# Experiment 02

## a. MongoDb query to select certain fields and ignore some feilds

### Select Attribute:-

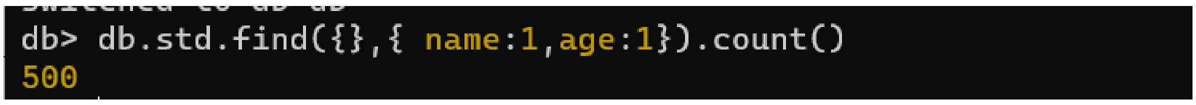
To develop a MongoDB query to select certain fields from documents in a collection, you can use the find method with a projection. The projection specifies which fields you want to include in the results.

**Basic Structure**

```
db.collection.find(
{ /* query criteria */ },
  {
    field1: 1,  // Include field1
    field2: 1,  // Include field2
    // Add more fields to include as needed
  }
)
```

Example:

Assume we have a collection named std and we want to select only the name and age fields. The query would look like this:



```
db> db.std.find({},{ name:1,age:1}).count()
500
```

**Including Fields Only**

Here’s a practical example. Suppose we have the following documents in the employees collection:

```
{
  "_id": ObjectId("60c72b2f4f1a2c1d5f8d27e1"),
  "name": "Alice Johnson",
  "email": "alice.johnson@example.com",
  "position": "Software Engineer",
  "salary": 95000,
  "department": "Engineering"
}
```

If we want to select only the name, email, and position fields, we use the query:

```
db.employees.find(
  { /* query criteria */ },
  {
    name: 1,      // Include the 'name' field
    email: 1,     // Include the 'email' field
    position: 1,  // Include the 'position' field
    _id: 0        // Exclude the '_id' field
  }
)
```

The result of this query will be:

```
{
  "name": "Alice Johnson",
  "email": "alice.johnson@example.com",
  "position": "Software Engineer"
}
```

<u>Notes</u>

1. *Inclusion and Exclusion Mix*: MongoDB does not allow mixing inclusion and exclusion in the same projection, with the exception of the _id field. When you specify fields to include (1), all other fields are excluded by default, except for _id which is included by default unless explicitly excluded.

2. *Exclusion of _id*: If you do not want the _id field in the results, you need to explicitly exclude it by setting _id: 0 in the projection.

3. *Default Behavior*: If no projection is specified, all fields of the document are returned by default.

## Ignore Attribute:-

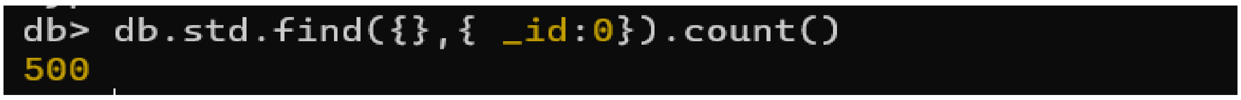
To develop a MongoDB query that ignores certain fields from documents in a collection, you can use the find method with a projection that excludes the fields. In MongoDB, you exclude fields by setting them to 0 in the projection.

**Basic Structure**

```
db.collection.find(
  { /* query criteria */ },
  {
    field1: 0,  // Exclude field1
    field2: 0,  // Exclude field2
    // Add more fields to exclude as needed
  }
)
```

Example

Get all student data but exclude the _id field



```
db> db.std.find({},{ _id:0}).count()
500
```

**Excluding Fields Only**

Here's a practical example. Suppose we have the following documents in the employees collection:

```
{
  "_id": ObjectId("60c72b2f4f1a2c1d5f8d27e1"),
  "name": "Alice Johnson",
  "email": "alice.johnson@example.com",
  "position": "Software Engineer",
  "salary": 95000,
  "department": "Engineering",
  "password": "secret123",
  "ssn": "123-45-6789"
}
```

If we want to exclude only the ‘password’ and ‘ssn’ fields, we use the query:

```
db.employees.find(
  { /* query criteria */ },
  {
    password: 0, // Exclude the 'password' field
    ssn: 0       // Exclude the 'ssn' field
  }
)
```

Example Result

The result of this query will be:

```
{
  "_id": ObjectId("60c72b2f4f1a2c1d5f8d27e1"),
  "name": "Alice Johnson",
  "email": "alice.johnson@example.com",
  "position": "Software Engineer",
  "salary": 95000,
  "department": "Engineering",
}
```

Notes

1. *Exclusion by Default*: When you specify fields to exclude (0), all other fields are included by default, except for _id which is included by default unless explicitly excluded.

