S. B. JAIN INSTITUTE OF TECHNOLOGY, MANAGEMENT & RESEARCH, NAGPUR.

Practical No. 7

Aim: To implement No-SQL database function: Count, Remove, Sort, Limit, Skip & aggregation using MongoDB.

Name of Student: Shrutika Pradeep Bagdi

Roll No.: CS22130

Semester/Year: 7th / 4th

Academic Session: 2025-2026

Date of Performance: __________

Date of Submission: __________

AIM: To implement No-SQL database function: Count, Remove, Sort, Limit, Skip & aggregation using MongoDB..

OBJECTIVE/EXPECTED LEARNING OUTCOME:

The objectives and expected learning outcome of this practical are:

- Gain a fundamental understanding of NoSQL databases, their characteristics, and how they differ from traditional relational databases.
- Acquire knowledge of MongoDB's document-based data model, collections, and documents, and understand how data is stored and managed in MongoDB
- Learn to manipulate data in MongoDB by implementing various operations like Count, Sort, Limit, and Skip to retrieve, filter, and organize data efficiently.

HARDWARE AND SOFTWARE REQUIRMENTS:

Hardware Requirement: High Configuration computer

Software Requirement: MongoDB-8.0, Mongo Shell

THEORY:

1) The MongoDB Limit() Method

To limit the records in MongoDB, you need to use limit() method. limit() method accepts one number type argument, which is number of documents that you want to displayed.

Syntax:

Basic syntax of **limit**() method is as follows

```
>db.COLLECTION_NAME.find().limit(NUMBER)
```

Example :Consider the collection myycol has the following data

```
{ "_id" : ObjectId(5983548781331adf45ec5), "title":"MongoDB Overview"}
{ "_id" : ObjectId(5983548781331adf45ec6), "title":"NoSQL Overview"}
{ "_id" : ObjectId(5983548781331adf45ec7), "title":"Tutorials Overview"}
```

Following example will display only 2 documents while quering the document.

```
>db.mycol.find({},{"title":1,_id:0}).limit(2)
{"title":"MongoDB Overview"}
{"title":"NoSQL Overview"}
```

If you don't specify number argument in limit() method then it will display all documents from the collection.

2) The MongoDB Skip() Method

Apart from limit() method there is one more method skip() which also accepts number type argument and used to skip number of documents.

Syntax:

Basic syntax of skip() method is as follows

```
>db.COLLECTION_NAME.find().limit(NUMBER).skip(NUMBER)
```

Example: Following example will only display only second document.

```
>db.mycol.find({},{"title":1,_id:0}).limit(1).skip(1)
{"title":"NoSQL Overview"}
```

Please note default value in skip() method is 0

3) The sort() Method

To sort documents in MongoDB, you need to use sort() method. sort() method accepts a document containing list of

fields along with their sorting order. To specify sorting order 1 and -1 are used. 1 is used for ascending order while -1 is used for descending order.

Syntax:

Basic syntax of sort() method is as follows

```
>db.COLLECTION_NAME.find().sort({KEY:1})
```

Example

Consider the collection myycol has the following data

```
{ "_id" : ObjectId(5983548781331adf45ec5), "title":"MongoDB Overview"}
{ "_id" : ObjectId(5983548781331adf45ec6), "title":"NoSQL Overview"}
{ "_id" : ObjectId(5983548781331adf45ec7), "title":"Tutorials Point Overview"}
```

Following example will display the documents sorted by title in descending order.

```
>db.mycol.find({},{"title":1,_id:0}).sort({"title":-1})
{"title":"Tutorials Point Overview"}
{"title":"NoSQL Overview"}
{"title":"MongoDB Overview"}
```

4) Remove() Function in MongoDB

In MongoDB, the db.collection.remove() method is used to remove documents from a collection. Either all of the documents can be removed from a collection or only those which matches a specific condition.

If you just issue the remove command, all of the documents will be removed from the collection.

