Experiment No:2

Name: Samruddhi Sangram Yadav
Roll No.560

it29@it29-OptiPlex-380:~$ mongo
MongoDB shell version v4.4.29
connecting to:
mongodb://127.0.0.1:27017/?compressors=disabled&gssapiServiceName=mongodb
Implicit session: session { "id" : UUID("5f32730d-100b-49af-824a-ca36e0ffe49e") }
MongoDB server version: 4.4.29

The server generated these startup warnings when booting:
2025-07-29T15:45:41.655+05:30: Using the XFS filesystem is strongly recommended with the WiredTiger storage engine. See http://dochub.mongodb.org/core/prodnotes-filesystem
2025-07-29T15:45:45.928+05:30: Access control is not enabled for the database. Read and write access to data and configuration is unrestricted
---
> show dbs
admin 0.000GB
config 0.000GB
local 0.000GB
teit 0.000GB
users 0.000GB
> use teit
switched to db teit

Example 1:

Create a collection
> db.createCollection("order")
{ "ok" : 1 }

Insert Data in Collection

> db.order.insertMany([
... {item:"apple",price:20,quantity:5,category:'fruit'},
... {item:"banana",price:10,quantity:10,category:'fruit'},
... {item:"milk",price:30,quantity:2,category:'dairy'},
... {item:"apple",price:20,quantity:5,category:'fruit'},
... {item:"cheese",price:50,quantity:1,category:'dairy'},
... {item:"carrot",price:15,quantity:4,category:'vegetable'}])
{

```
        "acknowledged" : true,
        "insertedIds" : [
                ObjectId("6888a3571a49238c48bb01c4"),
                ObjectId("6888a3571a49238c48bb01c5"),
                ObjectId("6888a3571a49238c48bb01c6"),
                ObjectId("6888a3571a49238c48bb01c7"),
                ObjectId("6888a3571a49238c48bb01c8"),
                ObjectId("6888a3571a49238c48bb01c9")
        ]
}
```

View Inserted data in collection

```
> db.order.find().pretty()
{
        "_id" : ObjectId("6888a3571a49238c48bb01c4"),
        "item" : "apple",
        "price" : 20,
        "quantity" : 5,
        "category" : "fruit"
}
{
        "_id" : ObjectId("6888a3571a49238c48bb01c5"),
        "item" : "banana",
        "price" : 10,
        "quantity" : 10,
        "category" : "fruit"
}
{
        "_id" : ObjectId("6888a3571a49238c48bb01c6"),
        "item" : "milk",
        "price" : 30,
        "quantity" : 2,
        "category" : "dairy"
}
{
        "_id" : ObjectId("6888a3571a49238c48bb01c7"),
        "item" : "apple",
        "price" : 20,
        "quantity" : 5,
        "category" : "fruit"
}
{
        "_id" : ObjectId("6888a3571a49238c48bb01c8"),
        "item" : "cheese",
```

```
        "price" : 50,
        "quantity" : 1,
        "category" : "dairy"
}
{
        "_id" : ObjectId("6888a3571a49238c48bb01c9"),
        "item" : "carrot",
        "price" : 15,
        "quantity" : 4,
        "category" : "vegetable"
}
```

$match aggredate function

```
> db.orders.aggregate([
... {$match:{category:"fruit"}}
... ]).pretty()
> db.order.aggregate([ {$match:{category:"fruit"}} ]).pretty()
{
        "_id" : ObjectId("6888a3571a49238c48bb01c4"),
        "item" : "apple",
        "price" : 20,
        "quantity" : 5,
        "category" : "fruit"
}
{
        "_id" : ObjectId("6888a3571a49238c48bb01c5"),
        "item" : "banana",
        "price" : 10,
        "quantity" : 10,
        "category" : "fruit"
}
{
        "_id" : ObjectId("6888a3571a49238c48bb01c7"),
        "item" : "apple",
        "price" : 20,
        "quantity" : 5,
        "category" : "fruit"
}

> db.order.aggregate([ {$match:{category:"vegetable"}} ]).pretty()
{
        "_id" : ObjectId("6888a3571a49238c48bb01c9"),
        "item" : "carrot",
        "price" : 15,
```

```
        "quantity" : 4,
        "category" : "vegetable"
}
```

