

## Add,Update & Delete Data:

1.To view all the data base using command

`"Show dbs"`

```
test> show dbs
admin      40.00 KiB
config    108.00 KiB
db         56.00 KiB
local     72.00 KiB
test> |
```

2.To create new database use command

`"Use db"`

```
test> use db
switched to db db
db> |
```

3.To find the data in the collection,use command

`"Show collection"`

```
db> show collections
stu
```

4.To create stud collections:

```
db.create collection("stu")
```

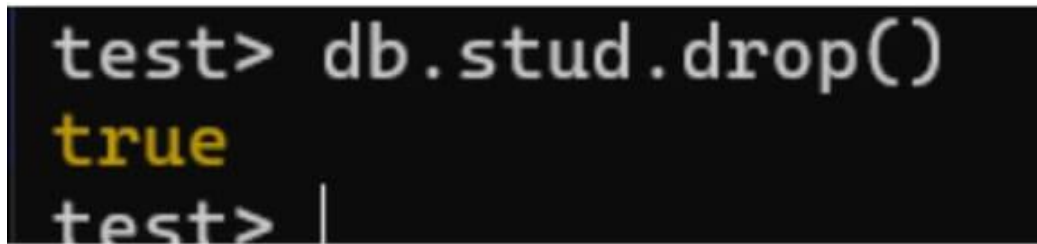
collection name is stu here

5.To insert a file to the collection:

```
db.stu.insert({name:"alice"})
```

6.To drop the stud collection:

```
db.stud.drop()
```



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

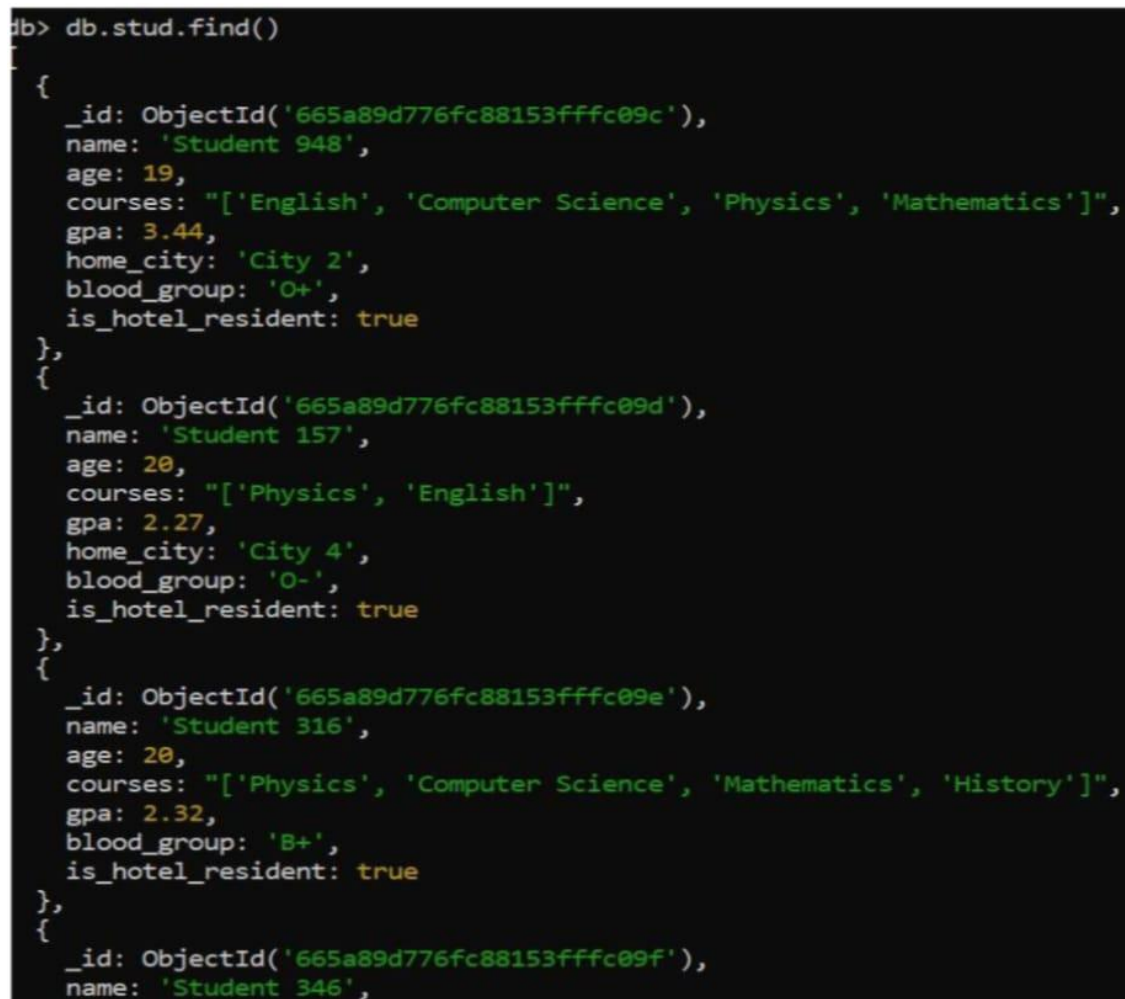
7.To find the total number of collection:

```
“db> db.stud.find().count()”
```