Example of MongoDB Remove() Function

The following code example demonstrate how to remove a specific document from the collection.

db.Employee.remove({Employeeid:22})

5) Count() Function in MongoDB

db.collection.count(query)

Returns the count of documents that would match a find() query. The db.collection.count() method does not perform the find() operation but instead counts and returns the number of results that match a query.

db.collection.find({ a: { $gt: 5 } }).count()

db.collection.find({ a: 5, b: { $gt: 10 } }).count()

6) Aggregation Functions in MongoDB:

a. $sum: Returns the sum of numeric values.

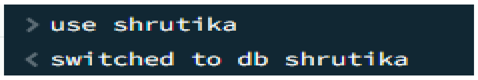
b. $avg : Returns the average of numeric values

c. $min : Returns the minimum value.

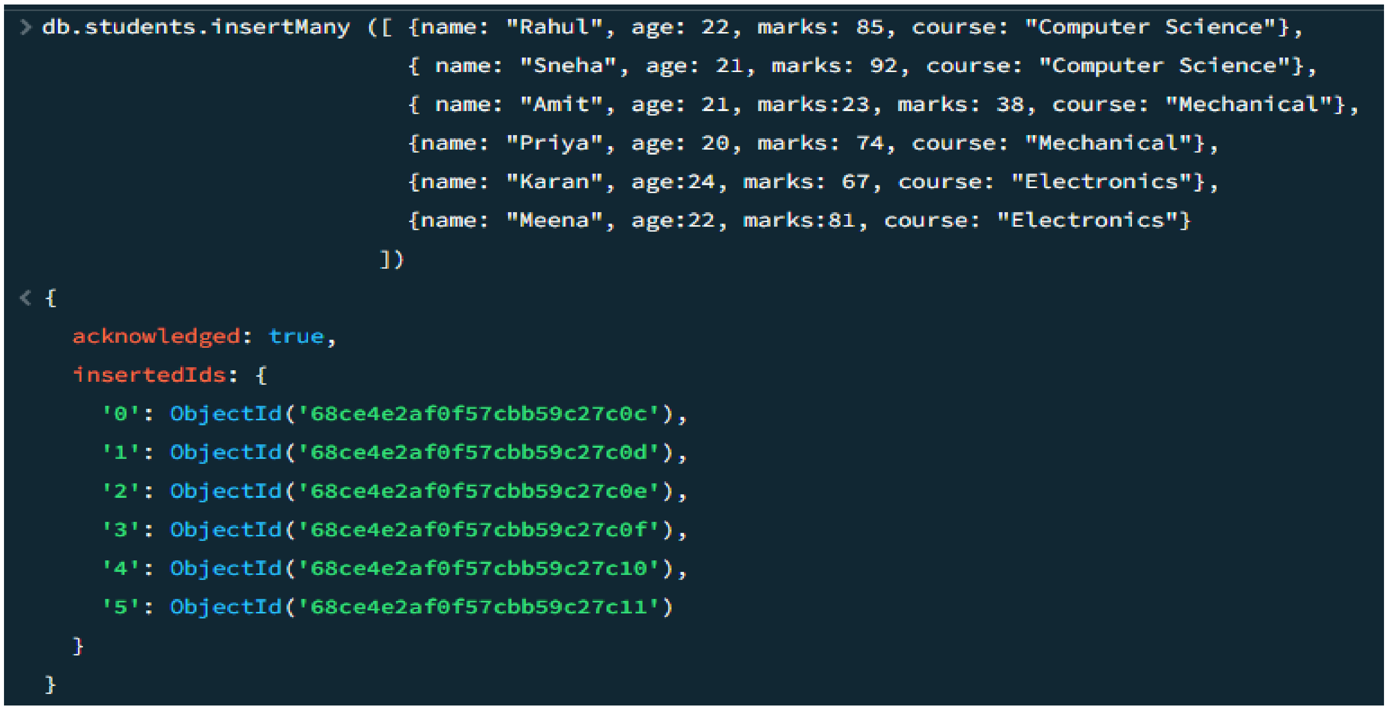
d. $max : Returns the maximum value.

e. $count : Counts the number of documents.

