Projection operations

Projection operators in MongoDB are used to shape the documents returned in the query results by including, excluding, or manipulating fields. Here are some commonly used projection operators.

1. Include Specific Fields: -

   db.collection.find({}, { field1: 1, field2: 1 })

2. Exclude Specific Fields:
   db.collection.find({}, { field1: 0, field2: 0 })

The examples for using projection operators in MongoDB is described below:

1.including the specific fields

{

"_id": 1,

"name": "Alice","age":

25,

"email": "alice@example.com"

}

Include only the "name" and "email" fields, excluding “age"db.users.find({}, { name: 1, email: 1, age: 0 });

In the above example name and email:1 indicates includes the nameand email.age:0 indicates excluded the age.

2.excluding the specific fields:

```
{
  "_id": 1,
  "name": "Alice",
  "age": 25,
  "email": "alice@example.com"
},
```

we want to retrieve all fields except the age field we use the command db.users.find({},{age:0}),The output is:

```
{
  "_id": 1,
  "name": "Alice",
  "email": "alice@example.com"
},
```

The {age:0} means it excludes the age field. And it retrieve rest of the data from the collections.

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.

Adding a new dataset called products:

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

The dataset is all about the new database as we are adding called product.

Retrieving the name and ratings the products:

```
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>
```

- db-Projects the first element in an array that matches the query condition.
- find ({}): This is the find method which is used to find documents in a collection.
- products: This refers to the collection named products within the current database.

The output of query has a seven documents from product collections.Each document has an _id field.

Excluding fields:

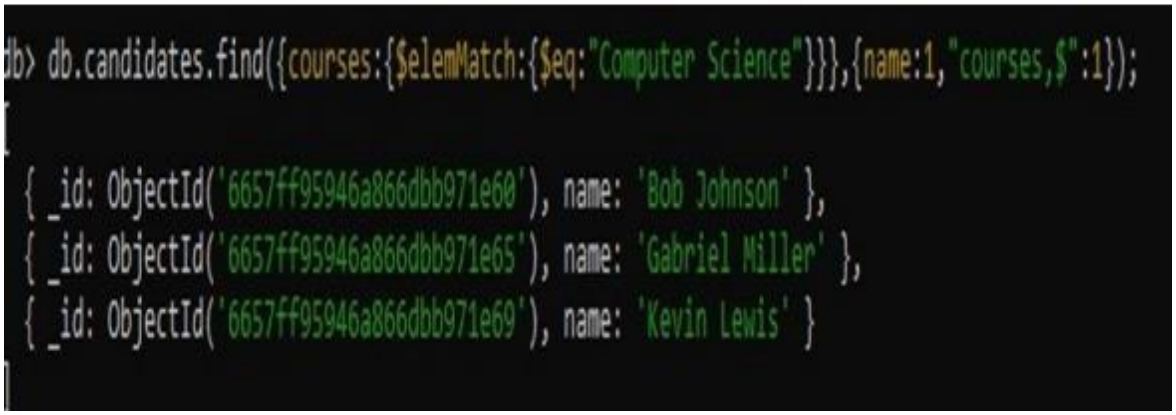
In MongoDB field is a part of a document that can store data. Fields can be top-level, embedded, or within embedded documents. When assigning values to fields at runtime, the values are converted to the specified type.

```
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'
  },
```

The output of the query shows seven documents from the products collection. However, it only displays specific fields for each documentbased on what wasn't excluded in the projection.

To get total number of counts we use:

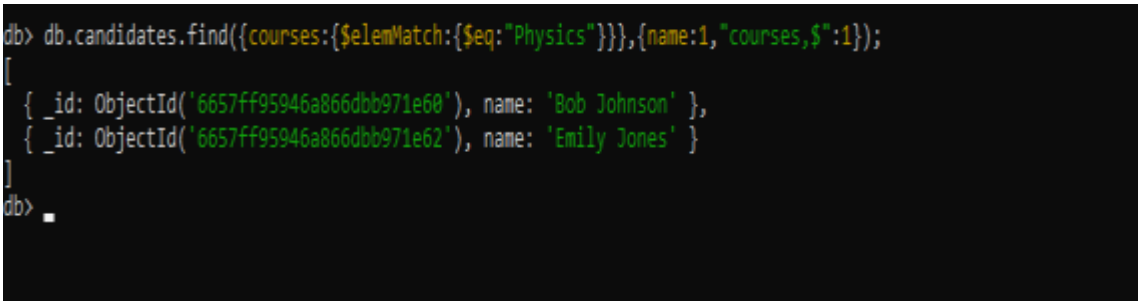
Now let us use candidates.json dataset:



The output of the query shows three documents where the "courses" field contains the value "Computer Science". The output only shows the "name" fieldand the matched course ("Computer Science") from the "courses" field for eachdocument.

$elemMatch:

In MongoDB, $elemMatch is an operator used to match documents that contain an array field with at least one element that matches allthe specified query criteria. It is commonly used in two scenarios:within the find query to filter documents and within projection to specify which array elements should be included in the returned documents.



