Projection Operators

Projection operators specify the fields returned by an operation.

Create new collection called candidates.

Projection operators in MongoDB are used in the queries to control the fields that should be included or excluded from the result set. They can either limit the fields to be returned or specify the fields to be excluded from the results. In this section, we will look at some common projection operators available in MongoDB, such as $, $elemMatch, and $slice.

find() operations on views do not support the following Query and Projection Operators operators:

Name	**Description**
$	Projects the first element in an array that matches the query condition.
$elemMatch	Projects the first element in an array that matches the specified $elemMatch condition.
$meta	Projects the available per-document metadata.
$slice	Limits the number of elements projected from an array. Supports skip and limit slices.

1. $

The $ operator is used to project the first element in an array that matches the specified condition. It is especially useful when dealing with large arrays, and you only need the first element matching a given condition.

Syntax:

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

Example 1:

```
db.collection.find({ grades: {$gte: 80 } }, { name: 1, 'grades.$': 1 });
```

This will return only the first grades element greater than or equal to 80 along with the name field.

Example 2:Retrive Name,Age, and GPA

```
db.candidates.find({}, { name: 1, age: 1, gpa: 1 });
```

Output:

```
[
  { "name": "Alice", "age": 25, "gpa": 3.8 },
  { "name": "Bob", "age": 30, "gpa": 3.5 }
]
```

```
db.candidates.find({}, { _id: 0,courses: 0 });
```

Output:

```
db> db.candidate.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
  },
  {
    name: 'David Williams',
    age: 23,
    gpa: 3,
    home_city: 'Phoenix',
    blood_group: 'A-',
    is_hotel_resident: true
  },
  {
    name: 'Fatima Brown',
    age: 18,
    gpa: 3.5,
    home_city: 'San Antonio',
db>
```

Example 3:Find candidates Enrolled in “Computer Science” with Specific Projection

```
db.candidates.find({ courses: { $elemMatch: { $eq: “Computer Science” }
{ name: 1, “courses.$”: 1 });
```

2. $elemMatch

The $elemMatch operator matches documents in a collection that contain an array field with at least one element that satisfies multiple given conditions.

Syntax:

```
{ <field>: { $elemMatch: { <query1>, <query2>, … } } }
```

Example 1:

```
db. collection. find({
   Subjects: { $elemMatch: { score: { $gte: 80 ), type: ‘exam’ } },
});
```

This will return documents that have at least one subjects element with a score greater than or equal to 80 and a type of “exam”.

Example 2:

```
db.players.find( {}, { games: { $elemMatch: { score: { $gt: 5 } } }, joined: 1, lastlogin: 1
```

Output:

```
[
{ "joined": <date_joined_value>, "lastlogin": <date_lastlogin_value>, "games": [ { <game_object1 with score > 5>, { <game_object2 with score > 5>, ... (other games with score > 5) ] }, { "joined": <date_joined_value>, "lastlogin": <date_lastlogin_value>, "games": [ { <game_object3 with score > 5>, ... (other games with score > 5) ] },
 ]
```

3. $slice

The $slice operator is used to limit the number of elements projected from an array. It can either return the first N elements, skip the first N elements, or return elements after skipping N elements.

Syntax:

```
{ <field>: { $slice: <num_elements> } }
```

Or

```
{ <field>: { $slice: [ <skip_count>, <num_element> ] } }
```

Example 1:

```
db.collection.find( {},{name: 1, grades: {$slice: 3 } } );
```

This will return the name field and the first 3 grades elements for all documents in the collection.

```
db.collection.find( {}, { name: 1, grades: { $slice: [1,2] } } );
```

This will return the name field and the 2 grades elements after skipping the first element for all documents in the collection.

In summary, projection operators play a crucial role in retrieving specific data from MongoDB collections as they allow you to get the desired output. Using the appropriate operator for your query can help optimize the performance and efficiency of your MongoDB queries.

Example 2:Retrive All Candidates with First Two Courses

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

Output:

```
[
  {
    _id: ObjectId('668d44cb206c424696c1f908'),
    name: 'Alice Smith',
    courses: [ 'English', 'Biology' ]
  },
  {
    _id: ObjectId('668d44cb206c424696c1f909'),
    name: 'Bob Johnson',
    courses: [ 'Computer Science', 'Mathematics' ]
  },
  {
    _id: ObjectId('668d44cb206c424696c1f90a'),
    name: 'Charlie Lee',
    courses: [ 'History', 'English' ]
  },
  {
    _id: ObjectId('668d44cb206c424696c1f90b'),
    name: 'Emily Jones',
    courses: [ 'Mathematics', 'Physics' ]
  },
  {
    _id: ObjectId('668d44cb206c424696c1f90c'),
    name: 'David Williams',
    courses: [ 'English', 'Literature' ]
  },
  {
    _id: ObjectId('668d44cb206c424696c1f90d'),
    name: 'Fatima Brown',
    courses: [ 'Biology', 'Chemistry' ]
  },
  {
    _id: ObjectId('668d44cb206c424696c1f90e'),
    name: 'Gabriel Miller',
    courses: [ 'Computer Science', 'Engineering' ]
  },
  {
    _id: ObjectId('668d44cb206c424696c1f90f'),
    name: 'Hannah Garcia',
    courses: [ 'History', 'Political Science' ]
  },
  {
    _id: ObjectId('668d44cb206c424696c1f910'),
    name: 'Isaac Clark',
    courses: [ 'English', 'Creative Writing' ]
  },
```