Assignment

**Document store NoSQL: MongoDB**

**MongoDB:**

1. [1 marks] Create a database called ‘movies’ and *write* a MongoDB query to select database as ‘movies’.

Code:

|  |
| --- |
| use movies |

Output:

|  |
| --- |
| switched to db movies |

1. [1 marks] Write a MongoDB query to display databases.

Code :

|  |
| --- |
| show dbs |

Output :

|  |
| --- |
| admin 0.000GB config 0.000GB local 0.000GB movies 0.000GB |

1. [1 marks] Create a collection called ‘mvr\_name’.(use capping and not capping)

Code(crapping):

|  |
| --- |
| db.createCollection("mvr\_name", { capped : true } ) |

Output:

|  |
| --- |
| {  "ok" : 0,  "errmsg" : "specify size:<n> when capped is true",  "code" : 14832,  "codeName" : "Location14832" } |

Code(Not Crapping):

|  |
| --- |
| db.createCollection("mvr\_name", { capped : false } ) |

Output:

|  |
| --- |
| { "ok" : 1 } |

1. [1 marks]Add 10 movies details to the collection named”mvr\_name”. Each document consists of following properties as Reviewer\_name, movie\_name, genre (Action, Crime, Horror, Comedy, Animation) as array, director, rating (out of 10), timestamp ( date and year release), reviewer\_occupation.

Code:

|  |
| --- |
| db.mvr\_name.insert([{Reviewer\_name:"Blue sattai",movie\_name:"bigil",genre:["Action","comedy"],Director:"atlee",Rating:7,timestamp:new Date("2019-11-27"),Reviewer\_occupation:"youtuber"},{Reviewer\_name:"Blue sattai",movie\_name:"kappan",genre:["Action","comedy"],Director:"K.V.Anand",Rating:6,timestamp:new Date("2019-08-14"),Reviewer\_occupation:"youtuber"},{Reviewer\_name:"Blue sattai",movie\_name:"Ner Konda paarvai",genre:["Action"],Director:"H.Vinoth",Rating:7,timestamp:new Date("2019-09-09"),Reviewer\_occupation:"youtuber"},{Reviewer\_name:"Blue sattai",movie\_name:"IT-2",genre:["Horror"],Director:"Andres Muschietti",Rating:8,timestamp:new Date("2019-04-07"),Reviewer\_occupation:"youtuber"},{Reviewer\_name:"Blue sattai",movie\_name:"Joker",genre:["Crime"],Director:"Todd phillips",Rating:9,timestamp:new Date("2019-09-08"),Reviewer\_occupation:"youtuber"},{Reviewer\_name:"Blue sattai",movie\_name:"Your Name",genre:["Animation"],Director:"Makoto Shinkai",Rating:9,timestamp:new Date("2019-11-27"),Reviewer\_occupation:"youtuber"},{Reviewer\_name:"Blue sattai",movie\_name:"Avengers endgame",genre:["Action"],Director:"Russo brothers",Rating:9,timestamp:new Date("2019-06-29"),Reviewer\_occupation:"youtuber"},{Reviewer\_name:"Blue sattai",movie\_name:"robo",genre:["Action","adventure"],Director:"shankar",Rating:8,timestamp:new Date("2010-09-17"),Reviewer\_occupation:"youtuber"},{Reviewer\_name:"Blue sattai",movie\_name:"robo 2.0",genre:["Action","adventure"],Director:"shankar",Rating:6,timestamp:new Date("2018-10-16"),Reviewer\_occupation:"youtuber"},{Reviewer\_name:"Blue sattai",movie\_name:"kaidhi",genre:["Action"],Director:"lokesh",Rating:8,timestamp:new Date("2019-10-27"),Reviewer\_occupation:"youtuber"}]) |

Output:

|  |
| --- |
| BulkWriteResult({  "writeErrors" : [ ],  "writeConcernErrors" : [ ],  "nInserted" : 10,  "nUpserted" : 0,  "nMatched" : 0,  "nModified" : 0,  "nRemoved" : 0,  "upserted" : [ ] }) |

1. [1 marks] Write a MongoDB query to display single document in a database and all the documents in a database. And also Use *pretty()* to display the results.

