MongoDB and MapReduce

import csv file

```
mongoimport --db users --type csv --collection users --headerline --file
/opt/backups/contacts.csv
```

with headers

```
mongoimport --db moviedb --type csv --collection users --fields
"userid,gender,age,movie" --file /opt/backups/contacts.csv

mongoimport --db nyseTest --type csv --collection stocks --headerline --file
NYSE_daily_prices_B.csv
```

// MapReduce with mongoDB Tutorial

import movie csv file into DB

```
mongoimport --db moviedb --type csv --collection users --fields
"userid,gender,age,occupation,zipcode"  --file user_dataset.csv

mongo

use moviedb

db.users.findOne() -- to show one entry from collection
```

1: map and reduce function to count gender

// map function

```
var map = function(){emit(this.gender,1)};

function (){
  emit(this.gender,1);
}
```

// reduce function

```
var reduce = function(key,values){}

function (key,values){
  var sum = 0;
  values.forEach(function(val){
  sum += val;
  });

  return sum;
}

// manual-- https://docs.mongodb.com/manual/core/map-reduce/index.html
/*

Ways to run map reduce

1----

db.runCommand(
               {
                 mapReduce: <collection>,
                 map: <function>,
                 reduce: <function>,
                 finalize: <function>,
                 out: <output>,
                 query: <document>,
                 sort: <document>,
                 limit: <number>,
                 scope: <document>,
                 jsMode: <boolean>,
                 verbose: <boolean>,
                 bypassDocumentValidation: <boolean>,
                 collation: <document>,
                 writeConcern: <document>
               }
             )


2----

db.collection.mapReduce(
                         <map>,
                         <reduce>,
                         {
                           out: <collection>,
```

```
                            query: <document>,
                            sort: <document>,
                            limit: <number>,
                            finalize: <function>,
                            scope: <document>,
                            jsMode: <boolean>,
                            verbose: <boolean>,
                            bypassDocumentValidation: <boolean>
                          }
                        )

*/

Mapreduce
Output in a collection--

db.users.mapReduce(map,reduce,{out:"MRGenderCount"})



/*

{
        "result" : "MRGenderCount",
        "timeMillis" : 95,
        "counts" : {
                "input" : 6040,
                "emit" : 6040,
                "reduce" : 122,
                "output" : 2
        },
        "ok" : 1
}

*/

db.MRGenderCount.find().pretty()
{ "_id" : "F", "value" : 1709 }
{ "_id" : "M", "value" : 4331 }

Output inline --

db.users.mapReduce(map,reduce,{out:{inline:1}})

// output
```

```
{
        "results" : [
                {
                        "_id" : "F",
                        "value" : 1709
                },
                {
                        "_id" : "M",
                        "value" : 4331
                }
        ],
        "timeMillis" : 52,
        "counts" : {
                "input" : 6040,
                "emit" : 6040,
                "reduce" : 122,
                "output" : 2
        },
        "ok" : 1
}
```

2: Changing map and reduce function to give label to count instead of values

MAP Function

```
function (){
  emit(this.gender,{"gender_count":1});
}
```

REDUCE Function

```
function (key,values){
  var sum = 0;
  values.forEach(function(val){
  sum += val;
  });

  return {"gender_count":sum};
}
```

When we run mapper--

```
db.users.mapReduce(map,reduce,{out:"MRGenderCountDesc"})
{
        "result" : "MRGenderCountDesc",
        "timeMillis" : 129,
        "counts" : {
                "input" : 6040,
                "emit" : 6040,
                "reduce" : 122,
                "output" : 2
        },
        "ok" : 1
}
```

Output--

```
db.MRGenderCountDesc.find().pretty()
{
        "_id" : "F",
        "value" : {
                "gender_count" : "0[object BSON][object BSON][object BSON][object
BSON][object BSON][object BSON][object BSON][object BSON][object BSON][object
BSON][object BSON][object BSON][object BSON][object BSON][object BSON][object
BSON][object BSON][object BSON]"
        }
}
{
        "_id" : "M",
        "value" : {
                "gender_count" : "0[object BSON][object BSON][object BSON][object
BSON][object BSON][object BSON][object BSON][object BSON][object BSON][object
BSON][object BSON][object BSON][object BSON][object BSON][object BSON][object
BSON][object BSON][object BSON][object BSON][object BSON][object BSON][object
BSON][object BSON][object BSON]"

        }
}
```

It is showing JSON because we added values in reduce function. but now the values are json document- {"gender_count":1}

So the correct REDUCE function will be--

```
function (key,values){
  var sum = 0;
  values.forEach(function(val){
```

```
  sum += val.gender_count;
  });

  return {"gender_count":sum};
}
```

val is a JSON document. We neet its value.

