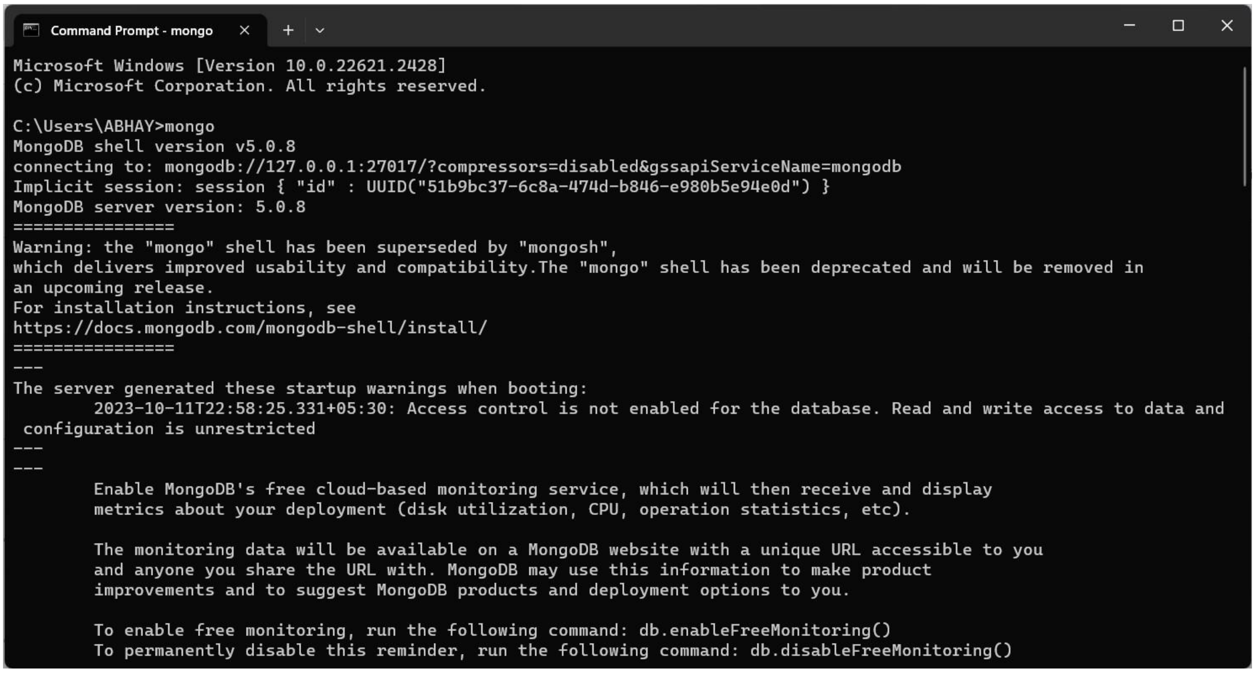
EXPERIMENT - 6

AIM:- To install and configure MongoDB and execute NoSQL commands

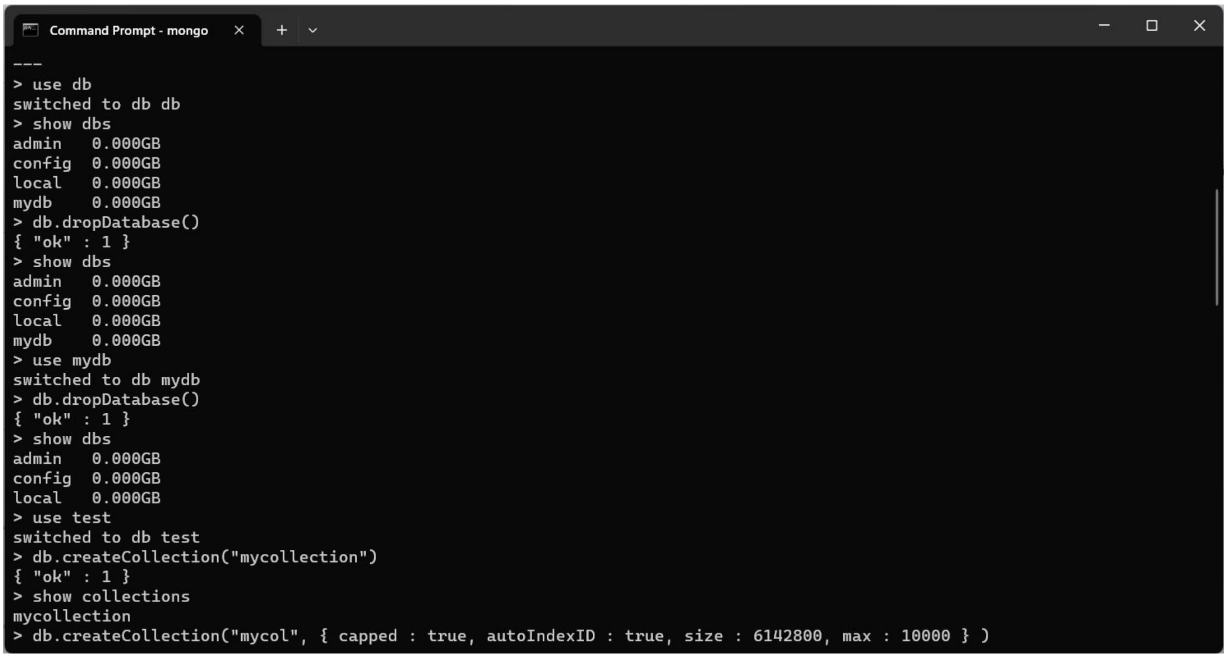
COMMANDS:-

