created by: Vinh Q. Dang

Eindhoven University of Technology

December 25, 2013

Homework 3.1

Download the file http://media.mongodb.org/zips.json. The example below uses "curl", but you can download with your web browser if you like. ("Zip codes" are the U.S. version of postal codes; they are 5 digits long.) Then import this file into a collection "zips".

```
$ curl -O http://media.mongodb.org/zips.json

                                 Dload  Upload   Total   Spent    Left
Speed
100 2803k  100 2803k    0     0  1807k      0  0:00:01  0:00:01 --:--
:-- 1848k
$
$ # this is just informational for you:
$ md5 zips.json
MD5 (zips.json) = 4854d69c2ac389f334c0abff03d96259
$
$ # now import it ...
```

After import you will find 29467 documents in your zips collection -- note this is slightly different than the .json file as the zips.json file is a bit noisy in its data and has some duplicate keys (duplicate _id's):

```
$ mongo
> db.zips.count()
29467
```

Question: consider the state with the 4th most zip codes in it. How many zip codes does that state have? Use the aggregation framework to query this.

Solution

1. Import:
    a. Copy zips.json file into the MongoDB directory (i.e: C\mongodb in MS Windows environment)
    b. Open Command Prompt
        i. Start Mongo DB server (i.e: mongod)
    c. Open the second Command Promt:
        i. mongoimport –db pcat –c zips < zips.json
    d. Check:
        i. db.zips.count ()
        ii. Should return 29467
2. Count:

a. Use:

   i.
```
db.zips.aggregate([
{$group:{_id:{state:"$state"},numberOfzipcodes:{$sum:1}}},
{$sort:{numberOfzipcodes:-1}}, {$limit:4}])
```

b. Answer:

   i. 1458

Homework 3.2

Use the same zips collection we imported in homework 3.1. You can confirm you have the right data (well, the count of documents at least) via:

```
> db.zips.count()
29467
```

Now use the aggregation framework to check the data quality. First let's check how many zip documents have a starting zip code digit of each possible character:

```
> db.zips.aggregate(
...    [
...      { $project : { _id : { $substr : ["$_id",0,1] } } } ,
...      { $group : { _id : "$_id", n : {$sum:1} } }
...    ]
... )
{
        "result" : [
                {
                        "_id" : "5",
                        "n" : 3624
                },
                {
                        "_id" : "4",
                        "n" : 3387
                },
                {
                        "_id" : "6",
                        "n" : 3540
                },
          …
```

Now check in a similar manner the "city" field for its first character. You will find there are cities in the collection that start with a number -- which indicates either a noisy data set or zip codes that have no corresponding city. Let's suppose we want to delete these zip code documents -- the ones that have a numeric city name -- from our data set. Do a remove operation to delete those documents.

After removing the documents indicated above, how many documents remain in the zips collection?

Solution

```
t.aggregate([

        { $project : { first : { $substr : ["$city",0,1] } } } ,

        { $group : { _id : "$first", n : {$sum:1}, id:{$addToSet:"$_id"} } },

         {$sort:{_id:1}},{$limit:10} ]).result.forEach( function(z){

                z.id.forEach(function(y){
```

```
            t.remove({_id:y});
        });
})
```

Answer: 23953

Homework 3.3

Download week3.js from the Download Handout link. Run this file in the shell to populate in the database of your choice a few documents in collections called:

```
- policies
- customers
- staffers
```

(Note some of these may be used in future weeks.)

Once loaded, query each collection above and verify they are non-empty. In particular run:

```
  db.policies.find().pretty()
```

to format the output of the one policy loaded nicely.

We have been asked to find all policies that are status!="expired", with liability coverage at or above $100. Note that current:true indicates the rate that is "current" and not historical.

The query to find those policies is which of the following?

- (●) db.policies.find({ status : { $ne : "expired" }, coverages : { $elemMatch : { type : "liability", rates : { $elemMatch : { rate : { $gte : 100 }, current : true } } } } })
- ( ) db.policies.find({ status : { $ne : "expired" }, coverages : { $elemMatch : { type : "liability", rates : { rate : { $gte : 100 }, current : true } } } })
- ( ) db.policies.find({ status : { $ne : "expired" }, coverages : { type : "liability", rates : { $elemMatch : { rate : { $gte : 100 }, current : true } } } })

Homework 3.4

In this homework, we're going to use the built-in Map/Reduce functionality in MongoDB to answer a simple question about zip codes in the U.S. state of Pennsylvania (PA). Zip codes are U.S. postal codes and may identify a city, region of a city, or rural area. The two largest cities in PA are Pittsburgh and Philadelphia. Use Map/Reduce to determine how many zip codes in PA are closer to Pittsburgh than to Philadelphia and how many are closer to Philadelphia than to Pittsburgh.

The map function is already written for you. It uses some lat/longs that are not exactly the center of Pittsburgh and Philadelphia. It also assumes the world is flat. For this problem, these simplifications are OK.

Note: We could solve this problem using MongoDB's geospatial features. However, for this homework we will use Map/Reduce.

The map function for our solution is provided in the aforementioned week3.js file. Thus if you run:

```
mongo --shell week3.js
```

The method map_closest() will already be defined. You can type it at the shell prompt without parameters to see its source code:

```
> map_closest
function map_closest() {
    var pitt = [-80.064879, 40.612044];
    var phil = [-74.978052, 40.089738];

    function distance(a, b) {
        var dx = a[0] - b[0];
        var dy = a[1] - b[1];
        return Math.sqrt(dx * dx + dy * dy);
    }

    if (distance(this.loc, pitt) < distance(this.loc, phil)) {
        emit("pitt", 1);
    } else {
        emit("phil", 1);
    }
}
>
```

Question: Given the map function above, how many zip codes in PA are closer to Philadelphia? Use map/reduce to find the answer.

Answer: 732