INPUT / OUTPUT (SCREENSHOTS):

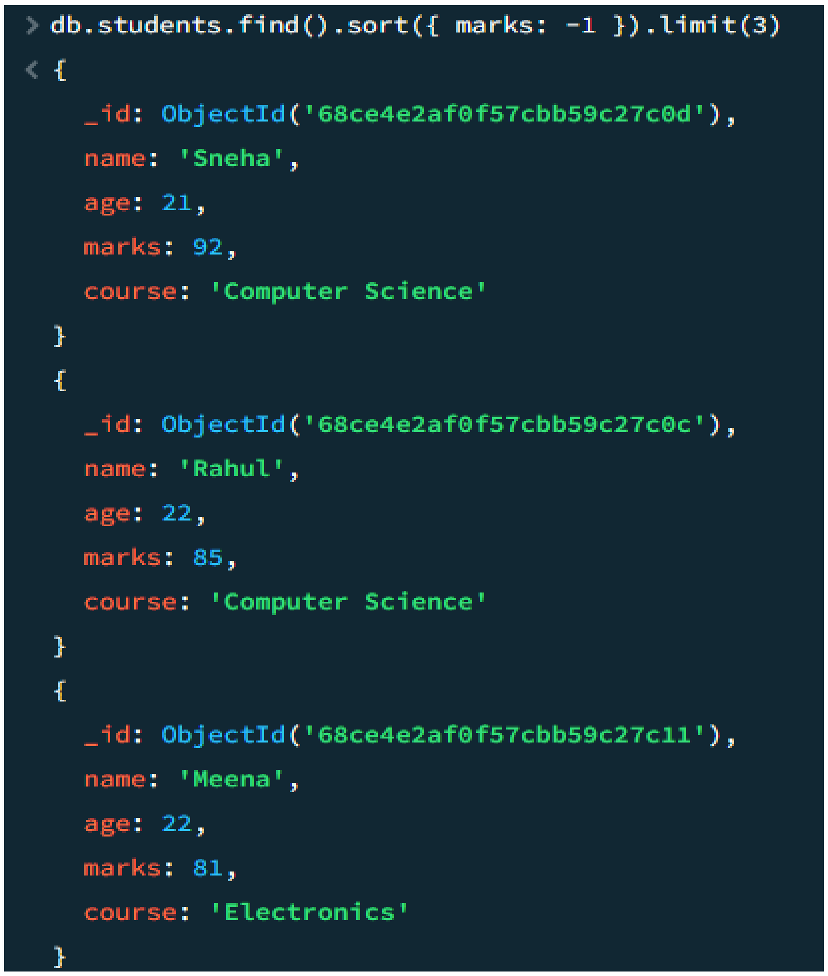


```
> use shrutika
< switched to db shrutika
```



