Experiment:4

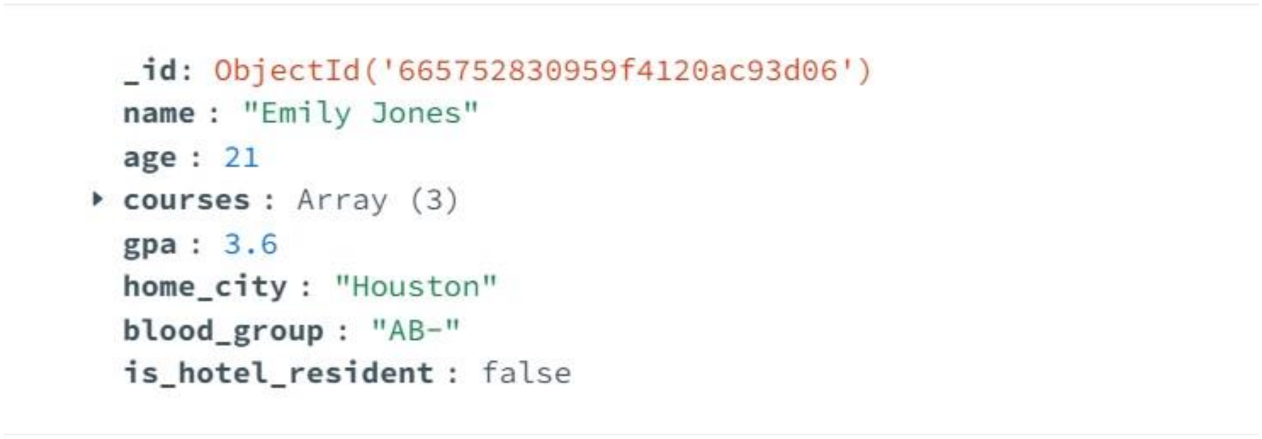
Projection Operators

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

Projection

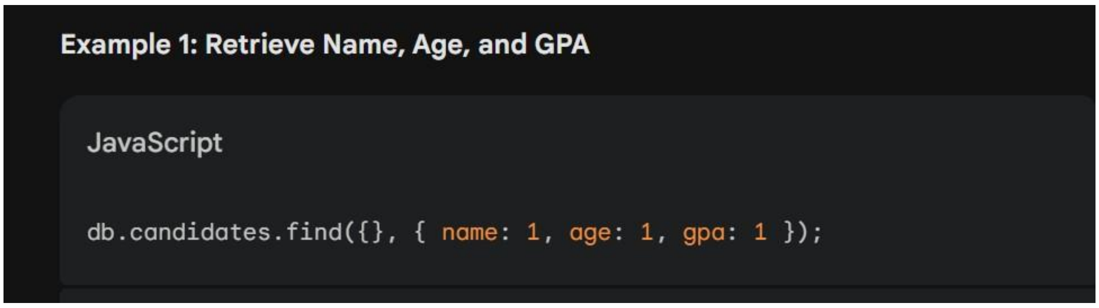
1.Create new collection called candidates.

2.Upload the dataset link.



```
_id: ObjectId('665752830959f4120ac93d06')
name : "Emily Jones"
age : 21
▸ courses : Array (3)
gpa : 3.6
home_city : "Houston"
blood_group : "AB-"
is_hotel_resident : false
```

Query:



Example 1: Retrieve Name, Age, and GPA

JavaScript

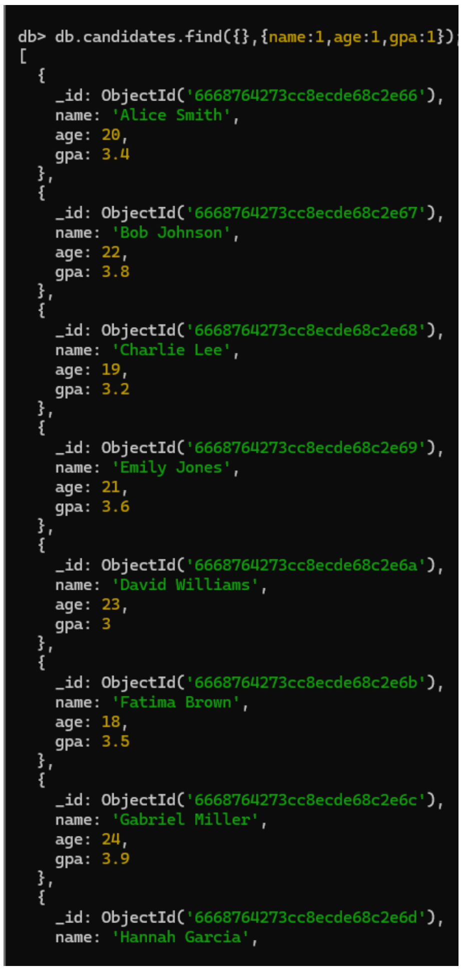
```javascript
db.candidates.find({}, { name: 1, age: 1, gpa: 1 });
```

