1BM17CS086 - SAIFUR RAHMAN

BDA LAB 4

Task 1:

ASSIGNMENTS FOR HANDS-ON PRACTICE

ASSIGNMENT 1

Objective: **To practice MapReduce programming in MongoDB.**

Step 1: Execute the below statements at the MongoDB shell prompt.

db.books.save({ _id:1,Category:"Machine Learning", Bookname:'Machine Learning for Hackers", Author:"Drew Conway",qty:25,price:400,rol:30,pages:350});
db.books.save({ _id:2,Category:"Business Intelligence", Bookname:"Fundamentals of Business Analytics", Author:"Seema Acharya",qty:55,price:500,rol:30,pages:250});
db.books.save({ _id:3,Category:"Analytics", Bookname:" Competing on Analytics", Author:"Thomas Davenport",qty:8,price:150,rol:20,pages:150});
db.books.save({ _id:4,Category:"Visualization", Bookname:"Visualizing Data", Author:"Ben Fry",qty:12, price:325,rol:6,pages:450});

170 •

db.books.save({_id:5,Category:"Web Mining", Bookname:" Learning R ", Author:" Richard C price:850,rol:10,pages:120});

Step 2: Confirm the presence of the above documents in the "books" collection.

Step 3: Write map and reduce functions to split the books into the following two cate
(a) Big books
(b) Small books
Books which have more than 300 pages should be in the big book category. have less than 300 pages should be in the small book category.

Step 4: Count the number of books in each category.

Step 5: Store the output as follows as documents in a new collection, called, "Book_R

Book Category	Count of the Books
(a) Big books	2
(b) Small books	3

```
db.books.save({_id:1,Category:"Machine Learning",BookName:"Machine Learning for Hackers",Author:"Drew Conway",qty:25,price:400,
db.books.save({_id:2,Category:"Business Intelligence",BookName:"Fundamentals of Business Analytics",Author:"Seema Acharya",qty:
db.books.save({_id:3,Category:"Analytics",BookName:"Competing on Analytics",Author:"Thomas",qty:8,price:150,rol:20,pages:150});
db.books.save({_id:4,Category:"Visualisation",BookName:"Visualising Data",Author:"Ben Fry",qty:12,price:325,rol:6,pages:450});
db.books.save({_id:5,Category:"Web Mining",BookName:"Learning R",Author:"Richard C",qty:10,price:850,rol:10,pages:120});

db.books.find();

db.books.mapReduce(
function(){
this.pages > 300 ? emit("Big books", 1) : emit("Small books", 1)},
function(key, values){
```

