

Microsoft Windows [Version 10.0.15063]

(c) 2017 Microsoft Corporation. All rights reserved.

C:\Users\19MCA44>mongo

MongoDB shell version v5.0.8

connecting to: mongodb://127.0.0.1:27017/?compressors=disabled&gssapiServiceName=mongodb

Implicit session: session { "id" : UUID("722c44e4-bc15-49cd-a7f6-e2e9f7d55965") }

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Warning: the "mongo" shell has been superseded by "mongosh",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:

2022-06-08T10:09:58.610+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()`

---

> show database

uncaught exception: Error: don't know how to show [database] :

shellHelper.show@src/mongo/shell/utils.js:1211:11

shellHelper@src/mongo/shell/utils.js:838:15

@(shellhelp2):1:1

> show dbs

admin 0.000GB

candidate 0.000GB

college 0.000GB

config 0.000GB

emp 0.000GB

employees 0.000GB

local 0.000GB

> use college

switched to db college

> show users

> db.createUser("bincy")

uncaught exception: Error: no 'user' field provided to 'createUser' function :

DB.prototype.createUser@src/mongo/shell/db.js:1335:15

```
@(shell):1:1
```

```
> db.createUser({"bincy"})
```

```
uncaught exception: SyntaxError: missing : after property id :
```

```
@(shell):1:22
```

```
> db.createuser({user:"bincy",pwd:"123",roles:[{role:"readWrite",db:"student"}]})
```

```
uncaught exception: TypeError: db.createuser is not a function :
```

```
@(shell):1:1
```

```
> db.createUser({user:"bincy",pwd:"123",roles:[{role:"readWrite",db:"student"}]})
```

```
Successfully added user: {
```

```
  "user" : "bincy",
```

```
  "roles" : [
```

```
    {
```

```
      "role" : "readWrite",
```

```
      "db" : "student"
```

```
    }
```

```
  ]
```

```
}
```

```
> show dbs
```

```
admin      0.000GB
```

```
candidate  0.000GB
```

```
college    0.000GB
```

```
config     0.000GB
```

```
emp        0.000GB
```

```
employees  0.000GB
```

```
local      0.000GB
```

```
> db.auth("bincy","123")
```

```
1
```

```
> show rows
```

```
uncaught exception: Error: don't know how to show [rows] :
```

```
shellHelper.show@src/mongo/shell/utils.js:1211:11
```

```
shellHelper@src/mongo/shell/utils.js:838:15
```

```
@(shellhelp2):1:1
```

```
> db.showUsers
```

```
college.showUsers
```

```
> show roles
```

```
{
```

```
  "role" : "read",
```

```
  "db" : "college",
```

```
  "isBuiltin" : true,
```

```
  "roles" : [ ],
```

```
  "inheritedRoles" : [ ]
```

```
}
```

```
{
```

```
  "role" : "enableSharding",
```

```
  "db" : "college",
```

```
  "isBuiltin" : true,
```

```
  "roles" : [ ],
```

```
  "inheritedRoles" : [ ]
```

```
}
```

```
{
```

```
    "role" : "readWrite",  
    "db" : "college",  
    "isBuiltin" : true,  
    "roles" : [ ],  
    "inheritedRoles" : [ ]
```

```
}
```

```
{
```

```
    "role" : "dbAdmin",  
    "db" : "college",  
    "isBuiltin" : true,  
    "roles" : [ ],  
    "inheritedRoles" : [ ]
```

```
}
```

```
{
```

```
    "role" : "dbOwner",  
    "db" : "college",  
    "isBuiltin" : true,  
    "roles" : [ ],  
    "inheritedRoles" : [ ]
```

```
}
```

```
{
```

```
    "role" : "userAdmin",  
    "db" : "college",  
    "isBuiltin" : true,  
    "roles" : [ ],
```

```
"inheritedRoles" : [ ]  
  
}
```



Microsoft Windows [Version 10.0.15063]

(c) 2017 Microsoft Corporation. All rights reserved.

