Data Storage and Retrieval – Assignment 11

Aaron Niskin

November 18, 2016

1. **Do Problem 1. See the group() finalize example on page 157.**

```
// Returns an object of prefix: count_for_prefix
db.phones.group({
  initial: {prefixes : {}},
  reduce: function(phone, output) {
    var out = output.prefixes,
        pre = phone.components.prefix;
    if (out.hasOwnProperty(pre)){
      out[pre] += 1;
    } else {
      out[pre] = 1;
    }
  },
  finalize: function(out) {
    return out;
  }
})[0].prefixes

// Returns the count of unique prefixes
db.phones.group({
  initial: { prefixes : {} },
  reduce: function(phone, output) {
    output.prefixes[phone.components.prefix] = 1;
  },
  finalize: function(out) {
    out.total = Object.keys(out.prefixes).length
  }
})[0].total
```

2. **Do Problem 2. Create your own database. Use python and the countries.json collection in /usr/share/databases/SevenDatabases/code/mongo/. Create a text index on the common name. Perform a query using the $text operator and show the results.**

   Install a Mongo driver for a language of your choice, and connect to the database. Populate a collection through it, and index one of the fields.

```
### See the code file: /usr/share/databases/aniskin/assignment11/q2.py
### and the accompanying results json file: /usr/share/databases/aniskin/assignment11/yentel.json
import pymongo
from bson.json_util import dumps
import json
from pprint import PrettyPrinter
pp = PrettyPrinter(indent = 2)

client = pymongo.MongoClient("localhost", 27017)
db = client.countries_aaron
```

```
db.countries_of_adawg.drop()
with open("/usr/share/databases/SevenDatabases/code/mongo/countries.json") as json_file:
        json_data = json.load(json_file)

ids = [db.countries_of_adawg.insert(x) for x in json_data]

db.countries_of_adawg.create_index([("name.common", 'text')], default_language='english')

kibbitzer = db.countries_of_adawg.find_one({"$text": { "$search": "Israel" } })
pp.pprint(kibbitzer)

with open('yentel.json', 'w') as kvetcher1:
        json.dump(json.loads(dumps(kibbitzer)), kvetcher1)

print("You're leaving? But I just made a meal. No, no... It's fine. I guess I can eat it all myself
```

3. **For the country collection you created in problem 2,**

   a. **Use group() to compute the area of each region**

```
db.countries_of_adawg.group({
  initial: {},
  reduce: function(country, out) {
    var region = country["region"],
        area = country["area"];
    if(out.hasOwnProperty(region)) {
      out[region] += parseInt(area);
    } else {
      out[region] = parseInt(area);
    }
  },
  finalize: function(out) {
    return out;
  }
})[0]
```

```
{
"Americas" : 42081497,

"Asia" : 32138141,

"Africa" : 30318023,

"Europe" : 23022897,

"Oceania" : 8515313,

"" : 14008208

}
```

   b. **Use MapReduce to find the number of countries that border N other countries for each N.**

```
db.countries_of_adawg.group({
  initial: {},
  reduce: function(country, out) {
    var borders = country.borders.length;
    if(out.hasOwnProperty(borders)) {
      out[borders] += 1;
    } else {
      out[borders] = 1;
    }
  },
  finalize: function(out) {
    return out;
  }
})[0]
```

{ “0” : 82, “1” : 24, “2” : 28, “3” : 30, “4” : 26, “5” : 24, “6” : 13, “7” : 9, “8” : 7, “9” : 2, “10” : 1, “14” : 1, “15” : 1 }