# *Experiment:2*

*a. Develop a MongoDB query to select certain fields and ignore some fields of the documents from any collection.*

*b. Develop a MongoDB query to display the first 5 documents from the results obtained in a.*

*[use of limit and find]*

## a. Select and ignore fields

**Develop a MongoDB query to select certain fields and ignore some fields of the documents from any collection.**

To select certain fields and ignore others in MongoDB, you use projections in your queries. Projections allow you to specify which fields to include or exclude in the returned documents.

### Basic Syntax for Projection

When using the `find()` method, the first parameter is the query filter, and the second parameter is the projection object. The projection object specifies the fields to include (using `1`) or exclude (using `0`).

## PROJECTION:

In MongoDB, projection is used to **specify or restrict the fields to return in query results**. It allows you to include or exclude specific fields from the documents returned by a query, enhancing efficiency by returning only necessary data.

For example, a projection can be used to return only the **`name` and `gpa`** fields from documents in the **`students` collection** while excluding the **`_id` field**.

## FIELD PROJECTION:

Use the projection document as the second argument to the find method Include field names with a **value of 1** to specify fields to be returned. Omit fields or set them to 0 to exclude them from the results.

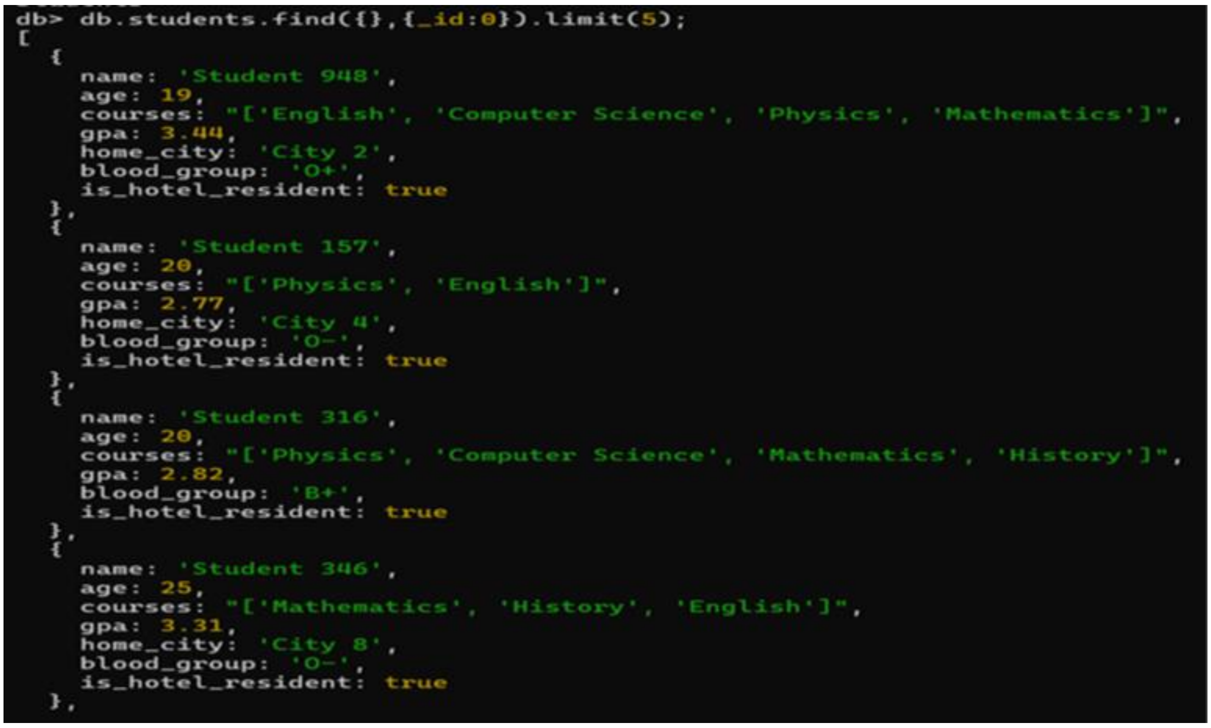
## BENEFITS:

Reduces data transferred between the database and your application.Improves query performance by retrieving only necessary data.Simplifies your code by focusing on the specific information you need.

