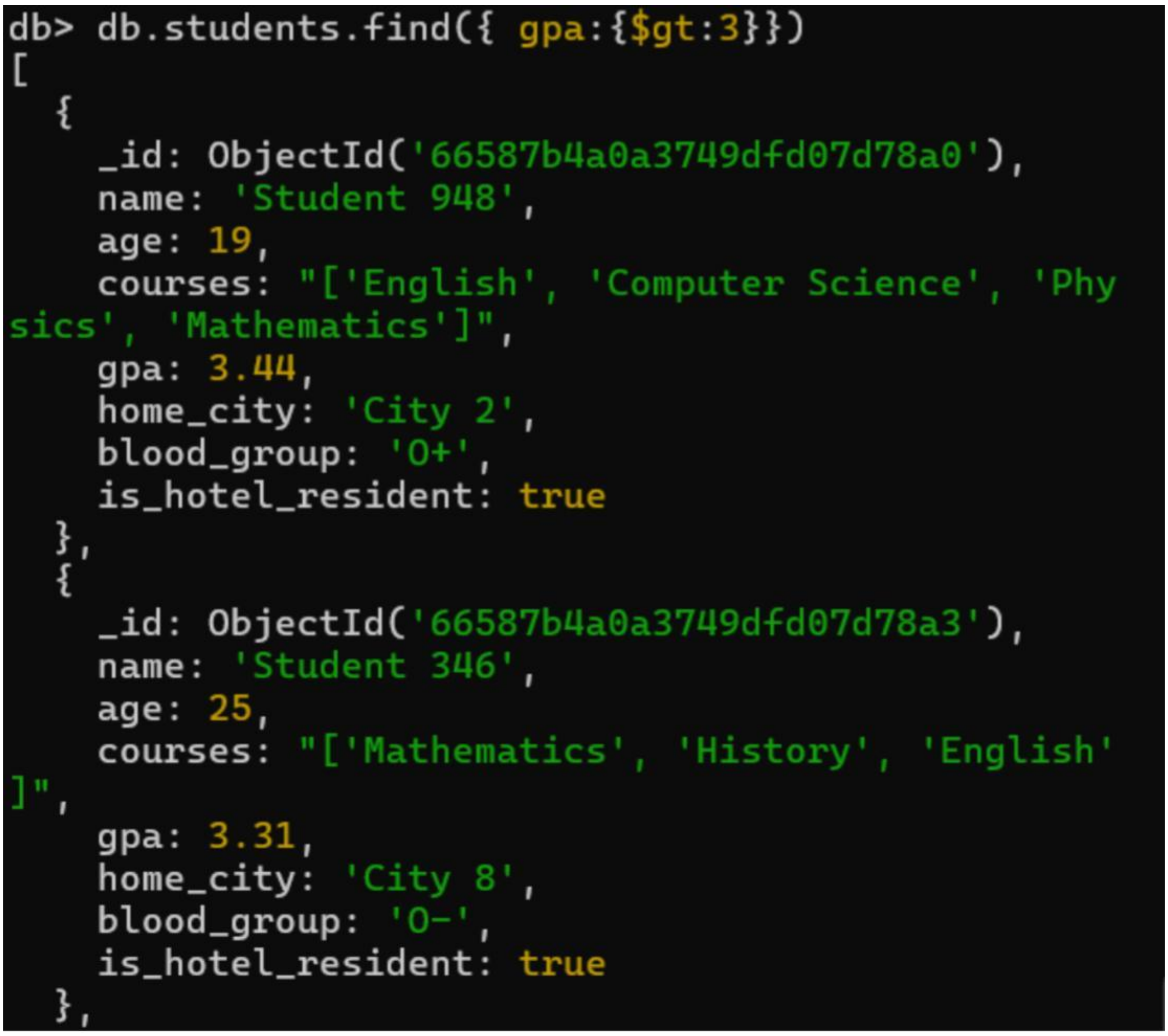
CLASS 3: WHERE, AND, OR & CURD

WHERE:

Given a Collection you want to FILTER a subset based on a condition. That is the place WHERE is used.



