MongoDB Lab

Use the NixOS Database Systems Virtual Machine [../nixos-dbs-vm/] to run the code in this lab.

Demonstration Example

Run MongoDB shell [https://docs.mongodb.com/manual/mongo/] by `mongo` and submit the following MongoDB statements (except for the first code to import a dataset).

See SQL to MongoDB Mapping Chart [https://docs.mongodb.com/manual/reference/sql-comparison/] to understand the MongoDB terminology and concepts.

Import a Dataset

```
curl -L http://media.mongodb.org/zips.json | mongoimport -d test -c
zipcodes
```

Query Data

Find data in a collection [https://docs.mongodb.com/manual/reference/method/db.collection.find/] based on a query with several filters described in a type bracketing.

```
db.zipcodes.find({})
db.zipcodes.find( {
    state: "MA",
    $or: [ { pop: { $lt: 200 } }, { pop: { $gt: 10000 } } ]
} )
```

Aggregate Data

Create an aggregation pipeline [https://docs.mongodb.com/manual/core/aggregation-pipeline/] to aggregate and filter data in several steps.

```
// returns all states with total population greater than 10 million
db.zipcodes.aggregate( [
    { $group: { _id: "$state", totalPop: { $sum: "$pop" } } },
    { $match: { totalPop: { $gte: 10*1000*1000 } } }
] )

// returns the average populations for cities in each state
db.zipcodes.aggregate( [
    { $group: { _id: { state: "$state", city: "$city" }, pop: {
$sum: "$pop" } } },
    { $group: { _id: "$_id.state", avgCityPop: { $avg: "$pop" } } }
] )

// returns the smallest and largest cities by population for each
state
db.zipcodes.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" }
        }
    },
    // the following $project is optional, and modifies the output
format.
    { $project:
        {
             _id: 0,
            state: "$_id",
            biggestCity: { name: "$biggestCity",  pop:
"$biggestPop" },
            smallestCity: { name: "$smallestCity", pop:
```

```
"$smallestPop" }
        }
    }
] )
```

Run Map-Reduce

Create and apply map and reduce functions [https://docs.mongodb.com/manual/reference/method/db.collection.mapReduce/] to process data.

```
// emit the state and the population for each city
var mapFunction1 = function() {
    emit(this.state, this.pop);
};

// reduce the populations array to the sum of its elements
var reduceFunction1 = function(keyCustId, valuesPops) {
    return Array.sum(valuesPops);
};

// calculate the average population per state
var finalizeFunction1 = function (key, reducedVal) {
    reducedVal.avg = reducedVal.qty/reducedVal.count;
    return reducedVal;
};

// apply
db.zipcodes.mapReduce(mapFunction1, reduceFunction1, {
    out: { merge: "states_populations" },
    query: { pop: { $gt: 10000 } },
    finalize: finalizeFunction1
})

// show
db.states_populations.find({})
```