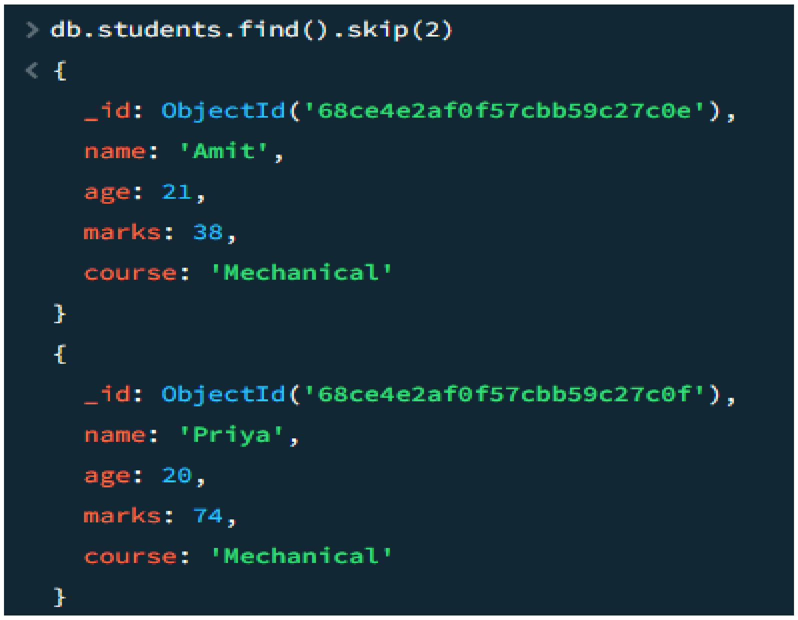
```
> db.students.insertMany ([ {name: "Rahul", age: 22, marks: 85, course: "Computer Science"},
                            { name: "Sneha", age: 21, marks: 92, course: "Computer Science"},
                            { name: "Amit", age: 21, marks:23, marks: 38, course: "Mechanical"},
                            {name: "Priya", age: 20, marks: 74, course: "Mechanical"},
                            {name: "Karan", age:24, marks: 67, course: "Electronics"},
                            {name: "Meena", age:22, marks:81, course: "Electronics"}
                          ])
< {
    acknowledged: true,
    insertedIds: {
      '0': ObjectId('68ce4e2af0f57cbb59c27c0c'),
      '1': ObjectId('68ce4e2af0f57cbb59c27c0d'),
      '2': ObjectId('68ce4e2af0f57cbb59c27c0e'),
      '3': ObjectId('68ce4e2af0f57cbb59c27c0f'),
      '4': ObjectId('68ce4e2af0f57cbb59c27c10'),
      '5': ObjectId('68ce4e2af0f57cbb59c27c11')
    }
  }
```

1) Limit()

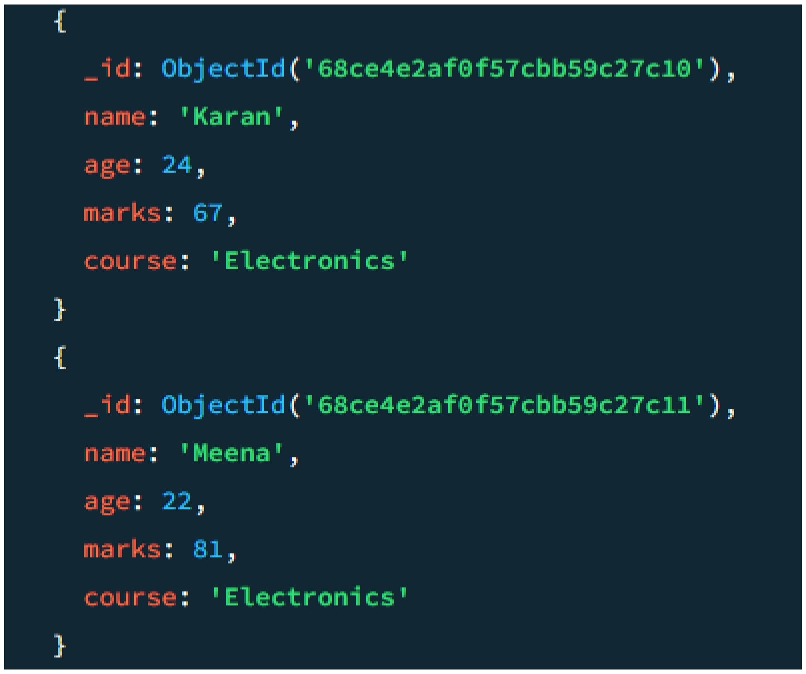


```
> db.students.find().sort({ marks: -1 }).limit(3)
< {
    _id: ObjectId('68ce4e2af0f57cbb59c27c0d'),
    name: 'Sneha',
    age: 21,
    marks: 92,
    course: 'Computer Science'
  }
  {
    _id: ObjectId('68ce4e2af0f57cbb59c27c0c'),
    name: 'Rahul',
    age: 22,
    marks: 85,
    course: 'Computer Science'
  }
  {
    _id: ObjectId('68ce4e2af0f57cbb59c27c11'),
    name: 'Meena',
    age: 22,
    marks: 81,
    course: 'Electronics'
  }
```

