3.Execute query selectors (comparison selectors, logical selectors) and list out the results on any collection

Selectors

In MongoDB, selectors are used to filter documents within a collection based on specific criteria. These criteria are defined using comparison operators and logical operators.

Comparison Operators in MongoDB Queries (Table)

Operator	Description	Example (find students with age > 20)
$gt	Greater Than	db.students.find({ age: { $gt: 20 } })
$gte	Greater Than or Equal To	db.students.find({ age: { $gte: 20 } })
$lt	Less Than	db.students.find({ age: { $lt: 20 } })
$lte	Less Than or Equal To	db.students.find({ age: { $lte: 20 } })
$eq	Equal To	db.students.find({ age: { $eq: 20 } })
$ne	Not Equal To	db.students.find({ age: { $ne: 20 } })

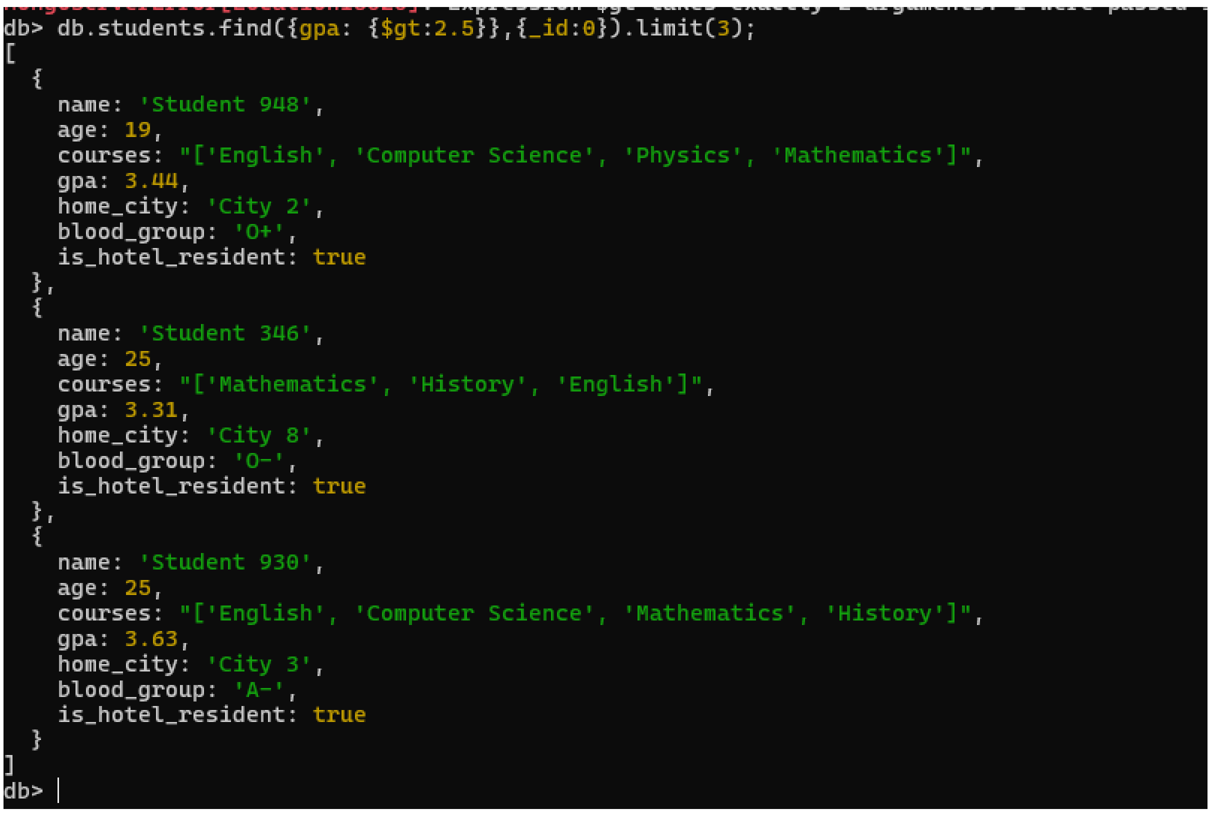
Logical Operators in MongoDB Queries (Table)