Total collections of students will be displayed in the database.

8.To find the data from the collection:

```
db.stud.find()
```



```
db> db.stud.find()  
[  
  {  
    _id: ObjectId('665a89d776fc88153fffc09c'),  
    name: 'Student 948',  
    age: 19,  
    courses: "['English', 'Computer Science', 'Physics', 'Mathematics']",  
    gpa: 3.44,  
    home_city: 'City 2',  
    blood_group: 'O+',  
    is_hotel_resident: true  
  },  
  {  
    _id: ObjectId('665a89d776fc88153fffc09d'),  
    name: 'Student 157',  
    age: 20,  
    courses: "['Physics', 'English']",  
    gpa: 2.27,  
    home_city: 'City 4',  
    blood_group: 'O-',  
    is_hotel_resident: true  
  },  
  {  
    _id: ObjectId('665a89d776fc88153fffc09e'),  
    name: 'Student 316',  
    age: 20,  
    courses: "['Physics', 'Computer Science', 'Mathematics', 'History']",  
    gpa: 2.32,  
    blood_group: 'B+',  
    is_hotel_resident: true  
  },  
  {  
    _id: ObjectId('665a89d776fc88153fffc09f'),  
    name: 'Student 346',  
    ...  
  }  
]
```

## WHERE, AND, OR & CRUD:

### WHERE-

The `$where` operator in MongoDB allows you to filter documents based on a JavaScript expression or function. It offers more flexibility for complex queries but comes with some drawbacks.

Given a collection you want to filter a subset based on condition that is the place “WHERE” is used.

```
db> db.stud.find({gpa:{$gt:3.5}});
[
  {
    _id: ObjectId('665a89d776fc88153fffc0a0'),
    name: 'Student 930',
    age: 25,
    courses: "['English', 'Computer Science', 'Mathematics', 'History']",
    gpa: 3.63,
    home_city: 'City 3',
    blood_group: 'A-',
    is_hotel_resident: true
  },
  {
    _id: ObjectId('665a89d776fc88153fffc0a2'),
    name: 'Student 268',
    age: 21,
    courses: "['Mathematics', 'History', 'Physics']",
    gpa: 3.98,
  }
]
db> db.stud.find({gpa:{$gt:3.5}});
[
  {
    is_hotel_resident: false
  },
  {
    _id: ObjectId('665a89d776fc88153fffc0a0'),
    name: 'Student 930',
    age: 25,
    courses: "['English', 'Computer Science', 'Mathematics', 'History']",
    gpa: 3.63,
    home_city: 'City 3',
    blood_group: 'A-',
    is_hotel_resident: true
  },
  {
    _id: ObjectId('665a89d776fc88153fffc0a2'),
    name: 'Student 268',
    age: 21,
    courses: "['Mathematics', 'History', 'Physics']",
    gpa: 3.98,
    home_city: 'City 3',
    blood_group: 'A-',
    is_hotel_resident: false
  }
]
```

To find the students based on gpa more than 3.5 with query

```
db.stud.find( {gpa:{$gt:3.5}});
```

or

```
test> db.stu.find({home_city:"City 3"}).count();
34
```

## AND-

Given a collection you want to filter a subset based on multiple condition. That is the place AND is used.

```
db> db.stud.find({
... $and:[
... {home_city:"City 5"},
... {blood_group:"A+"}
... ]
... });
[
  {
    _id: ObjectId('665a89d776fc88153fffc0d3'),
    name: 'Student 142',
    age: 24,
    courses: "['History', 'English', 'Physics', 'Computer Science']",
    gpa: 3.41,
    home_city: 'City 5',
    blood_group: 'A+',
    is_hotel_resident: false
  },
  {
    _id: ObjectId('665a89d776fc88153fffc1f3'),
    name: 'Student 947',
    age: 20,
    courses: "['Physics', 'History', 'English', 'Computer Science']",
    home_city: 'City 5',
    blood_group: 'A+',
    is_hotel_resident: true
  },
  {
    _id: ObjectId('665a89d776fc88153fffc265'),
    name: 'Student 567',
    age: 22,
    courses: "['Computer Science', 'History', 'English', 'Mathematics']",
    gpa: 2.01,
    home_city: 'City 5',
    blood_group: 'A+',
    is_hotel_resident: true
  }
]
```

```
db.students.find( {
  $and:[
    (home_city:"city b")
    {blood group: "A+"}
  ]
});
```