- **`db.candidates.find({}, ...)`**: This line uses the `find` method to query the `candidates` collection within a database (`db`). The curly braces `{}` represent an empty query object, which means it will return all documents in the collection.

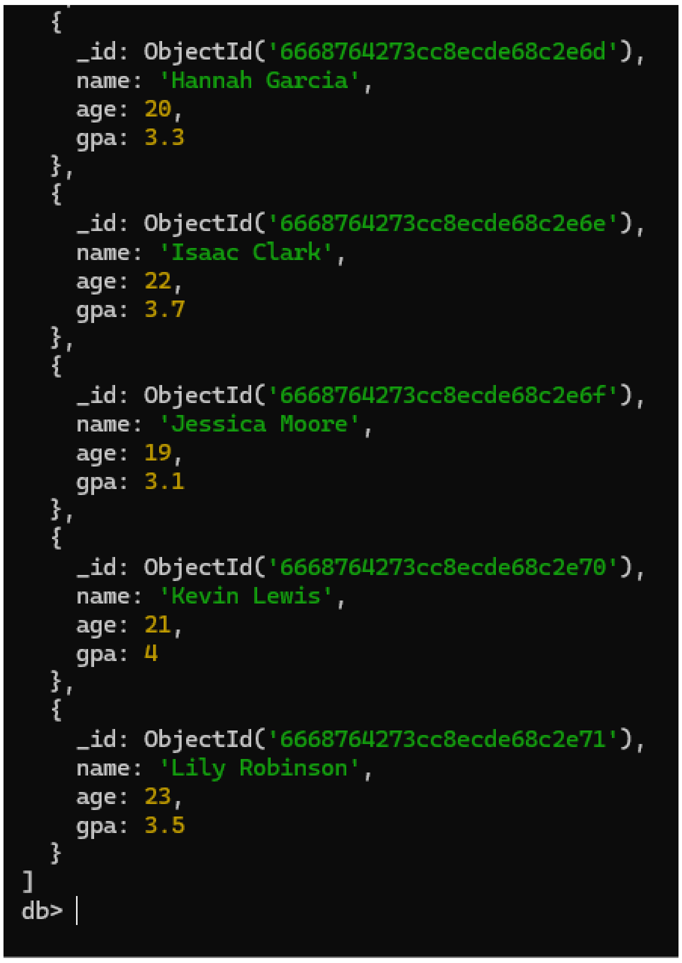
- **`{ name: 1, age: 1, gpa: 1 }`**: This part of the code specifies which fields you want to retrieve from the documents. In this case, you only want to retrieve the `name`, `age`, and `gpa` fields. The `1` values next to each field name indicate that you want to include those specific fields in the output.

Output :

