MONGO DB

EXPERIMENT-04

4. Create and demonstrate how projection operators ($, $elemMatch and $slice) would be used in the MongoDB.

Projection Operators in MongoDB

In MongoDB, projection operators are used within the find() method to control which fields are included or excluded in the query results. This allows you to retrieve only the specific data you need, improving performance and reducing network traffic.

Here are the three common projection operators you mentioned:

1.$
2.$elemMatch
3.$slice

1. $ (Field Selection):

- Used to specify which fields to include or exclude in the results.
- The most fundamental operator.
- Include a field by setting its value to 1.
- Exclude a field by setting its value to 0.
- Use `_id: 0` to exclude the `_id` field by default.

Syntax:

```
db.collection_name.find({},{condtion});
```

Example 1:

This query retrieves only the student having gpa greater than 3..5 and including name, gpa and the `_id` field is set -1 to sort in descending order with limit 3.

Using sort we can sort the data's in ascending or descending order.

```
db> db.student.find({ gpa:{$gte: 3.5}}).sort({ name: 1,gpa:1,_id:-1}).limit(3);
[
  {
    _id: ObjectId('665899c18fb275953df12838'),
    name: 'Bob Johnson',
    age: 22,
    courses: [ 'Computer Science', 'Mathematics', 'Physics' ],
    gpa: 3.8,
    home_city: 'Los Angeles',
    blood_group: 'O-',
    is_hotel_resident: false
  },
  {
    _id: ObjectId('665899c18fb275953df1283a'),
    name: 'Emily Jones',
    age: 21,
    courses: [ 'Mathematics', 'Physics', 'Statistics' ],
    gpa: 3.6,
    home_city: 'Houston',
    blood_group: 'AB-',
    is_hotel_resident: false
  },
  {
    _id: ObjectId('665899c18fb275953df1283c'),
    name: 'Fatima Brown',
    age: 18,
    courses: [ 'Biology', 'Chemistry', 'Environmental Science' ],
    gpa: 3.5,
    home_city: 'San Antonio',
    blood_group: 'B+',
    is_hotel_resident: false
  }
]
```

Example 2:

This query retrieves only the name, gpa and courses which it will show only 1st course and the count().

```
db> db.student.find({}, { name: 1, gpa: 1, courses: { $slice: 1 } }).count();
12
```

Example 3:

This query retrieves only the name, gpa and courses which it will show only 1st course.

- A value of `1` includes the field.

```
db> db.student.find({},{
... name:1,
... gpa:1,
... courses:{$slice:1}
... });
[
  {
    _id: ObjectId('665899c18fb275953df12837'),
    name: 'Alice Smith',
    courses: [ 'English' ],
    gpa: 3.4
  },
  {
    _id: ObjectId('665899c18fb275953df12838'),
    name: 'Bob Johnson',
    courses: [ 'Computer Science' ],
    gpa: 3.8
  },
  {
    _id: ObjectId('665899c18fb275953df12839'),
    name: 'Charlie Lee',
    courses: [ 'History' ],
    gpa: 3.2
  },
  {
    _id: ObjectId('665899c18fb275953df1283a'),
    name: 'Emily Jones',
    courses: [ 'Mathematics' ],
    gpa: 3.6
  },
```

Example 4:

This query retrieves the name, gpa less than 3.5 and _id is set 1 and the sort function is used.

