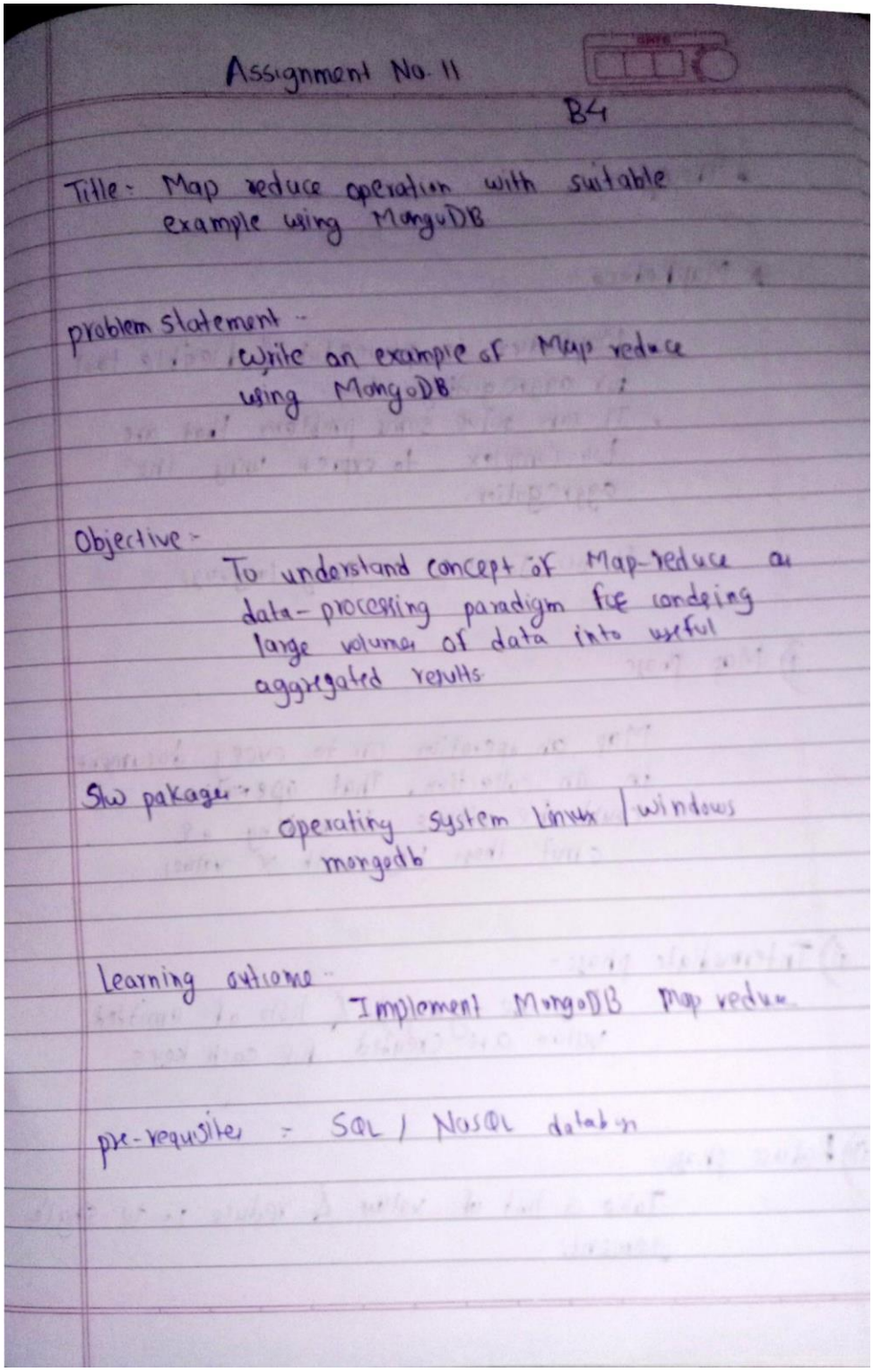
Mongo DB Assignment B4

Writeups, Code, Screenshots



Assignment No. 11

B4

Title - Map reduce operation with suitable example using MongoDB

problem statement -
write an example of Map reduce using MongoDB

Objective -
To understand concept of Map-reduce as data-processing paradigm for condeing large volumes of data into useful aggregated results

