Update Many

The db.collection.updateMany() method can accept an aggregation pipeline [<stage1>, <stage2>, ...] that specifies the modifications to perform.

Sharded Collections

operation that includes `upsert: true` and is on a sharded collection, you must include the full shard key in the `filter`.

Explainability

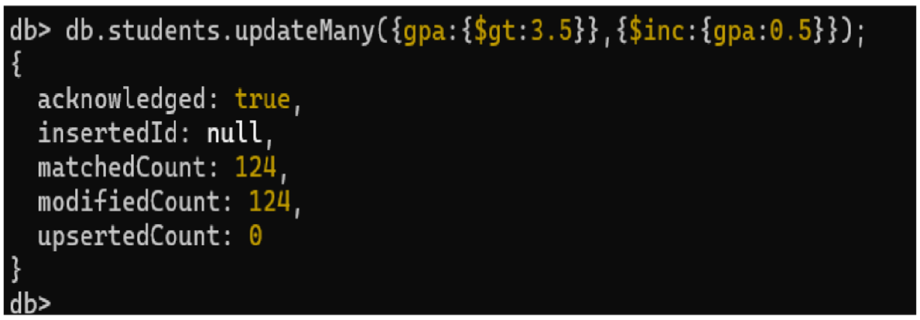
Update many is not compatible with db.collection.updatemany()

Transactions

db.collection.updatemany() can be used inside distributed transactions.

Example:

```
try {
   db.restaurant.updateMany(
      { violations: { $gt: 4 } },
      { $set: { "Review" : true } }
   );
} catch (e) {
   print(e);
}
```



```
db> db.students.updateMany({gpa:{$gt:3.5}},{$inc:{gpa:0.5}});
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 124,
  modifiedCount: 124,
  upsertedCount: 0
}
db>
```

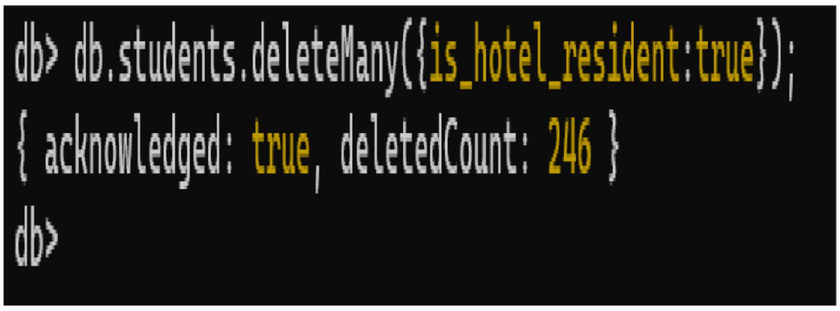
Delete many

The deleteMany() method allows you to remove all documents that match a condition from a collection.

In this syntax: filter is a document that specifies deletion criteria.

Syntax:

db.collection.deleteMany(filter, option)



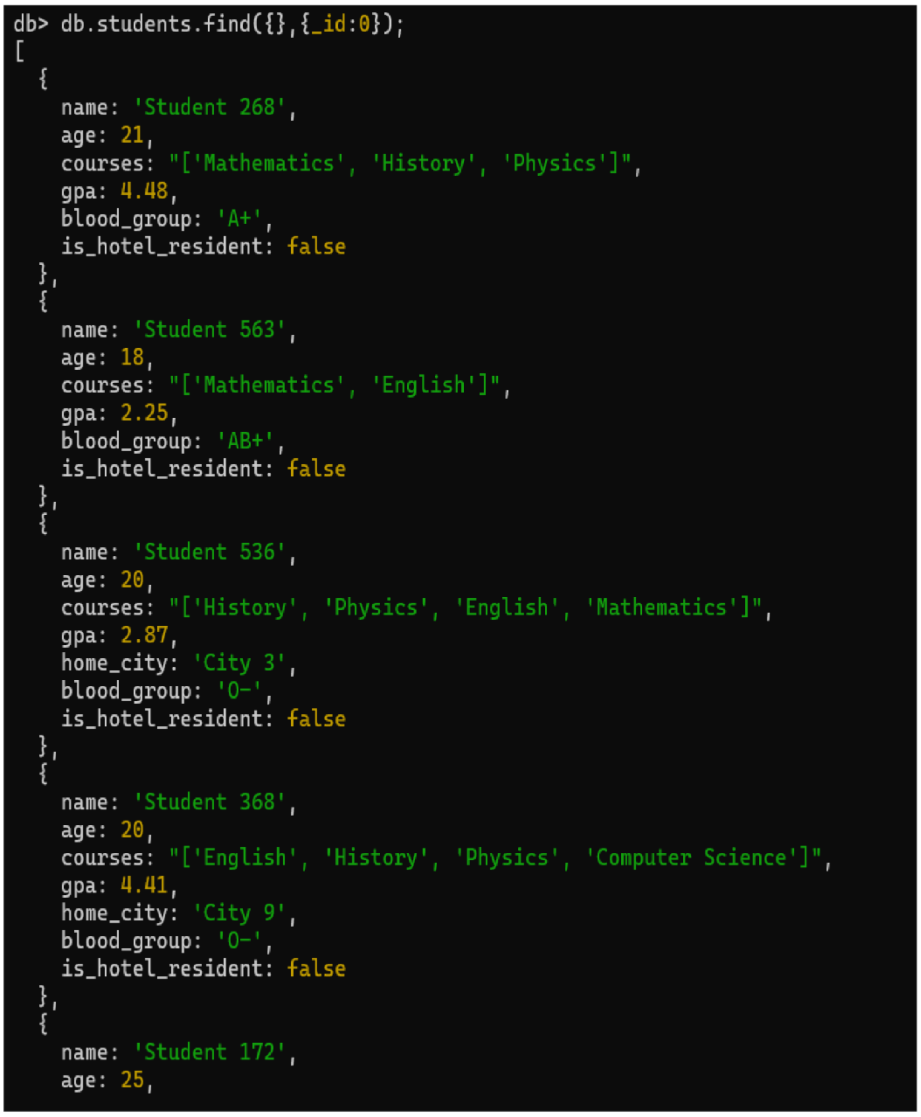
```
db> db.students.deleteMany({is_hotel_resident:true});
{ acknowledged: true, deletedCount: 246 }
db>
```