To check the mongoDB installation

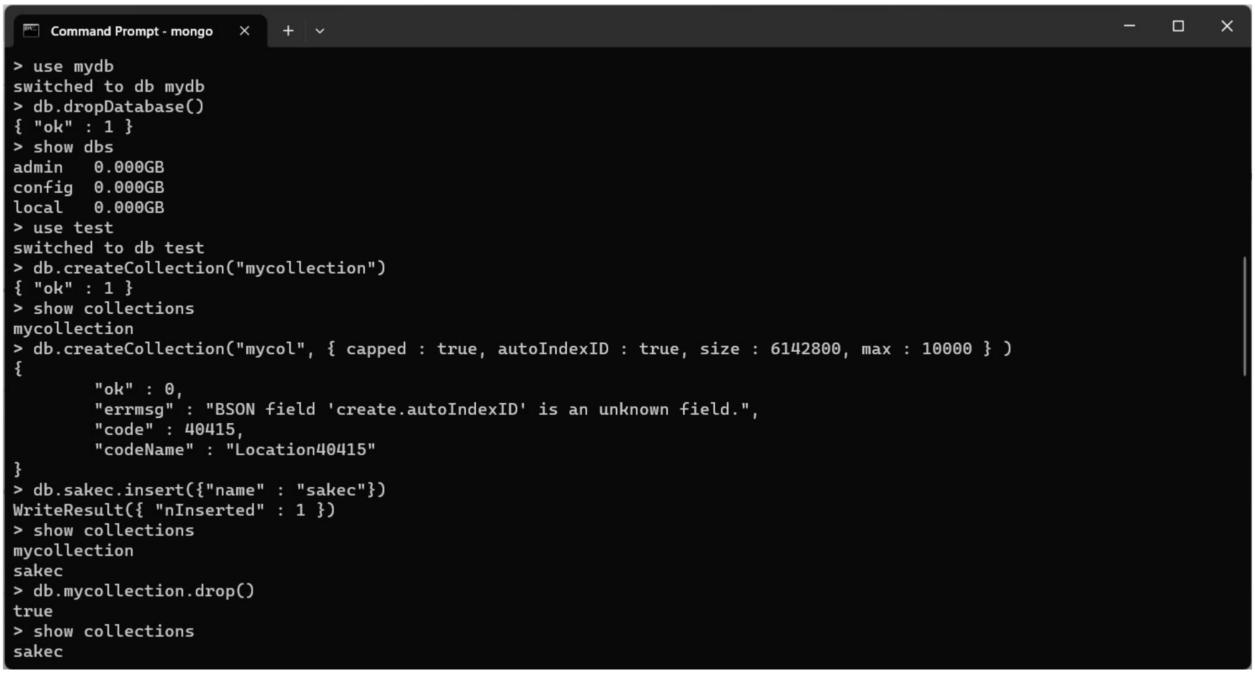


```
Microsoft Windows [Version 10.0.22621.2428]
(c) Microsoft Corporation. All rights reserved.