Sw pakages -
Operating system Linux / windows
mongodb

Learning outcome -
Implement MongoDB map reduce

pre-requisites - SQL / NoSQL database

• Theory :-

★ MapReduce :-

- MapReduce is poweeful & flexible tool for aggregating data.
- It can solve some problems that are too complex to express using the aggregation.
- It uses Javascript as query language

i) Map phase :-

Map an operation on to every document in an collection. That operation could be either do nothing or emit their Keys with X values.

ii) Intermediate phase :-

Keys are grouped & lists of emitted values are created for each keys.

iii) Reduce phase :-

Take a list of values & reduce it to element.

~~Ass10~~

Syntax :-

> db.collection.mapReduce (

function() { emit (key, value) } , // map function
function() (Keys, values) { return
reduceFunction } ,
// reduce function
{ out : collection ,
query : document ,
sort : document ,
limit : number
}
}

Example :-

db.orders.mapReduce (

map → function () { emit (this.custid, this.amount);
} ,

reduce → function(key, value) { return Array.sum
(values) } ,

{
~~collection~~
query → query : { status : "A" }
output → out : "order_totals"
}
)

{
cust_id = "A123",
amount = 500,
status = "A",
}

{
cust_id = "A123",
amount = 250,
status = "A",
}

{
cust_id = "B212",
amount = 200,
status = "A",
}

{
cust_id = "A123",
amount = 300,
status = "D",
}

query

{
cust_id = A123
amount = 500
status = A
}

{
cust_id = A123
amount = 250
status = A
}

{
cust_id = A123
amount = 300
status = A
}

reduce

{A123 [500, 250, 300]}

{B212 : 200}

Conclusion :-

Thus, we implemented MapReduce in MongoDB.

CODE :-

1.) Create collection Order

```
> db.createCollection("order")
{ "ok" : 1 }
> show tables
books
order
```

2) Insert Five documents in Order Schema having Fields (CustID, Amount, Status(pending, delivered, failed))

```
> db.order.insert({Custid :"A",amount :400, status:"pending"})
WriteResult({ "nInserted" : 1 })
> db.order.insert({Custid :"B",amount :300, status:"deleivered"})
WriteResult({ "nInserted" : 1 })
> db.order.insert({Custid :"A",amount :200, status:"failed"})
WriteResult({ "nInserted" : 1 })
> db.order.insert({Custid :"C",amount :200, status:"failed"})
WriteResult({ "nInserted" : 1 })
> db.order.insert({Custid :"B",amount :700, status:"pending"})
WriteResult({ "nInserted" : 1 })
> db.order.insert({Custid :"B",amount :800, status:"pending"})
WriteResult({ "nInserted" : 1 })
> db.order.find().pretty()
{
        "_id" : ObjectId("5fa8d74cf5005493af5be1d2"),
        "Custid" : "A",
        "amount" : 400,
        "status" : "pending"
}
{
        "_id" : ObjectId("5fa8d764f5005493af5be1d3"),
```

```
    "Custid" : "B",
    "amount" : 300,
    "status" : "deleivered"
}
{
    "_id" : ObjectId("5fa8d7c2f5005493af5be1d4"),
    "Custid" : "A",
    "amount" : 200,
    "status" : "failed"
}
{
    "_id" : ObjectId("5fa8d80cf5005493af5be1d5"),
    "Custid" : "C",
    "amount" : 200,
    "status" : "failed"
}
{
    "_id" : ObjectId("5fa8d827f5005493af5be1d6"),
    "Custid" : "B",
    "amount" : 700,
    "status" : "pending"
}
{
    "_id" : ObjectId("5fa8d843f5005493af5be1d7"),
    "Custid" : "B",
    "amount" : 800,
    "status" : "pending"
}
```

