Student Database Management with MongoDB

Dataset

In this project, I designed the dataset to simulate a real-world student database that stores academic and personal information for students across various courses and cities. The database is implemented using MongoDB, where the StudentDB database contains a collection named students.

Each student record consists of the following fields:

- **Roll:** A unique integer value assigned to each student for identification.
- **Name:** The full name of the student.
- **Course:** The academic program in which the student is enrolled, such as "Data Science", "Computer Science", "Electronics", "Mechanical", or "Information Technology".
- **Marks:** The numerical score the student has obtained in the course.
- **City:** The location where the student resides.
- **EmailID:** The contact email of the student.

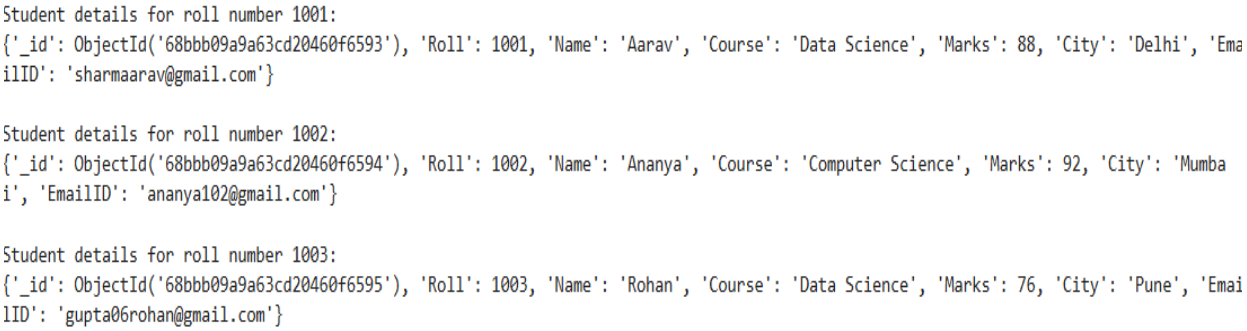
The dataset includes 15 records with diverse values to cover different courses and cities. The marks range from below passing to excellent performance, allowing meaningful filtering, sorting, and analysis.

Queries Executed

The following queries and operations were executed on the dataset:

1. **Displaying all students**

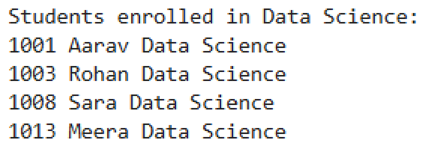
   The entire dataset was retrieved to view all student details.



```
Student details for roll number 1001:
{'_id': ObjectId('68bbb09a9a63cd20460f6593'), 'Roll': 1001, 'Name': 'Aarav', 'Course': 'Data Science', 'Marks': 88, 'City': 'Delhi', 'Ema
ilID': 'sharmaarav@gmail.com'}

Student details for roll number 1002:
{'_id': ObjectId('68bbb09a9a63cd20460f6594'), 'Roll': 1002, 'Name': 'Ananya', 'Course': 'Computer Science', 'Marks': 92, 'City': 'Mumba
i', 'EmailID': 'ananya102@gmail.com'}

Student details for roll number 1003:
{'_id': ObjectId('68bbb09a9a63cd20460f6595'), 'Roll': 1003, 'Name': 'Rohan', 'Course': 'Data Science', 'Marks': 76, 'City': 'Pune', 'Emai
lID': 'gupta06rohan@gmail.com'}
```

2. **Finding students by course**

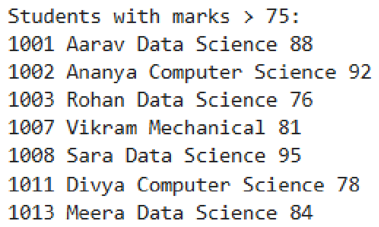
   Students enrolled in "Data Science" were filtered and displayed.



```
Students enrolled in Data Science:
1001 Aarav Data Science
1003 Rohan Data Science
1008 Sara Data Science
1013 Meera Data Science
```

