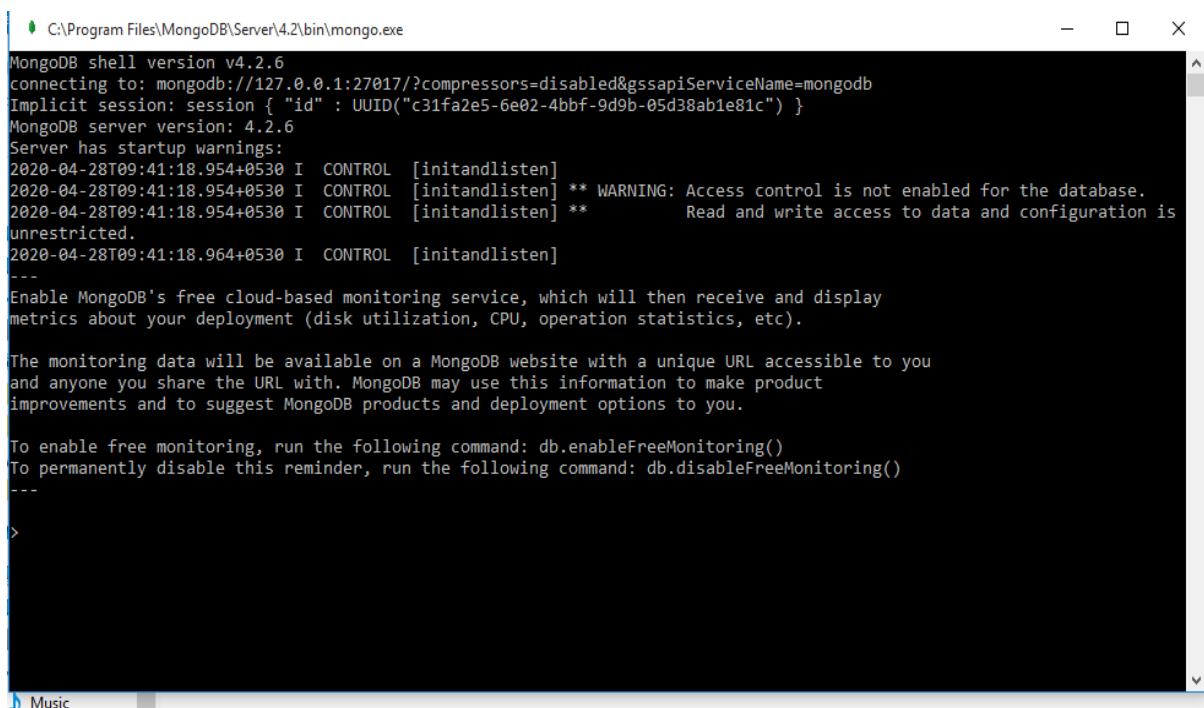


Commands in MongoDB

- All the commands will be written in MongoDB shell. Alternatively the operations may be executed in Compass GUI. We are using commands in shell.
- OPEN mongo shell by using the shell application file **mongo.exe** from at C:\Program Files\MongoDB\Server\4.2\bin (or the directory you have specified)
- Mongo screen will open



```
C:\Program Files\MongoDB\Server\4.2\bin\mongo.exe
MongoDB shell version v4.2.6
connecting to: mongodb://127.0.0.1:27017/?compressors=disabled&gssapiServiceName=mongodb
Implicit session: session { "id" : UUID("c31fa2e5-6e02-4bbf-9d9b-05d38ab1e81c") }
MongoDB server version: 4.2.6
Server has startup warnings:
2020-04-28T09:41:18.954+0530 I CONTROL [initandlisten]
2020-04-28T09:41:18.954+0530 I CONTROL [initandlisten] ** WARNING: Access control is not enabled for the database.
2020-04-28T09:41:18.954+0530 I CONTROL [initandlisten] **          Read and write access to data and configuration is
unrestricted.
2020-04-28T09:41:18.964+0530 I CONTROL [initandlisten]
---
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()
---
>
```

- All commands can be written and executed on this MongoDB Shell prompt.

1. Create database

use **DATABASE_NAME** command is used to create database. This will open/ or switch the specified database if exists otherwise will create a database with the given name and switch to that database.