2) Skip()

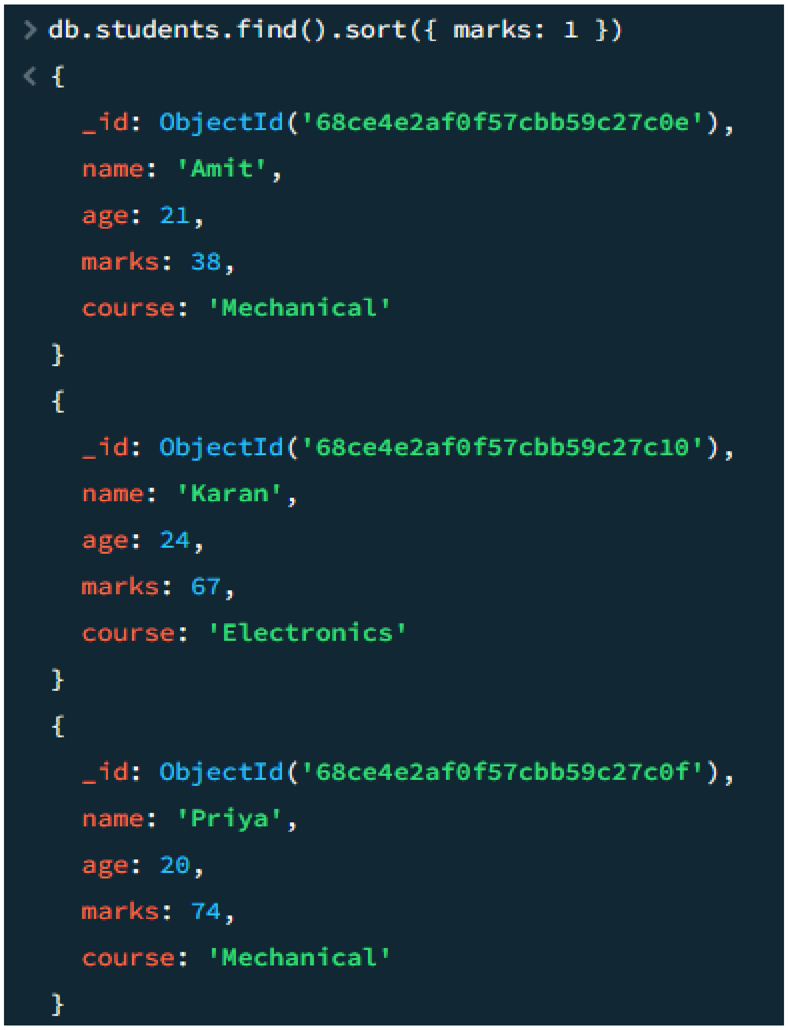


```
> db.students.find().skip(2)
< {
    _id: ObjectId('68ce4e2af0f57cbb59c27c0e'),
    name: 'Amit',
    age: 21,
    marks: 38,
    course: 'Mechanical'
  }
  {
    _id: ObjectId('68ce4e2af0f57cbb59c27c0f'),
    name: 'Priya',
    age: 20,
    marks: 74,
    course: 'Mechanical'
  }
```

