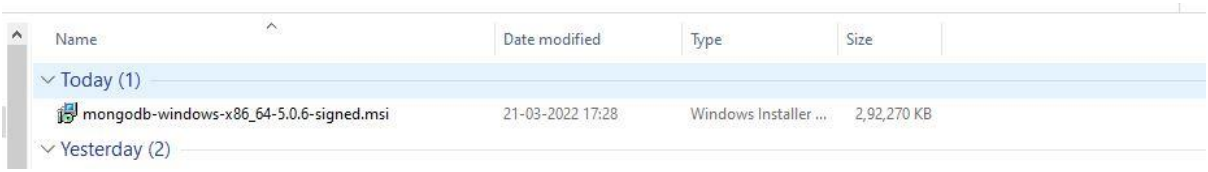
BIBD MINI PROJECT

Aim: Implementation of nosql database in mongodb.

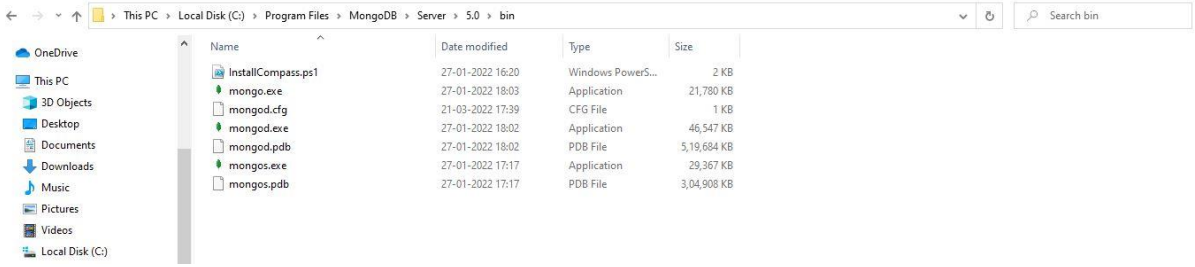
Steps:

We first need to install MongoDB in our pc. We will do so by installing the installer.



Name	Date modified	Type	Size
Today (1)			
mongodb-windows-x86_64-5.0.6-signed.msi	21-03-2022 17:28	Windows Installer ...	2,92,270 KB
Yesterday (2)			

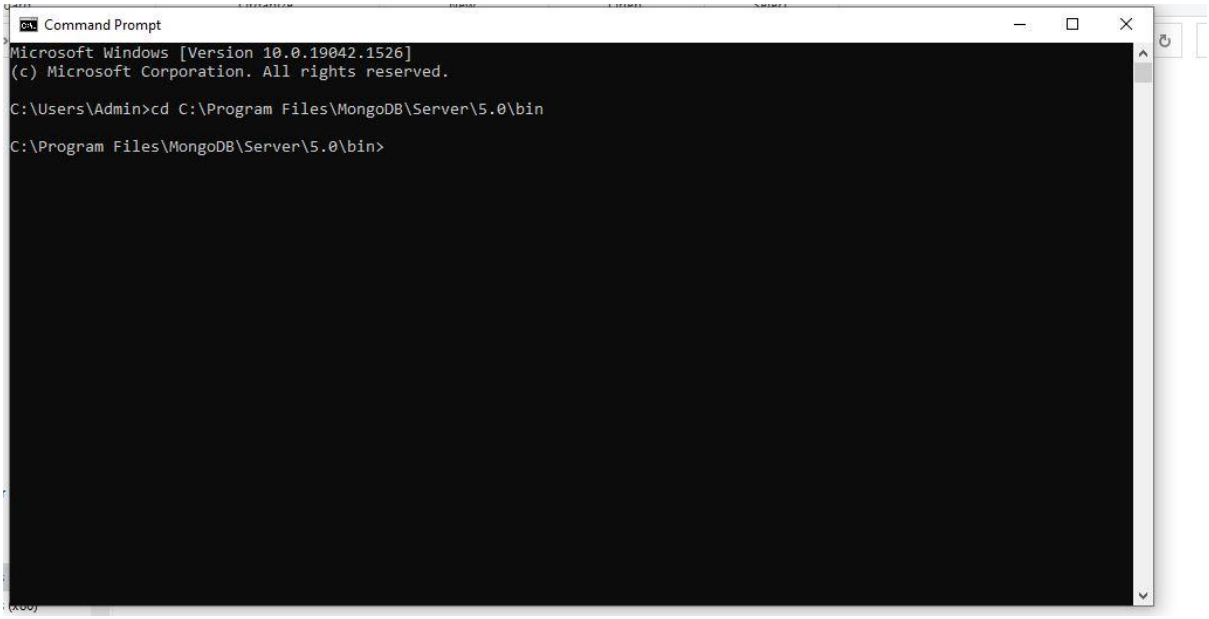
Then after installing installer successfully, we will run it on our pc and it will be downloaded successfully. In this case it is there on C drive.