C:\Users\19MCA44>mongo

MongoDB shell version v5.0.8

connecting to: mongodb://127.0.0.1:27017/?compressors=disabled&gssapiServiceName=mongodb

Implicit session: session { "id" : UUID("722c44e4-bc15-49cd-a7f6-e2e9f7d55965") }

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:



2022-06-08T10:09:58.610+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()`

---

> show database

uncaught exception: Error: don't know how to show [database] :

shellHelper.show@src/mongo/shell/utils.js:1211:11

shellHelper@src/mongo/shell/utils.js:838:15

@(shellhelp2):1:1

> show dbs

admin        0.000GB

candidate   0.000GB

college     0.000GB

config      0.000GB

emp         0.000GB

employees 0.000GB

local 0.000GB

> use college

switched to db college

> show users

> db.createUser("bincy")

uncaught exception: Error: no 'user' field provided to 'createUser' function :

DB.prototype.createUser@src/mongo/shell/db.js:1335:15

@(shell):1:1

> db.createUser({"bincy"})

uncaught exception: SyntaxError: missing : after property id :

@(shell):1:22

> db.createuser({user:"bincy",pwd:"123",roles:[{role:"readWrite",db:"student"}]})

uncaught exception: TypeError: db.createuser is not a function :

@(shell):1:1

> db.createUser({user:"bincy",pwd:"123",roles:[{role:"readWrite",db:"student"}]})

Successfully added user: {

"user" : "bincy",

"roles" : [

{

"role" : "readWrite",

"db" : "student"

}

]

}

```
> show dbs
```

```
admin      0.000GB
```

```
candidate  0.000GB
```

```
college    0.000GB
```

```
config     0.000GB
```

```
emp        0.000GB
```

```
employees  0.000GB
```

```
local      0.000GB
```

```
> db.auth("bincy","123")
```

```
1
```

```
> show rows
```

```
uncaught exception: Error: don't know how to show [rows] :
```

```
shellHelper.show@src/mongo/shell/utils.js:1211:11
```

```
shellHelper@src/mongo/shell/utils.js:838:15
```

```
@(shellhelp2):1:1
```

```
> db.showUsers
```

```
college.showUsers
```

```
> show roles
```

```
{  
  "role" : "read",  
  "db" : "college",  
  "isBuiltin" : true,  
  "roles" : [ ],  
  "inheritedRoles" : [ ]  
}
```

```
{  
  "role" : "enableSharding",  
  "db" : "college",  
  "isBuiltin" : true,  
  "roles" : [ ],  
  "inheritedRoles" : [ ]  
}
```

```
{  
  "role" : "readWrite",  
  "db" : "college",  
  "isBuiltin" : true,  
  "roles" : [ ],  
  "inheritedRoles" : [ ]  
}
```

```
{  
  "role" : "dbAdmin",  
  "db" : "college",  
  "isBuiltin" : true,  
  "roles" : [ ],  
  "inheritedRoles" : [ ]  
}
```

```
{  
  "role" : "dbOwner",  
  "db" : "college",  
  "isBuiltin" : true,  
  "roles" : [ ],  
  "inheritedRoles" : [ ]  
}
```

```
    "roles" : [ ],
    "inheritedRoles" : [ ]
  }
{
  "role" : "userAdmin",
  "db" : "college",
  "isBuiltin" : true,
  "roles" : [ ],
  "inheritedRoles" : [ ]
}
```

> show databases

```
admin      0.000GB
candidate  0.000GB
college    0.000GB
config     0.000GB
emp        0.000GB
employees  0.000GB
local      0.000GB
```

> use school

switched to db school

> db.createCollection("students")

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

> db.school.insert({regno:"101",name:"Bincy",mark:[chem:"80",phy:"78",bio:"90"]})

uncaught exception: SyntaxError: missing ] after element list :

@(shell):1:53

```
> db.students.insert({regno:"101",name:"Bincy",mark:[chem:"80",phy:"78",bio:"90"]})
```

uncaught exception: SyntaxError: missing ] after element list :

```
@(shell):1:55
```

```
> db.students.insert({regno:"101",name:"Bincy",mark:[{chem:"80"},{phy:"78"},{bio:"90"}]})
```

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

```
> db.students.insert({regno:"102",name:"sneha",mark:[{chem:"90"},{phy:"88"},{bio:"80"}]})
```

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

```
> db.students.insert({regno:"103",name:"Diya",mark:[{chem:"97"},{phy:"88"},{bio:"87"}]})
```