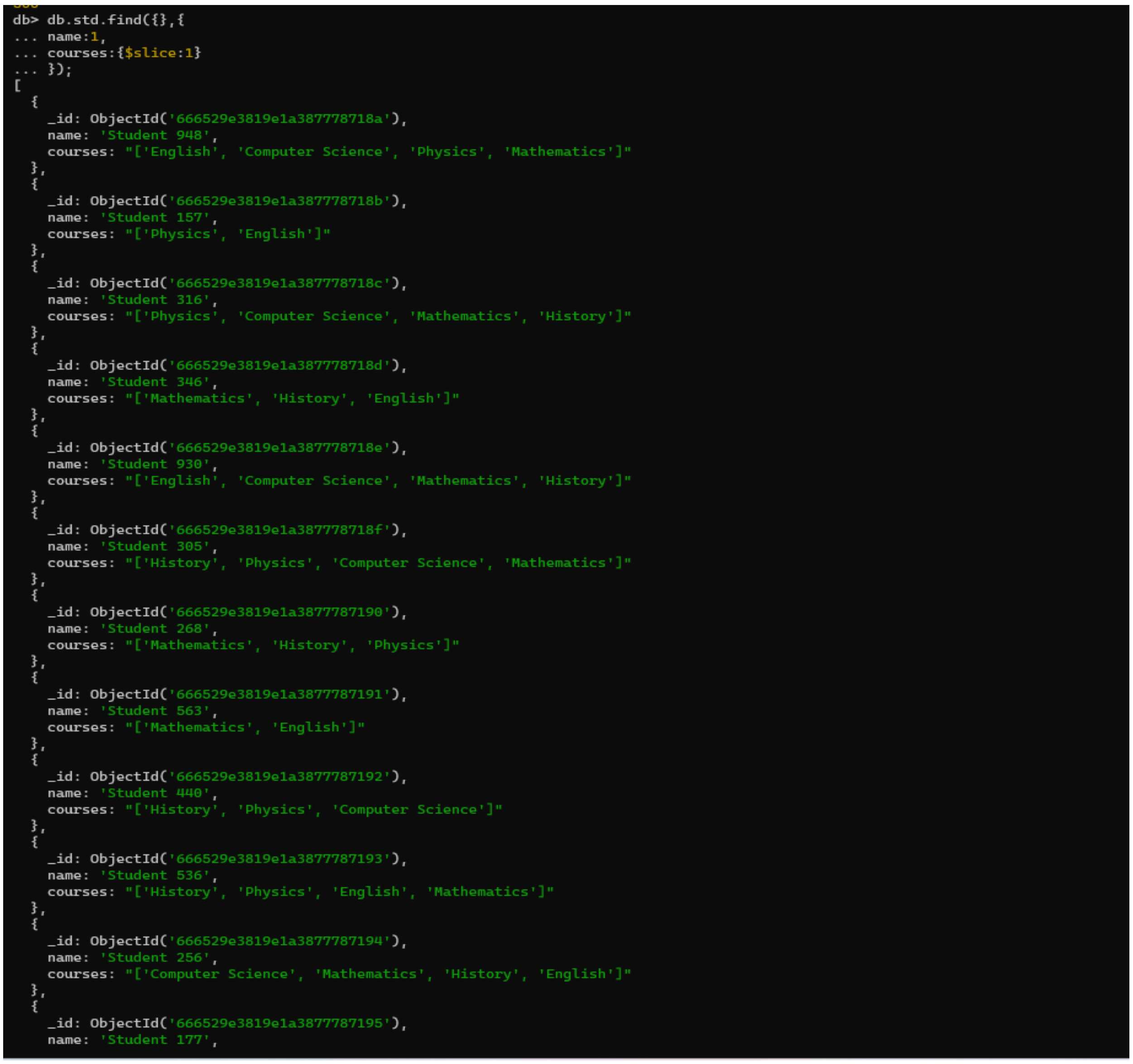
2. *Exclusion of _id*: If you do not want the _id field in the results, you need to explicitly exclude it by setting _id: 0 in the projection.