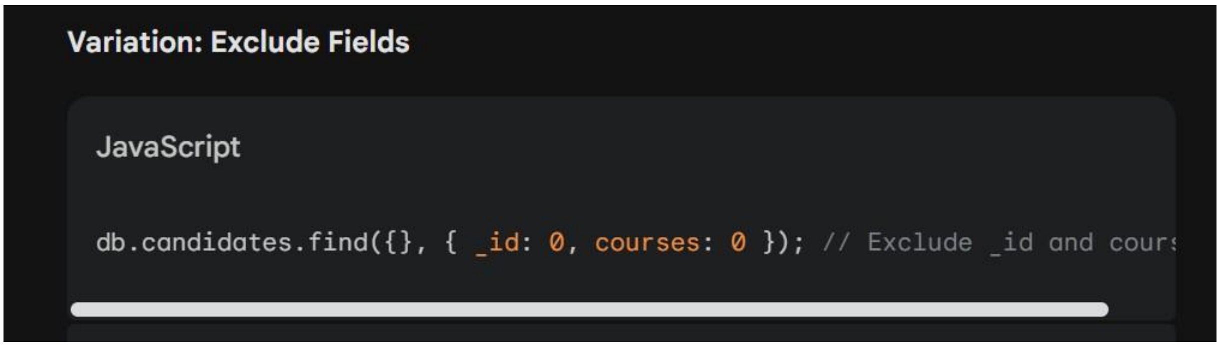
It will return a cursor containing all the documents in the `candidates` collection that include only the `name`, `age`, and `gpa` fields. You would need additional code to process the cursor and display the results.



```
db> db.candidates.find({},{name:1,age:1,gpa:1});
[
  {
    _id: ObjectId('6668764273cc8ecde68c2e66'),
    name: 'Alice Smith',
    age: 20,
    gpa: 3.4
  },
  {
    _id: ObjectId('6668764273cc8ecde68c2e67'),
    name: 'Bob Johnson',
    age: 22,
    gpa: 3.8
  },
  {
    _id: ObjectId('6668764273cc8ecde68c2e68'),
    name: 'Charlie Lee',
    age: 19,
    gpa: 3.2
  },
  {
    _id: ObjectId('6668764273cc8ecde68c2e69'),
    name: 'Emily Jones',
    age: 21,
    gpa: 3.6
  },
  {
    _id: ObjectId('6668764273cc8ecde68c2e6a'),
    name: 'David Williams',
    age: 23,
    gpa: 3
  },
  {
    _id: ObjectId('6668764273cc8ecde68c2e6b'),
    name: 'Fatima Brown',
    age: 18,
    gpa: 3.5
  },
  {
    _id: ObjectId('6668764273cc8ecde68c2e6c'),
    name: 'Gabriel Miller',
    age: 24,
    gpa: 3.9
  },
  {
    _id: ObjectId('6668764273cc8ecde68c2e6d'),
    name: 'Hannah Garcia',
```



```
  {
    _id: ObjectId('6668764273cc8ecde68c2e6d'),
    name: 'Hannah Garcia',
    age: 20,
    gpa: 3.3
  },
  {
    _id: ObjectId('6668764273cc8ecde68c2e6e'),
    name: 'Isaac Clark',
    age: 22,
    gpa: 3.7
  },
  {
    _id: ObjectId('6668764273cc8ecde68c2e6f'),
    name: 'Jessica Moore',
    age: 19,
    gpa: 3.1
  },
  {
    _id: ObjectId('6668764273cc8ecde68c2e70'),
    name: 'Kevin Lewis',
    age: 21,
    gpa: 4
  },
  {
    _id: ObjectId('6668764273cc8ecde68c2e71'),
    name: 'Lily Robinson',
    age: 23,
    gpa: 3.5
  }
]
db> 
```



Variation: Exclude Fields

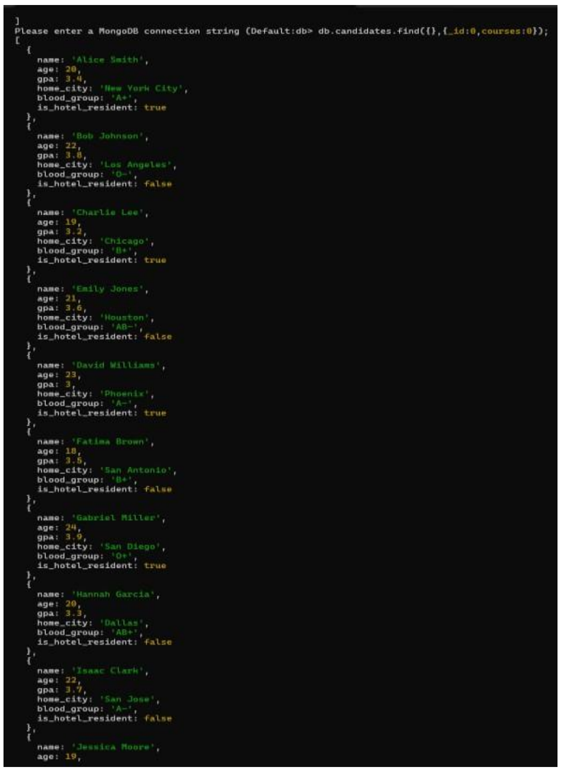
JavaScript

```javascript
db.candidates.find({}, { _id: 0, courses: 0 }); // Exclude _id and cour
```