3. **Finding students with marks greater than 75**

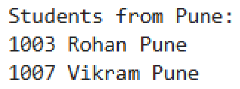
   A query was executed to list students performing above 75 marks.



```
Students with marks > 75:
1001 Aarav Data Science 88
1002 Ananya Computer Science 92
1003 Rohan Data Science 76
1007 Vikram Mechanical 81
1008 Sara Data Science 95
1011 Divya Computer Science 78
1013 Meera Data Science 84
```

4. **Finding students by city**

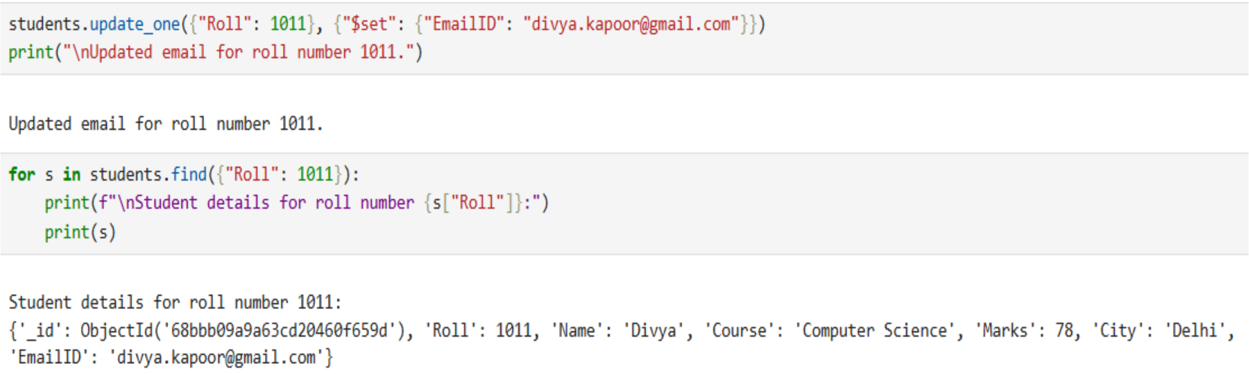
   Students from the city "Pune" were retrieved.



```
Students from Pune:
1003 Rohan Pune
1007 Vikram Pune
```

5. **Updating a student’s email**

   The email for the student with roll number 1011 was updated to a corrected value.



```
students.update_one({"Roll": 1011}, {"$set": {"EmailID": "divya.kapoor@gmail.com"}})
print("\nUpdated email for roll number 1011.")
```

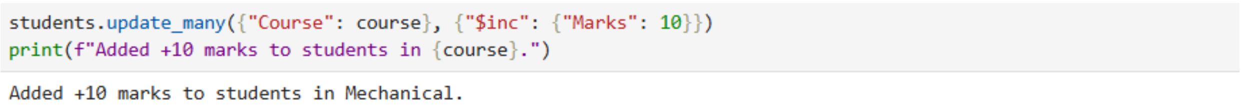
```
Updated email for roll number 1011.
```

```
for s in students.find({"Roll": 1011}):
    print(f"\nStudent details for roll number {s["Roll"]}:")
    print(s)
```

```
Student details for roll number 1011:
{'_id': ObjectId('68bbb09a9a63cd20460f659d'), 'Roll': 1011, 'Name': 'Divya', 'Course': 'Computer Science', 'Marks': 78, 'City': 'Delhi',
'EmailID': 'divya.kapoor@gmail.com'}
```

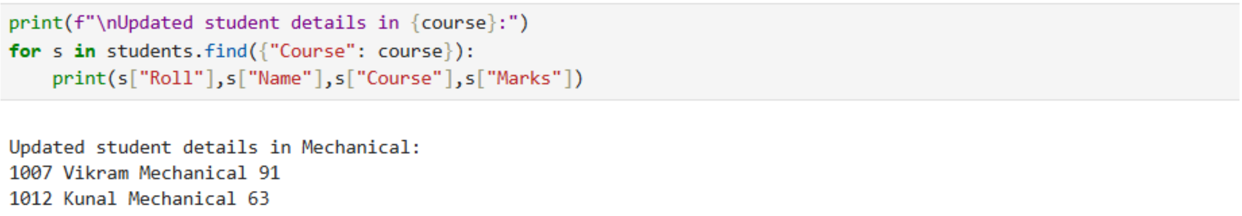
6. **Adding marks**

   All students enrolled in "Mechanical" were given an additional 10 marks using the $inc operator.



```
students.update_many({"Course": course}, {"$inc": {"Marks": 10}})
print(f"Added +10 marks to students in {course}.")
```

```
Added +10 marks to students in Mechanical.
```

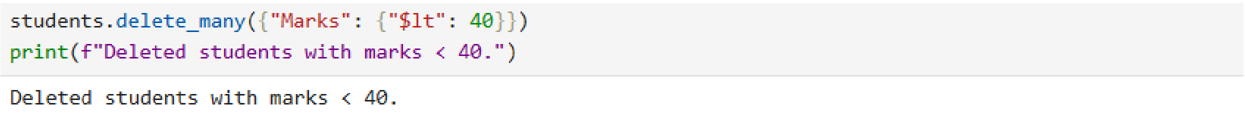


```
print(f"\nUpdated student details in {course}:")
for s in students.find({"Course": course}):
    print(s["Roll"],s["Name"],s["Course"],s["Marks"])
```

```
Updated student details in Mechanical:
1007 Vikram Mechanical 91
1012 Kunal Mechanical 63
```

7. Deleting students with marks less than 40

Students performing poorly were deleted from the database.

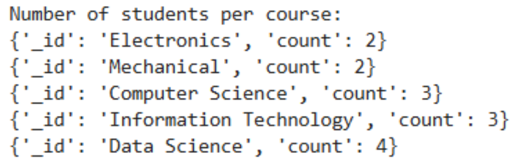


```
students.delete_many({"Marks": {"$lt": 40}})
print(f"Deleted students with marks < 40.")
```

```
Deleted students with marks < 40.
```

8. Counting students by course

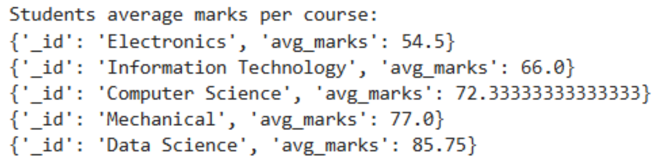
An aggregation query was executed to count the number of students in each course.



```
Number of students per course:
{'_id': 'Electronics', 'count': 2}
{'_id': 'Mechanical', 'count': 2}
{'_id': 'Computer Science', 'count': 3}
{'_id': 'Information Technology', 'count': 3}
{'_id': 'Data Science', 'count': 4}
```

9. Calculating average marks per course

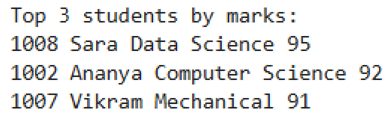
The average marks for each course were computed.



```
Students average marks per course:
{'_id': 'Electronics', 'avg_marks': 54.5}
{'_id': 'Information Technology', 'avg_marks': 66.0}
{'_id': 'Computer Science', 'avg_marks': 72.33333333333333}
{'_id': 'Mechanical', 'avg_marks': 77.0}
{'_id': 'Data Science', 'avg_marks': 85.75}
```