**Benefits of Projection**

1. Performance Improvement
2. Optimized Resource Usage
3. Simplified Data Handling
4. Better index Utilization
5. Flexible Query Result

**Retriveving Specific Fields from Nested Objects**:

The $slice operator in MongoDB is used to select a subset of an array. It is a particularly useful when you have large arrays stored in your documents and you only need to retrieve certain elements, optimizing data retrieval overhead.



```
db> db.std.find({},{
... name:1,
... courses:{$slice:1}
... });
[
  {
    _id: ObjectId('666529e3819e1a387778718a'),
    name: 'Student 948',
    courses: "['English', 'Computer Science', 'Physics', 'Mathematics']"
  },
  {
    _id: ObjectId('666529e3819e1a387778718b'),
    name: 'Student 157',
    courses: "['Physics', 'English']"
  },
  {
    _id: ObjectId('666529e3819e1a387778718c'),
    name: 'Student 316',
    courses: "['Physics', 'Computer Science', 'Mathematics', 'History']"
  },
  {
    _id: ObjectId('666529e3819e1a387778718d'),
    name: 'Student 346',
    courses: "['Mathematics', 'History', 'English']"
  },
  {
    _id: ObjectId('666529e3819e1a387778718e'),
    name: 'Student 930',
    courses: "['English', 'Computer Science', 'Mathematics', 'History']"
  },
  {
    _id: ObjectId('666529e3819e1a387778718f'),
    name: 'Student 305',
    courses: "['History', 'Physics', 'Computer Science', 'Mathematics']"
  },
  {
    _id: ObjectId('666529e3819e1a3877787190'),
    name: 'Student 268',
    courses: "['Mathematics', 'History', 'Physics']"
  },
  {
    _id: ObjectId('666529e3819e1a3877787191'),
    name: 'Student 563',
    courses: "['Mathematics', 'English']"
  },
  {
    _id: ObjectId('666529e3819e1a3877787192'),
    name: 'Student 440',
    courses: "['History', 'Physics', 'Computer Science']"
  },
  {
    _id: ObjectId('666529e3819e1a3877787193'),
    name: 'Student 536',
    courses: "['History', 'Physics', 'English', 'Mathematics']"
  },
  {
    _id: ObjectId('666529e3819e1a3877787194'),
    name: 'Student 256',
    courses: "['Computer Science', 'Mathematics', 'History', 'English']"
  },
  {
    _id: ObjectId('666529e3819e1a3877787195'),
    name: 'Student 177',
```

## b. Display the first 5 document from the result (use of limit and find)

To display the first 5 documents from the results obtained in a MongoDB query, you can use the find method combined with the limit method. The find method retrieves documents based on your query criteria, and the limit method restricts the number of documents returned.

