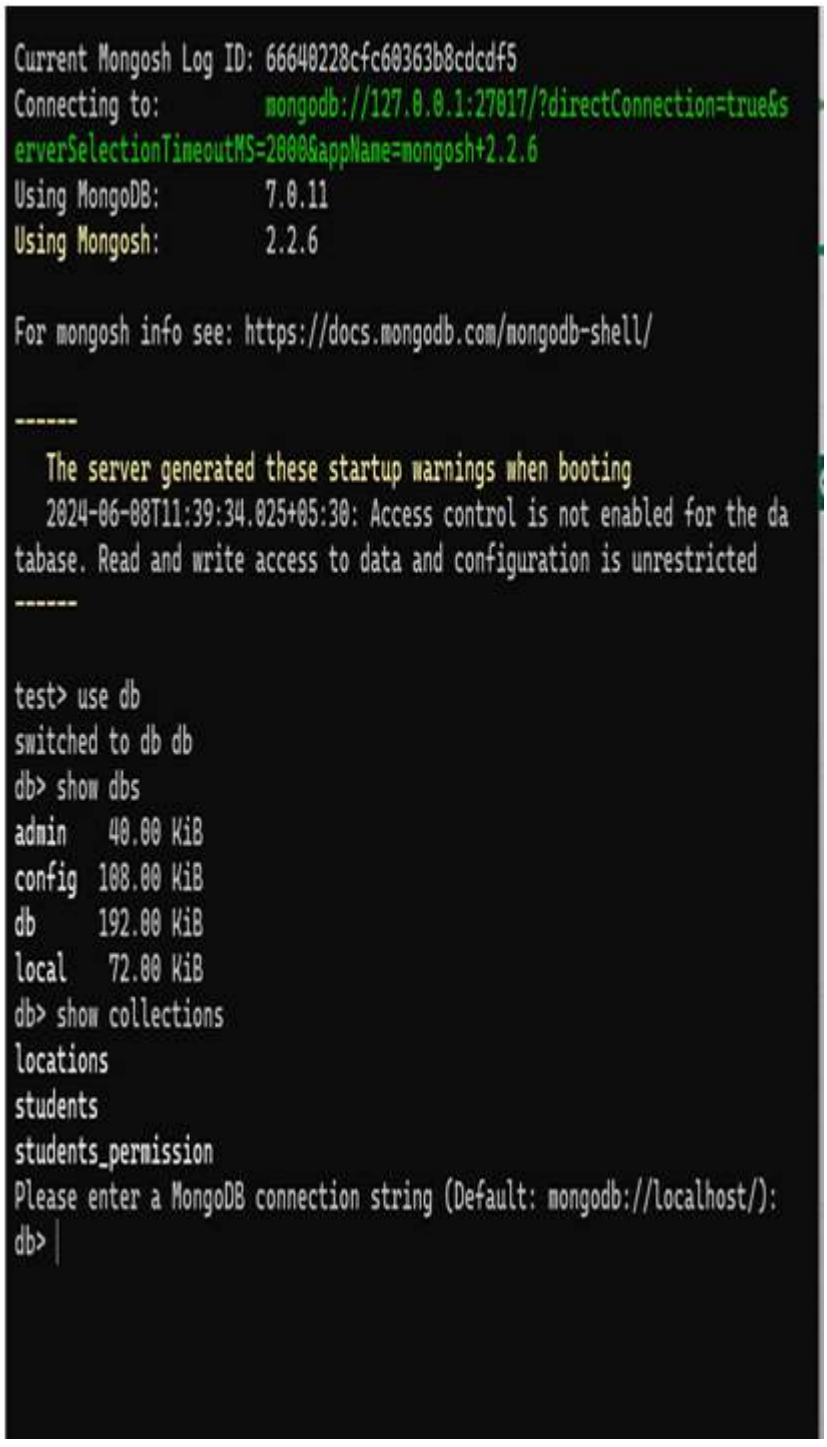
Geospatial query:

Need to upload a new collection called “location” in json format Follow the same steps to switch this collection to database.