This PC › Local Disk (C:) › Program Files › MongoDB › Server › 5.0 › bin

Name	Date modified	Type	Size
InstallCompass.ps1	27-01-2022 16:20	Windows PowerS...	2 KB
mongo.exe	27-01-2022 18:03	Application	21,780 KB
mongod.cfg	21-03-2022 17:39	CFG File	1 KB
mongod.exe	27-01-2022 18:02	Application	46,547 KB
mongod.pdb	27-01-2022 18:02	PDB File	5,19,684 KB
mongos.exe	27-01-2022 17:17	Application	29,367 KB
mongos.pdb	27-01-2022 17:17	PDB File	3,04,908 KB

After that we will open Command Prompt and we will assign the path folder on cmd.



```
Command Prompt
Microsoft Windows [Version 10.0.19042.1526]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Admin>cd C:\Program Files\MongoDB\Server\5.0\bin

C:\Program Files\MongoDB\Server\5.0\bin>
```

After this we will type command mongo
It is used to connect to the database server on localhost.

Start The MongoDB shell.

```
C:\Program Files\MongoDB\Server\5.0\bin\mongo.exe
MongoDB shell version v5.0.6
connecting to: mongodb://127.0.0.1:27017/?compressors=disabled&gssapiServiceName=mongodb
Implicit session: session { "id" : UUID("b74f20f2-e913-40af-a72a-4482315ba5e6") }
MongoDB server version: 5.0.6
================
Warning: the "mongo" shell has been superseded by "mongosh",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:
        2022-03-18T11:58:14.502+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()
---
>
```

Check for any existing databases.

```
> show dbs
admin   0.000GB
config  0.000GB
local   0.000GB
```

So, we do not have our own existing database, hence we'll create a new one.

```
> use school
switched to db school
> show dbs
admin   0.000GB
config  0.000GB
local   0.000GB
>
```

We've created a database named office here, but it is not displayed because its empty, so we need to create a collection first inside this database. To insert document into collection json format is followed.

```
> db.office.insertOne({Employee: "Anurag", Department: "IT"})
{
        "acknowledged" : true,
        "insertedId" : ObjectId("624c2c6296570e5d1032f19d")
}
>
```

Here, we've created a collection in the office database named Employee and added a department of Employee. So now if we check the databases on the system we can see the office database.

```
> show dbs
admin   0.000GB
config  0.000GB
local   0.000GB
office  0.000GB
>
```

Now, to check if the document is added in the collection we run If we want it in a more readable format we can use the pretty() function.

```
> db.office.find()
{ "_id" : ObjectId("624c2c6296570e5d1032f19d"), "Employee" : "Anurag", "Department" : "IT" }
> db.office.find().pretty()
{
        "_id" : ObjectId("624c2c6296570e5d1032f19d"),
        "Employee" : "Anurag",
        "Department" : "IT"
}
```

So, the document we inserted earlier is shown here. We know how to create a database. Now let's see how to delete/drop a database. Here, I've already created another sample database "sampledb" with document in it.

```
> use sampledb
switched to db sampledb
> db.test.insertOne({Name:"abc"})
{
        "acknowledged" : true,
        "insertedId" : ObjectId("623c4e455331ff28e5d5111c")
}
>
```

```
> show dbs
admin     0.000GB
config    0.000GB
local     0.000GB
sampledb  0.000GB
school    0.000GB
>
```

```
> use sampledb
switched to db sampledb
> db.dropDatabase()
{ "ok" : 1 }
> show dbs
admin   0.000GB
config  0.000GB
local   0.000GB
school  0.000GB
>
```

