CLASS 5

PROJECTION OPERATORS

PROJECTION:

MongoDB query and projection operators allow users to control queries and results. Query operators help to filter data based on specific conditions.

E.g., $eq, $and, $exists, etc. Projection operators in MongoDB control which fields should be shown in the results.

How MongoDB projection works

MongoDB projection 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
- Filter the data set without impacting the overall database performance

Because MongoDB projection is built on top of the existing **find()** method, you can use any projection query without significant modifications to the existing functions. Plus, projection is a key factor when finding user-specific data from a given data set.

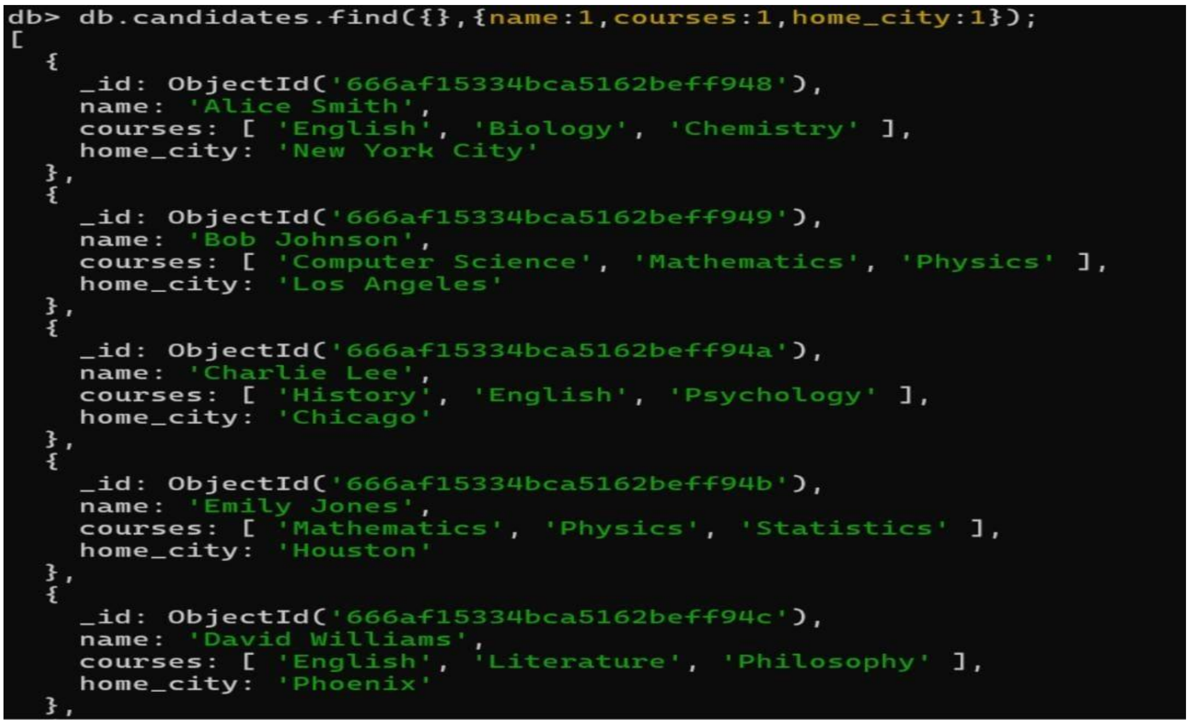
Syntax:

```
db.collection_name.find({},{<field> : <value>})
```

- If the value of the field is set to 1 or true, then it means the field will include in the return document.
- If the value of the field is set to 0 or false, then it means the field will not include in the return document.
- You are allowed to use projection operators, but find() method does not support following projection operators, i.e., $, $elemMatch, $slice, and $meta.
- There is no need to set _id field to 1 to return _id field, the find() method always return _id unless you set a _id field to 0.