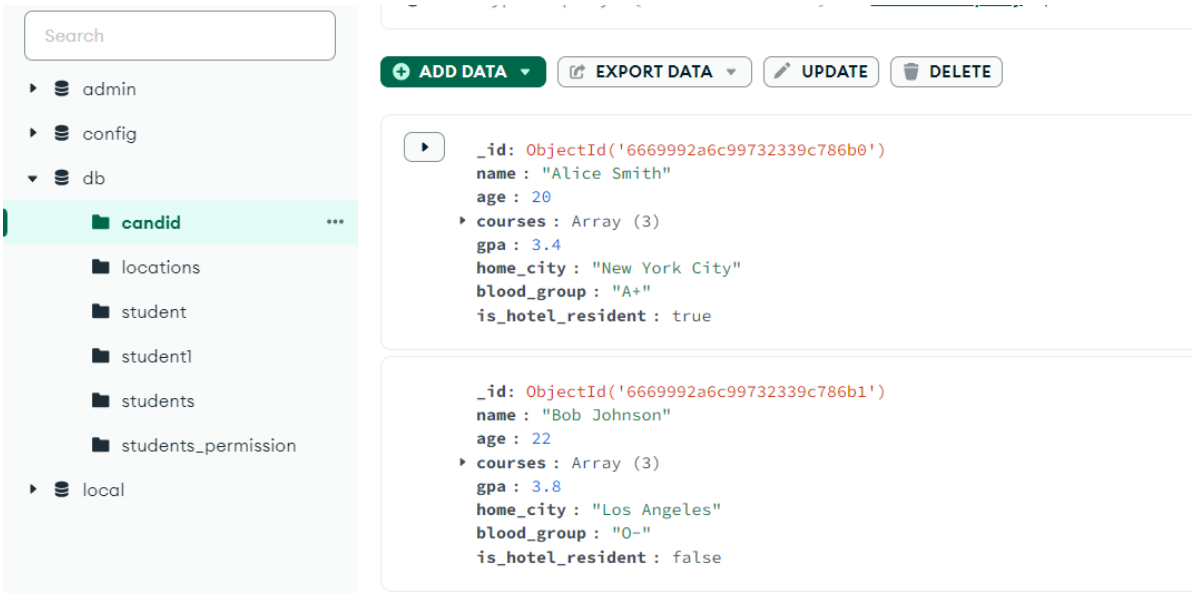
```
db> db.student.find({ gpa:{$lt: 3.5}}).sort({ name: 1,gpa:1,_id:1}).limit(3);
[
  {
    _id: ObjectId('665899c18fb275953df12837'),
    name: 'Alice Smith',
    age: 20,
    courses: [ 'English', 'Biology', 'Chemistry' ],
    gpa: 3.4,
    home_city: 'New York City',
    blood_group: 'A+',
    is_hotel_resident: true
  },
  {
    _id: ObjectId('665899c18fb275953df12839'),
    name: 'Charlie Lee',
    age: 19,
    courses: [ 'History', 'English', 'Psychology' ],
    gpa: 3.2,
    home_city: 'Chicago',
    blood_group: 'B+',
    is_hotel_resident: true
  },
  {
    _id: ObjectId('665899c18fb275953df1283b'),
    name: 'David Williams',
    age: 23,
    courses: [ 'English', 'Literature', 'Philosophy' ],
    gpa: 3,
    home_city: 'Phoenix',
    blood_group: 'A-',
    is_hotel_resident: true
  }
]
db>
```

Example 5:

This query retrieves only the name, gpa and courses which it will show only 1st two course in array.

```
db> db.student.find({}, { name: 1, gpa: 1, courses: { $slice: 2 } ,_id:0});
[
  { name: 'Alice Smith', courses: [ 'English', 'Biology' ], gpa: 3.4 },
  {
    name: 'Bob Johnson',
    courses: [ 'Computer Science', 'Mathematics' ],
    gpa: 3.8
  },
  { name: 'Charlie Lee', courses: [ 'History', 'English' ], gpa: 3.2 },
  {
    name: 'Emily Jones',
    courses: [ 'Mathematics', 'Physics' ],
    gpa: 3.6
  },
  {
    name: 'David Williams',
    courses: [ 'English', 'Literature' ],
    gpa: 3
  },
  {
    name: 'Fatima Brown',
    courses: [ 'Biology', 'Chemistry' ],
    gpa: 3.5
  },
  {
    name: 'Gabriel Miller',
    courses: [ 'Computer Science', 'Engineering' ],
    gpa: 3.9
  },
```

Now adding the new collections to data base with collection name-candid.