C:\Users\ABHAY>mongo
MongoDB shell version v5.0.8
connecting to: mongodb://127.0.0.1:27017/?compressors=disabled&gssapiServiceName=mongodb
Implicit session: session { "id" : UUID("51b9bc37-6c8a-474d-b846-e980b5e94e0d") }
MongoDB server version: 5.0.8
================
Warning: the "mongo" shell has been superseded by "mongosh",
which delivers improved usability and compatibility.The "mongo" shell has been deprecated and will be removed in
an upcoming release.
For installation instructions, see
https://docs.mongodb.com/mongodb-shell/install/
================
---
The server generated these startup warnings when booting:
        2023-10-11T22:58:25.331+05:30: Access control is not enabled for the database. Read and write access to data and
 configuration is unrestricted
---
---
        Enable MongoDB's free cloud-based monitoring service, which will then receive and display
        metrics about your deployment (disk utilization, CPU, operation statistics, etc).

        The monitoring data will be available on a MongoDB website with a unique URL accessible to you
        and anyone you share the URL with. MongoDB may use this information to make product
        improvements and to suggest MongoDB products and deployment options to you.

        To enable free monitoring, run the following command: db.enableFreeMonitoring()
        To permanently disable this reminder, run the following command: db.disableFreeMonitoring()
```

To create a database , collections, dropping the database and collections

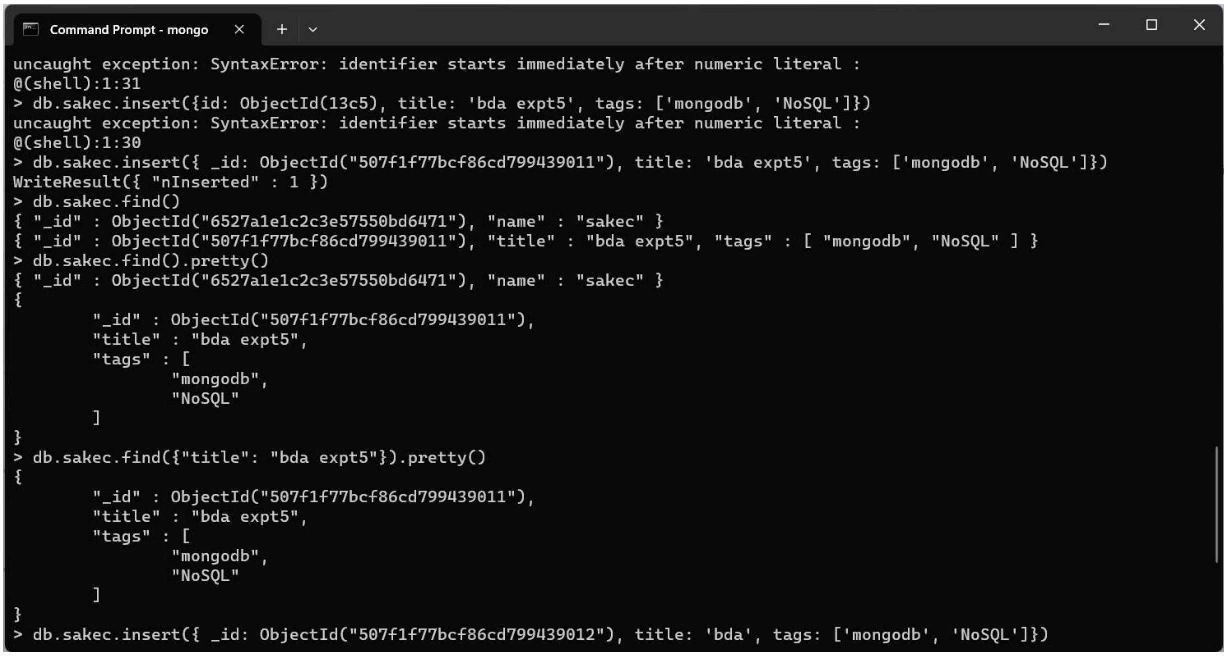


```
---
> use db
switched to db db
> show dbs
admin   0.000GB
config  0.000GB
local   0.000GB
mydb    0.000GB
> db.dropDatabase()
{ "ok" : 1 }
> show dbs
admin   0.000GB
config  0.000GB
local   0.000GB
mydb    0.000GB
> use mydb
switched to db mydb
> db.dropDatabase()
{ "ok" : 1 }
> show dbs
admin   0.000GB
config  0.000GB
local   0.000GB
> use test
switched to db test
> db.createCollection("mycollection")
{ "ok" : 1 }
> show collections
mycollection
> db.createCollection("mycol", { capped : true, autoIndexID : true, size : 6142800, max : 10000 } )
```

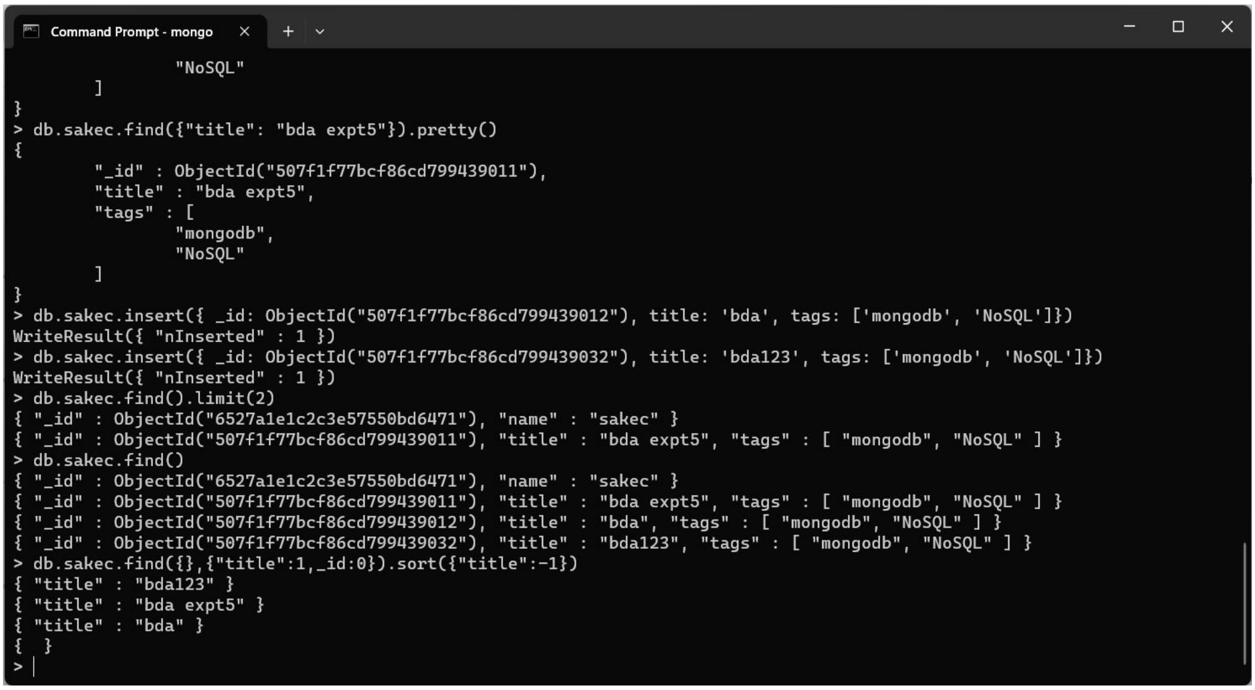


```
> use mydb
switched to db mydb
> db.dropDatabase()
{ "ok" : 1 }
> show dbs
admin   0.000GB
config  0.000GB
local   0.000GB
> use test
switched to db test
> db.createCollection("mycollection")
{ "ok" : 1 }
> show collections
mycollection
> db.createCollection("mycol", { capped : true, autoIndexID : true, size : 6142800, max : 10000 } )
{
        "ok" : 0,
        "errmsg" : "BSON field 'create.autoIndexID' is an unknown field.",
        "code" : 40415,
        "codeName" : "Location40415"
}
> db.sakec.insert({"name" : "sakec"})
WriteResult({ "nInserted" : 1 })
> show collections
mycollection
sakec
> db.mycollection.drop()
true
> show collections
sakec
```

To insert data in collection named ‘sakec’ and use sorting and find commands to access inserted data



```
uncaught exception: SyntaxError: identifier starts immediately after numeric literal :
@(shell):1:31
> db.sakec.insert({id: ObjectId(13c5), title: 'bda expt5', tags: ['mongodb', 'NoSQL']})
uncaught exception: SyntaxError: identifier starts immediately after numeric literal :
@(shell):1:30
> db.sakec.insert({ _id: ObjectId("507f1f77bcf86cd799439011"), title: 'bda expt5', tags: ['mongodb', 'NoSQL']})
WriteResult({ "nInserted" : 1 })
> db.sakec.find()
{ "_id" : ObjectId("6527a1e1c2c3e57550bd6471"), "name" : "sakec" }
{ "_id" : ObjectId("507f1f77bcf86cd799439011"), "title" : "bda expt5", "tags" : [ "mongodb", "NoSQL" ] }
> db.sakec.find().pretty()
