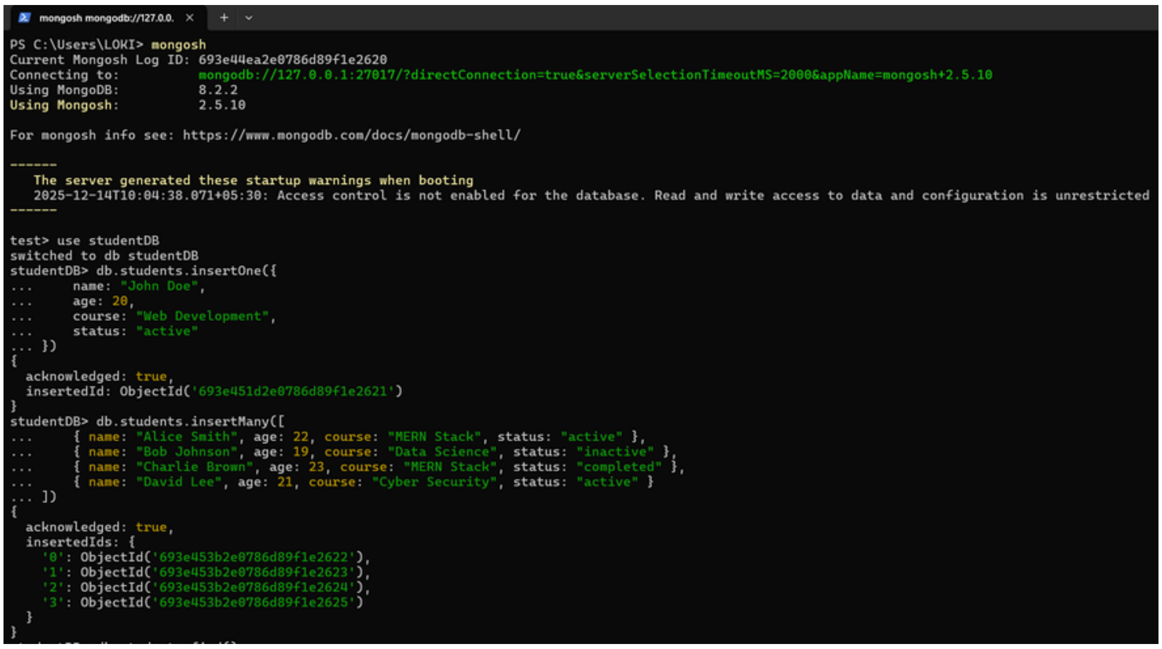
Task 1: Database Setup & Insert

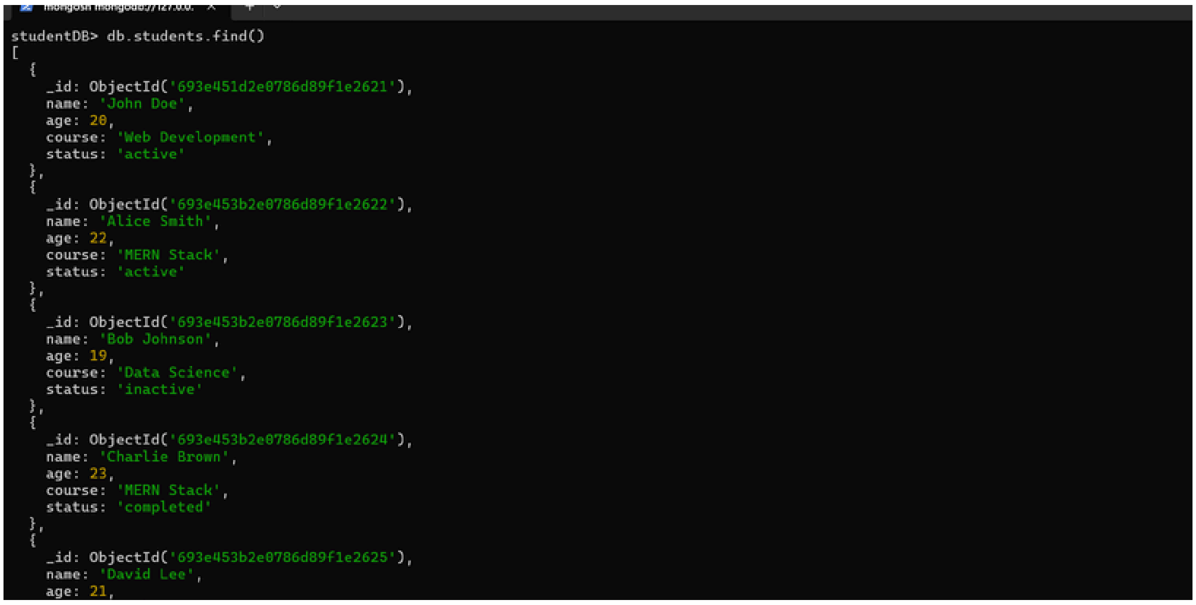
- use studentDB and creating the collection.
- insertOne, insertMany commands and the acknowledged: true output.



