**Session 2**

**31/03/2020**

**Objectives and tasks for the session:**

1. **Install Mongodb Compass and read it’s documentation**
2. **Repeat previous exercise of populating the db**

**Tasks completed:**

use Prelimnary\_database

db.createUser(

{

user: "craysri06",

pwd: passwordPrompt(), // or cleartext password

roles: [

{ role: "readWrite", db: "Prelimnary\_database" }

]

}

)

Install MongoDB Compass

<https://www.mongodb.com/try/download/compass>

Select Platform as Ubuntu 64-bit

Download and run deb Package

> use test

switched to db test

**//creates the db**

> show collections;

> db.help()

**//list the methods**

DB methods:

db.auth(username, password)

db.cloneDatabase(fromhost) - will only function with MongoDB 4.0 and below

db.commandHelp(name) returns the help for the command

db.copyDatabase(fromdb, todb, fromhost) - will only function with MongoDB 4.0 and below

db.createCollection(name, {size: ..., capped: ..., max: ...})

db.createUser(userDocument)

db.createView(name, viewOn, [{$operator: {...}}, ...], {viewOptions})

db.currentOp() displays currently executing operations in the db

db.dropDatabase(writeConcern)

db.dropUser(username)

db.eval() - deprecated

db.fsyncLock() flush data to disk and lock server for backups

db.fsyncUnlock() unlocks server following a db.fsyncLock()

db.getCollection(cname) same as db['cname'] or db.cname

db.getCollectionNames()

db.getLastError() - just returns the err msg string

db.getLastErrorObj() - return full status object

db.getLogComponents()

db.getMongo() get the server connection object

db.getMongo().setSecondaryOk() allow queries on a replication secondary server

db.getName()

db.getProfilingLevel() - deprecated

db.getProfilingStatus() - returns if profiling is on and slow threshold

db.getReplicationInfo()

db.getSiblingDB(name) get the db at the same server as this one

db.hostInfo() get details about the server's host

db.isMaster() check replica primary status

db.hello() check replica primary status

db.killOp(opid) kills the current operation in the db

db.listCommands() lists all the db commands

db.loadServerScripts() loads all the scripts in db.system.js

db.logout()

db.printCollectionStats()

db.printReplicationInfo()

db.printShardingStatus()

db.printSecondaryReplicationInfo()

db.resetError()

db.serverStatus()

db.setLogLevel(level,<component>)

db.setProfilingLevel(level,slowms) 0=off 1=slow 2=all

db.setVerboseShell(flag) display extra information in shell output

db.setWriteConcern(<write concern doc>) - sets the write concern for writes to the db

db.shutdownServer()

db.stats()

db.unsetWriteConcern(<write concern doc>) - unsets the write concern for writes to the db

db.version() current version of the server

**//Insert and implicitly create collection**

> show collections;

testcollection

> db.testcollection;

test.testcollection

> db.testcollection.find();

{ "\_id" : ObjectId("60643ca0b66c8781d3d48661"), "a" : "12" }

> db.testcollection.find().pretty();

{ "\_id" : ObjectId("60643ca0b66c8781d3d48661"), "a" : "12" }

> db.testcollection.find().pretty();

{ "\_id" : ObjectId("60643ca0b66c8781d3d48661"), "a" : "12" }

> db.testcollection.remove({a:"12"});

WriteResult({ "nRemoved" : 1 })

> db.testcollection.insert({a:"10"});

WriteResult({ "nInserted" : 1 })

> db.testcollection.insert({a:"15"});

WriteResult({ "nInserted" : 1 })

> db.testcollection.insert({a:"20"});

WriteResult({ "nInserted" : 1 })

> db.testcollection.find();

{ "\_id" : ObjectId("60643dfeb66c8781d3d48662"), "a" : "10" }

{ "\_id" : ObjectId("60643e0bb66c8781d3d48663"), "a" : "15" }

{ "\_id" : ObjectId("60643e0eb66c8781d3d48664"), "a" : "20" }

> db.testcollection.remove({a:"12"});

WriteResult({ "nRemoved" : 0 })

> db.testcollection.remove({a:"15"});

WriteResult({ "nRemoved" : 1 })

> db.testcollection.find();

{ "\_id" : ObjectId("60643dfeb66c8781d3d48662"), "a" : "10" }

{ "\_id" : ObjectId("60643e0eb66c8781d3d48664"), "a" : "20" }

> db.testcollection.insert({a:"20", b:"xyz", c:456});

WriteResult({ "nInserted" : 1 })

> db.testcollection.find();

{ "\_id" : ObjectId("60643dfeb66c8781d3d48662"), "a" : "10" }

{ "\_id" : ObjectId("60643e0eb66c8781d3d48664"), "a" : "20" }

{ "\_id" : ObjectId("60643e92b66c8781d3d48665"), "a" : "20", "b" : "xyz", "c" : 456 }

**//Use to have indented output**

db.testcollection.find().pretty();

{ "\_id" : ObjectId("60643dfeb66c8781d3d48662"), "a" : "10" }

{ "\_id" : ObjectId("60643e0eb66c8781d3d48664"), "a" : "20" }

{

"\_id" : ObjectId("60643e92b66c8781d3d48665"),

"a" : "2",

"b" : "xyz",

"c" : 456

}