Retrieve Name,Courses And Home Town:

Set a fields values to 1 in the projection document to include it in the returned results.



```
db> db.candidates.find({},{name:1,courses:1,home_city:1});
[
  {
    _id: ObjectId('666af15334bca5162beff948'),
    name: 'Alice Smith',
    courses: [ 'English', 'Biology', 'Chemistry' ],
    home_city: 'New York City'
  },
  {
    _id: ObjectId('666af15334bca5162beff949'),
    name: 'Bob Johnson',
    courses: [ 'Computer Science', 'Mathematics', 'Physics' ],
    home_city: 'Los Angeles'
  },
  {
    _id: ObjectId('666af15334bca5162beff94a'),
    name: 'Charlie Lee',
    courses: [ 'History', 'English', 'Psychology' ],
    home_city: 'Chicago'
  },
  {
    _id: ObjectId('666af15334bca5162beff94b'),
    name: 'Emily Jones',
    courses: [ 'Mathematics', 'Physics', 'Statistics' ],
    home_city: 'Houston'
  },
  {
    _id: ObjectId('666af15334bca5162beff94c'),
    name: 'David Williams',
    courses: [ 'English', 'Literature', 'Philosophy' ],
    home_city: 'Phoenix'
  },
```

To reduce the workload, MongoDB offers the below operators that can be used within a projection query:

- $elemMatch
- $slice

$elemMatch

The MongoDB $elemMatch operator is used to query documents that contain arrays and match the specified conditions within those arrays. It is particularly useful when dealing with arrays that have multiple elements, and you want to find documents that satisfy specific criteria on those elements.

The syntax for MongoDB $elemMatch operator is shown below:

{ <field>: { $elemMatch: { <query1>, <query2>, ... } } }

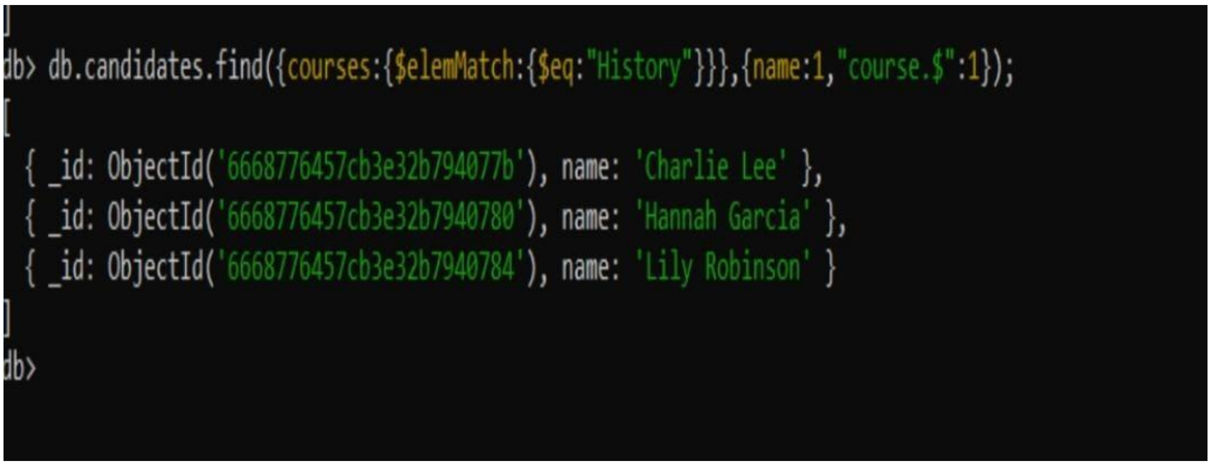
Limitations of the $elemMatch operator :

- Regardless of the ordering of fields in the document, the field to which $elemMatch projection is applied will be returned as the last field of the document.

- find() operations done on MongoDB views do not support $elemMatch projection operator.

- $text query expressions are not supported with the $elemMatch operator.

