1)Create Database:

use TEIT

2) show available databases

Show dbs

3) Create Collection:

db.createCollection('Student');

4) collections can be created just like databases, by writing a document to them.( Insert One)

db.student1.insert({name: "Ashwini", age: 30})

5) Insert Many Documents

r

6) Show collection name present in current database

show collections

7) Drop Collection

db.Student.drop():- It drops the student collection and all the indexes associated with the collection.

8) db.student1.remove({name: "Akshay"})

9) deleteOne

db.student1.deleteOne({age:20})

10) Replace the first matched document whose age is 30 and returns replaced document:

db.student1.findOneAndReplace({age:30},{name:"Mihir", age:30})

11) After Replacement

db.student1.find()

12) Replace the first matched document whose age is 30 and returns a new document:

db.student1.findOneAndReplace({age:30},{name:"Sagar", age:30},

{returnNewDocument:true})

13) Update Many

Update Multiple Documents

Set the “eligible” field to “true” for all students whose age is 18.

db.student.updateMany({age:18},{$set:{eligible:"true"}})

14) up inserts :

Let’s Update all documents matching the condition to set “eligible” to false and create a new document if no match is found.

db.student.updateMany({age: 18}, {$set: {eligible: false}}, {upsert: true})

15) Adding items at the start of the array:In this example, we are adding items, i.e., [“C#”, “Perl”] in the beginning(i.e., position 0 )of the language field.

db.student1.update({name: "Mihir"},

{$push: { language: { $each: ["C#", "Perl"],

$position: 0}}})

16) Cursors (Limit, skip, sort, advanced query options)

db.student1.find().skip(1)

db.student1.find().skip(2)

Limit: Here, we only want the first two documents in the result. So, we pass 2 in the limit method.

db.student1.find().limit(2)

Sort: Ascending Order

db.student1.find().sort({age:1})

where Query

Only apply the [$where](https://www.mongodb.com/docs/v4.2/reference/operator/query/where/#op._S_where) query operator to top-level documents. The [$where](https://www.mongodb.com/docs/v4.2/reference/operator/query/where/#op._S_where) query operator will not work inside a nested document, for instance, in an [$elemMatch](https://www.mongodb.com/docs/v4.2/reference/operator/query/elemMatch/#op._S_elemMatch) query.

Example

Consider the following documents in the players collection:

{

\_id: 12378,

name: "Steve",

username: "steveisawesome",

first\_login: "2017-01-01"

}

{

\_id: 2,

name: "Anya",

username: "anya",

first\_login: "2001-02-02"

}

The following example uses $where and the hex\_md5() JavaScript function to compare the value of the name field to an MD5 hash and returns any matching document

db.players.find( { $where: function() {

return (hex\_md5(this.name) == "9b53e667f30cd329dca1ec9e6a83e994")

} } );