## LIMIT:

The limit operator is used with the find method.It's chained after the filter criteria or any sorting operations. **Syntax: db.collection.find({filter}, {projection}).limit(number)**

## Get first 5 documents:



```
db> db.students.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: 2.77,
    home_city: 'City 4',
    blood_group: 'O-',
    is_hotel_resident: true
  },
  {
    name: 'Student 316',
    age: 20,
    courses: "['Physics', 'Computer Science', 'Mathematics', 'History']",
    gpa: 2.82,
    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
  },
```

The query `**db.students.find({},{_id:0}).limit(5)**;` is used to retrieve documents from the `students` collection in MongoDB.

**The query returns up to five student documents but excludes the `_id` field in the results.** The output includes details such as the student's name, age, courses, GPA, home city, blood group, and whether they are a hotel resident.

This demonstrates the use of **the `find` method with a projection** to **exclude specific fields** and the `**limit**` **method** to restrict the number of documents returned.

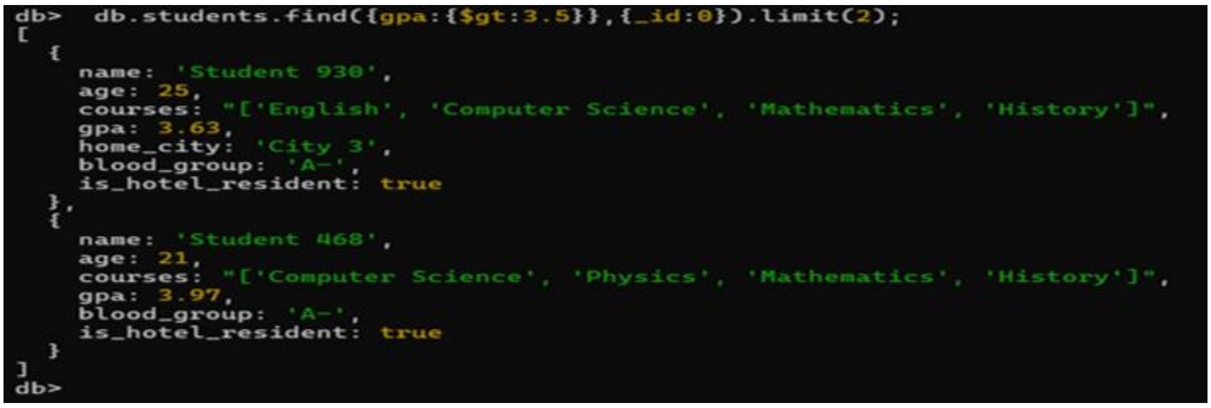
## LIMITING RESULTS:

The query `**db.students.find({gpa:{$gt:3.5}},{_id:0}).limit(2);**`

retrieves documents from the `students` collection in MongoDB where the **`gpa` field is greater than 3.5.**

It excludes the `_id` field from the results and limits the output to two documents.

The resulting documents include details such as the student's name, age, courses, GPA, home city, blood group, and whether they are a hotel resident, showcasing the use of the `find` method with filtering criteria, projection, and limit.