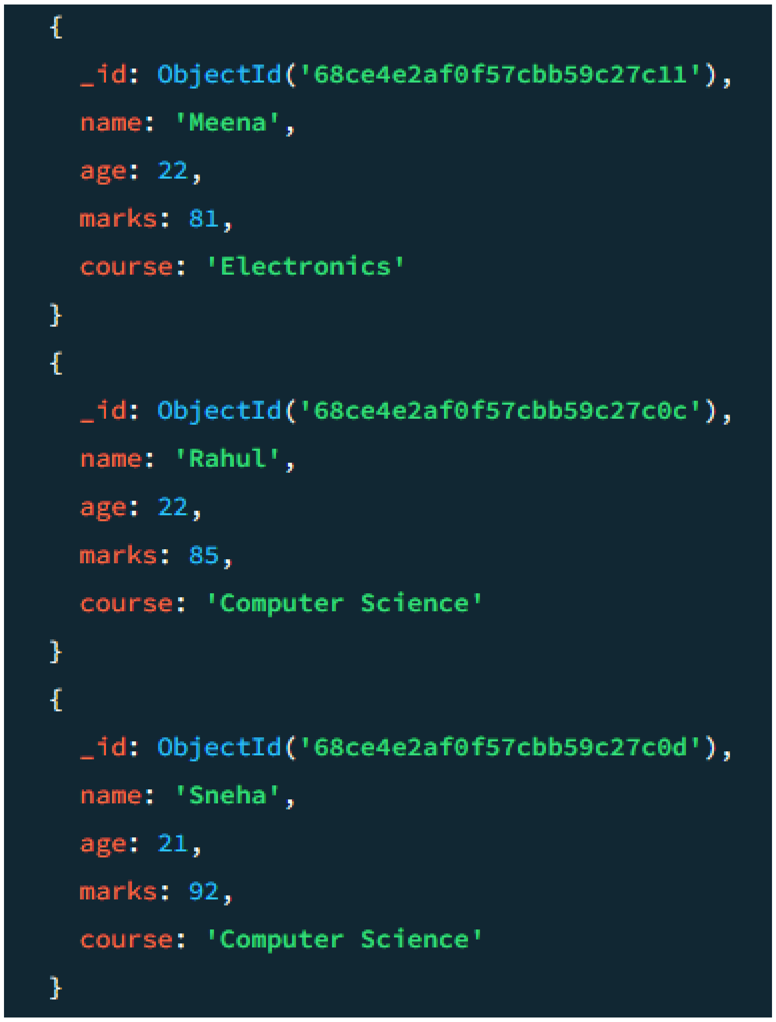


```
  {
    _id: ObjectId('68ce4e2af0f57cbb59c27c10'),
    name: 'Karan',
    age: 24,
    marks: 67,
    course: 'Electronics'
  }
  {
    _id: ObjectId('68ce4e2af0f57cbb59c27c11'),
    name: 'Meena',
    age: 22,
    marks: 81,
    course: 'Electronics'
  }
```

3) Sort()

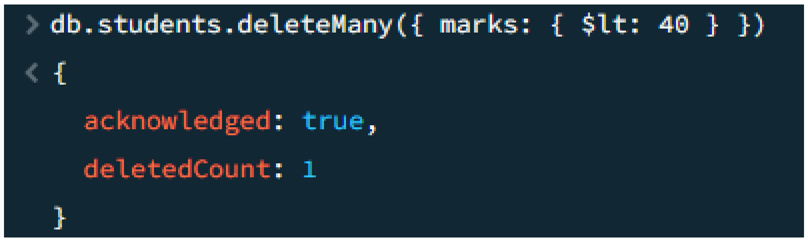


```
> db.students.find().sort({ marks: 1 })
< {
    _id: ObjectId('68ce4e2af0f57cbb59c27c0e'),
    name: 'Amit',
    age: 21,
    marks: 38,
    course: 'Mechanical'
  }
  {
    _id: ObjectId('68ce4e2af0f57cbb59c27c10'),
    name: 'Karan',
    age: 24,
    marks: 67,
    course: 'Electronics'
  }
  {
    _id: ObjectId('68ce4e2af0f57cbb59c27c0f'),
    name: 'Priya',
    age: 20,
    marks: 74,
    course: 'Mechanical'
  }
```



