# NBD Assignment 4 – MongoDB

Use database from previous assignment. Use map-reduce and aggregation frameworks to find the following information:

1. Average weight and height of people in the database, grouped by gender (so separate values for men and women)

*printjson(*

*db.people.aggregate(*

*[*

*{*

*$group:*

*{*

*\_id: "$sex",*

*avgHeight: { $avg: "$height" },*

*avgWeight: { $avg: "$weight" }*

*}*

*}*

*]*

*) .toArray())*

*var m = function() {*

*emit(this.sex, { sumH:this.height, sumW:this.weight,count:1});*

*};*

*var r = function(k, val) {*

*reducedVal = { sumH: 0, sumW: 0 , count:0};*

*for (var i = 0; i < val.length; i++) {*

*reducedVal.sumH +=val[i].sumH;*

*reducedVal.sumW+= val[i].sumW;*

*reducedVal.count +=val[i].count; }*

*return reducedVal;*

*};*

*function finalize(key, v) {*

*return {*

*avgHeight: v.sumH/ v.count,*

*avgWeight: v.sumW/ v.count*

*};*

*}*

*var options = {out: "avg\_height\_and\_weight", finalize : finalize };*

*db.people.mapReduce(m,r,options)*

*printjson(db.avg\_height\_and\_weight.find({}).toArray())*

1. Total amount of money left on credit cards of people in database, grouped by currency

*printjson(*

*db.people.aggregate(*

*[ { $unwind: "$credit" },*

*{*

*$group:*

*{*

*\_id: "$credit.currency",*

*total: { $sum: "$credit.balance" }*

*}*

*}, {$sort:{ \_id:1}}*

*]*

*).toArray())*

*var m = function() {*

*for(var i=0;i<this.credit.length;i++)*

*emit(this.credit[i].currency, this.credit[i].balance);*

*};*

*var r = function(k, val) {*

*return Array.sum(val);*

*};*

*var options = {out: "total\_balance" };*

*db.people.mapReduce(m,r,options)*

*printjson(db.total\_balance.find({}).toArray())*

1. List of unique jobs

*printjson(db.people.distinct("job").sort())*

*var m = function() {*

*emit(this.job,1);*

*};*

*var r = function(k, v) {*

*return null;*

*};*

*var options = {*

*out:"job\_list"*

*};*

*db.people.mapReduce(m,r,options)*

*printjson(db.job\_list.find({},{id:1}).toArray())*

1. Average, minimum and maximum BMI (weight/height^2) grouped by nationality

*printjson(*

*db.people.aggregate(*

*[*

*{*

*$addFields: {*

*BMI:{$divide:[ "$weight",{$pow:[ {$divide :["$height",100]},2]}]}*

*}},*

*{*

*$group:*

*{*

*\_id: "$sex",*

*avgBMI: { $avg: "$BMI" },*

*minBMI: { $min: "$BMI" },*

*maxBMI: { $max: "$BMI" }*

*}*

*}, {$sort:{ \_id:1}}*

*]*

*).toArray())*

*var m = function() {*

*emit(this.sex, { H:[this.height], W:[this.weight],count:1});*

*};*

*var r = function(k, val) {*

*reducedVal = { H:[] ,W:[] , count:0};*

*for (var i = 0; i < val.length; i++) {*

*reducedVal.H=val[i].H.concat(reducedVal.H);*

*reducedVal.W=val[i].W.concat(reducedVal.W);*

*reducedVal.count +=val[i].count; }*

*return reducedVal;*

*};*

*function finalize(key, v) {*

*var sumBMI=0,minBMI = v.W[0] /v.H[0] /v.H[0]\*10000 ,maxBMI= minBMI;*

*for (var i = 0; i < v.W.length; i++){*

*BMI=v.W[i] /v.H[i] /v.H[i]\*10000;*

*minBMI=BMI< minBMI?BMI: minBMI*

*maxBMI =BMI> maxBMI?BMI: maxBMI*

*sumBMI+=BMI*

*}*

*return {avgBMI:sumBMI/v.count, minBMI: minBMI , maxBMI : maxBMI };*

*};*

*db.people.mapReduce(m,r, { out:"BMI", finalize : finalize })*

*printjson(db.BMI.find().toArray())*

1. Average and total amount of money left on credit cards of polish women, grouped by currency

*printjson(*

*db.people.aggregate(*

*[ { $unwind: "$credit" },*

*{ $match : { sex : "Female" } },*

*{*

*$group:*

*{*

*\_id: "$credit.currency",*

*total: { $sum: "$credit.balance" },*

*avgBalance: { $avg: "$credit.balance" }*

*}*

*}, {$sort:{ \_id:1}}*

*]*

*).toArray())*

*var m = function() {*

*if(this.sex!="Female")*

*return*

*for(var i=0;i<this.credit.length;i++)*

*emit(this.credit[i].currency, {bal:this.credit[i].balance,count:1});*

*};*

*var r = function(k, val) {*

*var reducedVal={bal:0,count:0}*

*val.forEach(val=>{*

*reducedVal.count+=val.count;*

*reducedVal.bal+=val.bal;})*

*return reducedVal;*

*};*

*function finalize(key, v) {*

*return {totalBalance: v.bal, avgBalance: v.bal /v.count};*

*}*

*var options = {out: "avg\_and\_total\_balance", finalize : finalize };*

*db.people.mapReduce(m,r,options)*

*printjson(db.avg\_and\_total\_balance.find({}).toArray())*

In order to use the aggregation framework in older MongoDB versions you will need to convert some queried fields (e.g. weight and height) into numbers, examples here:

<https://stackoverflow.com/questions/29487351/how-to-convert-string-to-numerical-values-in-mongodb>

In newer versions (4.x) you can use appropriate aggregation framework operators to convert data on the fly or also make a conversion.

*db.people.find().forEach(function(person) {*

*db.people.update({*

*"\_id": person.\_id,*

*},{*

*"$set": {*

*"weight": parseFloat(person.weight),*

*"height": parseFloat(person.height),*

*}*

*});*

*})*

*db.people.find().forEach(function(person) {*

*person.credit.forEach( function(c) {*

*c.balance= parseFloat(c. balance)*

*db.people.save(person)*

*})*

*})*

Send solutions in the following format: for each task send 4 files – files with the aggregate and map-reduce query and files with results. Name files using the following pattern: query\_ag\_X.js, query\_mr\_X.js, result\_ag\_X.json, result\_mr\_X.json where X is the task #.

In order to save query results to file wrap the query with printjson function and add .toArray() after find if necessary – e.g. printjson(db.products.find().toArray()).

Put such query into a file, run mongo shell the following way:

mongo dbname queryfile >> resultfile

e.g.

mongo nbd query\_ag\_1.js >> result\_ag\_1.json

mongo nbd query\_mr\_1.js >> result\_mr\_1.json