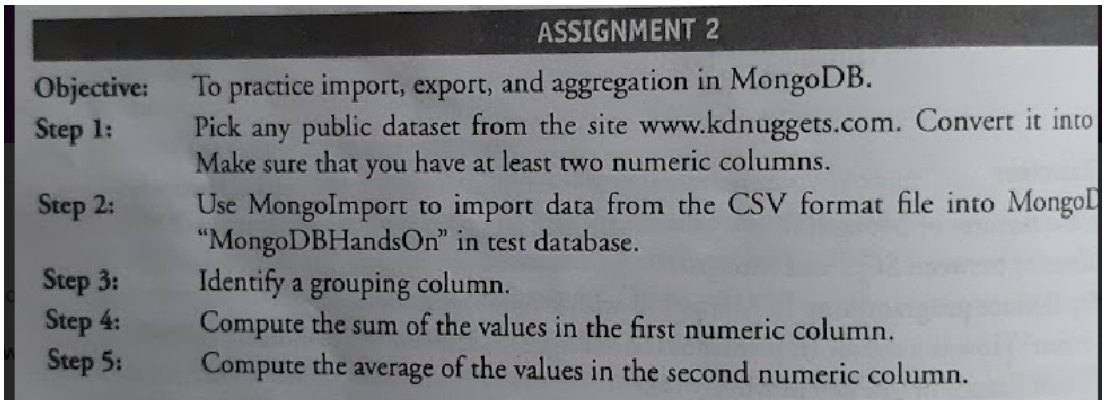
```
return values.length
},
{ out: "book_categories"});

db.book_categories.find();
```

```
> db.books.save({_id:1,Category:"Machine Learning",BookName:"Machine Learning for Hackers",Author:"Drew Conway",qty:25,price:400,rol:30,pages:350});
WriteResult({ "nMatched" : 0, "nUpserted" : 1, "nModified" : 0, "_id" : 1 })
> db.books.save({_id:2,Category:"Business Intelligence",BookName:"Fundamentals of Business Analytics",Author:"Seema Acharya",qty:55,price:500,rol:30,pages:250});
WriteResult({ "nMatched" : 0, "nUpserted" : 1, "nModified" : 0, "_id" : 2 })
> db.books.save({_id:3,Category:"Analytics",BookName:"Competing on Analytics",Author:"Thomas",qty:8,price:150,rol:20,pages:150});
WriteResult({ "nMatched" : 0, "nUpserted" : 1, "nModified" : 0, "_id" : 3 })
> db.books.save({_id:4,Category:"Visualisation",BookName:"Visualising Data",Author:"Ben Fry",qty:12,price:325,rol:6,pages:450});
WriteResult({ "nMatched" : 0, "nUpserted" : 1, "nModified" : 0, "_id" : 4 })
> db.books.save({_id:5,Category:"Web Mining",BookName:"Learning R",Author:"Richard C",qty:10,price:850,rol:10,pages:120});
WriteResult({ "nMatched" : 0, "nUpserted" : 1, "nModified" : 0, "_id" : 5 })
> db.books.find();
{ "_id" : 1, "Category" : "Machine Learning", "BookName" : "Machine Learning for Hackers", "Author" : "Drew Conway", "qty" : 25, "price" : 400, "rol" : 30, "pages" : 350 }
{ "_id" : 2, "Category" : "Business Intelligence", "BookName" : "Fundamentals of Business Analytics", "Author" : "Seema Acharya", "qty" : 55, "price" : 500, "rol" : 30, "pages" : 250 }
{ "_id" : 3, "Category" : "Analytics", "BookName" : "Competing on Analytics", "Author" : "Thomas", "qty" : 8, "price" : 150, "rol" : 20, "pages" : 150 }
{ "_id" : 4, "Category" : "Visualisation", "BookName" : "Visualising Data", "Author" : "Ben Fry", "qty" : 12, "price" : 325, "rol" : 6, "pages" : 450 }
{ "_id" : 5, "Category" : "Web Mining", "BookName" : "Learning R", "Author" : "Richard C", "qty" : 10, "price" : 850, "rol" : 10, "pages" : 120 }
> db.books.mapReduce(
... function(){
... this.pages > 300 ? emit("Big books", 1) : emit("Small books", 1)},
... function(key, values){
... return values.length
... },
... { out: "book_categories"});

{
        "result" : "book_categories",
        "timeMillis" : 834,
        "counts" : {
                "input" : 5,
                "emit" : 5,
                "reduce" : 2,
                "output" : 2
        },
        "ok" : 1
}
>
> db.book_categories.find();
{ "_id" : "Big books", "value" : 2 }
{ "_id" : "Small books", "value" : 3 }
>
```

Task 2:



ASSIGNMENT 2

Objective: To practice import, export, and aggregation in MongoDB.

Step 1: Pick any public dataset from the site www.kdnuggets.com. Convert it into
Make sure that you have at at least two numeric columns.

Step 2: Use MongoImport to import data from the CSV format file into MongoD
"MongoDBHandsOn" in test database.

Step 3: Identify a grouping column.

Step 4: Compute the sum of the values in the first numeric column.

Step 5: Compute the average of the values in the second numeric column.

```
mongoimport -d "lab4" -c "MongoDBHandsOn" --type csv --headerline --file "bank-data.csv"
```

```
db.MongoDBHandsOn.aggregate([
    { $group : { _id: null, sum: {$sum:"$children"} } }
])

db.MongoDBHandsOn.aggregate([
    { $group : { _id: "Average Age", avg: {$avg:"$age"} } }
])
```

```
saif@badger:~/college/bda$ mongoimport -d "lab4" -c "MongoDBHandsOn" --type csv --headerline --file "bank
-data.csv"
2020-10-16T04:39:17.349+0530    connected to: mongodb://localhost/
2020-10-16T04:39:17.804+0530    600 document(s) imported successfully. 0 document(s) failed to import.
```

```
> db.MongoDBHandsOn.aggregate([
...     { $group : { _id: null, sum: {$sum:"$children"} } }
... ])
{ "_id" : null, "sum" : 607 }
>
> db.MongoDBHandsOn.aggregate([
...     { $group : { _id: "Average Age", avg: {$avg:"$age"} } }
... ])
{ "_id" : "Average Age", "avg" : 42.395 }
```

Task 3:

1. Create an array called Cities in a collection called Country
2. Demonstrate the usage of following
3. Print first document
4. Print 3rd and 4th document
5. Push operation
6. Pop operation
7. Pull operation
8. Update using set and addtoset

```
db.createCollection("Country")

db.Country.insert({name:"India", cities:["Mumbai","New Delhi"]});
db.Country.insert({name:"Pakistan", cities:["Lahore","Karachi","Multan"]});
db.Country.insert({name:"Armenia", cities:["Gyumri","Vagharshapat"]});
db.Country.insert({name:"Kuwait", cities:["Al Jahra","Salmiya"]});

db.Country.find();

db.Country.find().limit(1);
db.Country.find().skip(2).limit(2);

db.Country.update( { name: "India" }, { $push: { cities: "Imphal" } });
db.Country.find({ name: "India" });

db.Country.update( { name: "Kuwait" }, { $pop: { cities: 1 } } );
db.Country.find({ name: "Kuwait" });

db.Country.update( { name: "Pakistan" }, { $pull: { cities:'Karachi' } } );
db.Country.find({ name: "Pakistan" });

db.Country.update({ name: "India" },{$set:{'cities.1':'Pune'}});

db.Country.update({ name: "India" },{$addToSet:{cities:"Mysore"}});
db.Country.find({ name: "India" });
```

```
> db.createCollection("Country")
{ "ok" : 1 }
> db.Country.insert({name:"India", cities:["Mumbai","New Delhi"]});
WriteResult({ "nInserted" : 1 })
> db.Country.insert({name:"Pakistan", cities:["Lahore","Karachi","Multan"]});
WriteResult({ "nInserted" : 1 })
> db.Country.insert({name:"Armenia", cities:["Gyumri","Vagharshapat"]});
WriteResult({ "nInserted" : 1 })
> db.Country.insert({name:"Kuwait", cities:["Al Jahra","Salmiya"]});
WriteResult({ "nInserted" : 1 })
> db.Country.find();
{ "_id" : ObjectId("5f88dc69b26d31f0f0835339"), "name" : "India", "cities" : [ "Mumbai", "New Delhi" ] }
{ "_id" : ObjectId("5f88dc69b26d31f0f083533a"), "name" : "Pakistan", "cities" : [ "Lahore", "Karachi", "Multan" ] }
{ "_id" : ObjectId("5f88dc69b26d31f0f083533b"), "name" : "Armenia", "cities" : [ "Gyumri", "Vagharshapat" ] }
{ "_id" : ObjectId("5f88dc69b26d31f0f083533c"), "name" : "Kuwait", "cities" : [ "Al Jahra", "Salmiya" ] }
> db.Country.find().limit(1);
{ "_id" : ObjectId("5f88dc69b26d31f0f0835339"), "name" : "India", "cities" : [ "Mumbai", "New Delhi" ] }
> db.Country.find().skip(2).limit(2);
{ "_id" : ObjectId("5f88dc69b26d31f0f083533b"), "name" : "Armenia", "cities" : [ "Gyumri", "Vagharshapat" ] }
{ "_id" : ObjectId("5f88dc69b26d31f0f083533c"), "name" : "Kuwait", "cities" : [ "Al Jahra", "Salmiya" ] }
> db.Country.update( { name: "India" }, { $push: { cities: "Imphal" } });
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
> db.Country.find({ name: "India" });
{ "_id" : ObjectId("5f88dc69b26d31f0f0835339"), "name" : "India", "cities" : [ "Mumbai", "New Delhi", "Imphal" ] }
> db.Country.update( { name: "Kuwait" }, { $pop: { cities: 1 } } );
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
> db.Country.find({ name: "Kuwait" });
{ "_id" : ObjectId("5f88dc69b26d31f0f083533c"), "name" : "Kuwait", "cities" : [ "Al Jahra" ] }
>
> db.Country.update( { name: "Pakistan" }, { $pull: { cities:'Karachi' } } );
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
> db.Country.find({ name: "Pakistan" });
{ "_id" : ObjectId("5f88dc69b26d31f0f083533a"), "name" : "Pakistan", "cities" : [ "Lahore", "Multan" ] }
>
> db.Country.update({ name: "India" },{$set:{'cities.1':'Pune'}});
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
>
> db.Country.update({ name: "India" },{$addToSet:{cities:"Mysore"}});
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
> db.Country.find({ name: "India" });
{ "_id" : ObjectId("5f88dc69b26d31f0f0835339"), "name" : "India", "cities" : [ "Mumbai", "Pune", "Imphal", "Mysore" ] }
```