In MongoDB, the $eq operator is used to match documents where thevalue of a field is equal to a specified value.

$slice:

In MongoDB, the $slice operator is used within a projection to limit thenumber of elements returned from an array field. This is useful when you want to retrieve only a subset of an array from a document.

Syntax:

```
db.collection.find(
 <query>,
{ <arrayField>: { $slice: <number> } }
);
```

```
db> db.candidates.find({courses:{$elemMatch:{$eq:"Physics"}}},{name:1,"courses.$":1});
[
  { _id: ObjectId('6657ff95946a866dbb971e60'), name: 'Bob Johnson' },
  { _id: ObjectId('6657ff95946a866dbb971e62'), name: 'Emily Jones' }
]
db>
```

$slice operation:

The $slice operator in MongoDB is used to project a subset of elementsfrom an array. It can be used both in queries and updates, as well as in aggregation pipelines to control which elements of an array field are included in the result.

```
db> db.candidates.find({},{courses:{$slice:1}})
[
  {
    _id: ObjectId('6657ff95946a866dbb971e5f'),
    name: 'Alice Smith',
    age: 20,
    courses: [ 'English' ],
    gpa: 3.4,
    home_city: 'New York City',
    blood_group: 'A+',
    is_hotel_resident: true
  },
  {
    _id: ObjectId('6657ff95946a866dbb971e60'),
    name: 'Bob Johnson',
    age: 22,
    courses: [ 'Computer Science' ],
    gpa: 3.8,
    home_city: 'Los Angeles',
    blood_group: 'O-',
    is_hotel_resident: false
  },
  {
    _id: ObjectId('6657ff95946a866dbb971e61'),
    name: 'Charlie Lee',
    age: 19,
    courses: [ 'History' ],
    gpa: 3.2,
    home_city: 'Chicago',
    blood_group: 'B+',
    is_hotel_resident: true
  },
  {
    _id: ObjectId('6657ff95946a866dbb971e62'),
    name: 'Emily Jones',
    age: 21,
    courses: [ 'Mathematics' ],
    gpa: 3.6,
    home_city: 'Houston',
    blood_group: 'AB-',
    is_hotel_resident: false
  },
  {
    _id: ObjectId('6657ff95946a866dbb971e63'),
    name: 'David Williams',
    age: 23,
    courses: [ 'English' ],
    gpa: 3,
    home_city: 'Phoenix',
    blood_group: 'A-',
    is_hotel_resident: true
  },
  {
    _id: ObjectId('6657ff95946a866dbb971e64'),
    name: 'Fatima Brown',
    age: 18,
    courses: [ 'Biology' ],
    gpa: 3.5,
    home_city: 'San Antonio',
    blood_group: 'B+',
```

db.candidates.find({}): This part of the query is selecting all documentsin the candidates collection.

Before slicing:

Consider a document collections and applying the slicing operations.

```
{
  "_id": 1,
  "name": "Alice",
  "scores": [85, 90, 78, 92, 88]
}
```

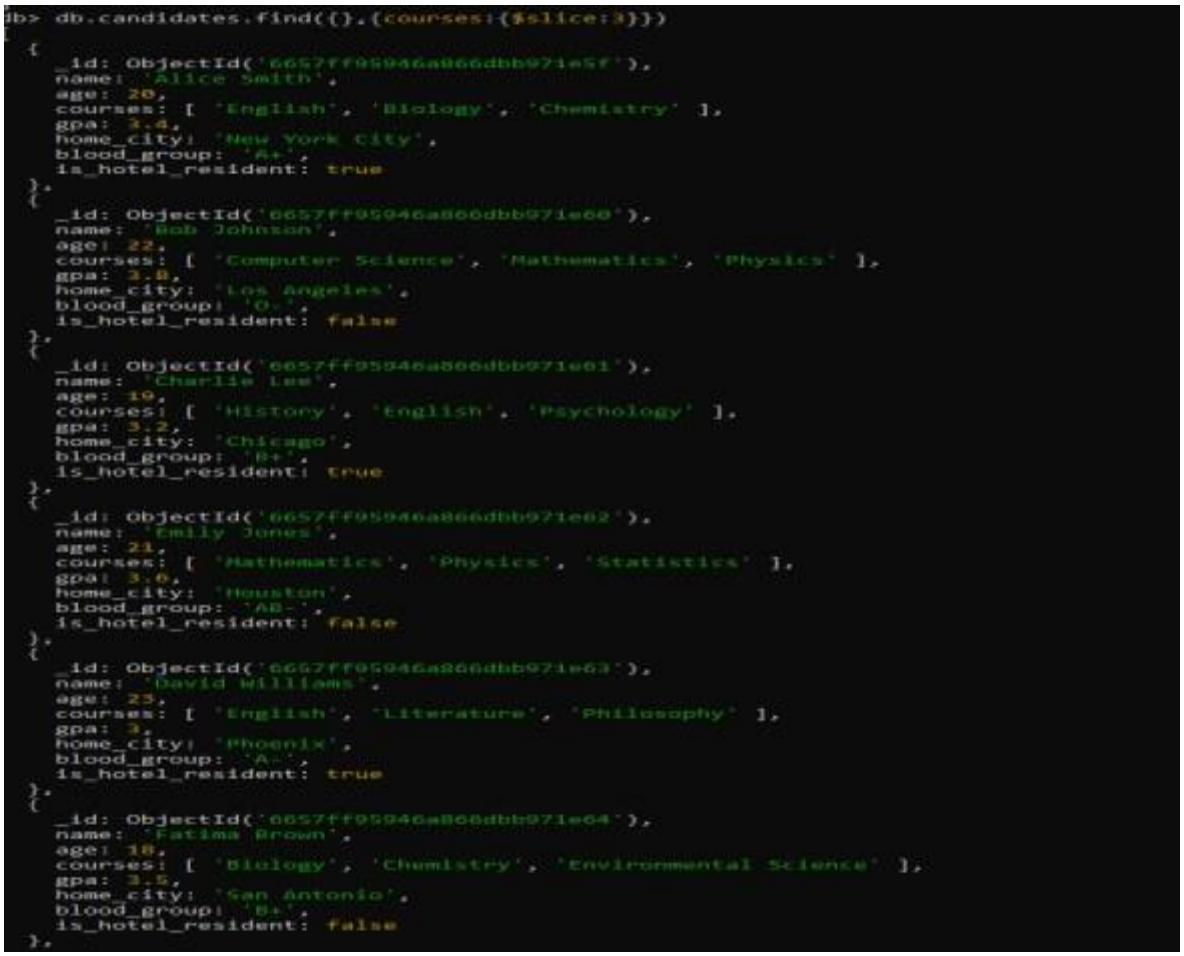
Return the first 3 elements of the score array:

```
db.students.find(
  { _id: 1 },
  { scores: { $slice: 3 } }
)
```