Example 01:

Retrieve all the candidates.

```
db> db.candid.find()
[
  {
    _id: ObjectId('6669992a6c99732339c786b0'),
    name: 'Alice Smith',
    age: 20,
    courses: [ 'English', 'Biology', 'Chemistry' ],
    gpa: 3.4,
    home_city: 'New York City',
    blood_group: 'A+',
    is_hotel_resident: true
  },
  {
    _id: ObjectId('6669992a6c99732339c786b1'),
    name: 'Bob Johnson',
    age: 22,
    courses: [ 'Computer Science', 'Mathematics', 'Physics' ],
    gpa: 3.8,
    home_city: 'Los Angeles',
    blood_group: 'O-',
    is_hotel_resident: false
  },
  {
    _id: ObjectId('6669992a6c99732339c786b2'),
    name: 'Charlie Lee',
    age: 19,
    courses: [ 'History', 'English', 'Psychology' ],
    gpa: 3.2,
    home_city: 'Chicago',
    blood_group: 'B+',
    is_hotel_resident: true
  },
```

Example 02:

Counting the candidates in db.

```
db> db.candid.find().count();
12
db>
```

Example 03:

This query retrieves only the name , age and count.

```
db> db.candid.find({},{name:1,age:1}).count();
12
db> db.candid.find({},{name:1,age:1})
[
  {
    _id: ObjectId('6669992a6c99732339c786b0'),
    name: 'Alice Smith',
    age: 20
  },
  {
    _id: ObjectId('6669992a6c99732339c786b1'),
    name: 'Bob Johnson',
    age: 22
  },
  {
    _id: ObjectId('6669992a6c99732339c786b2'),
    name: 'Charlie Lee',
    age: 19
  },
  {
    _id: ObjectId('6669992a6c99732339c786b3'),
    name: 'Emily Jones',
    age: 21
  },
  {
    _id: ObjectId('6669992a6c99732339c786b4'),
    name: 'David Williams',
    age: 23
  },
  {
    _id: ObjectId('6669992a6c99732339c786b5'),
    name: 'Fatima Brown',
    age: 18
  },
  {
    _id: ObjectId('6669992a6c99732339c786b6'),
    name: 'Gabriel Miller',
    age: 24
```

Example 04:

This query retrieves all the data , excluding the `_id` field and the courses .

```
db> db.candid.find({},{_id:0,courses:0})
[
  {
    name: 'Alice Smith',
    age: 20,
    gpa: 3.4,
    home_city: 'New York City',
    blood_group: 'A+',
    is_hotel_resident: true
  },
  {
    name: 'Bob Johnson',
    age: 22,
    gpa: 3.8,
    home_city: 'Los Angeles',
    blood_group: 'O-',
    is_hotel_resident: false
  },
  {
    name: 'Charlie Lee',
    age: 19,
    gpa: 3.2,
    home_city: 'Chicago',
    blood_group: 'B+',
    is_hotel_resident: true
  },
  {
    name: 'Emily Jones',
    age: 21,
    gpa: 3.6,
    home_city: 'Houston',
    blood_group: 'AB-',
    is_hotel_resident: false
  },
```

2. $elemMatch (Array Filtering):

- Used to project the first element in an array that matches a specified condition.
- Within the projection document, use the field name containing the array and the **$elemMatch operator.**
- Specify the condition for matching elements within the array.

Example 01:

This query retrieves the courses field array, it projects only the first element where the course is computer science.

```
db> db.candid.find({courses:{$elemMatch:{$eq:"Computer Science"}}},{name:1,"courses.$":1});
[
  {
    _id: ObjectId('6669992a6c99732339c786b1'),
    name: 'Bob Johnson',
    courses: [ 'Computer Science' ]
  },
  {
    _id: ObjectId('6669992a6c99732339c786b6'),
    name: 'Gabriel Miller',
    courses: [ 'Computer Science' ]
  },
  {
    _id: ObjectId('6669992a6c99732339c786ba'),
    name: 'Kevin Lewis',
    courses: [ 'Computer Science' ]
  }
]
db>
```

Example 02:

```
db> db.candid.find({courses:{$elemMatch:{$eq:"Computer Science"}}},{name:1,"courses.$":1}).count();
3
db>
```

This query retrieves the courses field array, it projects only the first element where the course is computer science count().