Opertor	Description	Example
$and	Matches documents that meet **all** specified conditions.	{ age: { $gt: 20 }, city: "New York" }
$or	Matches documents that meet **at least one** of the specified conditions.	{ gpa: { $gt: 3.0 }, major: "Computer Science" }

Example:

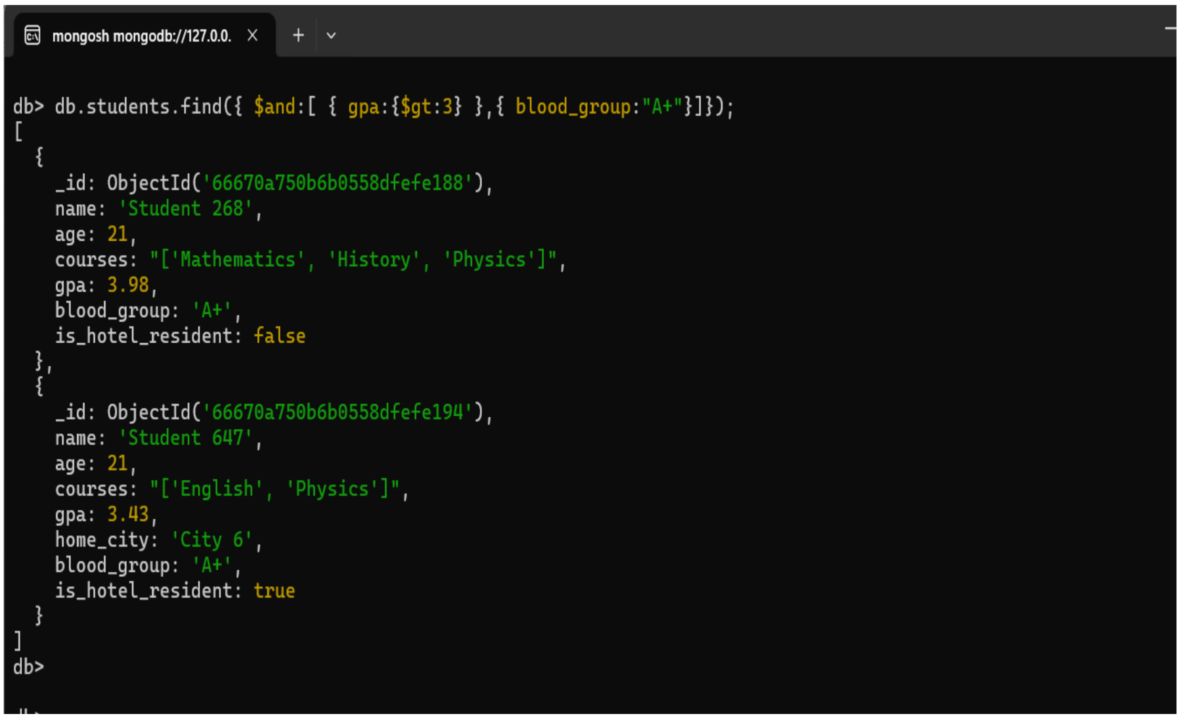
Collection name : students

1.Comparison Operator



```
db> db.students.find({gpa: {$gt:2.5}},{_id:0}).limit(3);
[
  {
    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
  },
  {
    name: 'Student 346',
    age: 25,
    courses: "['Mathematics', 'History', 'English']",
    gpa: 3.31,
    home_city: 'City 8',
    blood_group: 'O-',
    is_hotel_resident: true
  },
  {
    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
  }
]
db> |
```

2. **Logical Operator:**



```
db> db.students.find({ $and:[ { gpa:{$gt:3} },{ blood_group:"A+"}]});
[
  {
    _id: ObjectId('66670a750b6b0558dfefe188'),
    name: 'Student 268',
    age: 21,
    courses: "['Mathematics', 'History', 'Physics']",
    gpa: 3.98,
    blood_group: 'A+',
    is_hotel_resident: false
  },
  {
    _id: ObjectId('66670a750b6b0558dfefe194'),
    name: 'Student 647',
    age: 21,
    courses: "['English', 'Physics']",
    gpa: 3.43,
    home_city: 'City 6',
    blood_group: 'A+',
    is_hotel_resident: true
  }
]
db>
```

Execute query selectors (Geospatial selectors, Bitwise selectors) and list out the results on any collection.