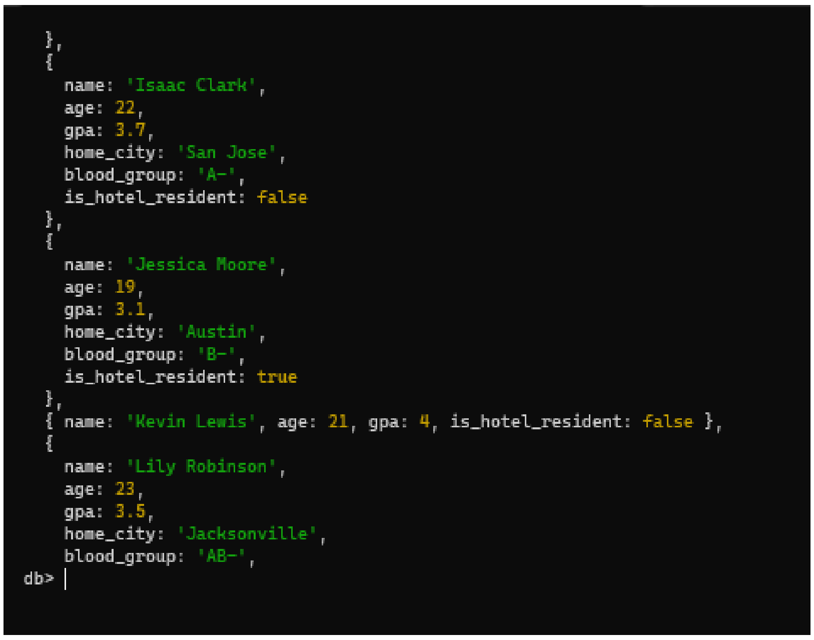
- **`db.candidates.find({}, ...)`**: This part uses the `find` method to query the `candidates` collection within a database (`db`). The empty curly braces `{}` represent an empty query object, which means it will return all documents in the collection by default.

- **`{ id: 0, courses: 0 }`**: This is the projection document that specifies which fields to exclude from the query results. In this case, setting the value of a field to `0` excludes that field from the documents returned. So here, the query will exclude the `_id` and `courses` fields from the results.

Output :

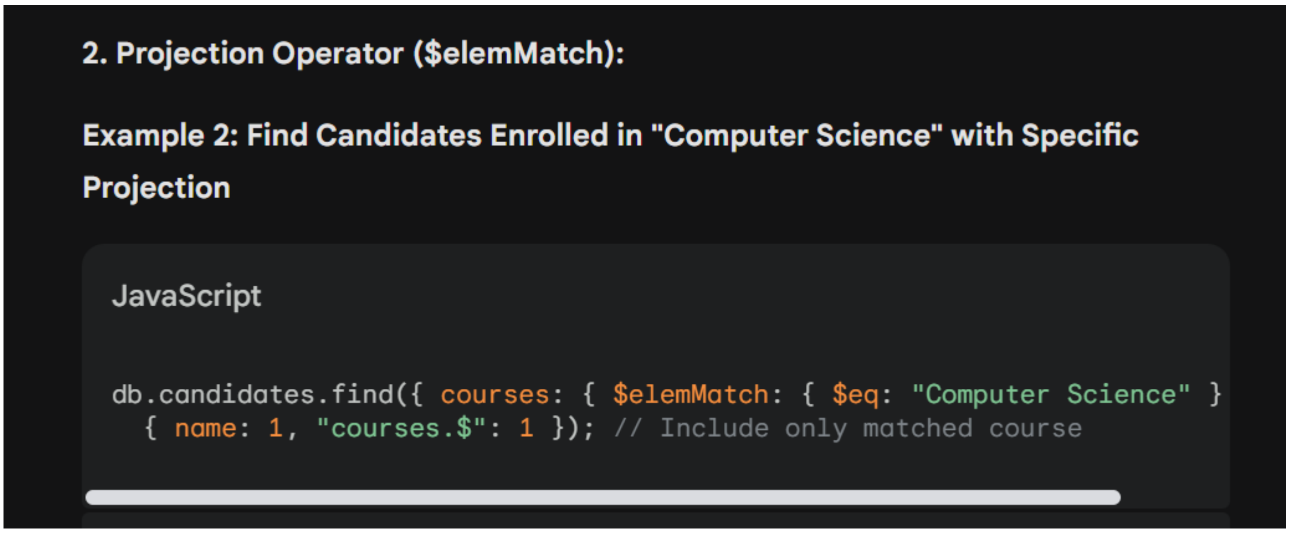


```
]
Please enter a MongoDB connection string (Default:db> db.candidates.find({},{_id:0,courses:0});
[
  {
    name: 'Alice Smith',
    age: 20,
    gpa: 3.4,
    home_city: 'New York City',
    blood_group: 'A+',
    is_hotel_resident: true
  },
  {
    name: 'Bob Johnson',
    age: 22,
    gpa: 3.8,
    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',
    blood_group: 'B+',
    is_hotel_resident: false
  },
  {
    name: 'Gabriel Miller',
    age: 24,
    gpa: 3.9,
    home_city: 'San Diego',
    blood_group: 'O+',
    is_hotel_resident: true
  },
  {
    name: 'Hannah Garcia',
    age: 20,
    gpa: 3.3,
    home_city: 'Dallas',
    blood_group: 'AB+',
    is_hotel_resident: false
  },
  {
    name: 'Isaac Clark',
    age: 22,
    gpa: 3.7,
    home_city: 'San Jose',
    blood_group: 'A-',
    is_hotel_resident: false
  },
  {
    name: 'Jessica Moore',
    age: 19,
```



```
  },
  {
    name: 'Isaac Clark',
    age: 22,
    gpa: 3.7,
    home_city: 'San Jose',
    blood_group: 'A-',
    is_hotel_resident: false
  },
  {
    name: 'Jessica Moore',
    age: 19,
    gpa: 3.1,
    home_city: 'Austin',
    blood_group: 'B-',
    is_hotel_resident: true
  },
  { name: 'Kevin Lewis', age: 21, gpa: 4, is_hotel_resident: false },
  {
    name: 'Lily Robinson',
    age: 23,
    gpa: 3.5,
    home_city: 'Jacksonville',
    blood_group: 'AB-',
db>
```