$project aggeregate function

```
> db.order.aggregate([ {$project:{ item:1,
total:{$multiply:["$price","$quantity"]} }} ]).pretty()
{
        "_id" : ObjectId("6888a3571a49238c48bb01c4"),
        "item" : "apple",
        "total" : 100
}
{
        "_id" : ObjectId("6888a3571a49238c48bb01c5"),
        "item" : "banana",
        "total" : 100
}
{
        "_id" : ObjectId("6888a3571a49238c48bb01c6"),
        "item" : "milk",
        "total" : 60
}
{
        "_id" : ObjectId("6888a3571a49238c48bb01c7"),
        "item" : "apple",
        "total" : 100
}
{
        "_id" : ObjectId("6888a3571a49238c48bb01c8"),
        "item" : "cheese",
        "total" : 50
}
{
        "_id" : ObjectId("6888a3571a49238c48bb01c9"),
        "item" : "carrot",
        "total" : 60
}
```

$group aggregate function

```
> db.order.aggregate([ { $group:{ _id:"$category",
totalQuantity:{$sum:"$quantity"}}}]).pretty()
{ "_id" : "vegetable", "totalQuantity" : 4 }
```

Example 1:

```
{ "_id" : "dairy", "totalQuantity" : 3 }
{ "_id" : "fruit", "totalQuantity" : 20 }
```

$sort aggregate function

```
>   db.order.aggregate([
... {$sort:{price:-1}} //descending order
... ])
{ "_id" : ObjectId("6888a3571a49238c48bb01c8"), "item" : "cheese", "price" : 50,
"quantity" : 1, "category" : "dairy" }
{ "_id" : ObjectId("6888a3571a49238c48bb01c6"), "item" : "milk", "price" : 30,
"quantity" : 2, "category" : "dairy" }
{ "_id" : ObjectId("6888a3571a49238c48bb01c4"), "item" : "apple", "price" : 20,
"quantity" : 5, "category" : "fruit" }
{ "_id" : ObjectId("6888a3571a49238c48bb01c7"), "item" : "apple", "price" : 20,
"quantity" : 5, "category" : "fruit" }
{ "_id" : ObjectId("6888a3571a49238c48bb01c9"), "item" : "carrot", "price" : 15,
"quantity" : 4, "category" : "vegetable" }
{ "_id" : ObjectId("6888a3571a49238c48bb01c5"), "item" : "banana", "price" : 10,
"quantity" : 10, "category" : "fruit" }
```

item quantity is greater then 3 and calculate total=price*quantity, sort by descending order.

```
> db.order.aggregate([
... {$match:{quantity:{$gt:3}}},
... {
... $project:{
... item:1,
... total:{$multiply:["$price","$quantity"]}
... }
... },
... {$sort:{total:-1}}
... ]).pretty()
{
        "_id" : ObjectId("6888a3571a49238c48bb01c4"),
        "item" : "apple",
        "total" : 100
}
{
        "_id" : ObjectId("6888a3571a49238c48bb01c5"),
        "item" : "banana",
        "total" : 100
}
{
```

```
        "_id" : ObjectId("6888a3571a49238c48bb01c7"),
        "item" : "apple",
        "total" : 100
}
{
        "_id" : ObjectId("6888a3571a49238c48bb01c9"),
        "item" : "carrot",
        "total" : 60
}
```

Example 2:
create collection

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

Insert recordes

```
> db.orders.insertMany([ {_id:1,customer:"Amit",amount:500,statues: "delivered"},
{_id:2,customer:"Neha",amount:300,statues:
"pending"},{_id:3,customer:"Amit",amount:200,statues:
"delivered"},{_id:4,customer:"vijay",amount:700,statues:
"delivered"},{_id:5,customer:"Neha",amount:150,statues: "delivered"}])
{ "acknowledged" : true, "insertedIds" : [ 1, 2, 3, 4, 5 ] }
```

Aggregation pipeline

```
> db.orders.aggregate([ {$match:{statues:"delivered"}}, { $group:{ _id:"$customer",
totalAmount:{$sum:"$amount"} } } ]).pretty()
{ "_id" : "Amit", "totalAmount" : 700 }
{ "_id" : "Neha", "totalAmount" : 150 }
{ "_id" : "vijay", "totalAmount" : 700 }
```

Map-Reduce

create colletion

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

Insert recordes

```
> db.sales.insertMany([ {item:"pen",quantity:10,price:5},
{item:"pen",quantity:15,price:5},
... {item:"book",quantity:5,price:20},
... {item:"book",quantity:2,price:20},
... {item:"pencil",quantity:7,price:2}]);
```

```
{
        "acknowledged" : true,
        "insertedIds" : [
                ObjectId("6888ae6d799a0b458dff7846"),
                ObjectId("6888ae6d799a0b458dff7847"),
                ObjectId("6888ae6d799a0b458dff7848"),
                ObjectId("6888ae6d799a0b458dff7849"),
                ObjectId("6888ae6d799a0b458dff784a")
        ]
}
>
```

Map function

```
> var mapfunction=function(){
... emit(this.item,this.quantity);
... };
```

Reduce Function

```
> var reduceFunction=function(keyItem,values){
... return Array.sum(values);
... };
```

Map-Reduce Operation

```
> db.sales.mapReduce
    ( mapFunction, reduceFunction, {
       out:"total_sales"
     });
{ "result" : "total_sales", "ok" : 1 }
```

Output collection:total_sales

```
> db.total_sales.find().pretty();
{ "_id" : "pen", "value" : 25 }
{ "_id" : "pencil", "value" : 7 }
{ "_id" : "book", "value" : 7 }
```

Create employee collection

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

Insert employee data

```
> db.employee.insertMany([ {_id:1,firstNAME:"ank",salary:1000},
{_id:2,firstNAME:"ank",salary:2000}
... ,{_id:3,firstNAME:"sag",salary:3000},
... {_id:4,firstNAME:"sag",salary:4000},
... {_id:5,firstNAME:"neh",salary:5000},
... {_id:6,firstNAME:"neh",salary:5000}])
{ "acknowledged" : true, "insertedIds" : [ 1, 2, 3, 4, 5, 6 ] }
```

Find the sum of salary of all employee with same firstName where salary < 5000 in employee collection.

```
> db.employee.mapReduce(
    function() { emit(this.firstNAME,this.salary);},
   function(key,values) {return Array.sum(values)},
   { query:{salary:{$lt:5000}},
   out:"result" });
  { "result" : "result", "ok" : 1 }
```

Output:

```
> db.result.find().pretty();
{ "_id" : "sag", "value" : 7000 }
{ "_id" : "ank", "value" : 3000 }
```

word Frequency counting using MapReduce in MongoDB

```
> use mydb
switched to db mydb
```

Create table Collection

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

Insert Data

```
> db.texts.insertMany([
... ... {"_id":1,"content":"Hello world hello"},
... {"_id":2,"content":"Hello MapReduce in MongoDB"},
... {"_id":3,"content":"world of hadoop and MapReduce"}
... ])
{ "acknowledged" : true, "insertedIds" : [ 1, 2, 3 ] }
```

Display Data

```
> db.texts.find().pretty()
{ "_id" : 1, "content" : "Hello world hello" }
{ "_id" : 2, "content" : "Hello MapReduce in MongoDB" }
{ "_id" : 3, "content" : "world of hadoop and MapReduce" }
>
```

Define MapReduce function

```
> var mapFunction=function(){
... var words=this.content.split("");
... for(var i=0;i<word.length;i++){
... emit(words[i].toLowerCase(),1);
... }
... };

> var reduceFunction=function(key,values){
... return Array.sum(values);
... };
```

Run MapReduce in MongoDB

```
> db.texts.mapReduce(
... mapFunction,
... reduceFunction,
... {out:"word_counts"}
... )
```

View result

```
> db.word_counts.find().sort({value:-1})
{ "_id": "hello", "value": 3 }
{ "_id": "world", "value": 2 }
{ "_id": "mapreduce", "value": 2 }
{ "_id": "mongodb", "value": 1 }
{ "_id": "hadoop", "value": 1 }
{ "_id": "of", "value": 1 }
{ "_id": "and", "value": 1 }
```

sample collection:student

```
> db.students.insertMany([
... {name:"Amit",age:21,marks:85,subjects:["Math","Physics"]},
... {name:"Neha",age:22,marks:92,subjects:["Chemistry","Biology"]},
... {name:"Ravi",age:21,marks:75,subjects:["Physics","Chemistry"]},
... {name:"Kiran",age:23,marks:88,subjects:["Math","Biology"]}
... ])
```

```
{
        "acknowledged" : true,
        "insertedIds" : [
                ObjectId("6888ab2f5406a030e762531f"),
                ObjectId("6888ab2f5406a030e7625320"),
                ObjectId("6888ab2f5406a030e7625321"),
                ObjectId("6888ab2f5406a030e7625322")
        ]
}
```

single Field index

```
> db.students.find({name:"Amit"})
{ "_id" : ObjectId("6888ab2f5406a030e762531f"), "name" : "Amit", "age" : 21,
"marks" : 85, "subjects" : [ "Math", "Physics" ] }
```

compound Index

```
> db.students.createIndex({age:1,marks:-1})
{
        "createdCollectionAutomatically" : false,
        "numIndexesBefore" : 2,
        "numIndexesAfter" : 3,
        "ok" : 1
}
> db.students.find({age:21}).sort({marks:-1})
{ "_id" : ObjectId("6888ab2f5406a030e762531f"), "name" : "Amit", "age" : 21,
"marks" : 85, "subjects" : [ "Math", "Physics" ] }
{ "_id" : ObjectId("6888ab2f5406a030e7625321"), "name" : "Ravi", "age" : 21,
"marks" : 75, "subjects" : [ "Physics", "Chemistry" ] }
```

Multikey index

```
> db.students.createIndex({subject:1})
{
        "createdCollectionAutomatically" : false,
        "numIndexesBefore" : 3,
        "numIndexesAfter" : 4,
        "ok" : 1
}

> db.students.find({subjects:'Math'})
{ "_id" : ObjectId("6888ab2f5406a030e762531f"), "name" : "Amit", "age" : 21,
"marks" : 85, "subjects" : [ "Math", "Physics" ] }
{ "_id" : ObjectId("6888ab2f5406a030e7625322"), "name" : "Kiran", "age" : 23,
"marks" : 88, "subjects" : [ "Math", "Biology" ] }
```

Text Index

```
> db.students.createIndex({name:"text"})
{
        "createdCollectionAutomatically" : false,
        "numIndexesBefore" : 4,
        "numIndexesAfter" : 5,
        "ok" : 1
}

> db.students.find({$text:{$search:"Amit"}})
{ "_id" : ObjectId("6888ab2f5406a030e762531f"), "name" : "Amit", "age" : 21,
"marks" : 85, "subjects" : [ "Math", "Physics" ] }
>
```

Hashed Index

```
> db.students.createIndex({name:"hashed"})
{
        "createdCollectionAutomatically" : false,
        "numIndexesBefore" : 5,
        "numIndexesAfter" : 6,
        "ok" : 1
}
> db.students.find({name:"Neha"})
{ "_id" : ObjectId("6888ab2f5406a030e7625320"), "name" : "Neha", "age" : 22,
"marks" : 92, "subjects" : [ "Chemistry", "Biology" ] }
```

wildcard Index

```
> db.dynamicData.createIndex({"$**":1})
{
        "createdCollectionAutomatically" : true,
        "numIndexesBefore" : 1,
        "numIndexesAfter" : 2,
        "ok" : 1
}
```

check and drop indexes

```
db.students.getIndexes()
[
        {
                "v" : 2,
                "key" : {
                        "_id" : 1
```

```
        },
        "name" : "_id_"
},
{
        "v" : 2,
        "key" : {
                "name" : 1
        },
        "name" : "name_1"
},
{
        "v" : 2,
        "key" : {
                "age" : 1,
                "marks" : -1
        },
        "name" : "age_1_marks_-1"
},
{
        "v" : 2,
        "key" : {
                "subject" : 1
        },
        "name" : "subject_1"
},
{
        "v" : 2,
        "key" : {
                "_fts" : "text",
                "_ftsx" : 1
        },
        "name" : "name_text",
        "weights" : {
                "name" : 1
        },
        "default_language" : "english",
        "language_override" : "language",
        "textIndexVersion" : 3
},
{
        "v" : 2,
        "key" : {
                "name" : "hashed"
        },
        "name" : "name_hashed"
```

```
        }
]
>

> db.students.dropIndex("name_1")
{ "nIndexesWas" : 6, "ok" : 1 }
```