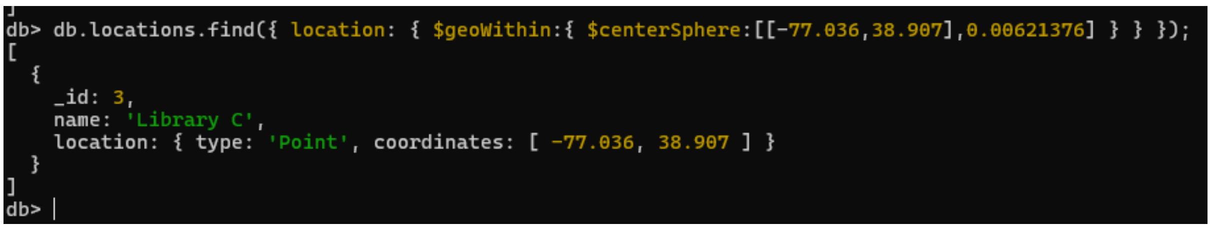
Geospatial selectors:

- In MongoDB, geospatial selectors are operators used within queries to filter documents based on their location data stored as GeoJSON geometries.
- These selectors work in conjunction with geospatial indexes (2dsphere or 2d) to efficiently perform location-based searches.

Geospatial selectors in MongoDB:

Selector	Description	Geometry	Index	Use Case
$geoIntersects	Finds documents where their GeoJSON geometry intersects with another specified GeoJSON geometry.	GeoJSON	2dsphere	Overlapping areas, complex spatial relationships
$geoWithin	Retrieves documents where their GeoJSON geometry falls entirely within a defined GeoJSON bounding area.	GeoJSON	2dsphere, **2d**	Points within a polygon, searching within a specific area
$near	Returns documents close to a specified geographic point (flat surface model). Requires a geospatial index.	Point	**2d**	Finding nearby locations (less accurate for Earth's curvature)
$nearSphere	Returns documents close to a specified geographic point (spherical model). Requires a geospatial index.	Point	2dsphere	Finding nearby locations (more accurate for Earth's curvature)

// Geospatial selectors


```
db> db.locations.find({ location: { $geoWithin:{ $centerSphere:[[-77.036,38.907],0.00621376] } } });
[
  {
    _id: 3,
    name: 'Library C',
    location: { type: 'Point', coordinates: [ -77.036, 38.907 ] }
  }
]
db> |
```