Example:



```
]
db> db.candidates.find({courses:{$elemMatch:{$eq:"History"}}},{name:1,"course.$":1});
[
  { _id: ObjectId('6668776457cb3e32b794077b'), name: 'Charlie Lee' },
  { _id: ObjectId('6668776457cb3e32b7940780'), name: 'Hannah Garcia' },
  { _id: ObjectId('6668776457cb3e32b7940784'), name: 'Lily Robinson' }
]
db>
```

$slice:

The $slice modifier limits the number of array elements during a $push operation. To project, or return, a specified number of array elements from a read operation, see the $slice projection operator instead.

To use the $slice modifier, it must appear with the $each modifier. You can pass an empty array [] to the $each modifier such that only the $slice modifier has an effect.

Syntax:

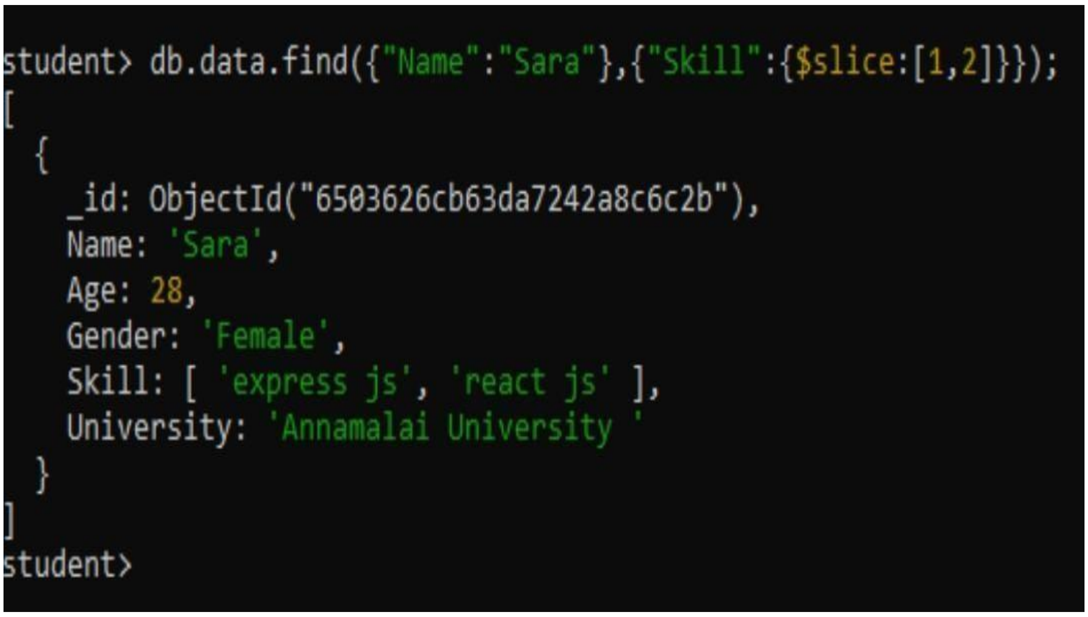
```
{
  $push: {
     <field>: {
       $each: [ <value1>, <value2>, ... ],
       $slice: <num>
     }
  }
}
```

Limitations in $slice operator:

- With the introduction of MongoDB 4.4, the $slice operator will only return the sliced element. It will not return any other item in a nested array.
- The $slice operator does not support the find() operation done on MongoDB views.
- The $slice operator cannot be used in conjunction with the $ projection operator due to the MongoDB restriction, where top-level fields can't consist of $ (dollar sign) as a part of the field name.
- Queries can't contain the $slice of an array and a field embedded in the array as part of the same statement to eliminate path collisions from MongoDB 4.4.

Example:

db.data.find({"Name":"Sara"},{"Skill":{$slice:[1,2]}})



```
student> db.data.find({"Name":"Sara"},{"Skill":{$slice:[1,2]}});
[
  {
    _id: ObjectId("6503626cb63da7242a8c6c2b"),
    Name: 'Sara',
    Age: 28,
    Gender: 'Female',
    Skill: [ 'express js', 'react js' ],
    University: 'Annamalai University '
  }
]
student>
```