Now the output will be labeled and correct as below-

```
> db.users.mapReduce(map,reduce,{out:"MRGenderCountDescCorrect"})
{
        "result" : "MRGenderCountDescCorrect",
        "timeMillis" : 125,
        "counts" : {
                "input" : 6040,
                "emit" : 6040,
                "reduce" : 122,
                "output" : 2
        },
        "ok" : 1
}
> db.MRGenderCountDescCorrect.find().pretty()
{ "_id" : "F", "value" : { "gender_count" : 1709 } }
{ "_id" : "M", "value" : { "gender_count" : 4331 } }
```

3- Find Oldest person for a gender

MAP

```
function (){
  emit(this.gender, this.age);
}
```

REDUCE

```
function (key,values){
  var maxage = 0;
  for(var i = 0; i<values.length;i++){
    if(values[i]>maxage){
      maxage=values[i];
```

```
    }
  }
return maxage;
}

Output
> db.users.mapReduce(map2,reduce2,{out:"MaxAge"})
{
        "result" : "MaxAge",
        "timeMillis" : 169,
        "counts" : {
                "input" : 6040,
                "emit" : 6040,
                "reduce" : 122,
                "output" : 2
        },
        "ok" : 1
}

> db.MaxAge.find().pretty()
{ "_id" : "F", "value" : 56 }
{ "_id" : "M", "value" : 56 }
```

4- Label the result for problem to find Oldest person for a gender

MAP

```
function (){
  emit({gender:this.gender}, {maxage:this.age});
}
```

REDUCE

```
function (key,values){
  var maxage = 0;
  for(var i = 0; i<values.length;i++){
    if(values[i].maxage>maxage){
      maxage=values[i].maxage;
    }
  }
```

```
return {maxage:maxage};
}
```

Output

```
> db.users.mapReduce(map2,reduce2,{out:"MaxAge"})
{
        "result" : "MaxAge",
        "timeMillis" : 217,
        "counts" : {
                "input" : 6040,
                "emit" : 6040,
                "reduce" : 122,
                "output" : 2
        },
        "ok" : 1
}
> db.MaxAge.find().pretty()
{ "_id" : { "gender" : "F" }, "value" : { "maxage" : 56 } }
{ "_id" : { "gender" : "M" }, "value" : { "maxage" : 56 } }
```

REDUCE2

```
function (key,values){
  var maxage = 0;
  for(var i = 0; i<values.length;i++){
    if(values[i].maxage>maxage){
      maxage=values[i].maxage;
    }
  }
return {gender:key,maxage:maxage};
}
```

Output

```
> db.users.mapReduce(map2,reduce2,{out:"MaxAge"})
{
        "result" : "MaxAge",
        "timeMillis" : 237,
        "counts" : {
                "input" : 6040,
                "emit" : 6040,
                "reduce" : 122,
```

```
                "output" : 2
        },
        "ok" : 1
}
> db.MaxAge.find().pretty()
{
        "_id" : {
                "gender" : "F"
        },
        "value" : {
                "gender" : {
                        "gender" : "F"
                },
                "maxage" : 56
        }
}
{
        "_id" : {
                "gender" : "M"
        },
        "value" : {
                "gender" : {
                        "gender" : "M"
                },
                "maxage" : 56
        }
}


> db.MaxAge.find()
{ "_id" : { "gender" : "F" }, "value" : { "gender" : { "gender" : "F" }, "maxage"
: 56 } }
{ "_id" : { "gender" : "M" }, "value" : { "gender" : { "gender" : "M" }, "maxage"
: 56 } }
```

REDUCE3

```
function (key,values){
  var maxage = 0;
  for(var i = 0; i<values.length;i++){
    if(values[i].maxage>maxage){
      maxage=values[i].maxage;
    }
```

```
  }
return {gender:key.gender,maxage:maxage};
}


output

> db.users.mapReduce(map2,reduce2,{out:"MaxAge"})
{
        "result" : "MaxAge",
        "timeMillis" : 237,
        "counts" : {
                "input" : 6040,
                "emit" : 6040,
                "reduce" : 122,
                "output" : 2
        },
        "ok" : 1
}
> db.MaxAge.find()
{ "_id" : { "gender" : "F" }, "value" : { "gender" : "F", "maxage" : 56 } }
{ "_id" : { "gender" : "M" }, "value" : { "gender" : "M", "maxage" : 56 } }


> db.users.mapReduce(map2,reduce2,{out:{inline:1}})
{
        "results" : [
                {
                        "_id" : {
                                "gender" : "F"
                        },
                        "value" : {
                                "gender" : "F",
                                "maxage" : 56
                        }
                },
                {
                        "_id" : {
                                "gender" : "M"
                        },
                        "value" : {
                                "gender" : "M",
                                "maxage" : 56
                        }
                }
```

```
        ],
        "timeMillis" : 151,
        "counts" : {
                "input" : 6040,
                "emit" : 6040,
                "reduce" : 122,
                "output" : 2
        },
        "ok" : 1
}
>
```