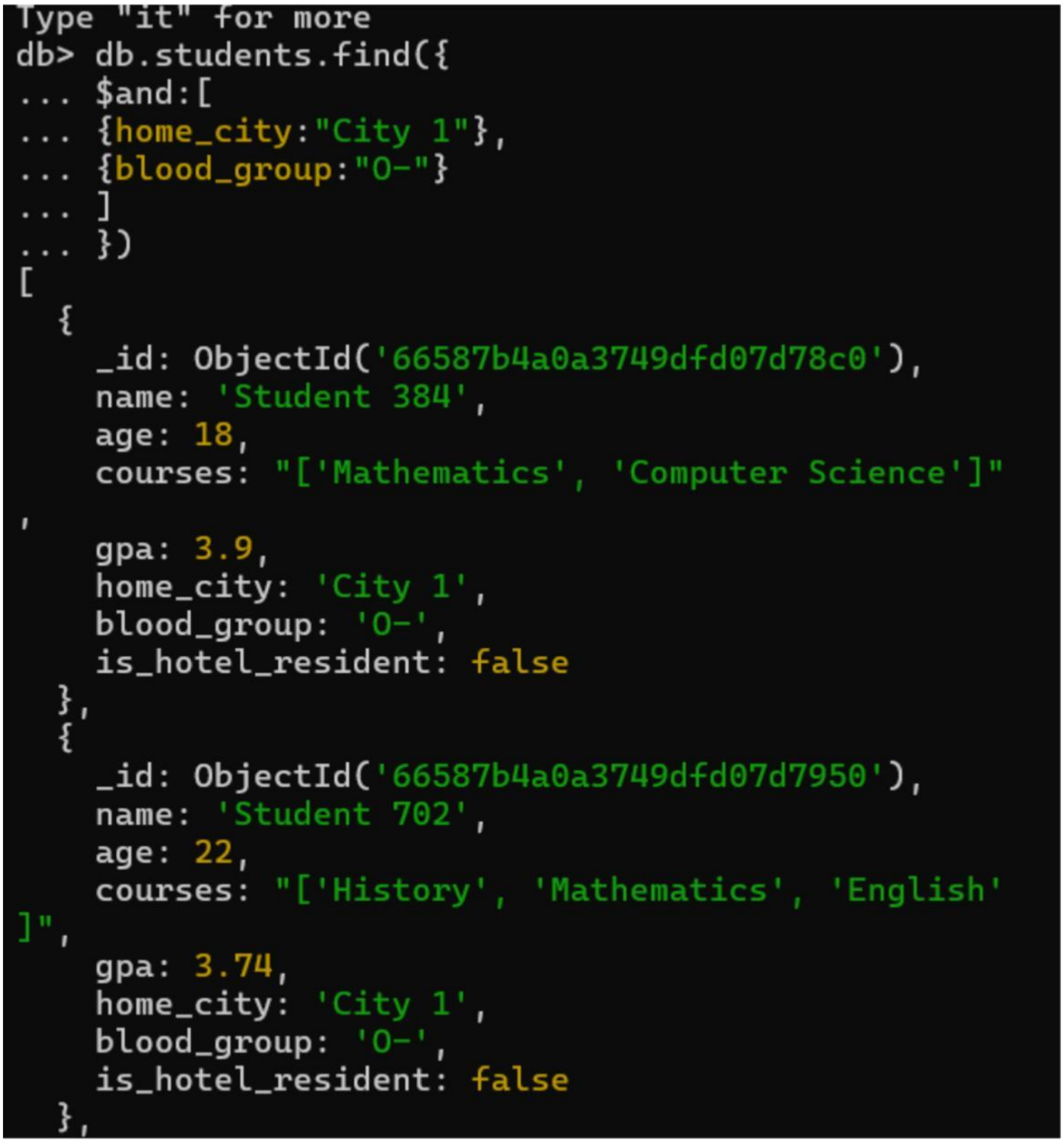
```
db> db.students.find({ gpa:{$gt:3}})
[
  {
    _id: ObjectId('66587b4a0a3749dfd07d78a0'),
    name: 'Student 948',
    age: 19,
    courses: "['English', 'Computer Science', 'Phy
sics', 'Mathematics']",
    gpa: 3.44,
    home_city: 'City 2',
    blood_group: 'O+',
    is_hotel_resident: true
  },
  {
    _id: ObjectId('66587b4a0a3749dfd07d78a3'),
    name: 'Student 346',
    age: 25,
    courses: "['Mathematics', 'History', 'English'
]",
    gpa: 3.31,
    home_city: 'City 8',
    blood_group: 'O-',
    is_hotel_resident: true
  },
```

In this example we use the condition gpa greater than 3.so the result is shown above is based on this condition.

Here '$gt' means greater than

AND:

Given a Collection you want to FILTER a subset based on multiple conditions.