The output will be

```
{
  "_id": 1,
  "name": "Alice",
  "scores": [85, 90, 78]
}
```

Slice operation [1:3]



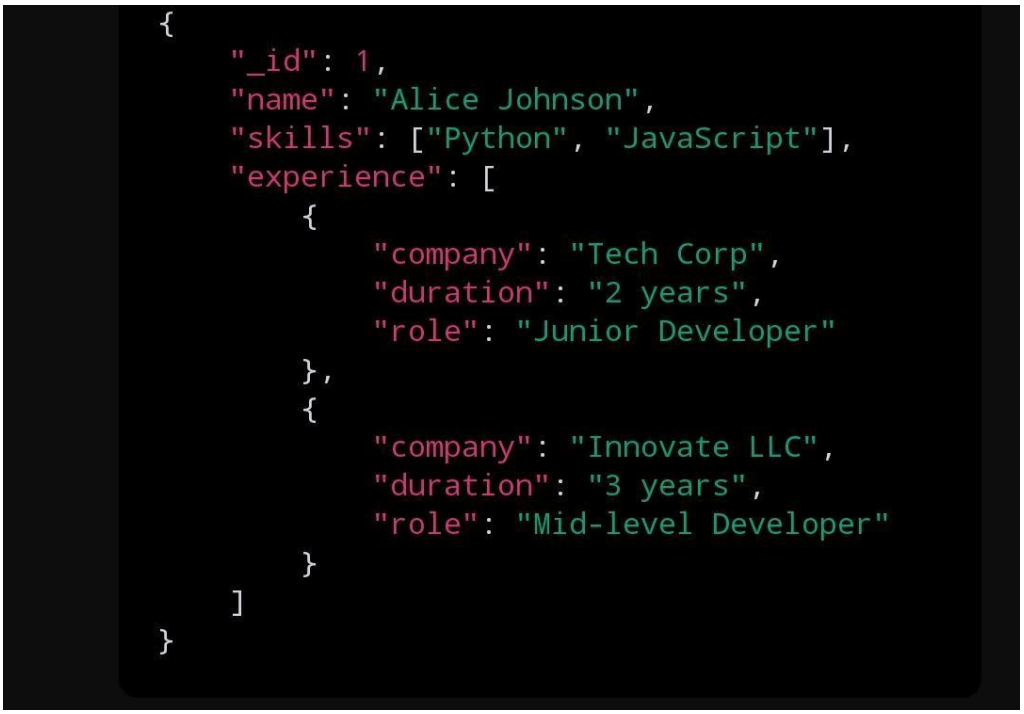
db.candidates.find({}, {courses:{$slice: [1,3]}}): This line uses the find method with two arguments to retrieve documents from the candidates collection andapply a projection to the output.

Before slicing:

Consider a document in the candidates collection before applying the $sliceoperation.

To retrieve only first two elements of the skills and experience array, the querry would be

The resulting document after applying the $slice:2 projection would be:



The query db. candidates. find({}, {courses: {$slice: 2}}) retrieves all documents from the candidates collection but limits the courses array in each document toonly include the first 2 elements.