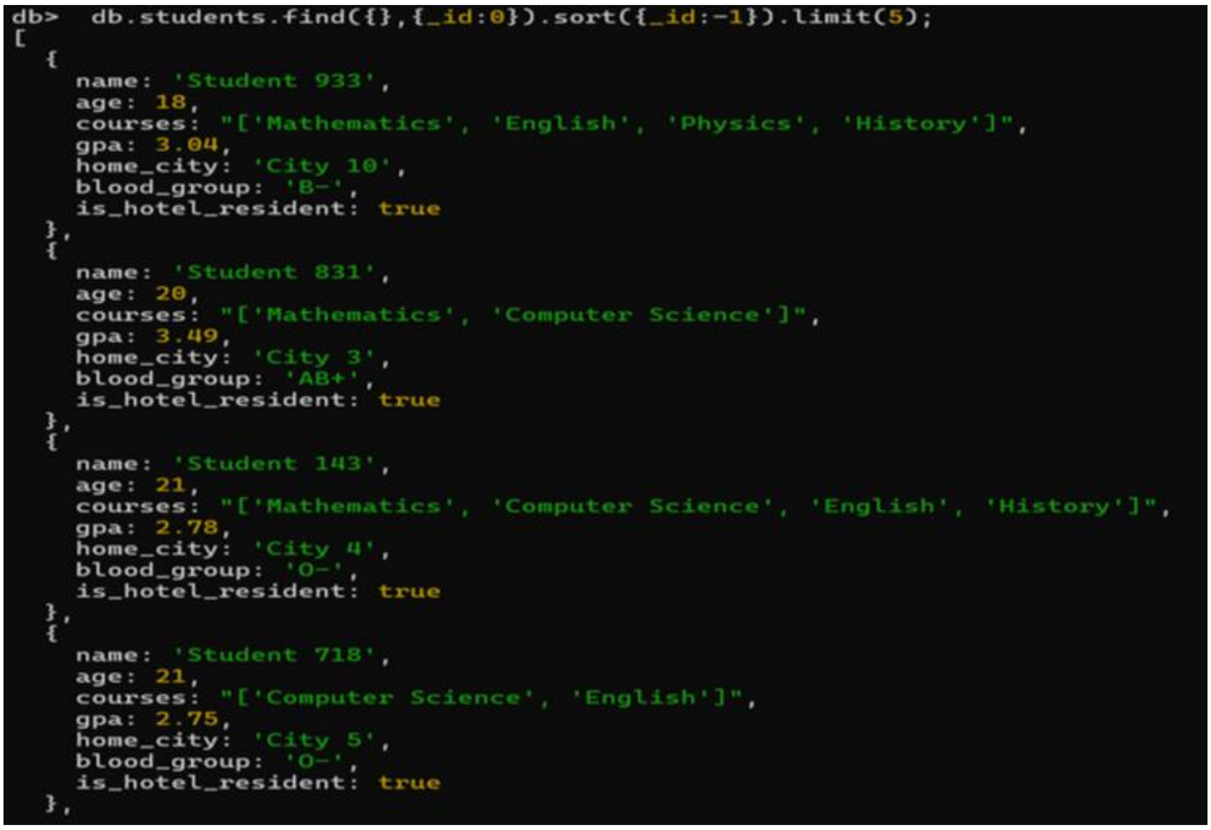


```
db>  db.students.find({gpa:{$gt:3.5}},{_id:0}).limit(2);
[
  {
    name: 'Student 930',
    age: 25,
    courses: "['English', 'Computer Science', 'Mathematics', 'History']",
    gpa: 3.63,
    home_city: 'City 3',
    blood_group: 'A-',
    is_hotel_resident: true
  },
  {
    name: 'Student 468',
    age: 21,
    courses: "['Computer Science', 'Physics', 'Mathematics', 'History']",
    gpa: 3.97,
    blood_group: 'A-',
    is_hotel_resident: true
  }
]
db>
```

## TO GET TOP 10 RESULTS:

The query `**db.students.find({},{_id:0}).sort({_id:-1}).limit(5);**` retrieves documents from the `students` collection, **excluding the `_id` field** from the results, and **sorts them in descending order by the `_id` field.**

The `**sort({_id:-1})**` part ensures that the documents are ordered from the most recent to the oldest based on their `_id` values.



```
db>  db.students.find({},{_id:0}).sort({_id:-1}).limit(5);
[
  {
    name: 'Student 933',
    age: 18,
    courses: "['Mathematics', 'English', 'Physics', 'History']",
    gpa: 3.04,
    home_city: 'City 10',
    blood_group: 'B-',
    is_hotel_resident: true
  },
  {
    name: 'Student 831',
    age: 20,
    courses: "['Mathematics', 'Computer Science']",
    gpa: 3.49,
    home_city: 'City 3',
    blood_group: 'AB+',
    is_hotel_resident: true
  },
  {
    name: 'Student 143',
    age: 21,
    courses: "['Mathematics', 'Computer Science', 'English', 'History']",
    gpa: 2.78,
    home_city: 'City 4',
    blood_group: 'O-',
    is_hotel_resident: true
  },
  {
    name: 'Student 718',
    age: 21,
    courses: "['Computer Science', 'English']",
    gpa: 2.75,
    home_city: 'City 5',
    blood_group: 'O-',
    is_hotel_resident: true
  },
```

The `**limit(5)**` clause restricts the output to the first five documents that meet these criteria. The resulting documents include various student details such as name, age, courses, GPA, home city, blood group, and whether they are hotel residents, demonstrating how sorting and limiting can be applied to refine query results.