3) Display Total Order amount of Customers Whose status in pending.

```
> var mapfunction=function() {if(this.status=='pending') emit(this.Custid,this.amount)};
> var reducefunction=function(key,values){return Array.sum(values);};
> db.order.mapReduce(mapfunction,reducefunction,{'out':'Order_total'})
{ "result" : "Order_total", "ok" : 1 }
> db.Order_total.find()
{ "_id" : "A", "value" : 400 }
{ "_id" : "B", "value" : 1500 }
```

4) Display averages of all customers.

```
> var mapfunction=function() {if(this.Custid=='A') emit(this.Custid,this.amount)};
> var reducefunction=function(key,values){return Array.avg(values);};
> db.order.mapReduce(mapfunction,reducefunction,{'out':'Order_avg_A'})
{ "result" : "Order_avg_A", "ok" : 1 }
> db.Order_avg_A.find()
{ "_id" : "A", "value" : 300 }

> var mapfunction=function() {if(this.Custid=='C') emit(this.Custid,this.amount)};
> var reducefunction=function(key,values){return Array.avg(values);};
> db.order.mapReduce(mapfunction,reducefunction,{'out':'Order_avg_C'})
{ "result" : "Order_avg_C", "ok" : 1 }
> db.Order_avg_C.find()
{ "_id" : "C", "value" : 200 }


> var mapfunction=function() {if(this.Custid=='B') emit(this.Custid,this.amount)};
> db.order.mapReduce(mapfunction,reducefunction,{'out':'Order_avg_B'})
{ "result" : "Order_avg_B", "ok" : 1 }
> db.Order_avg_B.find()
{ "_id" : "B", "value" : 600 }
```

```
> var mapfunction=function() { emit(this.Custid,this.amount)};
> db.order.mapReduce(mapfunction,reducefunction,{'out':'Order_avg'})
{ "result" : "Order_avg", "ok" : 1 }
> db.Order_avg.find()
{ "_id" : "C", "value" : 200 }
{ "_id" : "A", "value" : 300 }
{ "_id" : "B", "value" : 600 }
```

5) Find the minimum amount of Customers with all IDS

```
> var mapfunction=function() { emit(this.Custid,this.amount)};
> var reducefunction=function(key,values){return Math.min.apply(Math,values);};
> db.order.mapReduce(mapfunction,reducefunction,{'out':'Order_min'})
{ "result" : "Order_min", "ok" : 1 }
> db.Order_min.find()
{ "_id" : "A", "value" : 200 }
{ "_id" : "B", "value" : 300 }
{ "_id" : "C", "value" : 200 }
```