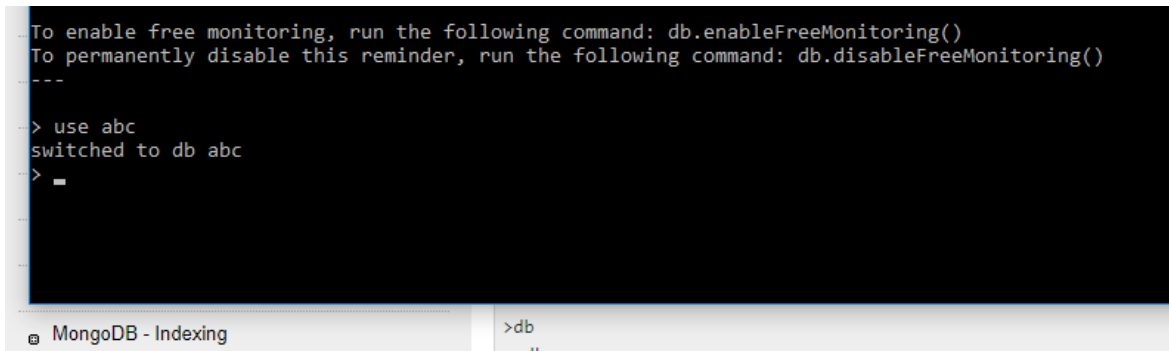
Syntax

use **DATABASE_NAME**

Example:

Use abc

```
To enable free monitoring, run the following command: db.enableFreeMonitoring()
To permanently disable this reminder, run the following command: db.disableFreeMonitoring()
---
> use abc
switched to db abc
>
_
```



2. Show database (show dbs) command

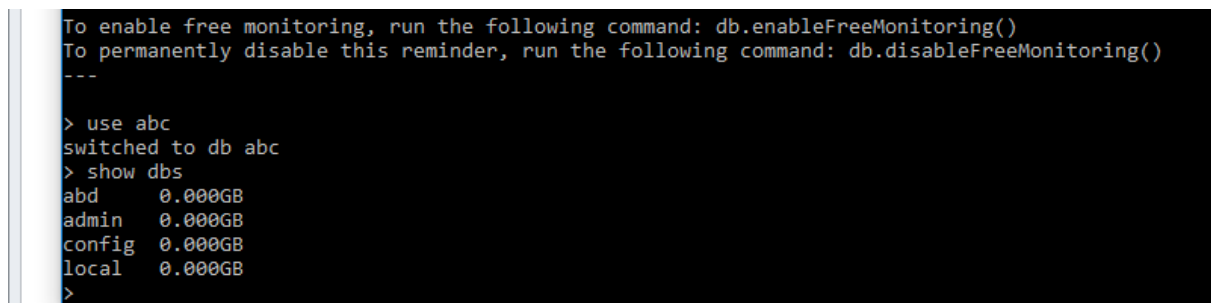
To display the list of databases, show command is used,

Syntax

Show dbs

- This will list all the non-empty databases.
- To display database, you need to insert at least one document into it.
- Just created database (abc) is not present in list shown below as it is not having any document in it.

```
To enable free monitoring, run the following command: db.enableFreeMonitoring()
To permanently disable this reminder, run the following command: db.disableFreeMonitoring()
---
> use abc
switched to db abc
> show dbs
abd    0.000GB
admin  0.000GB
config 0.000GB
local  0.000GB
>
```



3. Create Collection command

createCollection() method is used to create collection in the selected database.

Syntax

db.createCollection(name, options)

Parameter	Type	Description
-----------	------	-------------

Name	String	Name parameter specifies the name of the collection to be created. It is of String type and should be specified in quotes.
Options (it is optional parameter)	Document	Options parameter is used to specify options about memory size and indexing. This parameter is optional.

Example

To run this method / command database must be in use. We are using our newly created **abc** database.

db.createCollection("college")

this will result status as

```
{ "ok" : 1 }
```

```
> db.createCollection(college)
2020-05-04T10:42:41.353+0530 E QUERY [js] uncaught exception: ReferenceError: college is not defined :
@(shell):1:1
> db.createCollection("college")
{ "ok" : 1 }
>
```

Note: In the above screen, 1st we write collection name without quotes **db.createcollection(college)** which is wrong and in that case some exceptions has been raised.

In MongoDB, collection creation is not necessary before inserting document. MongoDB creates collection automatically, when you insert some document.

4. Show collections command

Show collections is used to display / list the created collection in the selected database.

Syntax

Show collections

If we run this command, it will display our recently created collection “college”.

```
> show collections
college
> db.insert("name", "NITESH")
```

5. Insert() command

Insert command is used to insert document in form of name/value combination to the specified collection. If collection is not already created, it will create the specified collection.

Syntax

db.COLLECTION_NAME.insert (document)

where

- COLLECTION_NAME** is the name of the collection in which document has to be inserted.
- document** is document(key/value combination) or array of documents to insert into the collection.

_id Field

- We may give an **_id** field while inserting the document and this shall be unique within the collection to avoid duplicate key error.
- If no **_id** field is specified, then MongoDB will add the **_id** field and assign a unique **ObjectId** to the document.