The comment // Exclude _id and courses clarifies the purpose of the projection document, which is to omit the `_id` and `courses` fields from the results.



2. Projection Operator ($elemMatch):

Example 2: Find Candidates Enrolled in "Computer Science" with Specific Projection

JavaScript

```javascript
db.candidates.find({ courses: { $elemMatch: { $eq: "Computer Science" } }
  { name: 1, "courses.$": 1 }); // Include only matched course
```

- **`db.candidates.find({...}, ...)`**: This line uses the `find` method to query the `candidates` collection within a database (`db`). The first curly braces `{...}` represent the query object that specifies conditions for selecting documents.

- **`{ courses: { $elemMatch: { $eq: "Computer Science" } } }`**: This part of the code defines the query condition. It targets the `courses` field in the documents and uses the $elemMatch operator to filter based on an element within an array.

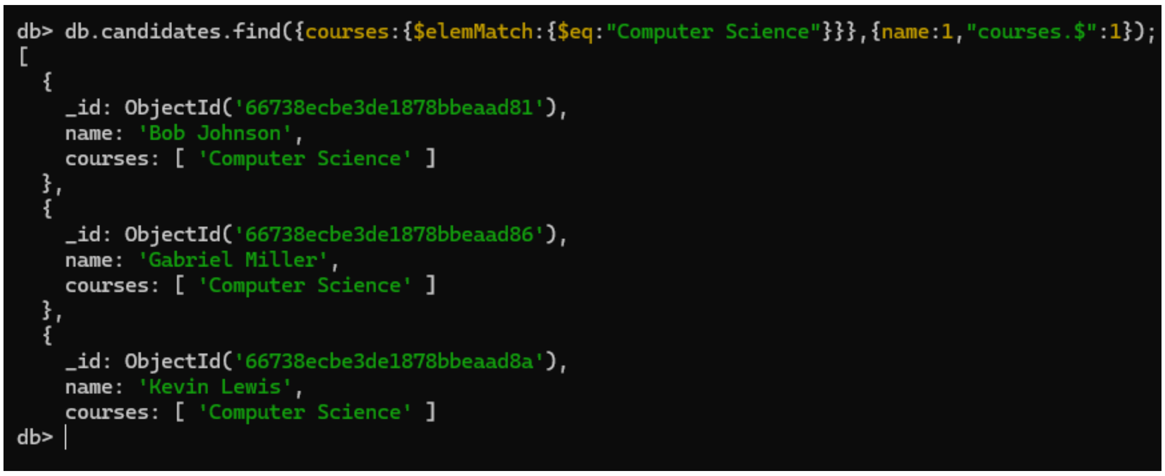
  - **`$elemMatch`**: This operator allows you to find documents containing an array where at least one element matches a specific condition.
  - **`{ $eq: "Computer Science" }`**: This is the condition within the $elemMatch operator. It uses the $eq operator to search for documents where an element in the `courses` array is equal to "Computer Science".