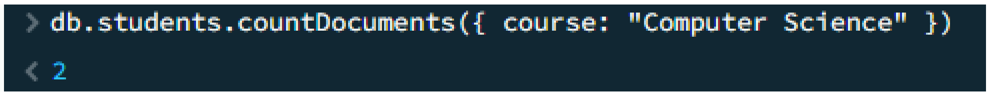
```
  {
    _id: ObjectId('68ce4e2af0f57cbb59c27c11'),
    name: 'Meena',
    age: 22,
    marks: 81,
    course: 'Electronics'
  }
  {
    _id: ObjectId('68ce4e2af0f57cbb59c27c0c'),
    name: 'Rahul',
    age: 22,
    marks: 85,
    course: 'Computer Science'
  }
  {
    _id: ObjectId('68ce4e2af0f57cbb59c27c0d'),
    name: 'Sneha',
    age: 21,
    marks: 92,
    course: 'Computer Science'
  }
```

4) **Remove()**



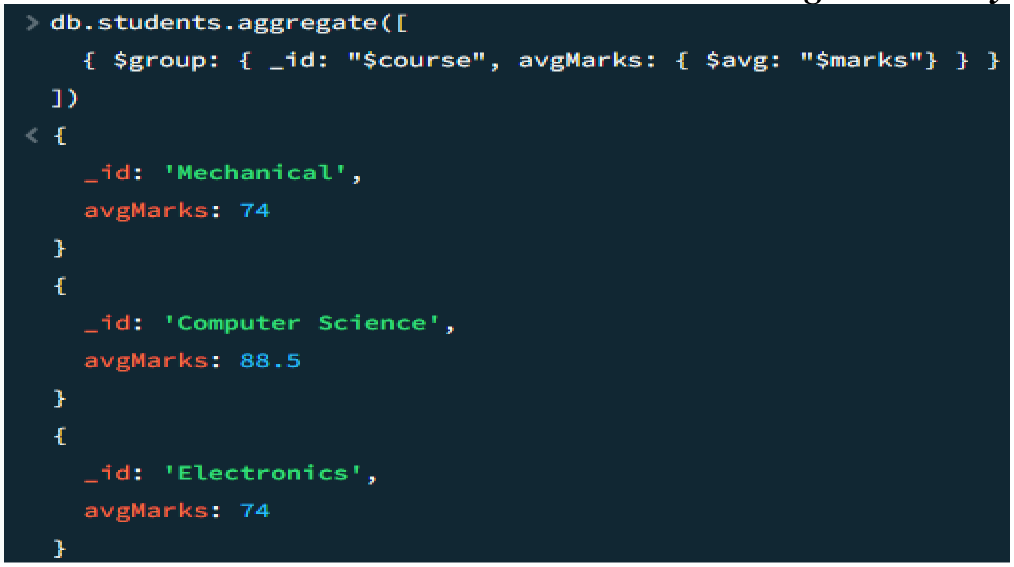
```
> db.students.deleteMany({ marks: { $lt: 40 } })
< {
    acknowledged: true,
    deletedCount: 1
  }
```