Example-1:

```
db.college.insert({"name": "RCC"})
```

This will show the following result after successful execution of the command:

```
WriteResult({ "nInserted" : 1 })
```

```
@(shell).1.1
> db.college.insert({name:"RCC"})
WriteResult({ "nInserted" : 1 })
> show collections
abc
college
```

Example-2:

```
db.college.insert({course:"A level",duration:"2 years",fees:"12000"})
```

This will show the following result after successful execution of the command:

```
WriteResult({ "nInserted" : 1 })
```

```
college
> db.college.insert({course:"A level",duration:"2 years",fees:"12000"})
WriteResult({ "nInserted" : 1 })
```

Example-3: Lets insert a document into a collection which has not been created. For example let collection name as **school**.

```
db.school.insert({"class": "10th", "section": "A section", "student count": "39"})
```

Once this command is executed, a new collection named as school will be automatically created.

```
C:\Program Files\MongoDB\Server\4.2\bin\mongo.exe
> show collections
abc
college
old
> db.school.insert({"class": "10th", "section": "A section", "student count": "39"})
WriteResult({ "nInserted" : 1 })
> show collections
abc
college
old
school
>
```

Example-4: Lets insert a record with specified `_id`

```
db.school.insert({"_id": "01", "class": "10th", "section": "B section", "student count": "27"})
```

```
> db.school.find()
{ "_id" : ObjectId("5eb0f690ffe73fbdd0e2b1e5"), "class" : "10th", "section" : "A section", "student count" : "39" }
>
> db.school.insert({"_id":"01","class":"10th","section":"B Section","student count":"27"})
WriteResult({ "nInserted" : 1 })
>
> db.school.find()
{ "_id" : ObjectId("5eb0f690ffe73fbdd0e2b1e5"), "class" : "10th", "section" : "A section", "student count" : "39" }
{ "_id" : "01", "class" : "10th", "section" : "B Section", "student count" : "27" }
>
```

Now a record with `_id` as `01` (specified in insert command) has been inserted in the collection. Find command used here has been described below.

6. Find() command

Find () command is used to see the records/documents in the specified collection.

Syntax

```
db.COLLECTION_NAME.find()
```

Example:

```
db.college.find()
```

```
> db.college.find()
{ "_id" : ObjectId("5eb0ee88ffe73fbdd0e2b1e1"), "name" : "RCC" }
{ "_id" : ObjectId("5eb0eec6ffe73fbdd0e2b1e2"), "course" : "A level", "duration" : "2 years", "fees" : "12000" }
>
```

7. Save() command

Save() command may also be used to insert the document. It has the same syntax as of insert() command.

Syntax

db.college.save(document)

If **_id** is not specified in the document then **save()** method will work same as **insert()** method. If **_id** filed is specified then it will replace whole data of document having the **_id** as specified in save() method if matched else Insert the document into the collection.

Example-1

Lets save a record in the school collection with **_id** as **02**, and then display the documents in the collection.

```
db.school.save({"_id":"02","class":"10th","section":"C Section","student count":"37"})
```

this will display

```
WriteResult({ "nMatched" : 0, "nUpserted" : 1, "nModified" : 0, "_id" : "02" })
```

*It will display details like "nMatched" : 0, i.e. no record is matched with the specified **_id**, so a new record is "nUpserted" : 1 with , "_id" : "02".*

```
> db.school.find()
{ "_id" : ObjectId("5eb0f690ffe73fbdd0e2b1e5"), "class" : "10th", "section" : "A sction", "student count" : "39" }
{ "_id" : "01", "class" : "10th", "section" : "B Section", "student count" : "27" }
>
> db.school.save({"_id":"02","class":"10th","section":"C Section","student count":"37"})
WriteResult({ "nMatched" : 0, "nUpserted" : 1, "nModified" : 0, "_id" : "02" })
>
> db.school.find()
{ "_id" : ObjectId("5eb0f690ffe73fbdd0e2b1e5"), "class" : "10th", "section" : "A sction", "student count" : "39" }
{ "_id" : "01", "class" : "10th", "section" : "B Section", "student count" : "27" }
{ "_id" : "02", "class" : "10th", "section" : "C Section", "student count" : "37" }
```

Example-2

Lets try to save a record in the school collection with **_id** as **01**, and then display the documents in the collection.

```
db.school.save({"_id":"01","class":"10th","section":"D Section","student count":"29"})
```

```
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
```

*It will display details like "nMatched" : 1 which matched with the specified **_id**, and the same is modified "nModified" : 1.*

```
> db.school.find()
{ "_id" : ObjectId("5eb0f690ffe73fbdd0e2b1e5"), "class" : "10th", "section" : "A section", "student count" : "39" }
{ "_id" : "01", "class" : "10th", "section" : "B Section", "student count" : "27" }
{ "_id" : "02", "class" : "10th", "section" : "C Section", "student count" : "37" }
>
> db.school.save({"_id":"01","class":"10th","section":"D Section","student count":"29"})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
>
> db.school.find()
{ "_id" : ObjectId("5eb0f690ffe73fbdd0e2b1e5"), "class" : "10th", "section" : "A section", "student count" : "39" }
{ "_id" : "01", "class" : "10th", "section" : "D Section", "student count" : "29" }
{ "_id" : "02", "class" : "10th", "section" : "C Section", "student count" : "37" }
>
```

Assignment

1. How to create database in mongoDB?
2. How to insert documents in mongoDB?
3. What is the difference between Save and Insert?