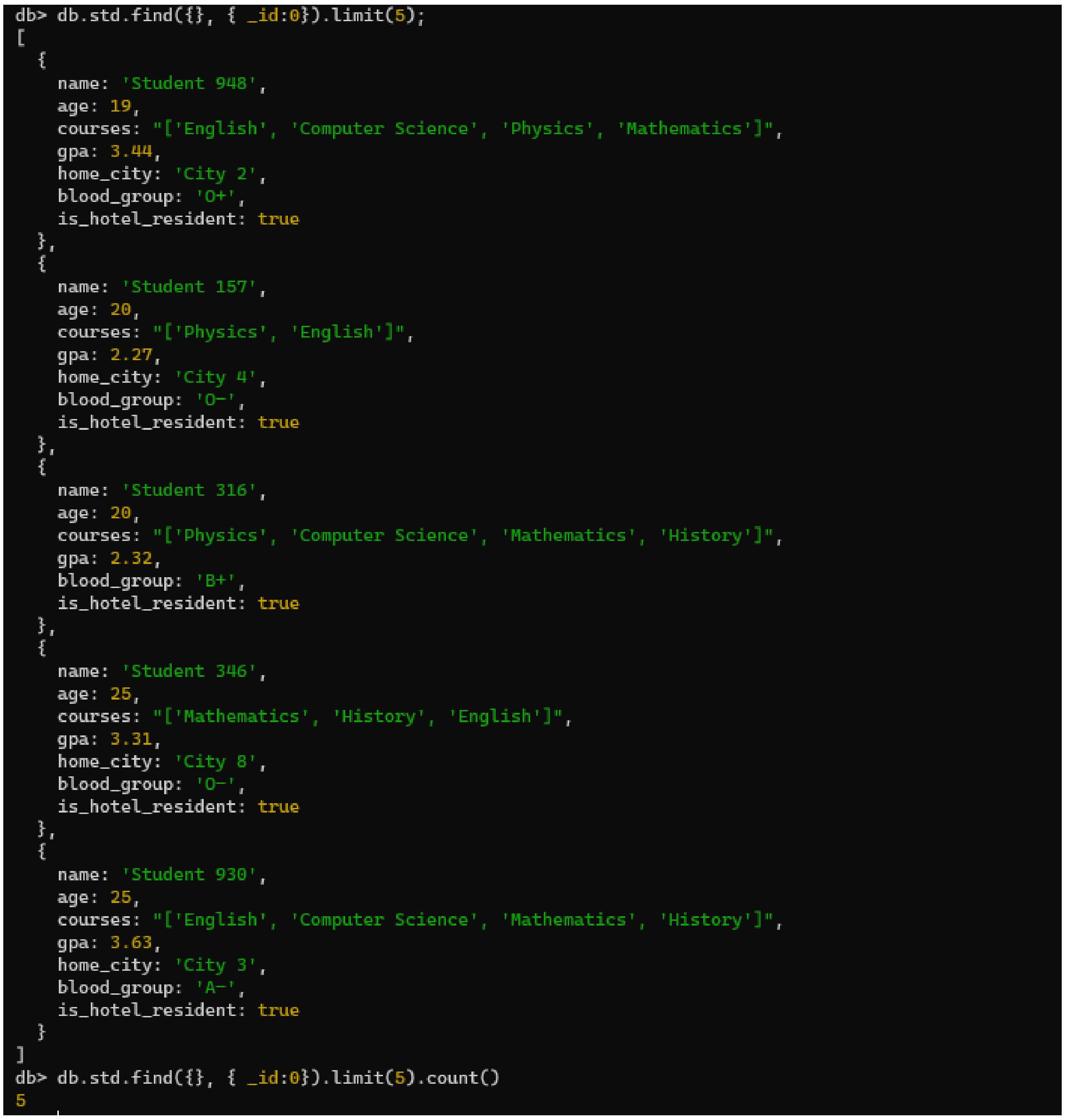
**Basic Structure:-**

```
db.collection.find(
  { /* query criteria */ },
  { /* projection */ }
).limit(5)
```