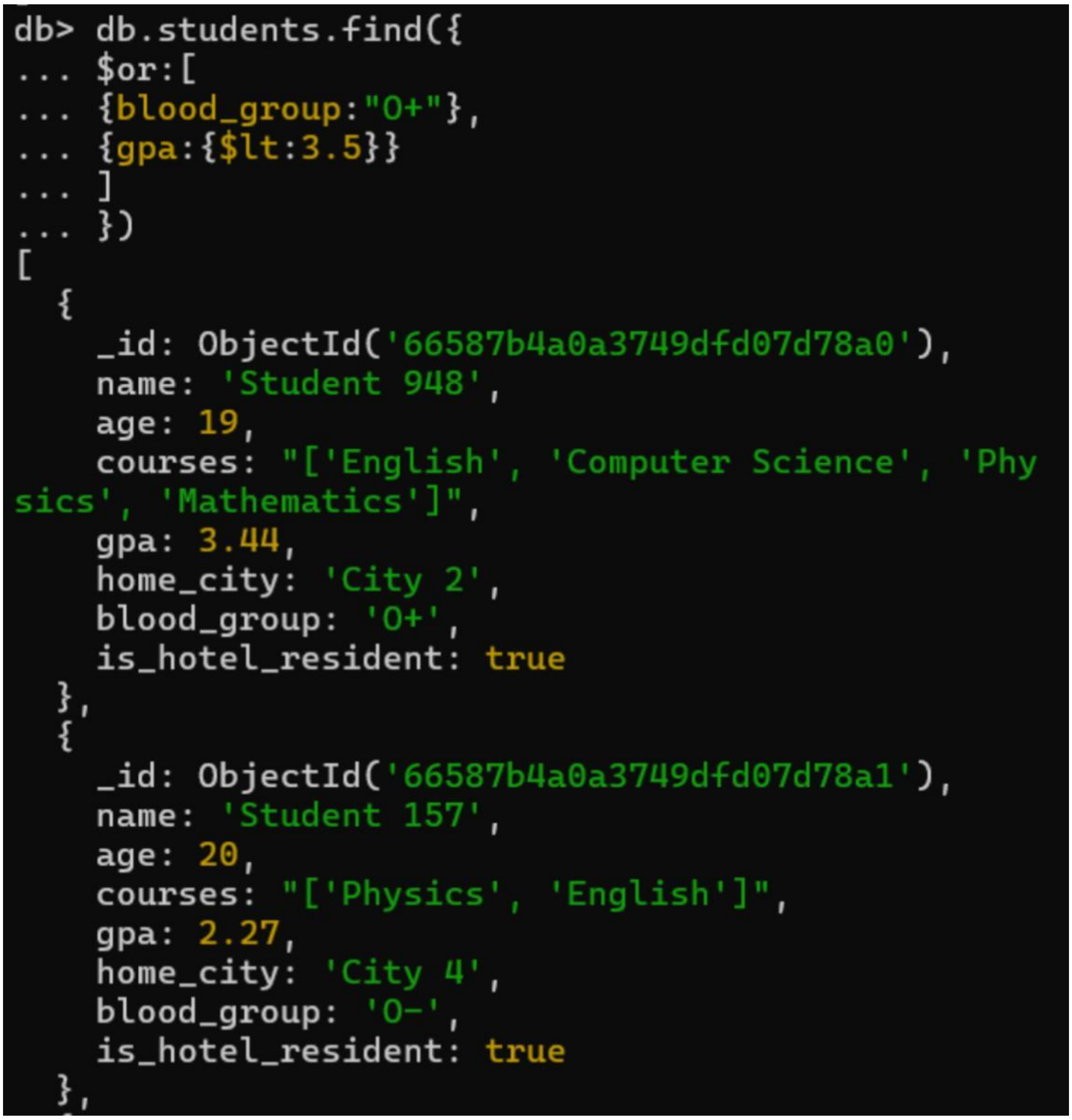
```
Type "it" for more
db> db.students.find({
... $and:[
... {home_city:"City 1"},
... {blood_group:"O-"}
... ]
... })
[
  {
    _id: ObjectId('66587b4a0a3749dfd07d78c0'),
    name: 'Student 384',
    age: 18,
    courses: "['Mathematics', 'Computer Science']"
,
    gpa: 3.9,
    home_city: 'City 1',
    blood_group: 'O-',
    is_hotel_resident: false
  },
  {
    _id: ObjectId('66587b4a0a3749dfd07d7950'),
    name: 'Student 702',
    age: 22,
    courses: "['History', 'Mathematics', 'English'
]",
    gpa: 3.74,
    home_city: 'City 1',
    blood_group: 'O-',
    is_hotel_resident: false
  },
```