Use db

Show dbs

Show collections



For 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>
```

Projection operators

Projection in mongodb is a powerful tool that can be used to extract only the fields you need from a document – not all fields.

- It enables you to project concise and transparent data.
- It filter the dataset without impacting the overall database performance**.**

Adding new dataset called **PRODUCTS**

```
test> use db
switched to db db
db> show collections
candidates
locations
products
std
students_permission
```

here. The new dataset **product** is added

now we are retriving name and ratings of the product

```
db> db.products.find({},{name:1,rating:1});
[
  { _id: 'ac3', name: 'AC3 Phone', rating: 3.8 },
  { _id: 'ac7', name: 'AC7 Phone', rating: 4 },
  {
    _id: ObjectId('507d95d5719dbef170f15bf9'),
    name: 'AC3 Series Charger',
    rating: 2.8
  },
  {
    _id: ObjectId('507d95d5719dbef170f15bfa'),
    name: 'AC3 Case Green',
    rating: 1
  },
  {
    _id: ObjectId('507d95d5719dbef170f15bfb'),
    name: 'Phone Extended Warranty',
    rating: 5
  },
  {
    _id: ObjectId('507d95d5719dbef170f15bfc'),
    name: 'AC3 Case Black',
    rating: 2
  },
  {
    _id: ObjectId('507d95d5719dbef170f15bfd'),
    name: 'AC3 Case Red',
    rating: 4
  },
  {
    _id: ObjectId('507d95d5719dbef170f15bfe'),
    name: 'Phone Service Basic Plan',
    rating: 3
  },
  {
    _id: ObjectId('507d95d5719dbef170f15bff'),
    name: 'Phone Service Core Plan',
    rating: 3
  },
  {
    _id: ObjectId('507d95d5719dbef170f15c00'),
    name: 'Phone Service Family Plan',
    rating: 4
  },
  {
    _id: ObjectId('507d95d5719dbef170f15c01'),
    name: 'Cable TV Basic Service Package',
    rating: 3.9
  }
]
db>
```

Product: refers to the collection named products within the current database.

To get excluding fields:

```
db> db.products.find({},{_id:0,type:0,limits:0,data:0});
[
  {
    name: 'AC3 Phone',
    brand: 'ACME',
    price: 200,
    rating: 3.8,
    warranty_years: 1,
    available: true
  },
  {
    name: 'AC7 Phone',
    brand: 'ACME',
    price: 320,
    rating: 4,
    warranty_years: 1,
    available: false
  },
  {
    name: 'AC3 Series Charger',
    price: 19,
    rating: 2.8,
    warranty_years: 0.25,
    for: [ 'ac3', 'ac7', 'ac9' ]
  },
  {
    name: 'AC3 Case Green',
    color: 'green',
    price: 12,
    rating: 1,
    warranty_years: 0
  },
  {
    name: 'Phone Extended Warranty',
    price: 38,
    rating: 5,
    warranty_years: 2,
    for: [ 'ac3', 'ac7', 'ac9', 'qp7', 'qp8', 'qp9' ]
  },
  {
    name: 'AC3 Case Black',
    color: 'black',
    price: 12.5,
    rating: 2,
    warranty_years: 0.25,
    available: false,
    for: 'ac3'
  },
  {
    name: 'AC3 Case Red',
    color: 'red',
    price: 12,
    rating: 4,
    warranty_years: 0.25,
    available: true,
    for: 'ac3'
```