$limit takes a number, n, and returns the first n resulting document

**Get first 5 document**

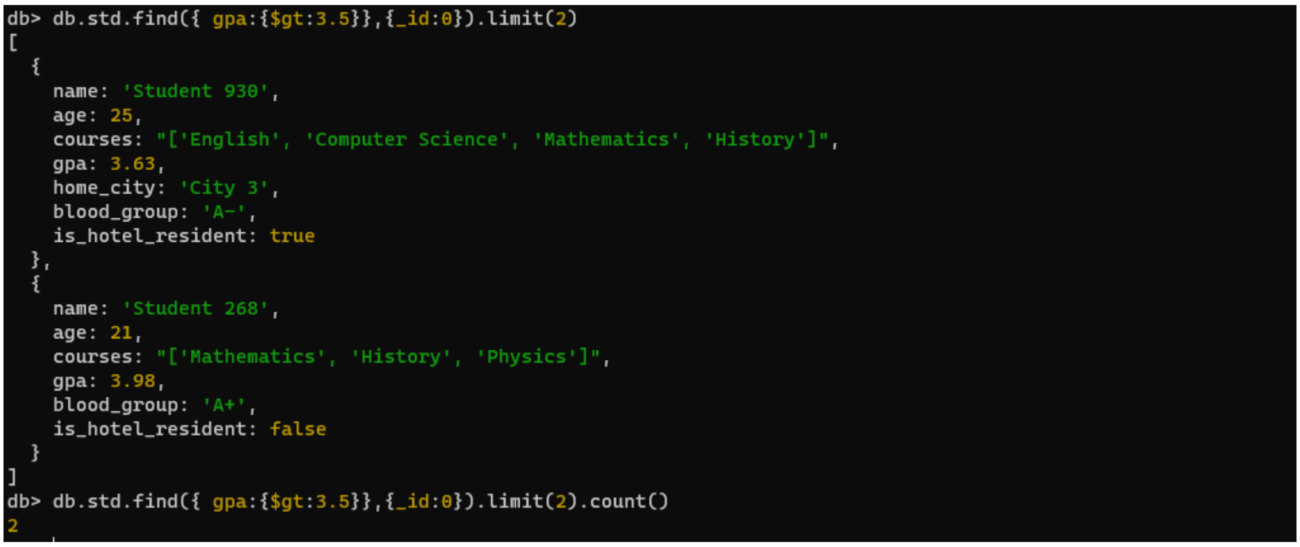
Limit the result to the first 5 document



```
db> db.std.find({}, { _id:0}).limit(5);
[
  {
    name: 'Student 948',
    age: 19,
    courses: "['English', 'Computer Science', 'Physics', 'Mathematics']",
    gpa: 3.44,
    home_city: 'City 2',
    blood_group: 'O+',
    is_hotel_resident: true
  },
  {
    name: 'Student 157',
    age: 20,
    courses: "['Physics', 'English']",
    gpa: 2.27,
    home_city: 'City 4',
    blood_group: 'O-',
    is_hotel_resident: true
  },
  {
    name: 'Student 316',
    age: 20,
    courses: "['Physics', 'Computer Science', 'Mathematics', 'History']",
    gpa: 2.32,
    blood_group: 'B+',
    is_hotel_resident: true
  },
  {
    name: 'Student 346',
    age: 25,
    courses: "['Mathematics', 'History', 'English']",
    gpa: 3.31,
    home_city: 'City 8',
    blood_group: 'O-',
    is_hotel_resident: true
  },
  {
    name: 'Student 930',
    age: 25,
    courses: "['English', 'Computer Science', 'Mathematics', 'History']",
    gpa: 3.63,
    home_city: 'City 3',
    blood_group: 'A-',
    is_hotel_resident: true
  }
]
db> db.std.find({}, { _id:0}).limit(5).count()
5
```