Selected attributes

To filter a subset of attributes from a collection we use projection to get only a needed attributes . db.students.

find({},{name:1,age1,courses:1});

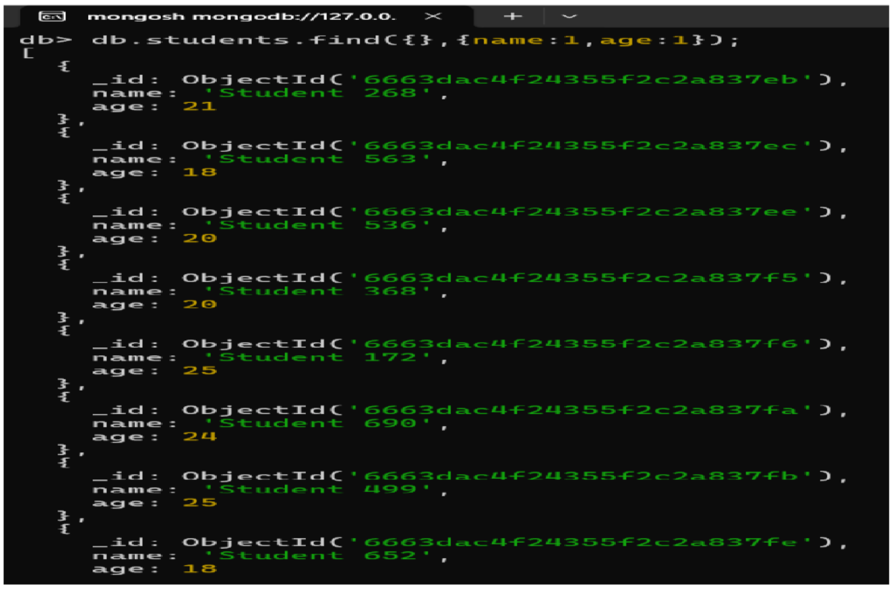
here we are trying to display only names ,age and courses of a students.



```
db> db.students.find({},{_id:0});
[
  {
    name: 'Student 268',
    age: 21,
    courses: "['Mathematics', 'History', 'Physics']",
    gpa: 4.48,
    blood_group: 'A+',
    is_hotel_resident: false
  },
  {
    name: 'Student 563',
    age: 18,
    courses: "['Mathematics', 'English']",
    gpa: 2.25,
    blood_group: 'AB+',
    is_hotel_resident: false
  },
  {
    name: 'Student 536',
    age: 20,
    courses: "['History', 'Physics', 'English', 'Mathematics']",
    gpa: 2.87,
    home_city: 'City 3',
    blood_group: 'O-',
    is_hotel_resident: false
  },
  {
    name: 'Student 368',
    age: 20,
    courses: "['English', 'History', 'Physics', 'Computer Science']",
    gpa: 4.41,
    home_city: 'City 9',
    blood_group: 'O-',
    is_hotel_resident: false
  },
  {
    name: 'Student 172',
    age: 25,
```

To display only name and age

db.students.find({},{name:1,age1});



