**Engineering Of Big Data Labs**

**Lab 1- 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

}

**Lab 2- MongoDB and MapReduce Contd.**

var map =function(){

      var value = this.price + ":" + this.date + ":" + this.volume;

      var key = this.stock\_symbol;

      emit(key,value);

    }

Or we can do sumthing better

var map =function(){

      var value = {price:this.price, date:this.date, volume:this.volume};

      var key = this.stock\_symbol;

      emit(key,value);

    }

Q- Find minimum, maximum , average, median and standard deviation of stock\_price\_close

MAP function

function (){

  var key = this.stock\_symbol;

  var value = {stock\_symbol:this.stock\_symbol,price:this.stock\_price\_close,avg:0.0,sum:this.stock\_price\_close,count:1};

  emit(key,value);

}

REDUCE function

function (key,values){

  var reducedObj = {stock\_symbol:key,price:0.0,avg:0.0,sum:0.0,count:0};

  values.forEach(function(val){

  if(!isNaN(val.price)){

    reducedObj.sum += val.sum;

    reducedObj.count += val.count;

    }

  });

  return reducedObj;

}

FINALIZER function

function (key, reducedValue){

  if(reducedValue.count>0){

    reducedValue.avg= reducedValue.sum/reducedValue.count;

  }

  return reducedValue;

}

> db.stocks.mapReduce(map,reduce,{finalize:finalizer,out:"MROutPriceCloseAvg"})

{

"result" : "MROutPriceCloseAvg",

"timeMillis" : 3299,

"counts" : {

"input" : 2586,

"emit" : 2586,

"reduce" : 26,

"output" : 1

},

"ok" : 1

}

> db.MROutPriceCloseAvg.find()

{ "\_id" : "AEA", "value" : { "stock\_symbol" : "AEA", "price" : 0, "avg" : 10.565622583139994, "sum" : 27322.700000000026, "count" : 2586 } }

PROF FUNCTIONS

MAP Function

function (){

  var key = this.stock\_symbol;

  var value = {stock\_symbol:this.stock\_symbol,price:this.stock\_price\_close,avg:0.0,sum:0.0,count:1};

  emit(key,value);

}

REDUCE Function

function (key,values){

  var reducedObj = {stock\_symbol:key,price:0.0,avg:0.0,sum:0.0,count:0};

  values.forEach(function(val){

    reducedObj.sum += val.price;

    reducedObj.price += val.price;

    reducedObj.count += val.count;

  });

  return reducedObj;

}

FINALIZER function

> db.stocks.mapReduce(map,reduce,{finalize:finalizer,out:"MROutPriceCloseAvg"})

{

"result" : "MROutPriceCloseAvg",

"timeMillis" : 3299,

"counts" : {

"input" : 2586,

"emit" : 2586,

"reduce" : 26,

"output" : 1

},

"ok" : 1

}

> db.MROutPriceCloseAvg.find()

{ "\_id" : "AEA", "value" : { "stock\_symbol" : "AEA", "price" : 0, "avg" : 10.565622583139994, "sum" : 27322.700000000026, "count" : 2586 } }