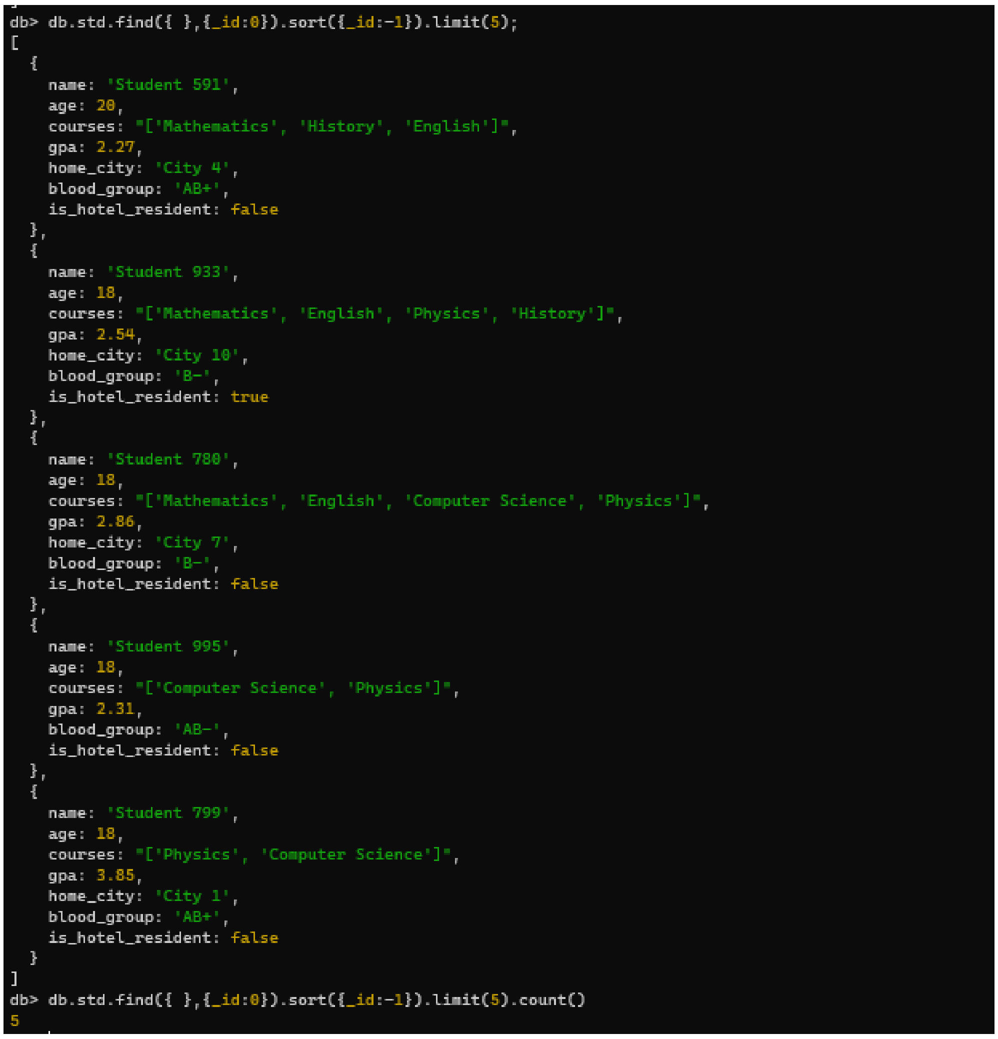
**Limiting result**

Find all students with GPA greater than 3.5 and limit to 2 documents


```
db> db.std.find({ gpa:{$gt:3.5}},{_id:0}).limit(2)
[
  {
    name: 'Student 930',
    age: 25,
    courses: "['English', 'Computer Science', 'Mathematics', 'History']",
    gpa: 3.63,
    home_city: 'City 3',
    blood_group: 'A-',
    is_hotel_resident: true
  },
  {
    name: 'Student 268',
    age: 21,
    courses: "['Mathematics', 'History', 'Physics']",
    gpa: 3.98,
    blood_group: 'A+',
    is_hotel_resident: false
  }
]
db> db.std.find({ gpa:{$gt:3.5}},{_id:0}).limit(2).count()
2
```

## I want top 10 Results

Sort documents in descending order by _id and limit to 5


```
db> db.std.find({ },{_id:0}).sort({_id:-1}).limit(5);
[
  {
    name: 'Student 591',
    age: 20,
    courses: "['Mathematics', 'History', 'English']",
    gpa: 2.27,
    home_city: 'City 4',
    blood_group: 'AB+',
    is_hotel_resident: false
  },
  {
    name: 'Student 933',
    age: 18,
    courses: "['Mathematics', 'English', 'Physics', 'History']",
    gpa: 2.54,
    home_city: 'City 10',
    blood_group: 'B-',
    is_hotel_resident: true
  },
  {
    name: 'Student 780',
    age: 18,
    courses: "['Mathematics', 'English', 'Computer Science', 'Physics']",
    gpa: 2.86,
    home_city: 'City 7',
    blood_group: 'B-',
    is_hotel_resident: false
  },
  {
    name: 'Student 995',
    age: 18,
    courses: "['Computer Science', 'Physics']",
    gpa: 2.31,
    blood_group: 'AB-',
    is_hotel_resident: false
  },
  {
    name: 'Student 799',
    age: 18,
    courses: "['Physics', 'Computer Science']",
    gpa: 3.85,
    home_city: 'City 1',
    blood_group: 'AB+',
    is_hotel_resident: false
  }
]
db> db.std.find({ },{_id:0}).sort({_id:-1}).limit(5).count()
5
```