To drop a single collection, you can do as follows

```
> db.test.drop()
true
>
```

Inserting Data through the **insertOne** and **insertMany** commands:
insertOne Command insert only one data in one time and **insertMany Command** insert many data at one time

```
> db.employee.insertOne({name:"Ravi", surname:"Joshi", email:"ravi@gmail.com", address:{city:"Mumbai", location:"Andheri"}, hobbies:["Cricket","Music"]})
{
        "acknowledged" : true,
        "insertedId" : ObjectId("623c7ca45331ff28e5d5111f")
}
```

```
> db.employee.insertMany([{name:"Harsh", surname:"Rai", email:"harsh@hotmail.com", address:{city:"Banglore", location:"AMK"}, hobbies:["Football"]}, {name:"Sandy", surname:"Singh", email:"sandys@yahoo.co.in", ad
dress:{city:"Pune", location: "Hinjewadi"}, hobbies:["Reading", "Traveling"]}])
{
        "acknowledged" : true,
        "insertedIds" : [
                ObjectId("623c7e7e5331ff28e5d51122"),
                ObjectId("623c7e7e5331ff28e5d51123")
        ]
}
>
```

Let us now check the database.

```
> show dbs
admin    0.000GB
company  0.000GB
config   0.000GB
local    0.000GB
school   0.000GB
> use company
switched to db company
> show collections
employee
```

Here check the records/document we have updated in the collection employee

```
> db.employee.find().pretty()
{
        "_id" : ObjectId("623c7feb5331ff28e5d51124"),
        "name" : "Ravi",
        "surname" : "Joshi",
        "email" : "ravi@gmail.com",
        "address" : {
                "city" : "Mumbai",
                "location" : "Andheri"
        },
        "hobbies" : [
                "Cricket",
                "Music"
        ]
}
{
        "_id" : ObjectId("623c7ff45331ff28e5d51125"),
        "name" : "Harsh",
        "surname" : "Rai",
        "email" : "harsh@hotmail.com",
        "address" : {
                "city" : "Banglore",
                "location" : "AMK"
        },
        "hobbies" : [
                "Football"
        ]
}
{
        "_id" : ObjectId("623c7ff45331ff28e5d51126"),
        "name" : "Sandy",
        "surname" : "Singh",
        "email" : "sandys@yahoo.co.in",
        "address" : {
                "city" : "Pune",
                "location" : "Hinjewadi"
        },
        "hobbies" : [
                "Reading",
                "Traveling"
        ]
}
>
```

Here, we've successfully executed the **insertOne** and **insertMany** commands and also Read the data in the Document.

Now let's try updating the location of Ravi to Ghatkopar in the document.

```
> db.employee.updateOne({_id : ObjectId("623c7feb5331ff28e5d51124")}, {$set: {"address.location" : "Ghatkopar"}})
{ "acknowledged" : true, "matchedCount" : 1, "modifiedCount" : 0 }
```

Check if the value is updated

```
> db.employee.find().pretty()
{
        "_id" : ObjectId("623c7feb5331ff28e5d51124"),
        "name" : "Ravi",
        "surname" : "Joshi",
        "email" : "ravi@gmail.com",
        "address" : {
                "city" : "Mumbai",
                "location" : "Ghatkopar"
        },
        "hobbies" : [
                "Cricket",
                "Music"
        ]
}
{
        "_id" : ObjectId("623c7ff45331ff28e5d51125"),
        "name" : "Harsh",
        "surname" : "Rai",
        "email" : "harsh@hotmail.com",
        "address" : {
                "city" : "Banglore",
                "location" : "AMK"
        },
        "hobbies" : [
                "Football"
        ]
}
{
        "_id" : ObjectId("623c7ff45331ff28e5d51126"),
        "name" : "Sandy",
        "surname" : "Singh",
        "email" : "sandys@yahoo.co.in",
        "address" : {
                "city" : "Pune",
                "location" : "Hinjewadi"
        },
        "hobbies" : [
                "Reading",
                "Traveling"
        ]
}
>
```

Now lets try **updateMany** command

```
> db.employee.updateMany({}, {$set: {relationshipStatus: "unknown"}})
{ "acknowledged" : true, "matchedCount" : 3, "modifiedCount" : 3 }
```