- **`{ name: 1, "courses.$": 1 }`**: This part of the code defines a projection document which tells MongoDB which fields to include and potentially modify when returning results.

  - **`name: 1`**: Here, you want to include the `name` field from the documents returned by the query. The `1` specifies to include this field.
  - **`courses.$`**: This is a special projection operator that allows you to include the matched element from the array field that satisfied the $elemMatch condition. In this case, it will only include the element that matched "Computer Science" within the `courses` array.

Output:

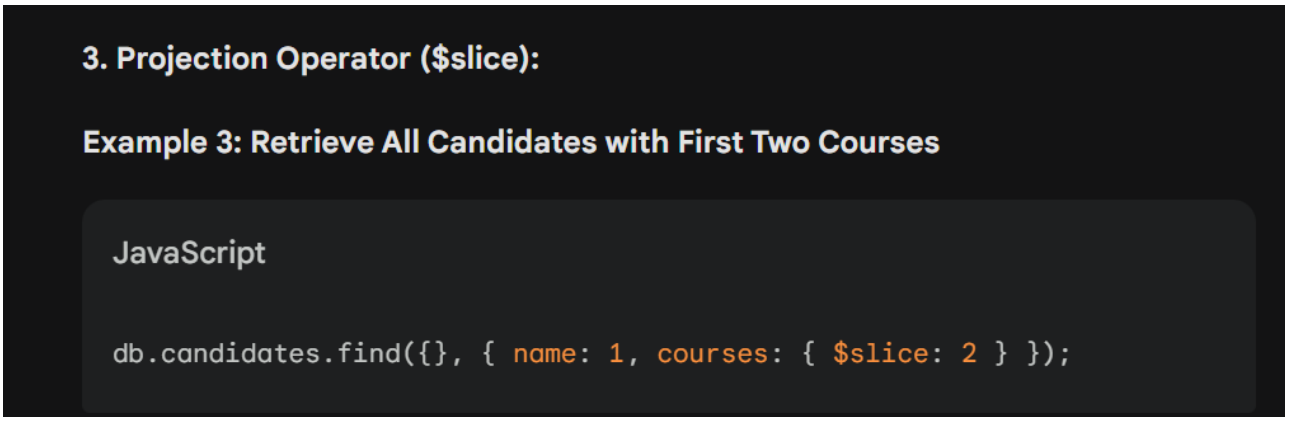
This demonstrates that the query found documents where "Computer Science" was in the `courses` array, and it only returned the `name` field and the matched course element from each document.



```
db> db.candidates.find({courses:{$elemMatch:{$eq:"Computer Science"}}},{name:1,"courses.$":1});
[
  {
    _id: ObjectId('66738ecbe3de1878bbeaad81'),
    name: 'Bob Johnson',
    courses: [ 'Computer Science' ]
  },
  {
    _id: ObjectId('66738ecbe3de1878bbeaad86'),
    name: 'Gabriel Miller',
    courses: [ 'Computer Science' ]
  },
  {
    _id: ObjectId('66738ecbe3de1878bbeaad8a'),
    name: 'Kevin Lewis',
    courses: [ 'Computer Science' ]
db> |
```

$elemMatch**:** [link]