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

```
> db.students.find().pretty()
```

```
{
  "_id" : ObjectId("62a06c6671761a55fd7551f9"),
  "regno" : "101",
  "name" : "Bincy",
  "mark" : [
    {
      "chem" : "80"
    },
    {
      "phy" : "78"
    },
    {
      "bio" : "90"
    }
  ]
}
```

```
}  
  
{  
  
  "_id" : ObjectId("62a06c9c71761a55fd7551fa"),  
  "regno" : "102",  
  "name" : "sneha",  
  "mark" : [  
    {  
      "chem" : "90"  
    },  
    {  
      "phy" : "88"  
    },  
    {  
      "bio" : "80"  
    }  
  ]  
}
```

```
{  
  
  "_id" : ObjectId("62a06cc171761a55fd7551fb"),  
  "regno" : "103",  
  "name" : "Diya",  
  "mark" : [  
    {  
      "chem" : "97"  
    },  
    {  
      "phy" : "88"  
    },  
    {  
      "bio" : "80"  
    }  
  ]  
}
```

```

        {
            "phy" : "88"
        },
        {
            "bio" : "87"
        }
    ]
}

> db.students.getIndices()

[ { "v" : 2, "key" : { "_id" : 1 }, "name" : "_id_" } ]

> db.students.createIndex({"101":1})

{
    "numIndexesBefore" : 1,
    "numIndexesAfter" : 2,
    "createdCollectionAutomatically" : false,
    "ok" : 1
}

> db.students.getIndices

function(filter) {
    var res = this.runCommand("listIndexes", filter);

    if (!res.ok) {
        if (res.code == ErrorCodes.NamespaceNotFound) {
            // For compatibility, return []
            return [];
        }
    }
}

```



```

    }

    throw _getErrorWithCode(res, "listIndexes failed: " + toJson(res));
}

return new DBCommandCursor(this._db, res).toArray();
}
> db.students.getIndices()
[
  {
    "v" : 2,
    "key" : {
      "_id" : 1
    },
    "name" : "_id_"
  },
  {
    "v" : 2,
    "key" : {
      "101" : 1
    },
    "name" : "101_1"
  }
]
> db.students.dropIndex({"regno":1})

```

```
{  
  "ok" : 0,  
  "errmsg" : "can't find index with key: { regno: 1.0 }",  
  "code" : 27,  
  "codeName" : "IndexNotFound"  
}
```

```
> db.students.getIndices()
```

```
[  
  {  
    "v" : 2,  
    "key" : {  
      "_id" : 1  
    },  
    "name" : "_id_"  
  },  
  {  
    "v" : 2,  
    "key" : {  
      "101" : 1  
    },  
    "name" : "101_1"  
  }  
]
```

```
> db.students.createIndex({"regno":1,"name":1})
```

```
{
```

```
    "numIndexesBefore" : 2,  
    "numIndexesAfter" : 3,  
    "createdCollectionAutomatically" : false,  
    "ok" : 1  
}
```

```
> db.students.getIndices()
```

```
[  
  {  
    "v" : 2,  
    "key" : {  
      "_id" : 1  
    },  
    "name" : "_id_"  
  },  
  {  
    "v" : 2,  
    "key" : {  
      "101" : 1  
    },  
    "name" : "101_1"  
  },  
  {  
    "v" : 2,  
    "key" : {  
      "regno" : 1,  

```

```

        "name" : 1
      },
      "name" : "regno_1_name_1"
    }
  ]
}

> db.students.dropIndex({"101":1})
{ "nIndexesWas" : 3, "ok" : 1 }

> db.students.getIndices()
[
  {
    "v" : 2,
    "key" : {
      "_id" : 1
    },
    "name" : "_id_"
  },
  {
    "v" : 2,
    "key" : {
      "regno" : 1,
      "name" : 1
    },
    "name" : "regno_1_name_1"
  }
]

```

```
> db.students.find()
```

```
{ "_id" : ObjectId("62a06c6671761a55fd7551f9"), "regno" : "101", "name" : "Bincy", "mark" : [ { "chem" : "80" }, { "phy" : "78" }, { "bio" : "90" } ] }
```

```
{ "_id" : ObjectId("62a06c9c71761a55fd7551fa"), "regno" : "102", "name" : "sneha", "mark" : [ { "chem" : "90" }, { "phy" : "88" }, { "bio" : "80" } ] }
```

```
{ "_id" : ObjectId("62a06cc171761a55fd7551fb"), "regno" : "103", "name" : "Diya", "mark" : [ { "chem" : "97" }, { "phy" : "88" }, { "bio" : "87" } ] }
```

```
> db.students.createIndex({"marks.phy":1})
```

```
{  
  "numIndexesBefore" : 2,  
  "numIndexesAfter" : 3,  
  "createdCollectionAutomatically" : false,  
  "ok" : 1  
}
```

```
> db.students.getIndices()
```

```
[  
  {  
    "v" : 2,  
    "key" : {  
      "_id" : 1  
    },  
    "name" : "_id_"  
  },  
  {  
    "v" : 2,  
    "key" : {  
      "regno" : 1,  

```

```

        "name" : 1
    },
    "name" : "regno_1_name_1"
},
{
    "v" : 2,
    "key" : {
        "marks.phy" : 1
    },
    "name" : "marks.phy_1"
}
]