Keeping the first parameter blank means updating all the entries.

```
> db.employee.find().pretty()
{
        "_id" : ObjectId("623c7feb5331ff28e5d51124"),
        "name" : "Ravi",
        "surname" : "Joshi",
        "email" : "ravi@gmail.com",
        "address" : {
                "city" : "Mumbai",
                "location" : "Ghatkopar"
        },
        "hobbies" : [
                "Cricket",
                "Music"
        ],
        "relationshipStatus" : "unknown"
}
{
        "_id" : ObjectId("623c7ff45331ff28e5d51125"),
        "name" : "Harsh",
        "surname" : "Rai",
        "email" : "harsh@hotmail.com",
        "address" : {
                "city" : "Banglore",
                "location" : "AMK"
        },
        "hobbies" : [
                "Football"
        ],
        "relationshipStatus" : "unknown"
}
{
        "_id" : ObjectId("623c7ff45331ff28e5d51126"),
        "name" : "Sandy",
        "surname" : "Singh",
        "email" : "sandys@yahoo.co.in",
        "address" : {
                "city" : "Pune",
                "location" : "Hinjewadi"
        },
        "hobbies" : [
                "Reading",
                "Traveling"
        ],
        "relationshipStatus" : "unknown"
}
>
```

Now let’s change status of one employee.

```
> db.employee.updateOne({_id : ObjectId("623c7ff45331ff28e5d51126")}, {$set: {"relationshipStatus" : "Married"}})
{ "acknowledged" : true, "matchedCount" : 1, "modifiedCount" : 1 }
```

```
> db.employee.find().pretty()
{
        "_id" : ObjectId("623c7feb5331ff28e5d51124"),
        "name" : "Ravi",
        "surname" : "Joshi",
        "email" : "ravi@gmail.com",
        "address" : {
                "city" : "Mumbai",
                "location" : "Ghatkopar"
        },
        "hobbies" : [
                "Cricket",
                "Music"
        ],
        "relationshipStatus" : "unknown"
}
{
        "_id" : ObjectId("623c7ff45331ff28e5d51125"),
        "name" : "Harsh",
        "surname" : "Rai",
        "email" : "harsh@hotmail.com",
        "address" : {
                "city" : "Banglore",
                "location" : "AMK"
        },
        "hobbies" : [
                "Football"
        ],
        "relationshipStatus" : "unknown"
}
{
        "_id" : ObjectId("623c7ff45331ff28e5d51126"),
        "name" : "Sandy",
        "surname" : "Singh",
        "email" : "sandys@yahoo.co.in",
        "address" : {
                "city" : "Pune",
                "location" : "Hinjewadi"
        },
        "hobbies" : [
                "Reading",
                "Traveling"
        ],
        "relationshipStatus" : "Married"
}
>
```

So now let’s delete an entry from employee using **deleteOne**() where relationship status is Married.

```
> db.employee.deleteOne({name: "Sandy"})
{ "acknowledged" : true, "deletedCount" : 1 }
```

```
> db.employee.find().pretty()
{
        "_id" : ObjectId("623c7feb5331ff28e5d51124"),
        "name" : "Ravi",
        "surname" : "Joshi",
        "email" : "ravi@gmail.com",
        "address" : {
                "city" : "Mumbai",
                "location" : "Ghatkopar"
        },
        "hobbies" : [
                "Cricket",
                "Music"
        ],
        "relationshipStatus" : "unknown"
}
{
        "_id" : ObjectId("623c7ff45331ff28e5d51125"),
        "name" : "Harsh",
        "surname" : "Rai",
        "email" : "harsh@hotmail.com",
        "address" : {
                "city" : "Banglore",
                "location" : "AMK"
        },
        "hobbies" : [
                "Football"
        ],
        "relationshipStatus" : "unknown"
}
> _
```

Now deleting users with **deleteMany**() operations where relationship Status is unknown.

```
> db.employee.deleteMany({relationshipStatus: "unknown"})
{ "acknowledged" : true, "deletedCount" : 2 }
> db.employee.find().pretty()
```

All records are deleted and hence we now have an empty collection.
To shutdown the server we will use..

```
> db.shutdownServer()
server should be down...
> exit
bye
```