5) **Count()**

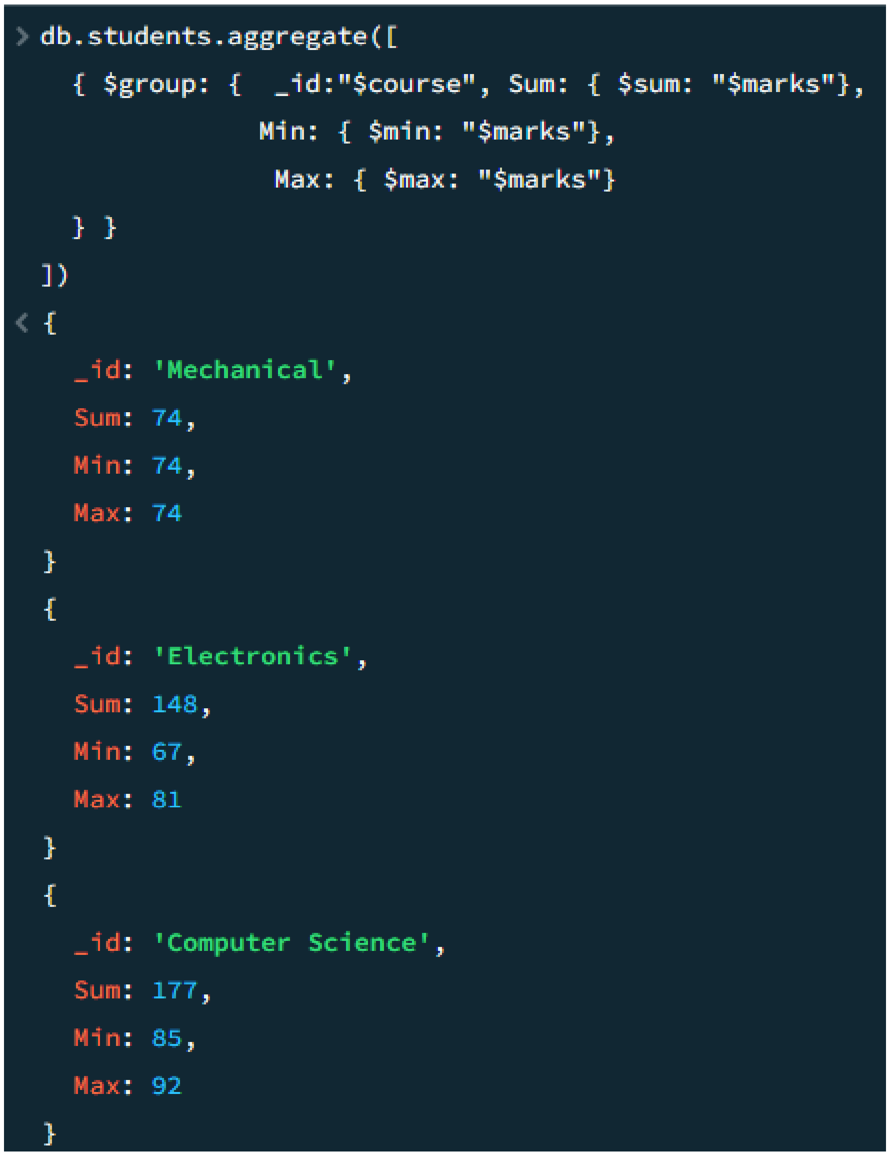


```
> db.students.countDocuments({ course: "Computer Science" })
< 2
```

6) **Aggregation Functions:**



```
> db.students.aggregate([
    { $group: { _id: "$course", avgMarks: { $avg: "$marks"} } }
  ])
< {
    _id: 'Mechanical',
    avgMarks: 74
  }
  {
    _id: 'Computer Science',
    avgMarks: 88.5
  }
  {
    _id: 'Electronics',
    avgMarks: 74
  }
```



```
> db.students.aggregate([
    { $group: {  _id:"$course", Sum: { $sum: "$marks"},
                Min: { $min: "$marks"},
                 Max: { $max: "$marks"}
    } }
  ])
< {
    _id: 'Mechanical',
    Sum: 74,
    Min: 74,
    Max: 74
  }
  {
    _id: 'Electronics',
    Sum: 148,
    Min: 67,
    Max: 81
  }
  {
    _id: 'Computer Science',
    Sum: 177,
    Min: 85,
    Max: 92
  }
```

CONCLUSION:

DISCUSSION AND VIVA VOCE:

- How do you count the total number of documents in a MongoDB collection?
- Which methods are used to remove documents from a MongoDB collection?
- How can you sort documents in ascending or descending order in MongoDB?
- Which functions are used to limit and skip documents while fetching data in MongoDB?
- How do you perform aggregation operations like sum, average, or group in MongoDB?

REFERENCE:

- https://blog.sqlauthority.com/2020/05/22/mongodb-fundamentals-crud-deleting-objects-day-5-of-6/
- https://www.tutorialspoint.com/mongodb/mongodb_quick_guide.htm
- https://dbdmg.polito.it/wordpress/wp-content/uploads/2019/11/02-MongoDB-query.pdf

Observation book: (3)	Viva-Voce (3)	Quality of Submission and timely Evaluation (4)
Total:		Sign with date: