❖ Assignment No:10

Aim: Implement aggregation and indexing with suitable example using MongoDB.

(base) sspm@sspm:~$ sudo su [sudo] password for sspm:
root@sspm:/home/sspm# service systemctl start mongod.service
Failed to start systemctl.service: Unit systemctl.service not found.
root@sspm:/home/sspm# mongo
MongoDB shell version v3.6.8
> **use lib**
switched to db lib

Create Collection:

> db.createCollection("Tutorial")
{ "ok" : 1 }

Insert:

> db.Tutorial.insert({title:"mongodb overview",description:"mongodb is easy to use",by_user:"tutorialpoint",url:"http://www.tuto.com",likes:"200"})
WriteResult({ "nInserted" : 1 })
> db.Tutorial.insert({title:"sql overview",description:"sql is very fast",by_user:"tutorialpoint",url:"http://www.tutorialpoint.com",likes:"450"})
WriteResult({ "nInserted" : 1 })
> db.Tutorial.insert({title:"Neo4j overview",description:"Neo4j is SQL database",by_user:"Neo4j",url:"http://www.neo4j.com",likes:"700"})
WriteResult({ "nInserted" : 1 })
> db.Tutorial.insert({title:"fb overview",description:"fb is SQL database",by_user:"fb",url:"http://www.facebook.com",likes:"650"})
WriteResult({ "nInserted" : 1 })

Read Operation:

> db.Tutorial.find().pretty() {
"_id" : ObjectId("670cded566f71e5764afd87a"),
"title" : "mongodb overview",
"description" : "mongodb is easy to use",
"by_user" : "tutorialpoint",
"url" : "http://www.tuto.com",
"likes" : "200" } {
"_id" : ObjectId("670cdf6566f71e5764afd87b"),
"title" : "sql overview",
"description" : "sql is very fast",
"by_user" : "tutorialpoint",
"url" : "http://www.tutorialpoint.com",
"likes" : "450" } {
"_id" : ObjectId("670cdff066f71e5764afd87c"),
"title" : "Neo4j overview",
"description" : "Neo4j is SQL database",
"by_user" : "Neo4j",
"url" : "http://www.neo4j.com",

"likes" : "700" } {
"_id" : ObjectId("670ce02f66f71e5764afd87d"),
"title" : "fb overview",
"description" : "fb is SQL database",
"by_user" : "fb",
"url" : "http://www.facebook.com",
"likes" : "650" }

Aggregate Function:

A.Group

1.Sum:

```
> db.Tutorial.aggregate([{$group:{_id:"$by_user",num_tutorial:{$sum:1}}}])
{ "_id" : "fb", "num_tutorial" : 1 }
{ "_id" : "Neo4j", "num_tutorial" : 1 }
{ "_id" : "tutorialpoint", "num_tutorial" : 2 }
> db.crateCollection("sales")
{ "ok" : 1 }
> db.Sales.insert({"_id":1,"item":"book","price":500,"quantity":2})
WriteResult({ "nInserted" : 1 })
> db.Sales.insert({"_id":2,"item":"pencil","price":10,"quantity":1})
WriteResult({ "nInserted" : 1 })
> db.Sales.insert({"_id":3,"item":"notebook","price":200,"quantity":5})
WriteResult({ "nInserted" : 1 })
> db.Sales.insert({"_id":4,"item":"scale","price":50,"quantity":3})
WriteResult({ "nInserted" : 1 })
> db.Sales.insert({"_id":5,"item":"bag","price":800,"quantity":1})
WriteResult({ "nInserted" : 1 })
> db.Sales.find().pretty()
{ "_id" : 1, "item" : "book", "price" : 500, "quantity" : 2 }
{ "_id" : 2, "item" : "pencil", "price" : 10, "quantity" : 1 }
{ "_id" : 3, "item" : "notebook", "price" : 200, "quantity" : 5 }
{ "_id" : 4, "item" : "scale", "price" : 50, "quantity" : 3 }
{ "_id" : 5, "item" : "bag", "price" : 800, "quantity" : 1 }
```

1. SUM:

```
> db.Sales.aggregate([{$group:{_id:"$item",total:{$sum:1}}}])
{ "_id" : "bag", "total" : 1 }
{ "_id" : "scale", "total" : 1 }
{ "_id" : "notebook", "total" : 1 }
{ "_id" : "pencil", "total" : 1 }
{ "_id" : "book", "total" : 1 }
```

2. MINIMUM:

```
> db.Sales.aggregate([{$group:{_id:"$item",minQuantity:{$min:"$quantity"}}}])
{ "_id" : "bag", "minQuantity" : 1 }
{ "_id" : "scale", "minQuantity" : 3 }
{ "_id" : "notebook", "minQuantity" : 5 }
{ "_id" : "pencil", "minQuantity" : 1 }
{ "_id" : "book", "minQuantity" : 2 }
```

3. MAXIMUM:

```
> db.Sales.aggregate([{$group:{_id:"$item",maxTotalAmount:{$max:{$multiply:
```

["$price","$quantity"]}},maxQuantity:{$max:"$quantity"}}}])
{ "_id" : "bag", "maxTotalAmount" : 800, "maxQuantity" : 1 }
{ "_id" : "scale", "maxTotalAmount" : 150, "maxQuantity" : 3 }
{ "_id" : "notebook", "maxTotalAmount" : 1000, "maxQuantity" : 5 }
{ "_id" : "pencil", "maxTotalAmount" : 10, "maxQuantity" : 1 }
{ "_id" : "book", "maxTotalAmount" : 1000, "maxQuantity" : 2 }

4. AVERAGE:

> db.Sales.aggregate([{$group:{_id:"$item",avgQuantity:{$avg:"$quantity"}}}]);
{ "_id" : "bag", "avgQuantity" : 1 }
{ "_id" : "scale", "avgQuantity" : 3 }
{ "_id" : "notebook", "avgQuantity" : 5 }
{ "_id" : "pencil", "avgQuantity" : 1 }
{ "_id" : "book", "avgQuantity" : 2 }

5. FIND FIRST:

> db.Sales.aggregate([{$group:{_id:"$item",FirstQuantity:{$first:"$quantity"}}}]);
{ "_id" : "bag", "FirstQuantity" : 1 }
{ "_id" : "scale", "FirstQuantity" : 3 }
{ "_id" : "notebook", "FirstQuantity" : 5 }
{ "_id" : "pencil", "FirstQuantity" : 1 }
{ "_id" : "book", "FirstQuantity" : 2 }

6. FIND LAST:

> db.Sales.aggregate([{$group:{_id:"$item",LastQuantity:{$last:"$quantity"}}}]);
{ "_id" : "bag", "LastQuantity" : 1 }
{ "_id" : "scale", "LastQuantity" : 3 }
{ "_id" : "notebook", "LastQuantity" : 5 }
{ "_id" : "pencil", "LastQuantity" : 1 }
{ "_id" : "book", "LastQuantity" : 2 }

7. PUSH:

> db.Sales.aggregate([{$group:{_id:"$item",pushQuantity:{$push:"$quantity"}}}]);
{ "_id" : "bag", "pushQuantity" : [1] }
{ "_id" : "scale", "pushQuantity" : [3] }
{ "_id" : "notebook", "pushQuantity" : [5] }
{ "_id" : "pencil", "pushQuantity" : [1] }
{ "_id" : "book", "pushQuantity" : [2] }

8. SORT:

> db.Sales.aggregate([{$sort:{'_id":1}},{$group:{_id:"$item",total:{$sum:1}}}])
2024-10-14T15:01:25.806+0530 E QUERY [thread1] SyntaxError: unterminated string literal
@(shell):1:28
> db.Sales.aggregate([{$sort:{'_id':1}},{$group:{_id:"$item",total:{$sum:1}}}])
{ "_id" : "bag", "total" : 1 }
{ "_id" : "scale", "total" : 1 }
{ "_id" : "notebook", "total" : 1 }
{ "_id" : "pencil", "total" : 1 }
{ "_id" : "book", "total" : 1 }

B.SORT:

1. SUM:

>
db.Sales.aggregate([{$sort:{'_id':1}},{$group:{_id:"$item",minQuantity:{$min:"quantity"}}}]);
{ "_id" : "bag", "minQuantity" : "quantity" }
{ "_id" : "scale", "minQuantity" : "quantity" }
{ "_id" : "notebook", "minQuantity" : "quantity" }
{ "_id" : "pencil", "minQuantity" : "quantity" }
{ "_id" : "book", "minQuantity" : "quantity" }

2. MIN:

>
db.Sales.aggregate([{$sort:{'_id':1}},{$group:{_id:"$item",maxQuantity:{$max:"quantity"}}}]);
{ "_id" : "bag", "maxQuantity" : "quantity" }
{ "_id" : "scale", "maxQuantity" : "quantity" }
{ "_id" : "notebook", "maxQuantity" : "quantity" }
{ "_id" : "pencil", "maxQuantity" : "quantity" }
{ "_id" : "book", "maxQuantity" : "quantity" }

3. MAX:

>
db.Sales.aggregate([{$sort:{'_id':1}},{$group:{_id:"$item",avgQuantity:{$avg:"$quantity"}}}]);
{ "_id" : "bag", "avgQuantity" : 1 }
{ "_id" : "scale", "avgQuantity" : 3 }
{ "_id" : "notebook", "avgQuantity" : 5 }
{ "_id" : "pencil", "avgQuantity" : 1 }
{ "_id" : "book", "avgQuantity" : 2 }

4. AVERAGE:

>
db.Sales.aggregate([{$sort:{'_id':1}},{$group:{_id:"$item",firstQuantity:{$first:"$quantity"}}}]);
{ "_id" : "bag", "firstQuantity" : 1 }
{ "_id" : "scale", "firstQuantity" : 3 }
{ "_id" : "notebook", "firstQuantity" : 5 }
{ "_id" : "pencil", "firstQuantity" : 1 }
{ "_id" : "book", "firstQuantity" : 2 }

5. FIRST:

>
db.Sales.aggregate([{$sort:{'_id':1}},{$group:{_id:"$item",lastQuantity:{$last:"$quantity"}}}]);
{ "_id" : "bag", "lastQuantity" : 1 }
{ "_id" : "scale", "lastQuantity" : 3 }
{ "_id" : "notebook", "lastQuantity" : 5 }
{ "_id" : "pencil", "lastQuantity" : 1 }
{ "_id" : "book", "lastQuantity" : 2 }

6. LAST:

> db.Sales.aggregate([{$sort:{'_id':1}},{$group:{_id:"$item",pushQuantity:
{$push:"$quantity"}}}]);
{ "_id" : "bag", "pushQuantity" : [1] }
{ "_id" : "scale", "pushQuantity" : [3] }
{ "_id" : "notebook", "pushQuantity" : [5] }
{ "_id" : "pencil", "pushQuantity" : [1] }
{ "_id" : "book", "pushQuantity" : [2] }

7. PUSH:

```
> db.Sales.aggregate([{$skip:1},{$group:{_id:"$item",totalQuantity:{$sum:"$quantity"}}}]);
{ "_id" : "bag", "totalQuantity" : 1 }
{ "_id" : "scale", "totalQuantity" : 3 }
{ "_id" : "notebook", "totalQuantity" : 5 }
{ "_id" : "pencil", "totalQuantity" : 1 }
```

C.SKIP:

1. SUM:

```
> db.Sales.aggregate([{$skip:1},{$group:{_id:"$item",minQuantity:{$min:"$quantity"}}}]);
{ "_id" : "bag", "minQuantity" : 1 }
{ "_id" : "scale", "minQuantity" : 3 }
{ "_id" : "notebook", "minQuantity" : 5 }
{ "_id" : "pencil", "minQuantity" : 1 }
```

2. MIN:

```
> db.Sales.aggregate([{$skip:1},{$group:{_id:"$item",maxQuantity:{$max:"$quantity"}}}]);
{ "_id" : "bag", "maxQuantity" : 1 }
{ "_id" : "scale", "maxQuantity" : 3 }
{ "_id" : "notebook", "maxQuantity" : 5 }
{ "_id" : "pencil", "maxQuantity" : 1 }
```

3. MAX:

```
> db.Sales.aggregate([{$skip:1},{$group:{_id:"$item",firstQuantity:{$first:"$quantity"}}}]);
{ "_id" : "bag", "firstQuantity" : 1 }
{ "_id" : "scale", "firstQuantity" : 3 }
{ "_id" : "notebook", "firstQuantity" : 5 }
{ "_id" : "pencil", "firstQuantity" : 1 }
```

4. FIRST:

```
> db.Sales.aggregate([{$skip:1},{$group:{_id:"$item",lastQuantity:{$last:"$quantity"}}}]);
{ "_id" : "bag", "lastQuantity" : 1 }
{ "_id" : "scale", "lastQuantity" : 3 }
{ "_id" : "notebook", "lastQuantity" : 5 }
{ "_id" : "pencil", "lastQuantity" : 1 }
```

5. LAST:

```
> db.Sales.aggregate([{$skip:1},{$group:{_id:"$item",avgQuantity:{$avg:"$quantity"}}}]);
{ "_id" : "bag", "avgQuantity" : 1 }
{ "_id" : "scale", "avgQuantity" : 3 }
{ "_id" : "notebook", "avgQuantity" : 5 }
{ "_id" : "pencil", "avgQuantity" : 1 }
```

6. AVERAGE:

```
> db.Sales.aggregate([{$skip:2},{$group:{_id:"$item",avgQuantity:{$avg:"$quantity"}}}]);
{ "_id" : "bag", "avgQuantity" : 1 }
{ "_id" : "scale", "avgQuantity" : 3 }
{ "_id" : "notebook", "avgQuantity" : 5 }
> db.Sales.aggregate([{$skip:3},{$group:{_id:"$item",avgQuantity:{$avg:"$quantity"}}}]);
```

{ "_id" : "bag", "avgQuantity" : 1 }
{ "_id" : "scale", "avgQuantity" : 3 }

7. PUSH:

> db.Sales.aggregate([{$skip:1},{$group:{_id:"$item",pushQuantity:{$push:"$quantity"}}}]);
{ "_id" : "bag", "pushQuantity" : [1] }
{ "_id" : "scale", "pushQuantity" : [3] }
{ "_id" : "notebook", "pushQuantity" : [5] }
{ "_id" : "pencil", "pushQuantity" : [1] }

D. LIMIT:

1. SUM:

> db.Sales.aggregate([{$limit:2},{$group:{_id:"$item",totalQuantity:{$sum:"$quantity"}}}]);
{ "_id" : "pencil", "totalQuantity" : 1 }
{ "_id" : "book", "totalQuantity" : 2 }

2. MAX :

> db.Sales.aggregate([{$limit:2},{$group:{_id:"$item",maxQuantity:{$max:"$quantity"}}}]);
{ "_id" : "pencil", "maxQuantity" : 1 }
{ "_id" : "book", "maxQuantity" : 2 }

3. MIN:

> db.Sales.aggregate([{$limit:2},{$group:{_id:"$item",minQuantity:{$min:"$quantity"}}}]);
{ "_id" : "pencil", "minQuantity" : 1 }
{ "_id" : "book", "minQuantity" : 2 }

4. AVERAGE:

> db.Sales.aggregate([{$limit:2},{$group:{_id:"$item",avgQuantity:{$avg:"$quantity"}}}]);
{ "_id" : "pencil", "avgQuantity" : 1 }
{ "_id" : "book", "avgQuantity" : 2 }

5. LIMIT:

> db.Sales.aggregate([{$limit:2},{$group:{_id:"$item",firstQuantity:{$first:"$quantity"}}}]);
{ "_id" : "pencil", "firstQuantity" : 1 }
{ "_id" : "book", "firstQuantity" : 2 }

6. FIRST:

> db.Sales.aggregate([{$limit:1},{$group:{_id:"$item",firstQuantity:{$first:"$quantity"}}}]);
{ "_id" : "book", "firstQuantity" : 2 }

7. LAST:

> db.Sales.aggregate([{$limit:2},{$group:{_id:"$item",lastQuantity:{$last:"$quantity"}}}]);
{ "_id" : "pencil", "lastQuantity" : 1 }
{ "_id" : "book", "lastQuantity" : 2 }
> db.Sales.aggregate([{$limit:1},{$group:{_id:"$item",lastQuantity:{$last:"$quantity"}}}]);
{ "_id" : "book", "lastQuantity" : 2 }

8. PUSH:

> db.Sales.aggregate([{$limit:2},{$group:{_id:"$item",pushQuantity:{$push:"$quantity"}}}]);
{ "_id" : "pencil", "pushQuantity" : [1] }
{ "_id" : "book", "pushQuantity" : [2] }

INDEX:

CREATE INDEX:

> db.Sales.ensureIndex({item:1}) {

"createdCollectionAutomatically" : false,
"numIndexesBefore" : 1,
"numIndexesAfter" : 2,
"ok" : 1 }
> db.sales.getIndexes()
[ ]

2. DISPLAY INDEX:

> db.Sales.getIndexes()
[{
"v" : 2,
"key" : {
"_id" : 1 },
"name" : "id",
"ns" : "lib.Sales" }, {
"v" : 2,
"key" : {
"item" : 1 },
"name" : "item_1",
"ns" : "lib.Sales" }]
> **db.Sales.ensureIndex({item:1,"price":-1})**{
"createdCollectionAutomatically" : false,
"numIndexesBefore" : 2,
"numIndexesAfter" : 3,
"ok" : 1}
> **db.Sales.getIndexes()** [{
"v" : 2,
"key" : {
"_id" : 1 },
"name" : "id",
"ns" : "lib.Sales" }, {
"v" : 2,
"key" : {
"item" : 1 },
"name" : "item_1",
"ns" : "lib.Sales" }, {
"v" : 2,
"key" : {
"item" : 1,
"price" : -1},
"name" : "item_1_price_-1",
"ns" : "lib.Sales" }]
>
db.Sales.ensureIndex({item:1,"price":1}){
"createdCollectionAutomatically" : false,
"numIndexesBefore" : 3,
"numIndexesAfter" : 4,
"ok" : 1 }

> **db.Sales.getIndexes()** [{
"v" : 2,
"key" : {
"_id" : 1 },
"name" : "id",
"ns" : "lib.Sales" }, {
"v" : 2,
"key" : {
"item" : 1 },
"name" : "item_1",
"ns" : "lib.Sales" }, {
"v" : 2,
"key" : {
"item" : 1,
"price" : -1 },
"name" : "item_1_price_-1",
"ns" : "lib.Sales" }, {
"v" : 2,
"key" : {
"item" : 1,
"price" : 1 },
"name" : "item_1_price_1",
"ns" : "lib.Sales" }]
> **db.Sales.dropIndex({item:1,"price":1})**
{ "nIndexesWas" : 4, "ok" : 1 }
> db.Sales.getIndexes() [{
"v" : 2,
"key" : {
"_id" : 1 },
"name" : "id",
"ns" : "lib.Sales" }, {
"v" : 2,
"key" : {
"item" : 1 },
"name" : "item_1",
"ns" : "lib.Sales" }, {
"v" : 2,
"key" : {
"item" : 1,
"price" : -1 },
"name" : "item_1_price_-1",
"ns" : "lib.Sales" }