## OR-

The `$or` operator in MongoDB performs a logical OR operation on an array of expressions within a query. It retrieves documents that satisfy at least one of the conditions specified in the `$or` array.

Given a collection you want to filter a subset based on multiple conditions but any one is sufficient.

```
db.stud.find({ $or: [ { blood_group: "A+" }, { gpa: { $gt: 3.5 } } ] })

{
  _id: ObjectId('665a89d776fc88153fffc0a0'),
  name: 'Student 930',
  age: 25,
  courses: "['English', 'Computer Science', 'Mathematics', 'History']",
  gpa: 3.63,
  home_city: 'City 3',
  blood_group: 'A-',
  is_hotel_resident: true
},
{
  _id: ObjectId('665a89d776fc88153fffc0a2'),
  name: 'Student 268',
  age: 21,
  courses: "['Mathematics', 'History', 'Physics']",
  gpa: 3.98,
  blood_group: 'A+',
  is_hotel_resident: false
},
{
  _id: ObjectId('665a89d776fc88153fffc0a7'),
  name: 'Student 177',
  age: 23,
  courses: "['Mathematics', 'Computer Science', 'Physics']",
  gpa: 2.52,
  home_city: 'City 10',
  blood_group: 'A+',
  is_hotel_resident: true
},
{
  _id: ObjectId('665a89d776fc88153fffc0ac'),
  name: 'Student 368',
  age: 20,
```

Above example is to find students belongs to “city 5” AND blood group “A+”

## CRUD-

- C - Create / Insert
- R - Remove
- U - update
- D – Delete

### Create/Insert:

The insert function in MongoDB is used to add documents to a collection. In the context of MongoDB ,a document is a set of key-value pairs ,and a collection is a group of documents.

```
Const studentsData={  
  "name":"Alice Smith",  
  "age":12,,  
  "coueses":["Mathematics","computer science"," English" ],  
  "gpa":3.5  
  "home_city":"New York",  
  "blood_group":"A+",  
  "is_hotel_resident":false  
};
```

```
test> const studentData = {  
...   "name":"Alice Smith",  
...   "age":22,  
...   "courses":["Mathematics","Computer Science","English"],  
...   "gpa":3.8,  
...   "home_city":"New York",  
...   "blood_group":"A+",  
...   "is_hotel_resident":false  
...   };  
  
test> db.stu.insertOne(studentData);  
{  
  acknowledged: true,  
  insertedId: ObjectId('665b529e49389824aecdcd7')  
}  
test> |
```

## UPDATE:

```
db.students.updateOne( { name: "Sam"} , {$set: {gpa:3.5} })
```

```
db> db.students.updateOne( { name:"Sam"} , {$set:{  
gpa:3} } )  
{  
  acknowledged: true,  
  insertedId: null,  
  matchedCount: 1,  
  modifiedCount: 1,  
  upsertedCount: 0  
}  
db> |
```



## DELETE:

```
db> db.students.deleteOne({ name:"Sam" })
{ acknowledged: true, deletedCount: 1 }
db> |
```

## DELETE MANY:

```
test> db.stu.deleteMany({is_hostel_resident:false});
{ acknowledged: true, deletedCount: 1 }
test> |
```

## Update Many:

```
test> db.stu.updateMany({gpa:{$lt:3.0}},{$inc:{gpa:0.5}});
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 261,
  modifiedCount: 261,
  upsertedCount: 0
}
```

## PROJECTION:

This is used when we don't need all columns/attributes

```
db> db.students.deleteOne({ name:"Sam" })
{ acknowledged: true, deletedCount: 1 }
db> db.students.find({}, {name:1, gpa:1})
[
  {
    _id: ObjectId('66587b4a0a3749dfd07d78a0'),
    name: 'Student 948',
    gpa: 3.44
  },
  {
    _id: ObjectId('66587b4a0a3749dfd07d78a1'),
    name: 'Student 157',
    gpa: 2.27
  },
  {
    _id: ObjectId('66587b4a0a3749dfd07d78a2'),
    name: 'Student 316',
    gpa: 2.32
  }
]
```