{ "_id" : ObjectId("6527a1e1c2c3e57550bd6471"), "name" : "sakec" }
{
        "_id" : ObjectId("507f1f77bcf86cd799439011"),
        "title" : "bda expt5",
        "tags" : [
                "mongodb",
                "NoSQL"
        ]
}
> db.sakec.find({"title": "bda expt5"}).pretty()
{
        "_id" : ObjectId("507f1f77bcf86cd799439011"),
        "title" : "bda expt5",
        "tags" : [
                "mongodb",
                "NoSQL"
        ]
}
> db.sakec.insert({ _id: ObjectId("507f1f77bcf86cd799439012"), title: 'bda', tags: ['mongodb', 'NoSQL']})
```



```
                "NoSQL"
        ]
}
> db.sakec.find({"title": "bda expt5"}).pretty()
{
        "_id" : ObjectId("507f1f77bcf86cd799439011"),
        "title" : "bda expt5",
        "tags" : [
                "mongodb",
                "NoSQL"
        ]
}
> db.sakec.insert({ _id: ObjectId("507f1f77bcf86cd799439012"), title: 'bda', tags: ['mongodb', 'NoSQL']})
WriteResult({ "nInserted" : 1 })
> db.sakec.insert({ _id: ObjectId("507f1f77bcf86cd799439032"), title: 'bda123', tags: ['mongodb', 'NoSQL']})
WriteResult({ "nInserted" : 1 })
> db.sakec.find().limit(2)
{ "_id" : ObjectId("6527a1e1c2c3e57550bd6471"), "name" : "sakec" }
{ "_id" : ObjectId("507f1f77bcf86cd799439011"), "title" : "bda expt5", "tags" : [ "mongodb", "NoSQL" ] }
> db.sakec.find()
{ "_id" : ObjectId("6527a1e1c2c3e57550bd6471"), "name" : "sakec" }
{ "_id" : ObjectId("507f1f77bcf86cd799439011"), "title" : "bda expt5", "tags" : [ "mongodb", "NoSQL" ] }
{ "_id" : ObjectId("507f1f77bcf86cd799439012"), "title" : "bda", "tags" : [ "mongodb", "NoSQL" ] }
{ "_id" : ObjectId("507f1f77bcf86cd799439032"), "title" : "bda123", "tags" : [ "mongodb", "NoSQL" ] }
> db.sakec.find({},{"title":1,_id:0}).sort({"title":-1})
{ "title" : "bda123" }
{ "title" : "bda expt5" }
{ "title" : "bda" }
{  }
>
```