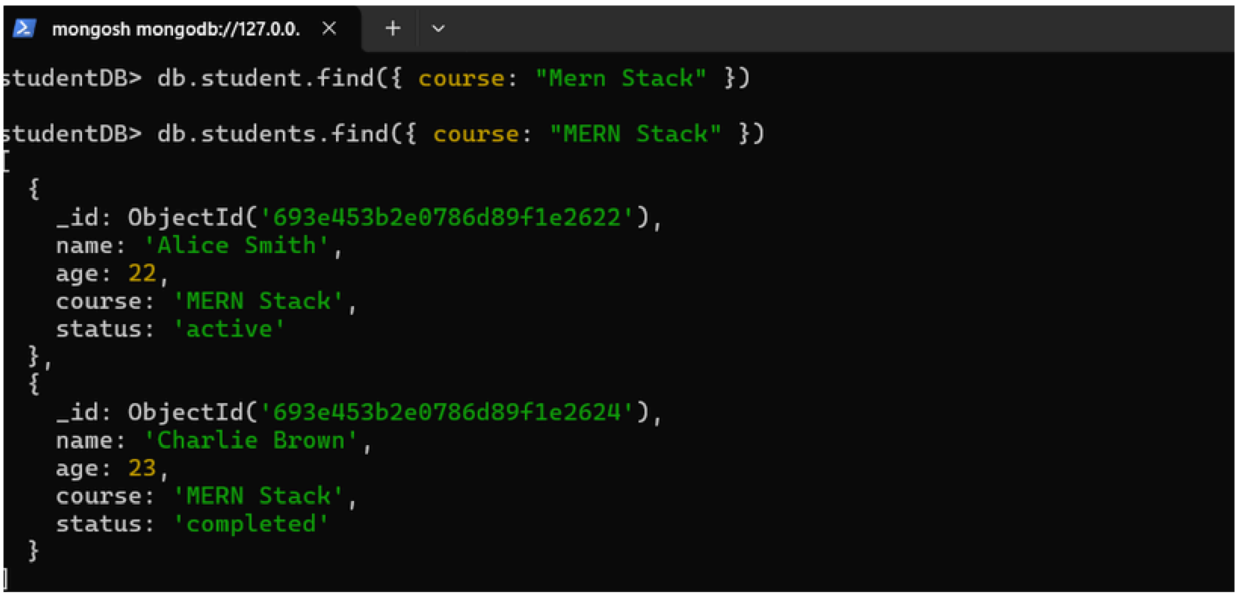
```
PS C:\Users\LOKI> mongosh
Current Mongosh Log ID: 693e44ea2e0786d89f1e2620
Connecting to:          mongodb://127.0.0.1:27017/?directConnection=true&serverSelectionTimeoutMS=2000&appName=mongosh+2.5.10
Using MongoDB:          8.2.2
Using Mongosh:          2.5.10

For mongosh info see: https://www.mongodb.com/docs/mongodb-shell/

------
   The server generated these startup warnings when booting
   2025-12-14T10:04:38.071+05:30: Access control is not enabled for the database. Read and write access to data and configuration is unrestricted
------

test> use studentDB
switched to db studentDB
studentDB> db.students.insertOne({
...     name: "John Doe",
...     age: 20,
...     course: "Web Development",
...     status: "active"
... })
{
  acknowledged: true,
  insertedId: ObjectId('693e451d2e0786d89f1e2621')
}
studentDB> db.students.insertMany([
...     { name: "Alice Smith", age: 22, course: "MERN Stack", status: "active" },
...     { name: "Bob Johnson", age: 19, course: "Data Science", status: "inactive" },
...     { name: "Charlie Brown", age: 23, course: "MERN Stack", status: "completed" },
...     { name: "David Lee", age: 21, course: "Cyber Security", status: "active" }
... ])
{
  acknowledged: true,
  insertedIds: {
    '0': ObjectId('693e453b2e0786d89f1e2622'),
    '1': ObjectId('693e453b2e0786d89f1e2623'),
    '2': ObjectId('693e453b2e0786d89f1e2624'),
    '3': ObjectId('693e453b2e0786d89f1e2625')
  }
}
```

Task 2: Read Operation

- ✴ **The list of students from db.students.find()**



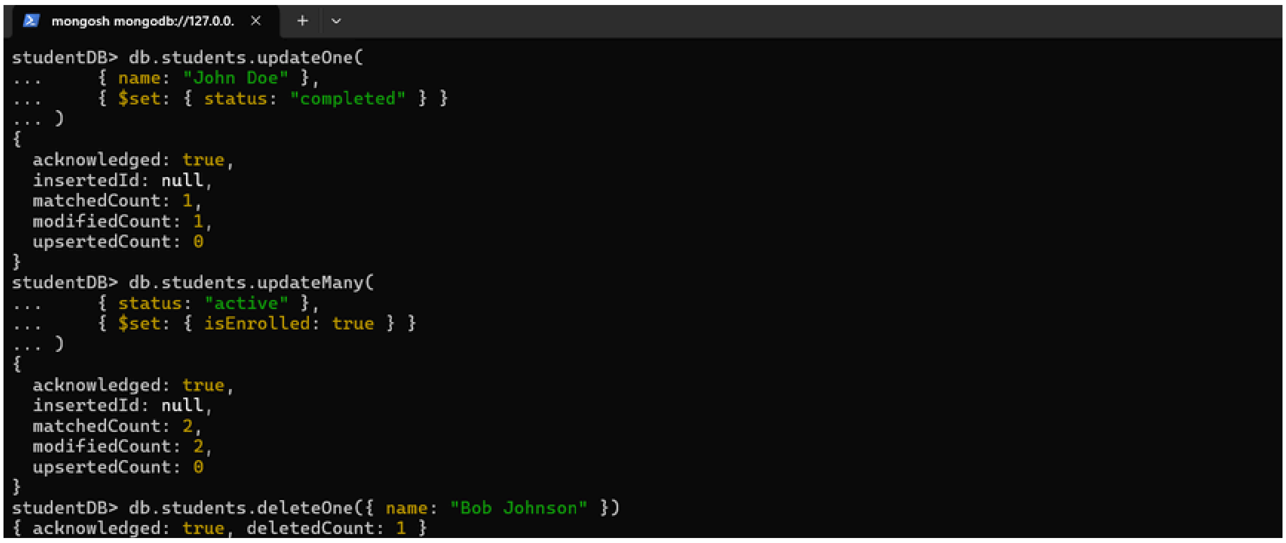
```
studentDB> db.students.find()
[
  {
    _id: ObjectId('693e451d2e0786d89f1e2621'),
    name: 'John Doe',
    age: 20,
    course: 'Web Development',
    status: 'active'
  },
  {
    _id: ObjectId('693e453b2e0786d89f1e2622'),
    name: 'Alice Smith',
    age: 22,
    course: 'MERN Stack',
    status: 'active'
  },
  {
    _id: ObjectId('693e453b2e0786d89f1e2623'),
    name: 'Bob Johnson',
    age: 19,
    course: 'Data Science',
    status: 'inactive'
  },
  {
    _id: ObjectId('693e453b2e0786d89f1e2624'),
    name: 'Charlie Brown',
    age: 23,
    course: 'MERN Stack',
    status: 'completed'
  },
  {
    _id: ObjectId('693e453b2e0786d89f1e2625'),
    name: 'David Lee',
    age: 21,
```



```
studentDB> db.student.find({ course: "Mern Stack" })

studentDB> db.students.find({ course: "MERN Stack" })
[
  {
    _id: ObjectId('693e453b2e0786d89f1e2622'),
    name: 'Alice Smith',
    age: 22,
    course: 'MERN Stack',
    status: 'active'
  },
  {
    _id: ObjectId('693e453b2e0786d89f1e2624'),
    name: 'Charlie Brown',
    age: 23,
    course: 'MERN Stack',
    status: 'completed'
  }
]
```

Task 3: Update & Delete

- The modifiedCount or deletedCount output.

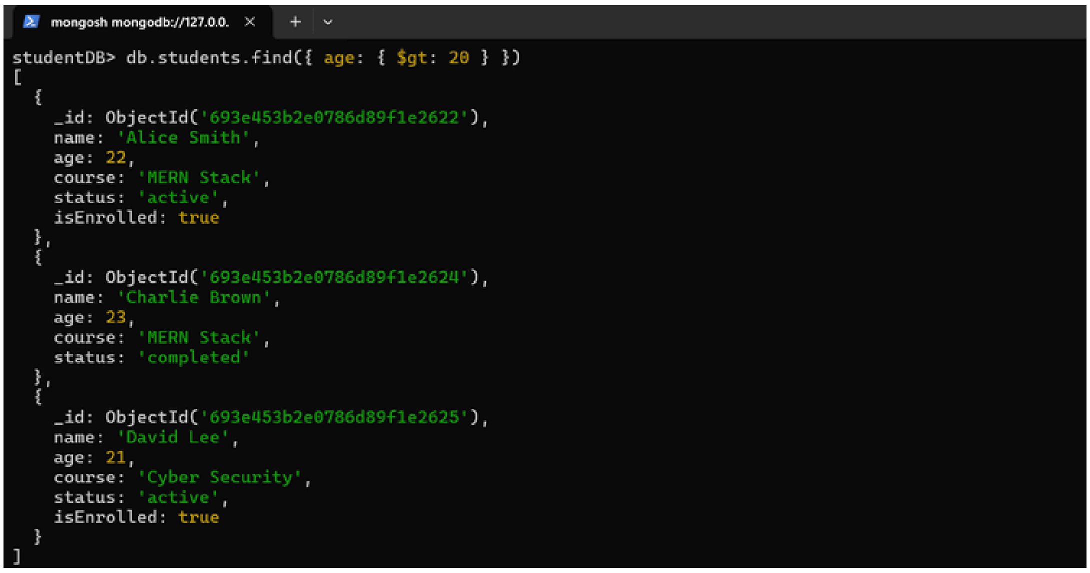


mongosh mongodb://127.0.0.

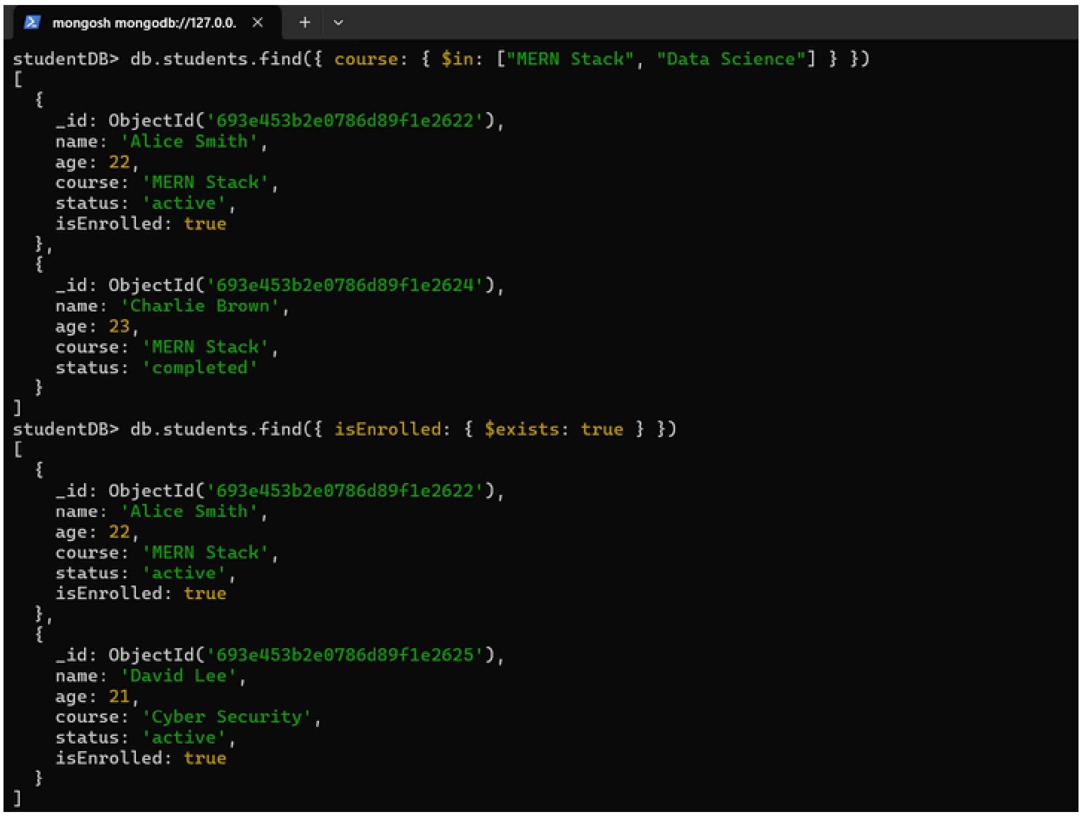
```
studentDB> db.students.updateOne(
...     { name: "John Doe" },
...     { $set: { status: "completed" } }
... )
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
studentDB> db.students.updateMany(
...     { status: "active" },
...     { $set: { isEnrolled: true } }
... )
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 2,
  modifiedCount: 2,
  upsertedCount: 0
}
studentDB> db.students.deleteOne({ name: "Bob Johnson" })
{ acknowledged: true, deletedCount: 1 }
```

Task 4: Query Operation

✴ **The results of queries like $gt, $in, $exists.**



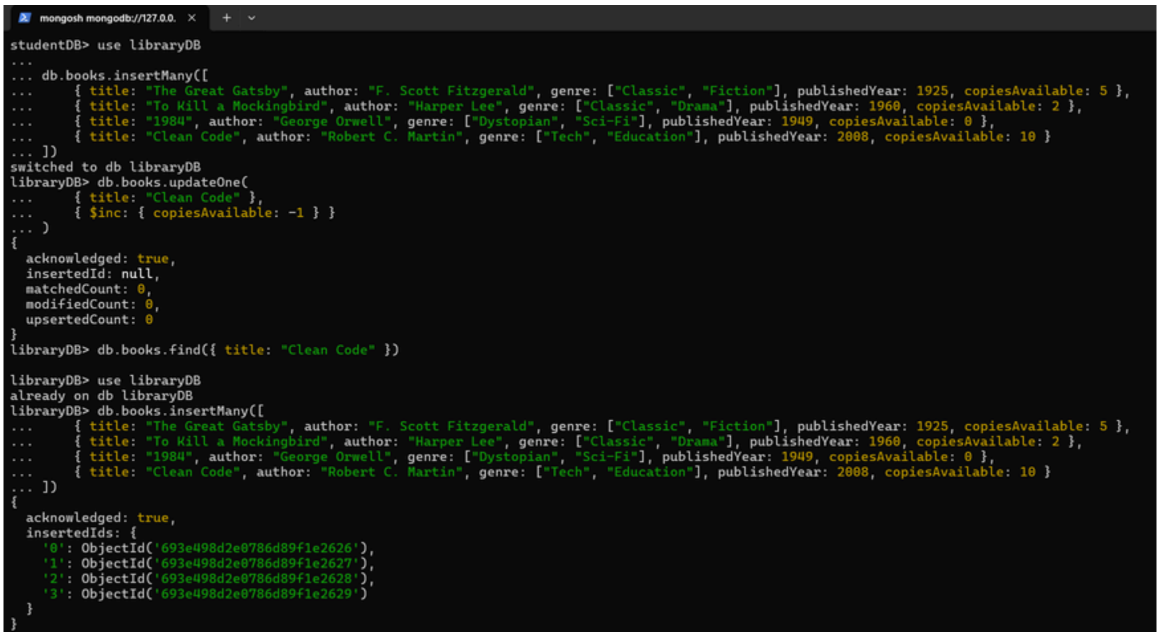
```
studentDB> db.students.find({ age: { $gt: 20 } })
[
  {
    _id: ObjectId('693e453b2e0786d89f1e2622'),
    name: 'Alice Smith',
    age: 22,
    course: 'MERN Stack',
    status: 'active',
    isEnrolled: true
  },
  {
    _id: ObjectId('693e453b2e0786d89f1e2624'),
    name: 'Charlie Brown',
    age: 23,
    course: 'MERN Stack',
    status: 'completed'
  },
  {
    _id: ObjectId('693e453b2e0786d89f1e2625'),
    name: 'David Lee',
    age: 21,
    course: 'Cyber Security',
    status: 'active',
    isEnrolled: true
  }
]
```



```
studentDB> db.students.find({ course: { $in: ["MERN Stack", "Data Science"] } })
[
  {
    _id: ObjectId('693e453b2e0786d89f1e2622'),
    name: 'Alice Smith',
    age: 22,
    course: 'MERN Stack',
    status: 'active',
    isEnrolled: true
  },
  {
    _id: ObjectId('693e453b2e0786d89f1e2624'),
    name: 'Charlie Brown',
    age: 23,
    course: 'MERN Stack',
    status: 'completed'
  }
]
studentDB> db.students.find({ isEnrolled: { $exists: true } })
[
  {
    _id: ObjectId('693e453b2e0786d89f1e2622'),
    name: 'Alice Smith',
    age: 22,
    course: 'MERN Stack',
    status: 'active',
    isEnrolled: true
  },
  {
    _id: ObjectId('693e453b2e0786d89f1e2625'),
    name: 'David Lee',
    age: 21,
    course: 'Cyber Security',
    status: 'active',
    isEnrolled: true
  }
]
```

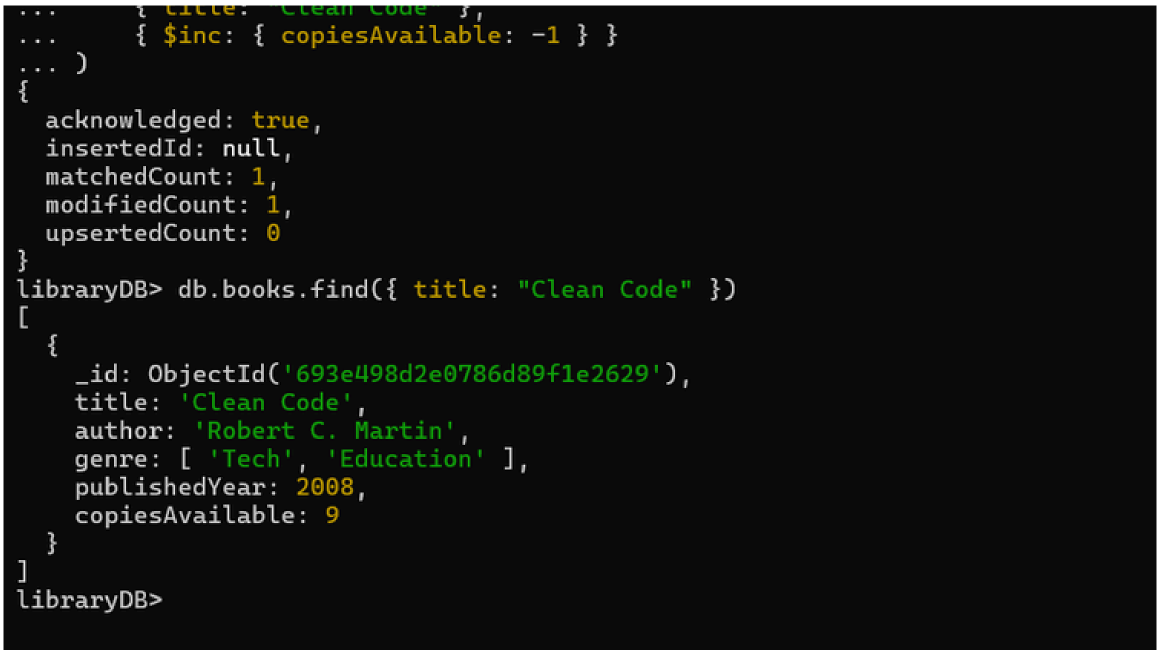
Task 5: Library Use Case

- **The libraryDB setup and the final "CleanCode" update showing 9 copies.**



mongosh mongodb://127.0.0.

```
studentDB> use libraryDB
...
... db.books.insertMany([
...     { title: "The Great Gatsby", author: "F. Scott Fitzgerald", genre: ["Classic", "Fiction"], publishedYear: 1925, copiesAvailable: 5 },
...     { title: "To Kill a Mockingbird", author: "Harper Lee", genre: ["Classic", "Drama"], publishedYear: 1960, copiesAvailable: 2 },
...     { title: "1984", author: "George Orwell", genre: ["Dystopian", "Sci-Fi"], publishedYear: 1949, copiesAvailable: 0 },
...     { title: "Clean Code", author: "Robert C. Martin", genre: ["Tech", "Education"], publishedYear: 2008, copiesAvailable: 10 }
... ])
switched to db libraryDB
libraryDB> db.books.updateOne(
...     { title: "Clean Code" },
...     { $inc: { copiesAvailable: -1 } }
... )
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 0,
  modifiedCount: 0,
  upsertedCount: 0
}
libraryDB> db.books.find({ title: "Clean Code" })

libraryDB> use libraryDB
already on db libraryDB
libraryDB> db.books.insertMany([
...     { title: "The Great Gatsby", author: "F. Scott Fitzgerald", genre: ["Classic", "Fiction"], publishedYear: 1925, copiesAvailable: 5 },
...     { title: "To Kill a Mockingbird", author: "Harper Lee", genre: ["Classic", "Drama"], publishedYear: 1960, copiesAvailable: 2 },
...     { title: "1984", author: "George Orwell", genre: ["Dystopian", "Sci-Fi"], publishedYear: 1949, copiesAvailable: 0 },
...     { title: "Clean Code", author: "Robert C. Martin", genre: ["Tech", "Education"], publishedYear: 2008, copiesAvailable: 10 }
... ])
{
  acknowledged: true,
  insertedIds: {
    '0': ObjectId('693e498d2e0786d89f1e2626'),
    '1': ObjectId('693e498d2e0786d89f1e2627'),
    '2': ObjectId('693e498d2e0786d89f1e2628'),
    '3': ObjectId('693e498d2e0786d89f1e2629')
  }
}
```



```
...     { title: "Clean Code" },
...     { $inc: { copiesAvailable: -1 } }
... )
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
libraryDB> db.books.find({ title: "Clean Code" })
[
  {
    _id: ObjectId('693e498d2e0786d89f1e2629'),
    title: 'Clean Code',
    author: 'Robert C. Martin',
    genre: [ 'Tech', 'Education' ],
    publishedYear: 2008,
    copiesAvailable: 9
  }
]
libraryDB>
```