Projection & Limit

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

Projection:

- In MongoDB, projections are a way to control which fields get returned from a query.
- By default, MongoDB returns all fields in a document when you use the find method.
- Projections allow you to specify exactly which data you need, improving efficiency and reducing the amount of data transferred.
- projections are a powerful tool in MongoDB for optimizing queries and reducing data transfer by only returning the specific fields you need.

how projections work:

- **Specifying Fields:** You use a document as the second argument to the find method to specify which fields to include or exclude.
- **Including Fields:** Set a field name to 1 or true to include that field in the results.
- **Excluding Fields:** Set a field name to 0 or false to exclude that field from the results.
- **Default _id Inclusion:** The _id field is always included by default unless you explicitly set it to 0 for exclusion.

Selected Attributes:

- You add a projection document as the second argument to the find method.
- In the projection document, specify the fields you want to include by name and set their value to 1.

Ignore Attributes:

- You add a projection document as the second argument to the find method.
- In the projection document, specify the fields you want to exclude by _id and set their value to 0.

//Get name and gpa and ignore the_id

```
db.students.find({}, { name: 1, gpa: 1 ,_id:0});
```

```
db> db.students.find({},{name:1,gpa:1,_id:0});
[
  { name: 'Student 948', gpa: 4.34 },
  { name: 'Student 157', gpa: 2.27 },
  { name: 'Student 316', gpa: 2.32 },
  { name: 'Student 346', gpa: 3.31 },
  { name: 'Student 930', gpa: 3.63 },
  { name: 'Student 305', gpa: 3.4 },
  { name: 'Student 268', gpa: 3.98 },
  { name: 'Student 563', gpa: 2.25 },
  { name: 'Student 440', gpa: 2.06 },
  { name: 'Student 536', gpa: 2.87 },
  { name: 'Student 256', gpa: 2.94 },
  { name: 'Student 177', gpa: 2.52 },
  { name: 'Student 871', gpa: 3.2 },
  { name: 'Student 487', gpa: 2.1 },
  { name: 'Student 213', gpa: 2.39 },
  { name: 'Student 690', gpa: 2.25 },
  { name: 'Student 368', gpa: 3.91 },
  { name: 'Student 172', gpa: 2.46 },
  { name: 'Student 647', gpa: 3.43 },
  { name: 'Student 232', gpa: 2.54 }
]
Type "it" for more
```

- ✓ **name: 1:** This specifies that you want to include the **name** field in the results. The value 1 here indicates inclusion.
- ✓ **gpa: 1:** Similarly, this includes the **gpa** field.
- ✓ **_id: 0:** This explicitly excludes the **_id** field, Setting the value to 0 excludes it.

2.b.Develop a MongoDB query to display the first 5 documents from the results obtained in a.[use of limit and find]

Limit:

- In MongoDB, limit is a method used to restrict the number of documents returned by a query. It essentially sets a maximum cap on the results fetched from the database
- Using limit can significantly improve query performance, especially when dealing with large collections.
- It reduces the amount of data processed and transferred from the server.
- limit is a valuable tool for efficiently retrieving specific subsets of data from your MongoDB collections.

how limit works:

- U**sage:** The limit method is typically applied to a cursor object obtained using the find method.
- **Positive Limit:** Pass a positive integer value to limit to specify the maximum number of documents you want to retrieve. For example, .limit(10) will return a maximum of 10 documents.
- **Negative Limit:** While less common, you can use a negative value with limit. This instructs the server to close the cursor after returning a single batch of results.

//Get the First 5 documents

db.students.find({}, {_id:0}).limit(5);

```
mongosh mongodb://127.0.0.
]
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: 2.27,
    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>
```

- ✓ **.find({}):** This is the find method used to retrieve documents from the collection. The empty curly braces {} act as a filter, currently including all documents
- ✓ **_id: 0:** This explicitly excludes the _id field, which is MongoDB's default field for document identification. Setting the value to 0 excludes it.
- ✓ **.limit(5)**: limits the number of documents returned in the result set to 5.

//get all students with gpa greater than 2.5 and limit to 3 documents

```
db> db.students.find({gpa: {$gt:2.5}},{_id:0}).limit(3);
[
  {
    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 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>
```

- ✓ **gpa:** {$gt:2.5} is the filtering criteria within the document.
- ✓ **gpa**: This specifies the field to filter on, which is the "gpa" field of each document.
- ✓ **$gt:2.5:** This is a comparison operator.
- ✓ This filter essentially finds only those documents where the "gpa" field is greater than 2.5.
- ✓ **{_id:0}.limit(3):**
    - **{_id:0}:** This is the projection document that controls which fields are returned in the results.
    - **_id: 0:** This explicitly excludes the _id field, Setting the value to 0 excludes it.
    - **.limit(3)** limits the number of documents returned in the result set to 3. This acts on the documents that passed the initial filtering stage.

//get Top 5 Results

```
]
db> db.students.find({},{_id:0}).sort({_id:-1}).limit(5);
[
  {
    name: 'Student 871',
    age: 19,
    courses: "['Computer Science', 'English', 'History']",
    gpa: 3.33,
    blood_group: 'O+',
    is_hotel_resident: false
  },
  {
    name: 'Student 873',
    age: 21,
    courses: "['History', 'Mathematics', 'Physics']",
    gpa: 3.94,
    home_city: 'City 8',
    blood_group: 'O+',
    is_hotel_resident: false
  },
  {
    name: 'Student 111',
    age: 18,
    courses: "['Physics', 'Computer Science']",
    gpa: 2.99,
    blood_group: 'O+',
    is_hotel_resident: false
  },
  {
    name: 'Student 404',
    age: 25,
    courses: "['Mathematics', 'Physics', 'Computer Science', 'English']",
    gpa: 2.59,
    home_city: 'City 10',
    blood_group: 'AB-',
    is_hotel_resident: false
  },
  {
    name: 'Student 347',
    age: 22,
    courses: "['English', 'Computer Science', 'Physics']",
    gpa: 2.17,
    blood_group: 'B-',
    is_hotel_resident: true
  }
]
```

Sorting in Descending Order:

.sort({_id:-1}):

- This part is chained to the find operation and applies sorting.
- **.sort({}):** This specifies the sorting criteria within curly braces.
- **{_id:-1}:** This defines how to sort the documents.

- **_id:** This specifies the field to sort by, which is the _id field in this case.
- **-1:** This indicates descending order. Documents with higher (more recent) _id values will appear later in the results.
- **.limit(5)** restricts the number of documents returned in the final result set to 5. This acts on the documents that passed the filtering (which included all documents here) and sorting stages.