```

```
> db.students.dropIndex({"marks.phy":1})
```

```
{ "nIndexesWas" : 3, "ok" : 1 }
```

```
> db.students.getIndices()
```

```

[
    {
        "v" : 2,
        "key" : {
            "_id" : 1
        },
        "name" : "_id_"
    },
    {
        "v" : 2,

```

```

        "key" : {
            "regno" : 1,
            "name" : 1
        },
        "name" : "regno_1_name_1"
    }
]

> db.db.createCollection("comment")

uncaught exception: TypeError: db.db.createCollection is not a function :

@(shell):1:1

> db.createCollection("comment")

{ "ok" : 1 }

> db.comment.insert({name:"sony","post":"hai there"})

WriteResult({ "nInserted" : 1 })

> db.comment.insert({name:"bincy","post":"hello there"})

WriteResult({ "nInserted" : 1 })

> db.comment.insert({name:"sneha","post":"thankyou"})

WriteResult({ "nInserted" : 1 })

> db.students.find()

{ "_id" : ObjectId("62a06c6671761a55fd7551f9"), "regno" : "101", "name" : "Bincy", "mark" : [ { "chem" : "80" }, { "phy" : "78" }, { "bio" : "90" } ] }

{ "_id" : ObjectId("62a06c9c71761a55fd7551fa"), "regno" : "102", "name" : "sneha", "mark" : [ { "chem" : "90" }, { "phy" : "88" }, { "bio" : "80" } ] }

{ "_id" : ObjectId("62a06cc171761a55fd7551fb"), "regno" : "103", "name" : "Diya", "mark" : [ { "chem" : "97" }, { "phy" : "88" }, { "bio" : "87" } ] }

> db.comment.find()

{ "_id" : ObjectId("62a072a871761a55fd7551fc"), "name" : "sony", "post" : "hai there" }

```

```
{ "_id" : ObjectId("62a072c671761a55fd7551fd"), "name" : "bincy", "post" : "hello there" }
```

```
{ "_id" : ObjectId("62a072dc71761a55fd7551fe"), "name" : "sneha", "post" : "thankyou" }
```

```
> db.comment.createIndex({"post":"text"})
```

```
{
```

```
    "numIndexesBefore" : 1,
```

```
    "numIndexesAfter" : 2,
```

```
    "createdCollectionAutomatically" : false,
```

```
    "ok" : 1
```

```
}
```

```
> db.comment.find()
```

```
{ "_id" : ObjectId("62a072a871761a55fd7551fc"), "name" : "sony", "post" : "hai there" }
```

```
{ "_id" : ObjectId("62a072c671761a55fd7551fd"), "name" : "bincy", "post" : "hello there" }
```

```
{ "_id" : ObjectId("62a072dc71761a55fd7551fe"), "name" : "sneha", "post" : "thankyou" }
```

```
> db.students.getIndices()
```

```
[
```

```
{
```

```
    "v" : 2,
```

```
    "key" : {
```

```
        "_id" : 1
```

```
    },
```

```
    "name" : "_id_"
```

```
},
```

```
{
```

```
    "v" : 2,
```

```
    "key" : {
```



```

        "regno" : 1,
        "name" : 1
    },
    "name" : "regno_1_name_1"
}

]

> db.comment.find({$text:{ $search : "\"hai there\"" }})
{ "_id" : ObjectId("62a072a871761a55fd7551fc"), "name" : "sony", "post" : "hai there" }

> db.comment.find({$text:{ $search : "\" there\"" }})

> db.comment.find({$text:{ $search : "\" thank\"" }})

> db.comment.find({$text:{ $search : "\" thankyou\"" }})

> db.comment.find({$text:{ $search : "\" hello there\"" }})

>

>

>

> db.comment.find({$text:{ $search : "\"hai there\"" }})
{ "_id" : ObjectId("62a072a871761a55fd7551fc"), "name" : "sony", "post" : "hai there" }

> db.comment.find({$text:{ $search : "\"hello there\"" }})
{ "_id" : ObjectId("62a072c671761a55fd7551fd"), "name" : "bincy", "post" : "hello there" }

> db.comment.find({$text:{ $search : "\"thankyou\"" }})
{ "_id" : ObjectId("62a072dc71761a55fd7551fe"), "name" : "sneha", "post" : "thankyou" }

> db.comment.find({$text:{ $search : "\"thank\"" }})

>

> db.comment.find({$text:{ $search : "\"thankyou\"" }})
{ "_id" : ObjectId("62a072dc71761a55fd7551fe"), "name" : "sneha", "post" : "thankyou" }

```