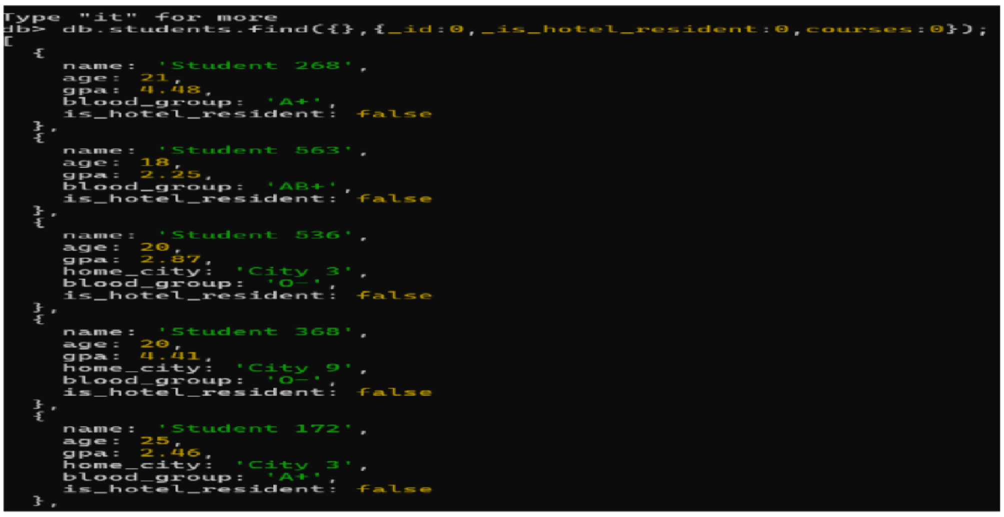
mongosh mongodb://127.0.0.

```
db> db.students.find({},{name:1,age:1});
[
  {
    _id: ObjectId('6663dac4f24355f2c2a837eb'),
    name: 'Student 268',
    age: 21
  },
  {
    _id: ObjectId('6663dac4f24355f2c2a837ec'),
    name: 'Student 563',
    age: 18
  },
  {
    _id: ObjectId('6663dac4f24355f2c2a837ee'),
    name: 'Student 536',
    age: 20
  },
  {
    _id: ObjectId('6663dac4f24355f2c2a837f5'),
    name: 'Student 368',
    age: 20
  },
  {
    _id: ObjectId('6663dac4f24355f2c2a837f6'),
    name: 'Student 172',
    age: 25
  },
  {
    _id: ObjectId('6663dac4f24355f2c2a837fa'),
    name: 'Student 690',
    age: 24
  },
  {
    _id: ObjectId('6663dac4f24355f2c2a837fb'),
    name: 'Student 499',
    age: 25
  },
  {
    _id: ObjectId('6663dac4f24355f2c2a837fe'),
    name: 'Student 652',
    age: 18
```

Ignore attributes

To get all the students data except id ,hotel resident and courses have to set all of three has 0 that is

db.students.find({},{id:0,_is_hotel_resident:0,courses:0});

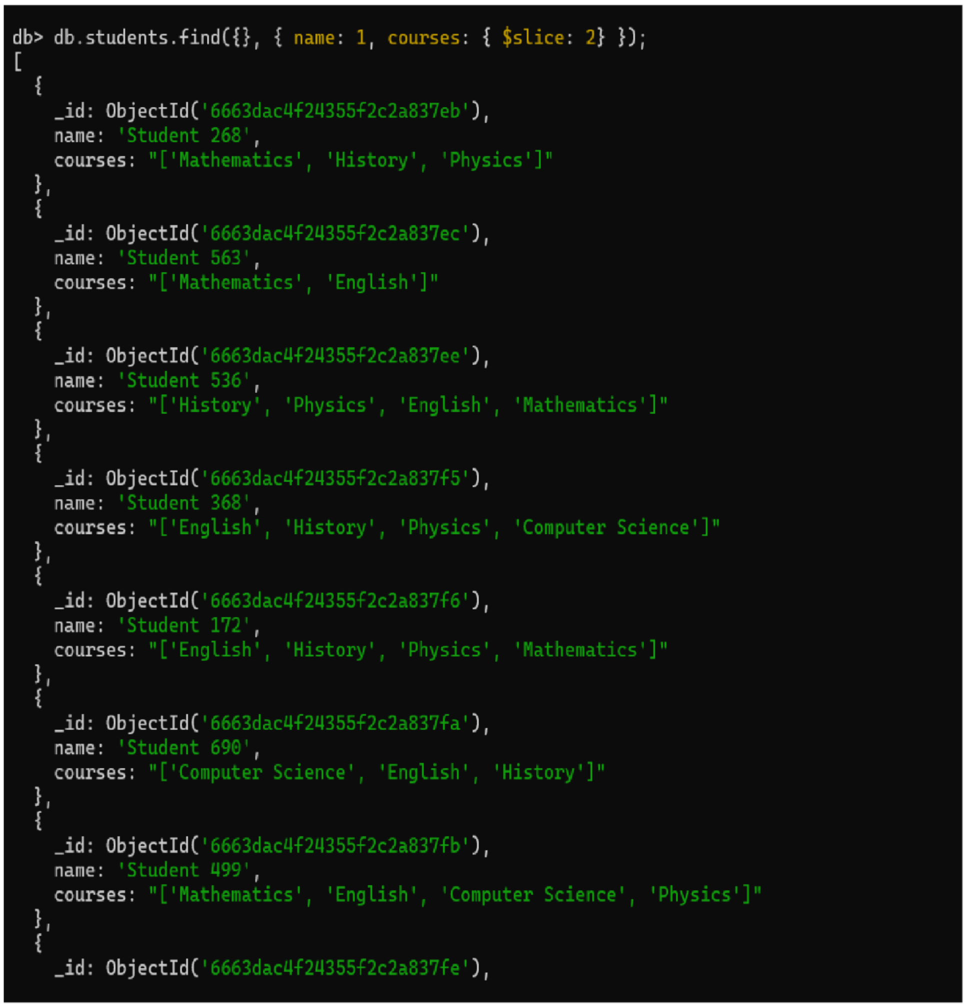


```
Type "it" for more
db> db.students.find({},{_id:0,_is_hotel_resident:0,courses:0});
[
  {
    name: 'Student 268',
    age: 21,
    gpa: 4.48,
    blood_group: 'A+',
    is_hotel_resident: false
  },
  {
    name: 'Student 563',
    age: 18,
    gpa: 2.25,
    blood_group: 'AB+',
    is_hotel_resident: false
  },
  {
    name: 'Student 536',
    age: 20,
    gpa: 2.87,
    home_city: 'City 3',
    blood_group: 'O-',
    is_hotel_resident: false
  },
  {
    name: 'Student 368',
    age: 20,
    gpa: 4.41,
    home_city: 'City 9',
    blood_group: 'O-',
    is_hotel_resident: false
  },
  {
    name: 'Student 172',
    age: 25,
    gpa: 2.46,
    home_city: 'City 3',
    blood_group: 'A+',
    is_hotel_resident: false
  },
```

Slice

The $slice projection operator specifies the number of elements in an array to return in the query result.

db.students.find({},{name:1,courses:{$slice:2}});



```
db> db.students.find({}, { name: 1, courses: { $slice: 2} });
[
  {
    _id: ObjectId('6663dac4f24355f2c2a837eb'),
    name: 'Student 268',
    courses: "['Mathematics', 'History', 'Physics']"
  },
  {
    _id: ObjectId('6663dac4f24355f2c2a837ec'),
    name: 'Student 563',
    courses: "['Mathematics', 'English']"
  },
  {
    _id: ObjectId('6663dac4f24355f2c2a837ee'),
    name: 'Student 536',
    courses: "['History', 'Physics', 'English', 'Mathematics']"
  },
  {
    _id: ObjectId('6663dac4f24355f2c2a837f5'),
    name: 'Student 368',
    courses: "['English', 'History', 'Physics', 'Computer Science']"
  },
  {
    _id: ObjectId('6663dac4f24355f2c2a837f6'),
    name: 'Student 172',
    courses: "['English', 'History', 'Physics', 'Mathematics']"
  },
  {
    _id: ObjectId('6663dac4f24355f2c2a837fa'),
    name: 'Student 690',
    courses: "['Computer Science', 'English', 'History']"
  },
  {
    _id: ObjectId('6663dac4f24355f2c2a837fb'),
    name: 'Student 499',
    courses: "['Mathematics', 'English', 'Computer Science', 'Physics']"
  },
  {
    _id: ObjectId('6663dac4f24355f2c2a837fe'),
```

This is used to get student name and only the mentioned number course from the course array.

Using projection offers several benefits:

➢ By retrieving only the necessary fields, projection reduces the amount of data leads to faster query performance.

➢ When working with complex documents, projection allows to focus on relevant data.

➢ Projection can help in optimizing queries by allowing MongoDB to make better use of indexes, especially when the indexed fields are the only ones being retrieved.

Limit:

In MongoDB, the limit method limits the number of records or documents that you want. It basically defines the max limit of records/documents that you want.

example

```
db.gfg.find().limit(2)
```

Benefits of Using limit() in MongoDB:

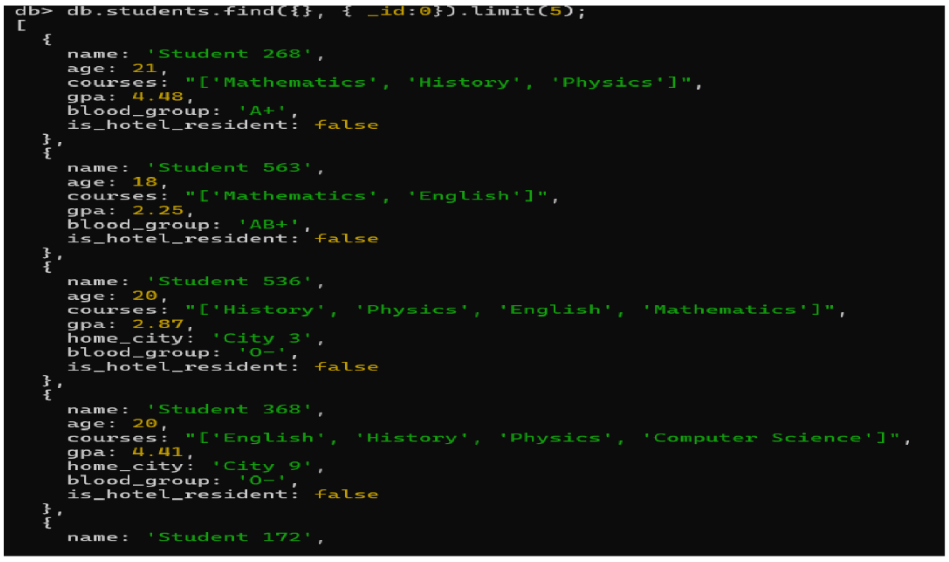
➢ By restricting the number of documents returned, limit() can improve query performance.

➢ For applications with user interfaces, returning a smaller subset of data can enhance responsiveness and provide a better user experience.

➢ Handling a limited number of documents at a time simplifies data processing.

To find only data of 5 students without an id

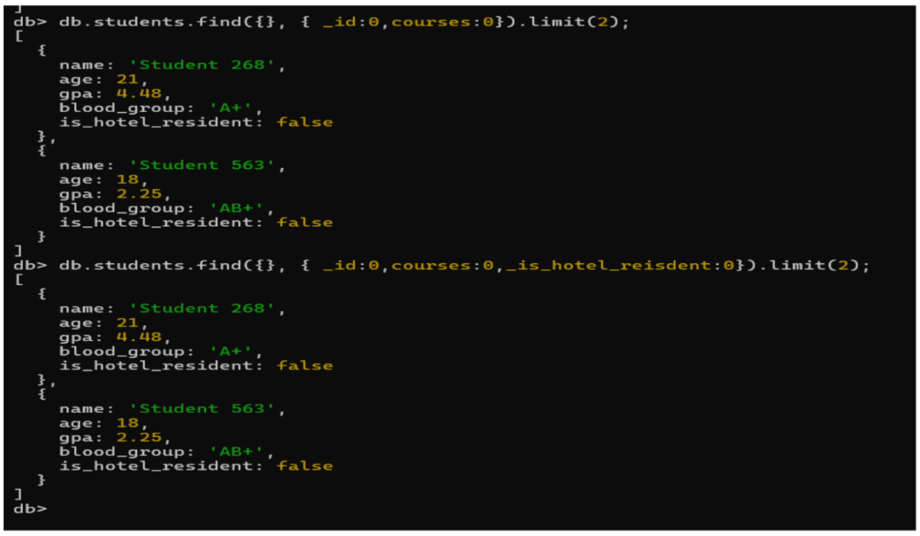
db.students.find({},{_id:0}).limit(5);



```
db> db.students.find({}, { _id:0}).limit(5);
[
  {
    name: 'Student 268',
    age: 21,
    courses: "['Mathematics', 'History', 'Physics']",
    gpa: 4.48,
    blood_group: 'A+',
    is_hotel_resident: false
  },
  {
    name: 'Student 563',
    age: 18,
    courses: "['Mathematics', 'English']",
    gpa: 2.25,
    blood_group: 'AB+',
    is_hotel_resident: false
  },
  {
    name: 'Student 536',
    age: 20,
    courses: "['History', 'Physics', 'English', 'Mathematics']",
    gpa: 2.87,
    home_city: 'City 3',
    blood_group: 'O-',
    is_hotel_resident: false
  },
  {
    name: 'Student 368',
    age: 20,
    courses: "['English', 'History', 'Physics', 'Computer Science']",
    gpa: 4.41,
    home_city: 'City 9',
    blood_group: 'O-',
    is_hotel_resident: false
  },
  {
    name: 'Student 172',
```

To find only first two members data without an id and courses

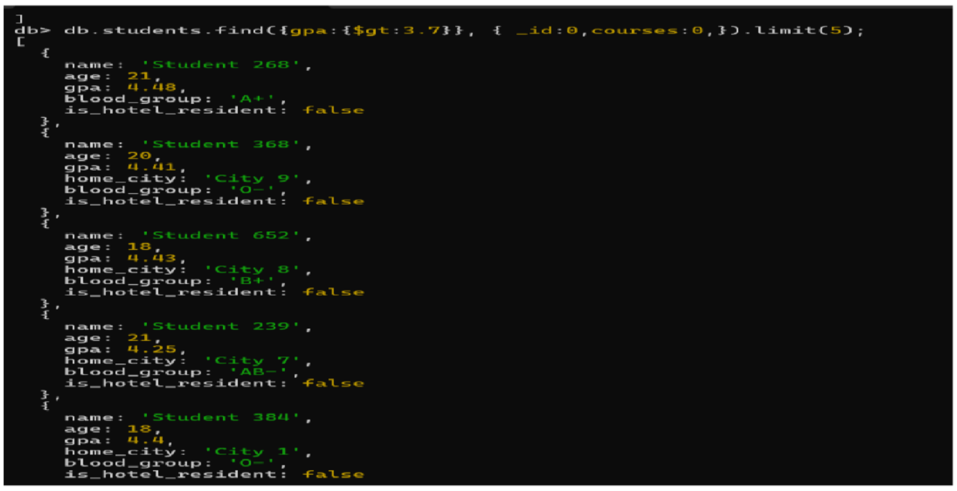
db.students.find({gpa:{$gt:3.7}},{_id:0,courses:0}).limit(2);



```
]
db> db.students.find({}, { _id:0,courses:0}).limit(2);
[
  {
    name: 'Student 268',
    age: 21,
    gpa: 4.48,
    blood_group: 'A+',
    is_hotel_resident: false
  },
  {
    name: 'Student 563',
    age: 18,
    gpa: 2.25,
    blood_group: 'AB+',
    is_hotel_resident: false
  }
]
db> db.students.find({}, { _id:0,courses:0,_is_hotel_reisdent:0}).limit(2);
[
  {
    name: 'Student 268',
    age: 21,
    gpa: 4.48,
    blood_group: 'A+',
    is_hotel_resident: false
  },
  {
    name: 'Student 563',
    age: 18,
    gpa: 2.25,
    blood_group: 'AB+',
    is_hotel_resident: false
  }
]
db>
```

here finding all the students with gpa greater than 3.7 without displaying an id and course limit to 5 documents

db.students.find({gpa:$gt:3.7}}.{_id:0,courses:0}).limit(5);



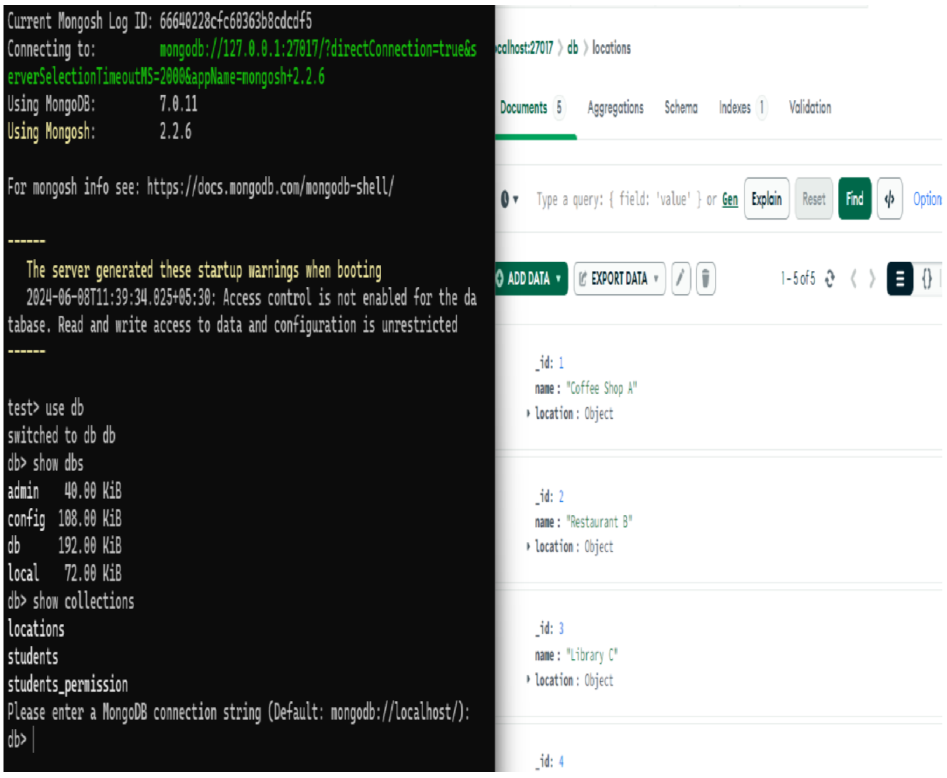
```
]
db> db.students.find({gpa:{$gt:3.7}}, { _id:0,courses:0,}).limit(5);
[
  {
    name: 'Student 268',
    age: 21,
    gpa: 4.48,
    blood_group: 'A+',
    is_hotel_resident: false
  },
  {
    name: 'Student 368',
    age: 20,
    gpa: 4.41,
    home_city: 'City 9',
    blood_group: 'O-',
    is_hotel_resident: false
  },
  {
    name: 'Student 652',
    age: 18,
    gpa: 4.43,
    home_city: 'City 8',
    blood_group: 'B+',
    is_hotel_resident: false
  },
  {
    name: 'Student 239',
    age: 21,
    gpa: 4.25,
    home_city: 'City 7',
    blood_group: 'AB-',
    is_hotel_resident: false
  },
  {
    name: 'Student 384',
    age: 18,
    gpa: 4.4,
    home_city: 'City 1',
    blood_group: 'O-',
    is_hotel_resident: false
```

Geospatial Query

MongoDB geospatial queries can interpret geometry on a flat surface or a sphere.

It follow the same step to switch this collection to database.

Use db
Show dbs
Show collections



```
Current Mongosh Log ID: 66640228cfc60363b8cdcdf5
Connecting to:          mongodb://127.0.0.1:27017/?directConnection=true&s
erverSelectionTimeoutMS=2000&appName=mongosh+2.2.6
Using MongoDB:          7.0.11
Using Mongosh:          2.2.6

For mongosh info see: https://docs.mongodb.com/mongodb-shell/

------
   The server generated these startup warnings when booting
   2024-06-08T11:39:34.025+05:30: Access control is not enabled for the da
tabase. Read and write access to data and configuration is unrestricted
------

test> use db
switched to db db
db> show dbs
admin    40.00 KiB
config  108.00 KiB
db      192.00 KiB
local    72.00 KiB
db> show collections
locations
students
students_permission
Please enter a MongoDB connection string (Default: mongodb://localhost/):
db>
```

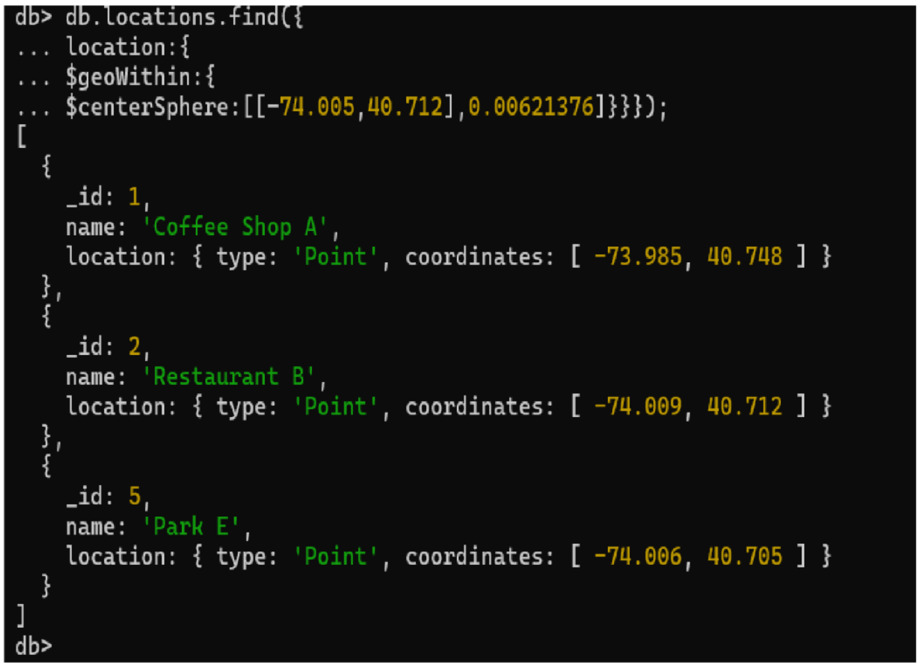
```
_id: 1
name : "Coffee Shop A"
▸ location : Object

_id: 2
name : "Restaurant B"
▸ location : Object

_id: 3
name : "Library C"
▸ location : Object

_id: 4
```

Here to find a location



```
db> db.locations.find({
... location:{
... $geoWithin:{
... $centerSphere:[[-74.005,40.712],0.00621376]}}});
[
  {
    _id: 1,
    name: 'Coffee Shop A',
    location: { type: 'Point', coordinates: [ -73.985, 40.748 ] }
  },
  {
    _id: 2,
    name: 'Restaurant B',
    location: { type: 'Point', coordinates: [ -74.009, 40.712 ] }
  },
  {
    _id: 5,
    name: 'Park E',
    location: { type: 'Point', coordinates: [ -74.006, 40.705 ] }
  }
]
db>
```

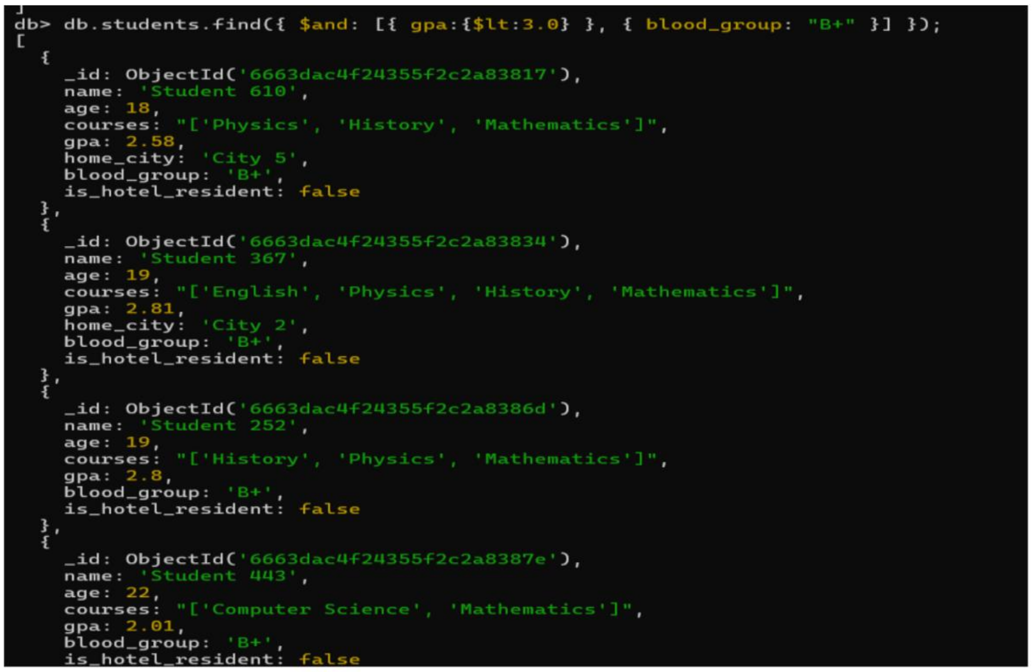
Operators

Logical operators:

And operator:

The logical AND operator ($and) is used to combine multiple query conditions. The $and operator is particularly useful when you need to match documents that satisfy all the specified conditions.

db. student. find({$and :{{home_city: “City 2”}, {blood_ group: “B+”}]});

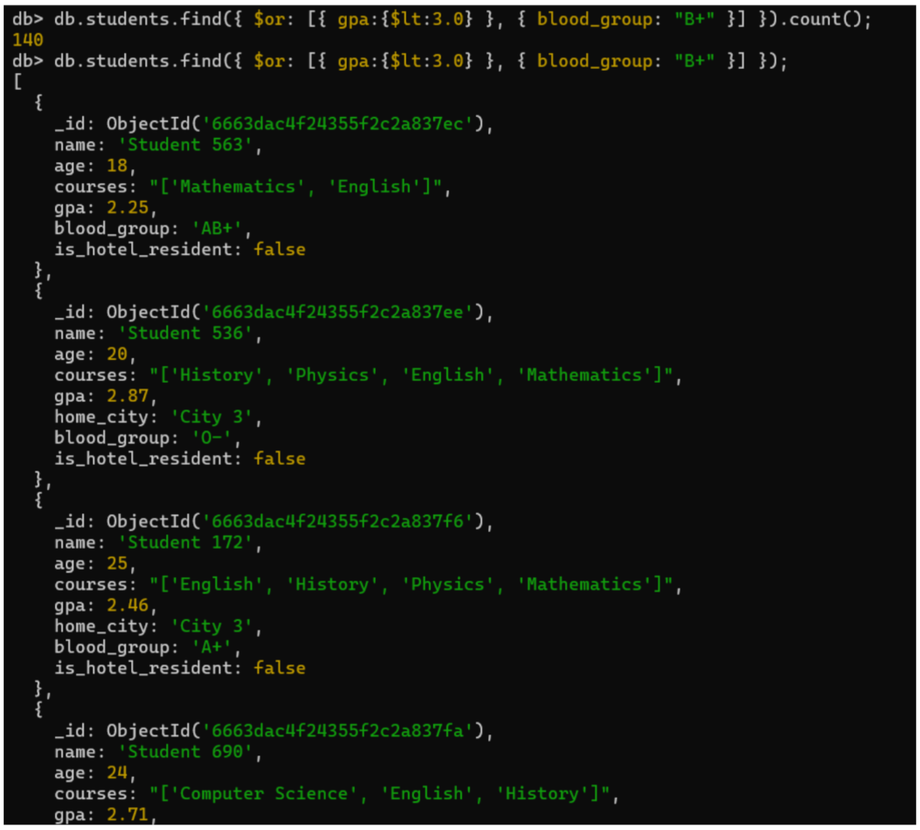


```
db> db.students.find({ $and: [{ gpa:{$lt:3.0} }, { blood_group: "B+" }] });
[
  {
    _id: ObjectId('6663dac4f24355f2c2a83817'),
    name: 'Student 610',
    age: 18,
    courses: "['Physics', 'History', 'Mathematics']",
    gpa: 2.58,
    home_city: 'City 5',
    blood_group: 'B+',
    is_hotel_resident: false
  },
  {
    _id: ObjectId('6663dac4f24355f2c2a83834'),
    name: 'Student 367',
    age: 19,
    courses: "['English', 'Physics', 'History', 'Mathematics']",
    gpa: 2.81,
    home_city: 'City 2',
    blood_group: 'B+',
    is_hotel_resident: false
  },
  {
    _id: ObjectId('6663dac4f24355f2c2a8386d'),
    name: 'Student 252',
    age: 19,
    courses: "['History', 'Physics', 'Mathematics']",
    gpa: 2.8,
    blood_group: 'B+',
    is_hotel_resident: false
  },
  {
    _id: ObjectId('6663dac4f24355f2c2a8387e'),
    name: 'Student 443',
    age: 22,
    courses: "['Computer Science', 'Mathematics']",
    gpa: 2.01,
    blood_group: 'B+',
    is_hotel_resident: false
```

OR Operator:

MongoDB provides different types of logical query operators and $or operator is one of them. This operator is used to perform logical OR operation on the array of two or more expressions and select or retrieve only those documents that match at least one of the given expression in the array.

Syntax: { $or: [{ Expression1 }, { Expression2 }, ..., { ExpressionN }] }



```
db> db.students.find({ $or: [{ gpa:{$lt:3.0} }, { blood_group: "B+" }] }).count();
140
db> db.students.find({ $or: [{ gpa:{$lt:3.0} }, { blood_group: "B+" }] });
[
  {
    _id: ObjectId('6663dac4f24355f2c2a837ec'),
    name: 'Student 563',
    age: 18,
    courses: "['Mathematics', 'English']",
    gpa: 2.25,
    blood_group: 'AB+',
    is_hotel_resident: false
  },
  {
    _id: ObjectId('6663dac4f24355f2c2a837ee'),
    name: 'Student 536',
    age: 20,
    courses: "['History', 'Physics', 'English', 'Mathematics']",
    gpa: 2.87,
    home_city: 'City 3',
    blood_group: 'O-',
    is_hotel_resident: false
  },
  {
    _id: ObjectId('6663dac4f24355f2c2a837f6'),
    name: 'Student 172',
    age: 25,
    courses: "['English', 'History', 'Physics', 'Mathematics']",
    gpa: 2.46,
    home_city: 'City 3',
    blood_group: 'A+',
    is_hotel_resident: false
  },
  {
    _id: ObjectId('6663dac4f24355f2c2a837fa'),
    name: 'Student 690',
    age: 24,
    courses: "['Computer Science', 'English', 'History']",
    gpa: 2.71,
```