Databases and information systems laboratory CS313

IIT Dharwad

Handout 10
25 – 10 – 2023

MongoDB

Follow instructions in the class to set up *MongoDB*.

- Set up an account on `https://www.mongodb.com`
- Create a cluster on a cloud server
- Install mongosh on your computer
- Connect to mongo shell

Questions on Document database. Connect to the cluster that you have created using *Mongoshell* and do the following: [1]

Create and list database

1. List the databases in the cluster show dbs
2. Create a new database called *CompanyDB*. use companydb
if no present creates new db else switches to db
3. (⋆) Now list the databases in the cluster. Does it show *CompanyDB*?

No because it is empty

[1](⋆) refers to questions that you can try on your own

Insert

1. Use *CompanyDB* from now onwards. use companydb
2. Create a new collection called *customers* and insert a document with the following details: db.customer.insertOne({'name':'Alice','age':24})

name	Alice
age	24

insertOne():- one document as argument
insertMany():- List as an argument

3. (⋆) List the databases in the cluster. Does it show *CompanyDB*?

```
show dbs
companydb    8.00 KiB
admin      348.00 KiB
local       18.87 GiB
```

4. List all the collections in the database *CompanyDB* show collections
5. Insert the following documents into the collection *customers*

name	Bob
name	Charles
age	26
level	1
name	Darshan
age	27

```
db.customer.insertMany([{
'name':'Bob'} ,
{'name':'Charles','Age':26,'
level':1},
{'name':'Darshan','Age':27
}])
```

Find Output is list of json file

1. Find all the documents in the collection *customers* db.customer.find()
2. Find all *customers* whose age is 27. companydb> db.customer.find({'age':27})
3. Find all *customers* whose age is ≥ 25. companydb> db.customer.find({'age':{$gte:25} })

for except use {field:0}

4. (⋆) Find all *customers* whose age is < 27. db.customer.find({'age':{$lt:27} })
5. (⋆) Find all *customers* whose age is ≤ 27. db.customer.find({'age':{$lte:27} })

***
db.customer.find({'age':{$lte:27}} ,{'name':1},{'age':0})
inclusion exclusion shouldn't be in same query

6. Find all *customers* whose age is ≤ 27. Display only name db.customer.find({'age':{$lte:27}} , {'name':1})
7. Find all *customers* whose age is ≤ 27. Display only name and age db.customer.find({'age': {$lte:27}} ,{'name':1,'age':1})

in between: companydb> db.customer.find({'age':{$lte:27,$gt:20}})

Nested documents

db.customer.find({'age':{$gte:22}}, {'name':1,'age':1})

1. Insert the following nested document into the collection *customers*

name		Harry
age		25
address	street	75, Bd. Saint Germain
	city	Paris
	Country	France

db.customer.insertOne({'name':'Harry','age':25,'address':{'street':'Paud Road','city':'Pune','state':'Maharastra','pin':411038}})

Use dots to keep going inside

2. Find all *customer documents* who live in *Paris*

db.customer.find({'address.city':'Dharwad'})

Update

1. Change age of the customer *Alice* to 29

db.customer.update({'name':'Alice'},{$set:{'age':29}})

2. (⋆)What happens if you do not use `$set` in the previous query?

db.customer.update({'name':'Alice'},{'age': 29})

Delete

deleteOne: Deletes first record
deleteMany: Deletes all records

1. Delete the document with *name: Bob.*

db.customer.deleteOne({'name':'Bob'})

2. (⋆) List all documents in the collection *customers* to verify the successful execution of the previous command.

db.customer.find()

3. Delete the collection *customers* from the database

db.customer.drop()

4. Delete the databse *companyDB* form the database

If all collections and data records are deleted then entire database is alos deleted

Samples

1. Load the sample data set onto the cluster

2. Explore the samle databases in your *mongo shell*

Exercise (Redis)

1. Insert the following keys and values with the appropriately specified data type for the values

Key	Value	Data type
`course:1:title`	Data Management	String
`course:1:NumberOfStudents`	3	Integer
`course:1:textbooks`	Fundamentals of Database Systems No SQL for Mere Mortals	Set

2. Add the following students with the key `course:1:students` where are values form an ordered sets. Use the grade points (given below) as the score.

Student Name	grade points
Alice	9.1
Bob	8.9
Charles	9.0

3. Retrieve all the student names in the key `course:1:students`
4. Find the size of the values with the key `course:1:textbooks`
5. Rename the key `course:1:textbooks` to `course:1:materials`
6. Add a new value *slides* to the key `course:1:materials`
7. Add a new key `course:1:assignment4` with value *Redis and MongoDB*
8. Set the expiration time for the key `course:1:handout10` to 100 seconds
9. List all the keys in the database
10. Delete the key `course:1:NumberOfStudents` along with its value.

Exercise (MongoDB)

1. List all the databases in the cluster — show dbs
2. List all the collections in the database `sample_mflix`
3. List the *id and name* of all the houses in `sample_airbnb` databse (there is only one collection) that are in *Australia*
4. List the *id and name* of all the houses in `sample_airbnb` databse (there is only one collection) that have 2 *or more bedrooms*
5. List the *id, name and address* of all the houses in `sample_airbnb` databse (there is only one collection) whose location is exact.

db.collection_name.find({'address.country':'Australia'},{"_id":1,"name":1}) :- Find the database.

db.listingsAndReviews.find({"address.country": "Australia","property_type": " House",'bedrooms':{$gt:2}},{"_id": 1,"name": 1})

db.listingsAndReviews.find({"property_type": "House","address.location.is_location_exact": true,},{"_id": 1,"name": 1,"address":1})