Above example is filtered based on 2 condition

‘home_city:City1’ and ‘blood_group: O-‘

OR:

Given a Collection you want to FILTER a subset based on multiple conditions but Any One is Sufficient.



```
db> db.students.find({
... $or:[
... {blood_group:"O+"},
... {gpa:{$lt:3.5}}
... ]
... })
[
  {
    _id: ObjectId('66587b4a0a3749dfd07d78a0'),
    name: 'Student 948',
    age: 19,
    courses: "['English', 'Computer Science', 'Phy
sics', 'Mathematics']",
    gpa: 3.44,
    home_city: 'City 2',
    blood_group: 'O+',
    is_hotel_resident: true
  },
  {
    _id: ObjectId('66587b4a0a3749dfd07d78a1'),
    name: 'Student 157',
    age: 20,
    courses: "['Physics', 'English']",
    gpa: 2.27,
    home_city: 'City 4',
    blood_group: 'O-',
    is_hotel_resident: true
  },
```

Above example,the student database is filtered based on either ‘blood_group:O+’or”gpa less than 3.5

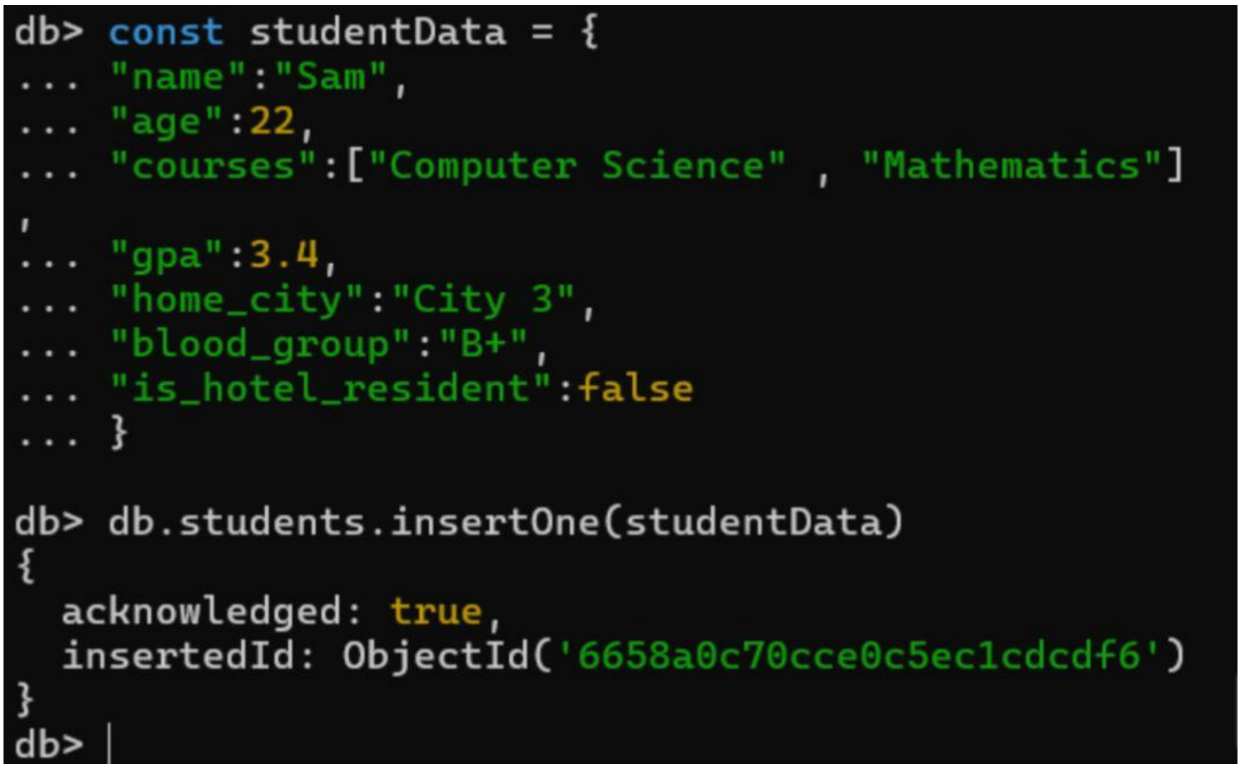
CURD:

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

This is applicable for a Collection (Table) or a Document (Row)

INSERT:

We can insert the single document and also multiple document into a collection.