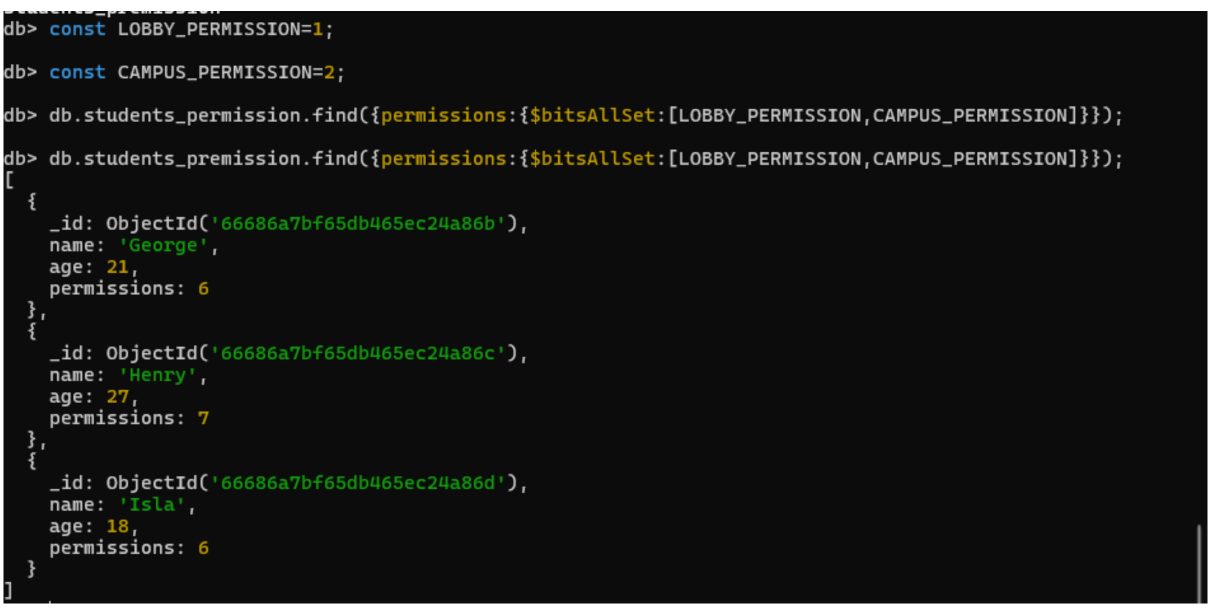
- ✓ location: This specifies that the query should target the location field within documents.
- ✓ $geoWithin: This is a geospatial selector used to filter based on location data.
- ✓ $centerSphere: This shape operator defines a circle on a sphere for the geospatial search. It requires a 2dsphere index for efficient execution.
- ✓ [[-77.036,38.907]]: This is the center point of the sphere defined by an array containing longitude and latitude in that order ([-77.036] for longitude, [38.907] for latitude).
- ✓ 0.00621376: This value specifies the radius of the circle in radians (approximately 350 meters in this case).

Bitwise selectors:

MongoDB doesn't support bitwise selectors directly for filtering documents. However, it provides the `$bit` operator to perform bitwise operations on integer fields (32-bit or 64-bit) within documents.

Operator	Description
$bit	Updates or performs bitwise operations (AND, OR, XOR) on integer fields within documents
$bitsAllSet	Matches numberic or binary values in which a set bit positions all have a value of 1
$bitsAnyClear	Matches numberic or binary values in which any bit from a set of bit positions has a value of 0
$bitsAnySet	Matches numberic or binary values in which any bit from a set of bit positions has a value of 1
$bitsAllClear	Matches numberic or binary values in which a set bit positions all have a value of 0

//bit positions for permissions



```
db> const LOBBY_PERMISSION=1;

db> const CAMPUS_PERMISSION=2;

db> db.students_permission.find({permissions:{$bitsAllSet:[LOBBY_PERMISSION,CAMPUS_PERMISSION]}});

db> db.students_premission.find({permissions:{$bitsAllSet:[LOBBY_PERMISSION,CAMPUS_PERMISSION]}});
[
  {
    _id: ObjectId('66686a7bf65db465ec24a86b'),
    name: 'George',
    age: 21,
    permissions: 6
  },
  {
    _id: ObjectId('66686a7bf65db465ec24a86c'),
    name: 'Henry',
    age: 27,
    permissions: 7
  },
  {
    _id: ObjectId('66686a7bf65db465ec24a86d'),
    name: 'Isla',
    age: 18,
    permissions: 6
  }
]
```

The find operation uses a query document to filter results:

permissions: { $bitsAllSet: [LOBBY_PERMISSION, CAMPUS_PERMISSION] }

- ✓ This part checks the permissions field of documents in the collection.
- ✓ It uses the $bitsAllSet operator to ensure that all the bits specified in the array ([LOBBY_PERMISSION, CAMPUS_PERMISSION]) are set in the permissions field.
- ✓ In this case, it searches for documents where both the LOBBY_PERMISSION bit (which has the value 1) and the CAMPUS_PERMISSION bit (which has the value 2) are set.