```
  },
  {
    name: 'AC3 Case Red',
    color: 'red',
    price: 12,
    rating: 4,
    warranty_years: 0.25,
    available: true,
    for: 'ac3'
  },
  {
    name: 'Phone Service Basic Plan',
    monthly_price: 40,
    rating: 3,
    term_years: 2
  },
  {
    name: 'Phone Service Core Plan',
    monthly_price: 60,
    rating: 3,
    term_years: 1
  },
  {
    name: 'Phone Service Family Plan',
    monthly_price: 90,
    rating: 4,
    sales_tax: true,
    term_years: 2
  },
  {
    name: 'Cable TV Basic Service Package',
    monthly_price: 50,
    rating: 3.9,
    term_years: 2,
    cancel_penalty: 25,
    sales_tax: true,
    additional_tarriffs: [
      { kind: 'federal tarriff', amount: { percent_of_service: 0.06 } },
      { kind: 'misc tarriff', amount: 2.25 }
    ]
  }
```

Lets use json for candidates (candidates.json) to dataset:

```
db> db.candidates.find({courses:{$elemMatch:{$eq:"Computer Science"}}},{name:1,"courses.$":1});
[
  { _id: ObjectId('6657f95546a856db971e68'), name: 'Bob Johnson' },
  { _id: ObjectId('6657f95546a856db971e65'), name: 'Gabriel Miller' },
  { _id: ObjectId('6657f95546a856db971e69'), name: 'Kevin Lewis' }
]
```

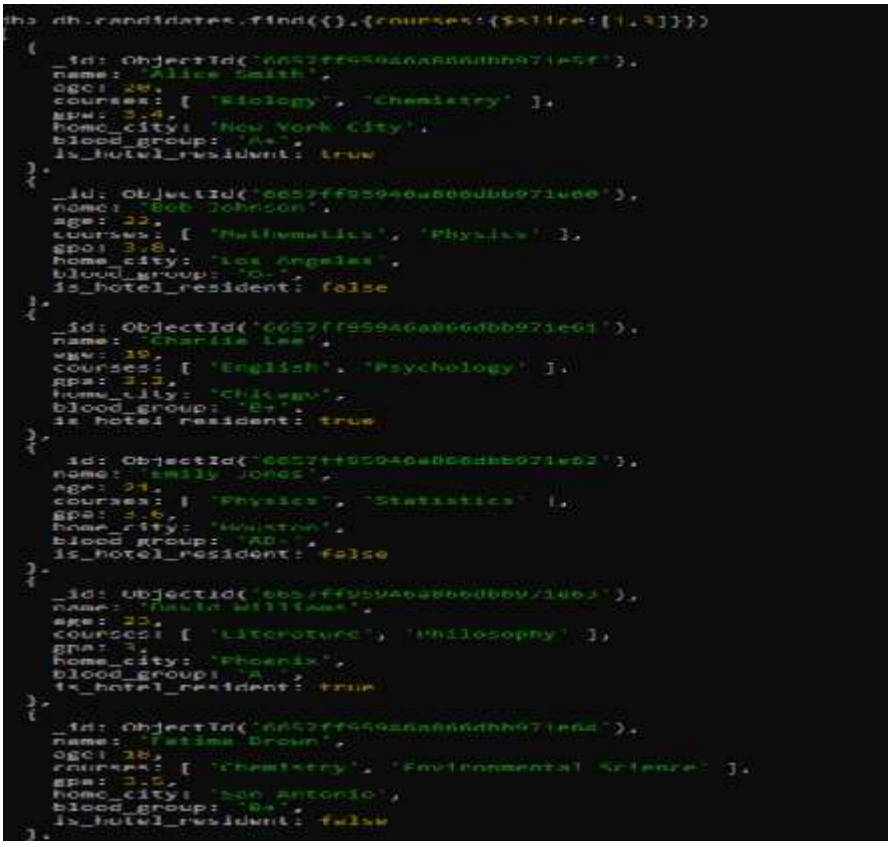
$elemMatch

This is a MongoDB operator that allows you to find documents within an array that contain an element that matches a specified condition

$slice operation:

This operator limits the number of elements returned from an array. It takes two arguments: o The first argument specifies the starting index of the slice (zero-based). The second argument (optional) specifies the number of elements to return. It returns all elements from the starting index to the end of the array.

This part defines a projection document using curly braces {}.



$slice operation [1:3]