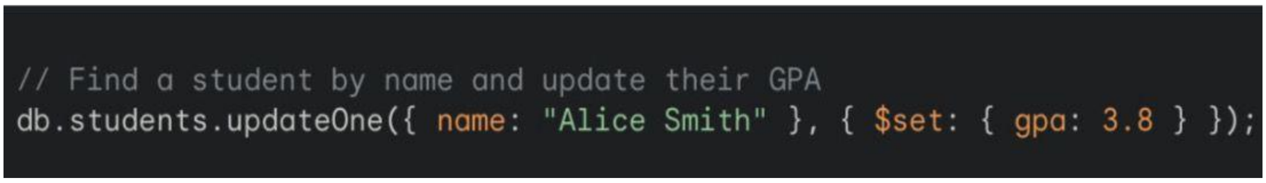
```
db> const studentData = {
... "name":"Sam",
... "age":22,
... "courses":["Computer Science" , "Mathematics"]
,
... "gpa":3.4,
... "home_city":"City 3",
... "blood_group":"B+",
... "is_hotel_resident":false
... }

db> db.students.insertOne(studentData)
{
  acknowledged: true,
  insertedId: ObjectId('6658a0c70cce0c5ec1cdcdf6')
}
db> |
```

In this above example,single document is inserted

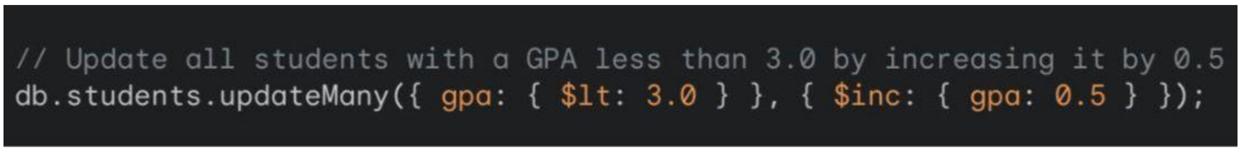
UPDATE:

We can update any data that is present in the collection.



```
// Find a student by name and update their GPA
db.students.updateOne({ name: "Alice Smith" }, { $set: { gpa: 3.8 } });
```

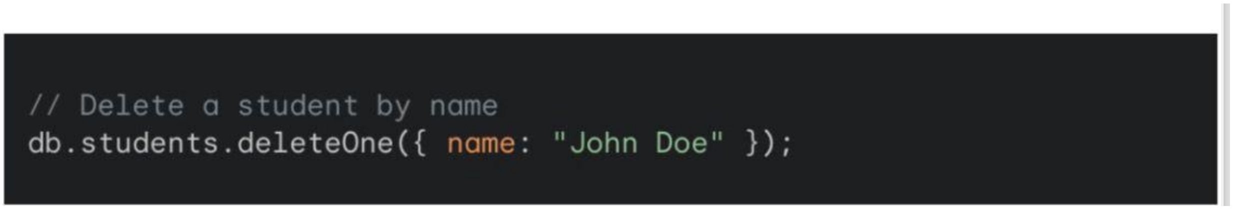
In this above example we can able to “updateOne”



```
// Update all students with a GPA less than 3.0 by increasing it by 0.5
db.students.updateMany({ gpa: { $lt: 3.0 } }, { $inc: { gpa: 0.5 } });
```

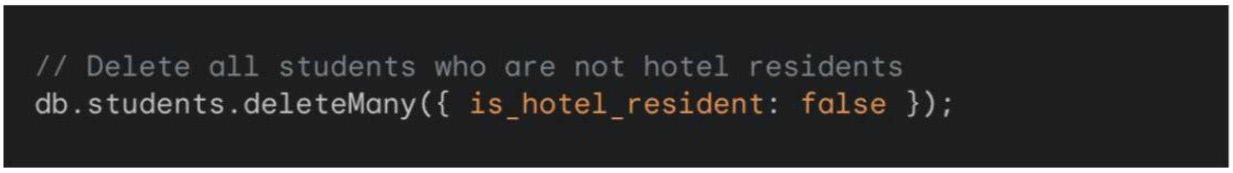
In this above example we can able to update many time “updateMany”

DELETE:



```
// Delete a student by name
db.students.deleteOne({ name: "John Doe" });
```

In this above example we can able to “deleteonce”



```
// Delete all students who are not hotel residents
db.students.deleteMany({ is_hotel_resident: false });
```

In this above example we can able to delete many time “deleteMany”