6) Find Maximum Amount of customer whose status in pending

```
> var mapfunction=function() { if(this.status='pending')emit(this.Custid,this.amount)};
> var reducefunction=function(key,values){return Math.max.apply(Math,values);};
> db.order.mapReduce(mapfunction,reducefunction,{'out':'Order_Max_P'})
{ "result" : "Order_Max_P", "ok" : 1 }
> db.Order_Max_P.find()
{ "_id" : "C", "value" : 200 }
{ "_id" : "A", "value" : 400 }
{ "_id" : "B", "value" : 800 }
>
```

```
Administrator: Command Prompt - "C:\Program Files\MongoDB\Server\4.4\bin\mongo.exe"
> db.createcollection("order")
uncaught exception: TypeError: db.createcollection is not a function :
@(shell):1:1
> db.createCollection("order")
{ "ok" : 1 }
> show tables
books
order
> db.order.insert({Custid :"A",amount :400, status:"pending"})
WriteResult({ "nInserted" : 1 })
> db.order.insert({Custid :"B",amount :300, status:"deleivered"})
WriteResult({ "nInserted" : 1 })
> db.order.insert({Custid :"A",amount :200, status:"failed"})
WriteResult({ "nInserted" : 1 })
> db.order.insert({Custid :"C",amount :200, status:"failed"})
WriteResult({ "nInserted" : 1 })
> db.order.insert({Custid :"B",amount :700, status:"pending"})
WriteResult({ "nInserted" : 1 })
> db.order.insert({Custid :"B",amount :800, status:"pending"})
WriteResult({ "nInserted" : 1 })
> db.order.find().pretty()
{
        "_id" : ObjectId("5fa8d74cf5005493af5be1d2"),
        "Custid" : "A",
        "amount" : 400,
        "status" : "pending"
}
{
        "_id" : ObjectId("5fa8d764f5005493af5be1d3"),
        "Custid" : "B",
        "amount" : 300,
        "status" : "deleivered"
}
{
        "_id" : ObjectId("5fa8d7c2f5005493af5be1d4"),
        "Custid" : "A",
        "amount" : 200,
        "status" : "failed"
}
{
        "_id" : ObjectId("5fa8d80cf5005493af5be1d5"),
        "Custid" : "C",
        "amount" : 200,
        "status" : "failed"
}
{
        "_id" : ObjectId("5fa8d827f5005493af5be1d6"),
        "Custid" : "B",
        "amount" : 700,
        "status" : "pending"
```

```
> var mapfunction=function() {if(this.status=='pending') emit(this.Custid,this.amount)};
> var reducefunction=function(key,values){return Array.sum(values);};
> db.order.mapReduce(mapfunction,reducefunction,{'out':'Order_total'})
{ "result" : "Order_total", "ok" : 1 }
> db.Order_total.find()
{ "_id" : "A", "value" : 400 }
{ "_id" : "B", "value" : 1500 }
> var mapfunction=function() {if(this.Custid=='A') emit(this.Custid,this.amount)};
> var reducefunction=function(key,values){return Array.avg(values);};
> db.order.mapReduce(mapfunction,reducefunction,{'out':'Order_avg_A'})
{ "result" : "Order_avg_A", "ok" : 1 }
> db.Order_avg_A.find()
{ "_id" : "A", "value" : 300 }
> var mapfunction=function() {if(this.Custid=='C') emit(this.Custid,this.amount)};
> var reducefunction=function(key,values){return Array.avg(values);};
> db.order.mapReduce(mapfunction,reducefunction,{'out':'Order_avg_C'})
{ "result" : "Order_avg_C", "ok" : 1 }
> db.Order_avg_C.find()
{ "_id" : "C", "value" : 200 }
> var mapfunction=function() {if(this.Custid=='B') emit(this.Custid,this.amount)};
> db.order.mapReduce(mapfunction,reducefunction,{'out':'Order_avg_B'})
{ "result" : "Order_avg_B", "ok" : 1 }
> db.Order_avg_B.find()
{ "_id" : "B", "value" : 600 }
> var mapfunction=function() { emit(this.Custid,this.amount)};
> db.order.mapReduce(mapfunction,reducefunction,{'out':'Order_avg'})
{ "result" : "Order_avg", "ok" : 1 }
> db.Order_avg.find()
{ "_id" : "C", "value" : 200 }
{ "_id" : "A", "value" : 300 }
{ "_id" : "B", "value" : 600 }
> var mapfunction=function() { emit(this.Custid,this.amount)};
> var reducefunction=function(key,values){return Math.min.apply(Math,values);};
> db.order.mapReduce(mapfunction,reducefunction,{'out':'Order_min'})
{ "result" : "Order_min", "ok" : 1 }
> db.Order_min.find()
{ "_id" : "A", "value" : 200 }
{ "_id" : "B", "value" : 300 }
{ "_id" : "C", "value" : 200 }
> var mapfunction=function() { if(this.status='pending')emit(this.Custid,this.amount)};
> var reducefunction=function(key,values){return Math.max.apply(Math,values);};
> db.order.mapReduce(mapfunction,reducefunction,{'out':'Order_Max_P'})
{ "result" : "Order_Max_P", "ok" : 1 }
> db.Order_Max_P.find()
{ "_id" : "C", "value" : 200 }
{ "_id" : "A", "value" : 400 }
{ "_id" : "B", "value" : 800 }
>
```