Code 1:

|  |
| --- |
| db.mvr;\_name.find() |

Output:

|  |
| --- |
| { "\_id" : ObjectId("5defd3ffe24626f63a775490"), "Reviewer\_name" : "Blue sattai", "movie\_name" : "bigil", "genre" : [ "Action", "comedy" ], "Director" : "atlee", "Rating" : 7, "timestamp" : ISODate("2019-11-27T00:00:00Z"), "Reviewer\_occupation" : "youtuber" } { "\_id" : ObjectId("5defd3ffe24626f63a775491"), "Reviewer\_name" : "Blue sattai", "movie\_name" : "kappan", "genre" : [ "Action", "comedy" ], "Director" : "K.V.Anand", "Rating" : 6, "timestamp" : ISODate("2019-08-14T00:00:00Z"), "Reviewer\_occupation" : "youtuber" } { "\_id" : ObjectId("5defd3ffe24626f63a775492"), "Reviewer\_name" : "Blue sattai", "movie\_name" : "Ner Konda paarvai", "genre" : [ "Action" ], "Director" : "H.Vinod", "Rating" : 7, "timestamp" : ISODate("2019-09-09T00:00:00Z"), "Reviewer\_occupation" : "youtuber" } { "\_id" : ObjectId("5defd3ffe24626f63a775493"), "Reviewer\_name" : "Blue sattai", "movie\_name" : "IT-2", "genre" : [ "Horror" ], "Director" : "Andres Muschietti", "Rating" : 8, "timestamp" : ISODate("2019-04-07T00:00:00Z"), "Reviewer\_occupation" : "youtuber" } { "\_id" : ObjectId("5defd3ffe24626f63a775494"), "Reviewer\_name" : "Blue sattai", "movie\_name" : "Joker", "genre" : [ "Crime" ], "Director" : "Todd phillips", "Rating" : 9, "timestamp" : ISODate("2019-09-08T00:00:00Z"), "Reviewer\_occupation" : "youtuber" } { "\_id" : ObjectId("5defd3ffe24626f63a775495"), "Reviewer\_name" : "Blue sattai", "movie\_name" : "Your Name", "genre" : [ "Animation" ], "Director" : "Makoto Shinkai", "Rating" : 9, "timestamp" : ISODate("2019-11-27T00:00:00Z"), "Reviewer\_occupation" : "youtuber" } { "\_id" : ObjectId("5defd3ffe24626f63a775496"), "Reviewer\_name" : "Blue sattai", "movie\_name" : "Avengers endgame", "genre" : [ "Action" ], "Director" : "Russo brothers", "Rating" : 9, "timestamp" : ISODate("2019-06-29T00:00:00Z"), "Reviewer\_occupation" : "youtuber" } { "\_id" : ObjectId("5defd3ffe24626f63a775497"), "Reviewer\_name" : "Blue sattai", "movie\_name" : "robo", "genre" : [ "Action", "adventure" ], "Director" : "shankar", "Rating" : 8, "timestamp" : ISODate("2010-09-17T00:00:00Z"), "Reviewer\_occupation" : "youtuber" } { "\_id" : ObjectId("5defd3ffe24626f63a775498"), "Reviewer\_name" : "Blue sattai", "movie\_name" : "robo 2.0", "genre" : [ "Action", "adventure" ], "Director" : "shankar", "Rating" : 6, "timestamp" : ISODate("2018-10-16T00:00:00Z"), "Reviewer\_occupation" : "youtuber" } { "\_id" : ObjectId("5defd3ffe24626f63a775499"), "Reviewer\_name" : "Blue sattai", "movie\_name" : "kaidhi", "genre" : [ "Action" ], "Director" : "lokesh", "Rating" : 8, "timestamp" : ISODate("2019-10-27T00:00:00Z"), "Reviewer\_occupation" : "youtuber" } |

Code 2(using pretty) :

|  |
| --- |
| db.mvr\_name.find().pretty() |

Output:

|  |
| --- |
| {  "\_id" : ObjectId("5defd3ffe24626f63a775490"),  "Reviewer\_name" : "Blue sattai",  "movie\_name" : "bigil",  "genre" : [  "Action",  "comedy"  ],  "Director" : "atlee",  "Rating" : 7,  "timestamp" : ISODate("2019-11-27T00:00:00Z"),  "Reviewer\_occupation" : "youtuber" } {  "\_id" : ObjectId("5defd3ffe24626f63a775491"),  "Reviewer\_name" : "Blue sattai",  "movie\_name" : "kappan",  "genre" : [  "Action",  "comedy"  ],  "Director" : "K.V.Anand",  "Rating" : 6,  "timestamp" : ISODate("2019-08-14T00:00:00Z"),  "Reviewer\_occupation" : "youtuber" } {  "\_id" : ObjectId("5defd3ffe24626f63a775492"),  "Reviewer\_name" : "Blue sattai",  "movie\_name" : "Nerkonda parvai",  "genre" : [  "Action"  ],  "Director" : "H.Vinod",  "Rating" : 7,  "timestamp" : ISODate("2019-09-09T00:00:00Z"),  "Reviewer\_occupation" : "youtuber" } {  "\_id" : ObjectId("5defd3ffe24626f63a775493"),  "Reviewer\_name" : "Blue sattai",  "movie\_name" : "IT-2",  "genre" : [  "Horrer"  ],  "Director" : "Andres Muschietti",  "Rating" : 8,  "timestamp" : ISODate("2019-04-07T00:00:00Z"),  "Reviewer\_occupation" : "youtuber" } {  "\_id" : ObjectId("5defd3ffe24626f63a775494"),  "Reviewer\_name" : "Blue sattai",  "movie\_name" : "Joker",  "genre" : [  "Crime"  ],  "Director" : "Todd phillips",  "Rating" : 9,  "timestamp" : ISODate("2019-09-08T00:00:00Z"),  "Reviewer\_occupation" : "youtuber" } {  "\_id" : ObjectId("5defd3ffe24626f63a775495"),  "Reviewer\_name" : "Blue sattai",  "movie\_name" : "Your Name",  "genre" : [  "Animation"  ],  "Director" : "Makoto Shinkai",  "Rating" : 9,  "timestamp" : ISODate("2019-11-27T00:00:00Z"),  "Reviewer\_occupation" : "youtuber" } {  "\_id" : ObjectId("5defd3ffe24626f63a775496"),  "Reviewer\_name" : "Blue sattai",  "movie\_name" : "Avengers endgame",  "genre" : [  "Action"  ],  "Director" : "Russo brothers",  "Rating" : 9,  "timestamp" : ISODate("2019-06-29T00:00:00Z"),  "Reviewer\_occupation" : "youtuber" } {  "\_id" : ObjectId("5defd3ffe24626f63a775497"),  "Reviewer\_name" : "Blue sattai",  "movie\_name" : "robo",  "genre" : [  "Action",  "adventure"  ],  "Director" : "shankar",  "Rating" : 8,  "timestamp" : ISODate("2010-09-17T00:00:00Z"),  "Reviewer\_occupation" : "youtuber" } {  "\_id" : ObjectId("5defd3ffe24626f63a775498"),  "Reviewer\_name" : "Blue sattai",  "movie\_name" : "robo 2.0",  "genre" : [  "Action",  "adventure"  ],  "Director" : "shankar",  "Rating" : 6,  "timestamp" : ISODate("2018-10-16T00:00:00Z"),  "Reviewer\_occupation" : "youtuber" } {  "\_id" : ObjectId("5defd3ffe24626f63a775499"),  "Reviewer\_name" : "Blue sattai",  "movie\_name" : "kaidhi",  "genre" : [  "Action"  ],  "Director" : "lokesh",  "Rating" : 8,  "timestamp" : ISODate("2019-10-27T00:00:00Z"),  "Reviewer\_occupation" : "youtuber" } |

1. [1 marks] find the highest and lowest rating movie and update the movie rating as maximum- minimum and minimum to Maximum.

Code:

|  |
| --- |
| db.mvr\_name.aggregate([ { "$group": { "\_id": null, "MaximumValue":{ "$max": "$Rating" }, "MinimumValue": { "$min": "$Rating" }}} ]); |

Output:

|  |
| --- |
| { "\_id" : null, "MaximumValue" : 9, "MinimumValue" : 6 } |

1. [4 marks] MongoDB Querying:

* Find the total number of rating movies rated by user belonging to specific occupation.

Code:

|  |
| --- |
| db.mvr\_name.aggregate([{$group : {\_id : "$Reviewer\_occupation", sum : {$sum : 1}}}]) |

Output:

|  |
| --- |
| { "\_id" : "youtuber", "sum" : 20 } |

* Find the average rating by reviewer and movie.

Code:

|  |
| --- |
| db.mvr\_name.aggregate([{ "$group": {"\_id": null, "average rating":{"$avg": "$Rating" }}} ]); |

Output:

|  |
| --- |
| { "\_id" : null, "average rating" : 7.7 } |

* Find the movie name filmed between 2015 and 2017 and also find the 3rd highest rated movie. [Hint: use sort, limit, skip]

Code for movie filmed between 2015 and 2017:

|  |
| --- |
| db.mvr\_name.find({"timestamp":{$gte:new ISODate("2015-01-01T00:00:00Z"),$lte:new ISODate("2017-12-30T00:00:00Z")}}) |

Output:

|  |
| --- |
| { "\_id" : ObjectId("5defd3ffe24626f63a775498"), "Reviewer\_name" : "Blue sattai", "movie\_name" : "robo 2.0", "genre" : [ "Action", "adventure" ], "Director" : "shankar", "Rating" : 6, "timestamp" : ISODate("2018-10-16T00:00:00Z"), "Reviewer\_occupation" : "youtuber" } |

Code for 3rd highest:

db.mvr\_name.find({},{"movie\_name":1,\_id:0}).sort({"Rating":-1}).limit(1).skip(2)

Output:

{ "movie\_name" : "Avengers endgame" }

**Practice(tutorials point/mongo docs)**

1.Creating database

Code :

use movies

Output :

|  |
| --- |
| switched to movies mydb |

2.Write a MongoDB query to display databases.

Code :

|  |
| --- |
| show dbs |

Output :

|  |
| --- |
| admin 0.000GB  config 0.000GB  db 0.000GB  local 0.000GB |

3.Drop database

Code :

|  |
| --- |
| db.dropDatabase |

Output:

|  |
| --- |
| function () {  if (arguments.length)  throw Error("dropDatabase doesn't take arguments");  return this.\_dbCommand({dropDatabase: 1});  } |

4..Inserting new document in student collection.

Code : db.student.insertOne({name:"Goutham",age:19,branch:"swe",subjects:{statistics:2,nosqldb:4,AI:4}})

Output :

|  |
| --- |
| WriteResult({ "nInserted" : 1 }) |

5.Collection list

Code:

|  |
| --- |
| show collection |

Output:

|  |
| --- |
| student |

6.Dropping collection

Code:

|  |
| --- |
| db.student.drop() |

Output:

|  |
| --- |
| True |

7.Inserting multiple documents

Code:

|  |
| --- |
| db.student.insert([{name:"harish",age:19,branch:"cse",subjects:{law:3,nosqldb:4,AI:4}},{name:"vageesan",age:19,branch:"ECE",subjects:{digital\_logics:3,embeded\_prog:4,vlsi:4}}]) |

Output:

|  |
| --- |
| BulkWriteResult({  "writeErrors" : [ ],  "writeConcernErrors" : [ ],  "nInserted" : 2,  "nUpserted" : 0,  "nMatched" : 0,  "nModified" : 0,  "nRemoved" : 0,  "upserted" : [ ]  }) |

8.Display all the documents in a non-structured way-**find()** method.

Code:

|  |
| --- |
| db.student.find() |

Output :

|  |
| --- |
| { "\_id" : ObjectId("5dee11a91be6a0a74301aff6"), "name" : "Goutham", "age" : 19, "branch" : "swe", "subjects" : { "statistics" : 2, "nosqldb" : 4, "AI" : 4 } }  { "\_id" : ObjectId("5dee11de1be6a0a74301aff7"), "name" : "yokesh", "age" : 19, "branch" : "cse", "subjects" : { "cloud\_cmp" : 2, "nosqldb" : 4, "AI" : 4 } }  { "\_id" : ObjectId("5dee12a31be6a0a74301aff8"), "name" : "harish", "age" : 19, "branch" : "cse", "subjects" : { "law" : 3, "nosqldb" : 4, "AI" : 4 } }  { "\_id" : ObjectId("5dee12a31be6a0a74301aff9"), "name" : "vageesan", "age" : 19, "branch" : "ECE", "subjects" : { "digital\_logics" : 3, "embeded\_prog" : 4, "vlsi" : 4 } } |

9.Display the results in a formatted way

Code:

|  |
| --- |
| db.student.find().pretty() |

Output:

|  |
| --- |
| {  "\_id" : ObjectId("5dee11a91be6a0a74301aff6"),  "name" : "Goutham",  "age" : 19,  "branch" : "swe",  "subjects" : {  "statistics" : 2,  "nosqldb" : 4,  "AI" : 4  } } {  "\_id" : ObjectId("5dee11de1be6a0a74301aff7"),  "name" : "yokesh",  "age" : 19,  "branch" : "cse",  "subjects" : {  "cloud\_cmp" : 2,  "nosqldb" : 4,  "AI" : 4  } } {  "\_id" : ObjectId("5dee12a31be6a0a74301aff8"),  "name" : "harish",  "age" : 19,  "branch" : "cse",  "subjects" : {  "law" : 3,  "nosqldb" : 4,  "AI" : 4  } } {  "\_id" : ObjectId("5dee12a31be6a0a74301aff9"),  "name" : "vageesan",  "age" : 19,  "branch" : "ECE",  "subjects" : {  "digital\_logics" : 3,  "embeded\_prog" : 4,  "vlsi" : 4  } } |