Example 03:

This query retrieves the courses field array, it projects only the first element where the course is not equal to computer science with count.

```
db> db.candid.find({courses:{$elemMatch:{$ne:"Computer Science"}}},{name:1,"courses.$":1}).count();
12
db> db.candid.find({courses:{$elemMatch:{$ne:"Computer Science"}}},{name:1,"courses.$":1});
[
  {
    _id: ObjectId('6669992a6c99732339c786b0'),
    name: 'Alice Smith',
    courses: [ 'English' ]
  },
  {
    _id: ObjectId('6669992a6c99732339c786b1'),
    name: 'Bob Johnson',
    courses: [ 'Mathematics' ]
  },
  {
    _id: ObjectId('6669992a6c99732339c786b2'),
    name: 'Charlie Lee',
    courses: [ 'History' ]
  },
  {
    _id: ObjectId('6669992a6c99732339c786b3'),
    name: 'Emily Jones',
    courses: [ 'Mathematics' ]
  },
  {
    _id: ObjectId('6669992a6c99732339c786b4'),
    name: 'David Williams',
    courses: [ 'English' ]
  },
  {
    _id: ObjectId('6669992a6c99732339c786b5'),
    name: 'Fatima Brown',
    courses: [ 'Biology' ]
  },
```

3. $slice (Array Subset):

- Used to project a specific subset (slice) of elements from an array.
- Within the projection document, use the field name containing the array and the `$slice` operator.
- Provide a positive integer (n) to return the first n elements.
- Provide a negative integer (-n) to return the last n elements.
- Optionally, you can specify a skip value to offset the starting position within the array.

Example 01:

Return the first 2 elements of the "courses" array ,including name and excluding _id.

```
db> db.candid.find({},{_id:0,name:1,courses:{$slice:2}});
[
  { name: 'Alice Smith', courses: [ 'English', 'Biology' ] },
  {
    name: 'Bob Johnson',
    courses: [ 'Computer Science', 'Mathematics' ]
  },
  { name: 'Charlie Lee', courses: [ 'History', 'English' ] },
  { name: 'Emily Jones', courses: [ 'Mathematics', 'Physics' ] },
  { name: 'David Williams', courses: [ 'English', 'Literature' ] },
  { name: 'Fatima Brown', courses: [ 'Biology', 'Chemistry' ] },
  {
    name: 'Gabriel Miller',
    courses: [ 'Computer Science', 'Engineering' ]
  },
  {
    name: 'Hannah Garcia',
    courses: [ 'History', 'Political Science' ]
  },
  { name: 'Isaac Clark', courses: [ 'English', 'Creative Writing' ] },
  { name: 'Jessica Moore', courses: [ 'Biology', 'Ecology' ] },
  {
    name: 'Kevin Lewis',
    courses: [ 'Computer Science', 'Artificial Intelligence' ]
  },
  { name: 'Lily Robinson', courses: [ 'History', 'Art History' ] }
]
```

Example 02:

Return the first 1 element of the "courses" array ,including name and excluding _id.

```
db> db.candid.find({},{_id:0,name:1,courses:{$slice:1}});
[
  { name: 'Alice Smith', courses: [ 'English' ] },
  { name: 'Bob Johnson', courses: [ 'Computer Science' ] },
  { name: 'Charlie Lee', courses: [ 'History' ] },
  { name: 'Emily Jones', courses: [ 'Mathematics' ] },
  { name: 'David Williams', courses: [ 'English' ] },
  { name: 'Fatima Brown', courses: [ 'Biology' ] },
  { name: 'Gabriel Miller', courses: [ 'Computer Science' ] },
  { name: 'Hannah Garcia', courses: [ 'History' ] },
  { name: 'Isaac Clark', courses: [ 'English' ] },
  { name: 'Jessica Moore', courses: [ 'Biology' ] },
  { name: 'Kevin Lewis', courses: [ 'Computer Science' ] },
  { name: 'Lily Robinson', courses: [ 'History' ] }
]
db>
```