- *Insert few players above*



3. Projection Operator ($slice):

Example 3: Retrieve All Candidates with First Two Courses

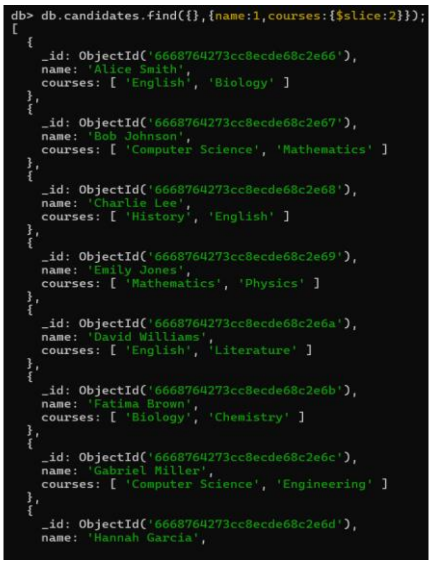
JavaScript

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

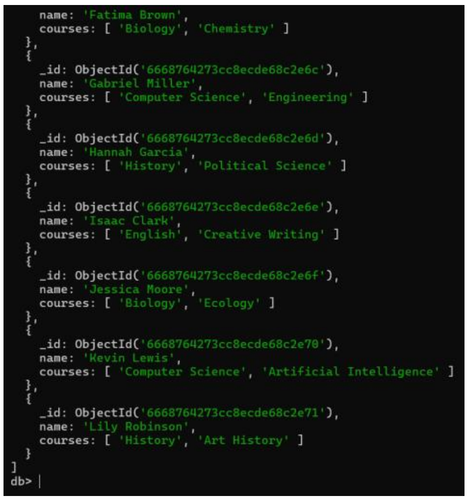
1. **`db.candidates.find({}, ... )`**: This line uses the `find` method to query the `candidates` collection within a database (`db`). The empty curly braces `{}` represent an empty query object, which means it will return all documents in the collection by default.
2. **`{ name: 1, courses: { $slice: 2 } }`**: This part of the code defines a projection document which tells MongoDB which fields to include and how to modify them when returning results.
   - **`name: 1`**: Here, you want to include the `name` field from the documents returned by the query. The `1` specifies to include this field.
   - **`courses: { $slice: 2 }`**: This part targets the `courses` field. It uses the MongoDB projection operator `$slice` to limit the number of elements returned in an array. In this case, it will return only the first two elements from the `courses` field

Output :

This demonstrates that the projection document limited the `courses` field to only the first two elements using the `$slice` operator.



```
db> db.candidates.find({},{name:1,courses:{$slice:2}});
[
  {
    _id: ObjectId('6668764273cc8ecde68c2e66'),
    name: 'Alice Smith',
    courses: [ 'English', 'Biology' ]
  },
  {
    _id: ObjectId('6668764273cc8ecde68c2e67'),
    name: 'Bob Johnson',
    courses: [ 'Computer Science', 'Mathematics' ]
  },
  {
    _id: ObjectId('6668764273cc8ecde68c2e68'),
    name: 'Charlie Lee',
    courses: [ 'History', 'English' ]
  },
  {
    _id: ObjectId('6668764273cc8ecde68c2e69'),
    name: 'Emily Jones',
    courses: [ 'Mathematics', 'Physics' ]
  },
  {
    _id: ObjectId('6668764273cc8ecde68c2e6a'),
    name: 'David Williams',
    courses: [ 'English', 'Literature' ]
  },
  {
    _id: ObjectId('6668764273cc8ecde68c2e6b'),
    name: 'Fatima Brown',
    courses: [ 'Biology', 'Chemistry' ]
  },
  {
    _id: ObjectId('6668764273cc8ecde68c2e6c'),
    name: 'Gabriel Miller',
    courses: [ 'Computer Science', 'Engineering' ]
  },
  {
    _id: ObjectId('6668764273cc8ecde68c2e6d'),
    name: 'Hannah Garcia',
```



```
    name: 'Fatima Brown',
    courses: [ 'Biology', 'Chemistry' ]
  },
  {
    _id: ObjectId('6668764273cc8ecde68c2e6c'),
    name: 'Gabriel Miller',
    courses: [ 'Computer Science', 'Engineering' ]
  },
  {
    _id: ObjectId('6668764273cc8ecde68c2e6d'),
    name: 'Hannah Garcia',
    courses: [ 'History', 'Political Science' ]
  },
  {
    _id: ObjectId('6668764273cc8ecde68c2e6e'),
    name: 'Isaac Clark',
    courses: [ 'English', 'Creative Writing' ]
  },
  {
    _id: ObjectId('6668764273cc8ecde68c2e6f'),
    name: 'Jessica Moore',
    courses: [ 'Biology', 'Ecology' ]
  },
  {
    _id: ObjectId('6668764273cc8ecde68c2e70'),
    name: 'Kevin Lewis',
    courses: [ 'Computer Science', 'Artificial Intelligence' ]
  },
  {
    _id: ObjectId('6668764273cc8ecde68c2e71'),
    name: 'Lily Robinson',
    courses: [ 'History', 'Art History' ]
  }
]
db> |
```