Example 03:

Return the last 1 element of the "courses" array ,including name and excluding _id.

```
db> db.candid.find({},{_id:0,name:1,courses:{$slice:-1}});
[
  { name: 'Alice Smith', courses: [ 'Chemistry' ] },
  { name: 'Bob Johnson', courses: [ 'Physics' ] },
  { name: 'Charlie Lee', courses: [ 'Psychology' ] },
  { name: 'Emily Jones', courses: [ 'Statistics' ] },
  { name: 'David Williams', courses: [ 'Philosophy' ] },
  { name: 'Fatima Brown', courses: [ 'Environmental Science' ] },
  { name: 'Gabriel Miller', courses: [ 'Robotics' ] },
  { name: 'Hannah Garcia', courses: [ 'Sociology' ] },
  { name: 'Isaac Clark', courses: [ 'Film Studies' ] },
  { name: 'Jessica Moore', courses: [ 'Marine Science' ] },
  { name: 'Kevin Lewis', courses: [ 'Cybersecurity' ] },
  { name: 'Lily Robinson', courses: [ 'Music History' ] }
]
```

Example 04:

Return the first 3 elements of the "courses" array ,including name and excluding _id.

```
db> db.candid.find({},{_id:0,name:1,courses:{$slice:3}});
[
  {
    name: 'Alice Smith',
    courses: [ 'English', 'Biology', 'Chemistry' ]
  },
  {
    name: 'Bob Johnson',
    courses: [ 'Computer Science', 'Mathematics', 'Physics' ]
  },
  {
    name: 'Charlie Lee',
    courses: [ 'History', 'English', 'Psychology' ]
  },
  {
    name: 'Emily Jones',
    courses: [ 'Mathematics', 'Physics', 'Statistics' ]
  },
  {
    name: 'David Williams',
    courses: [ 'English', 'Literature', 'Philosophy' ]
  },
  {
    name: 'Fatima Brown',
    courses: [ 'Biology', 'Chemistry', 'Environmental Science' ]
  },
  {
    name: 'Gabriel Miller',
    courses: [ 'Computer Science', 'Engineering', 'Robotics' ]
  },
  {
    name: 'Hannah Garcia',
    courses: [ 'History', 'Political Science', 'Sociology' ]
  },
  {
    name: 'Isaac Clark',
    courses: [ 'English', 'Creative Writing', 'Film Studies' ]
  },
```

Example 05:

Return the last 2 elements of the "courses" array ,including name and excluding _id.

```
db> db.candid.find({},{_id:0,name:1,courses:{$slice:-2}});
[
  { name: 'Alice Smith', courses: [ 'Biology', 'Chemistry' ] },
  { name: 'Bob Johnson', courses: [ 'Mathematics', 'Physics' ] },
  { name: 'Charlie Lee', courses: [ 'English', 'Psychology' ] },
  { name: 'Emily Jones', courses: [ 'Physics', 'Statistics' ] },
  { name: 'David Williams', courses: [ 'Literature', 'Philosophy' ] },
  {
    name: 'Fatima Brown',
    courses: [ 'Chemistry', 'Environmental Science' ]
  },
  { name: 'Gabriel Miller', courses: [ 'Engineering', 'Robotics' ] },
  {
    name: 'Hannah Garcia',
    courses: [ 'Political Science', 'Sociology' ]
  },
  {
    name: 'Isaac Clark',
    courses: [ 'Creative Writing', 'Film Studies' ]
  },
  { name: 'Jessica Moore', courses: [ 'Ecology', 'Marine Science' ] },
  {
    name: 'Kevin Lewis',
    courses: [ 'Artificial Intelligence', 'Cybersecurity' ]
  },
  {
    name: 'Lily Robinson',
    courses: [ 'Art History', 'Music History' ]
  }
]
```

Projection operators are applied after the query filters documents. They control the fields that are returned for the matching documents. By effectively using these operators, you can optimize your MongoDB queries to retrieve only the data you need, enhancing performance and data transfer efficiency.