10. Finding top 3 students by marks

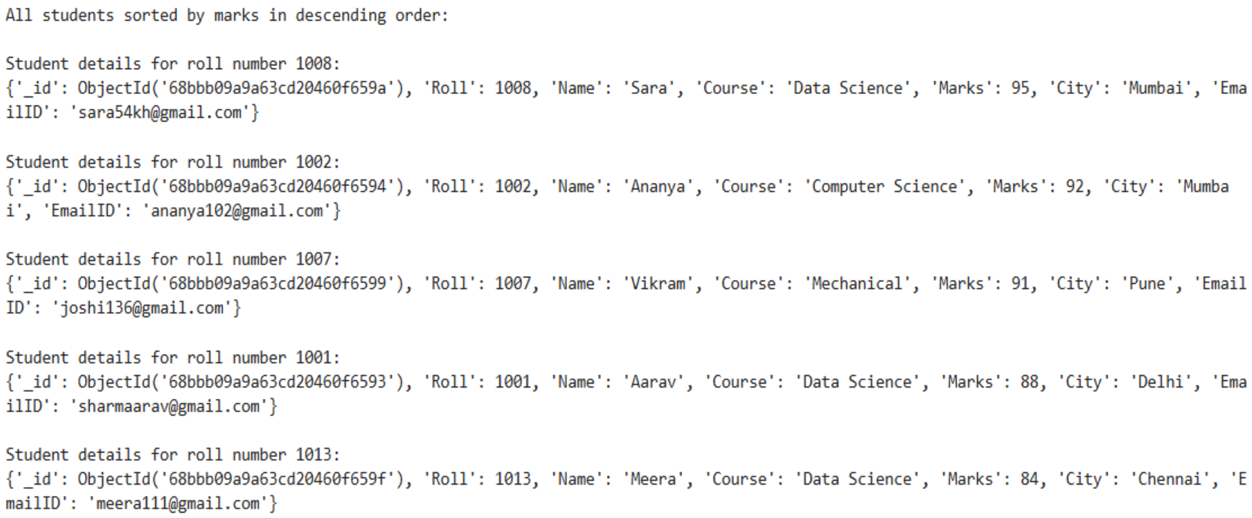
A query sorted by descending marks was used to identify the highest-scoring students.



```
Top 3 students by marks:
1008 Sara Data Science 95
1002 Ananya Computer Science 92
1007 Vikram Mechanical 91
```

11. Sorting all students by marks

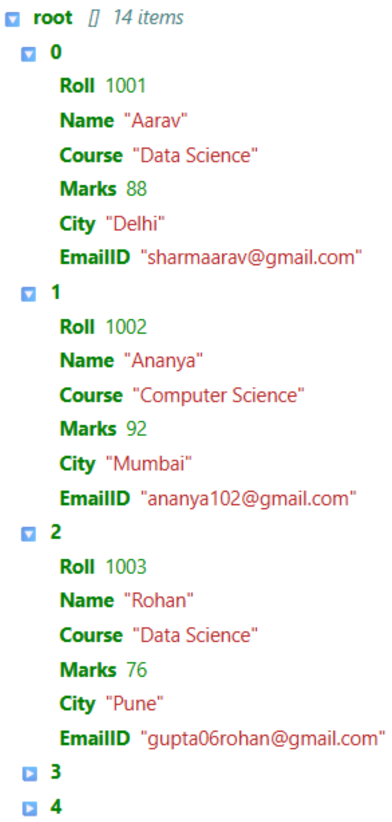
The entire dataset was sorted in descending order based on marks.



```
All students sorted by marks in descending order:

Student details for roll number 1008:
{'_id': ObjectId('68bbb09a9a63cd20460f659a'), 'Roll': 1008, 'Name': 'Sara', 'Course': 'Data Science', 'Marks': 95, 'City': 'Mumbai', 'Ema
ilID': 'sara54kh@gmail.com'}

Student details for roll number 1002:
{'_id': ObjectId('68bbb09a9a63cd20460f6594'), 'Roll': 1002, 'Name': 'Ananya', 'Course': 'Computer Science', 'Marks': 92, 'City': 'Mumba
i', 'EmailID': 'ananya102@gmail.com'}

Student details for roll number 1007:
{'_id': ObjectId('68bbb09a9a63cd20460f6599'), 'Roll': 1007, 'Name': 'Vikram', 'Course': 'Mechanical', 'Marks': 91, 'City': 'Pune', 'Email
ID': 'joshi136@gmail.com'}

Student details for roll number 1001:
{'_id': ObjectId('68bbb09a9a63cd20460f6593'), 'Roll': 1001, 'Name': 'Aarav', 'Course': 'Data Science', 'Marks': 88, 'City': 'Delhi', 'Ema
ilID': 'sharmaarav@gmail.com'}

Student details for roll number 1013:
{'_id': ObjectId('68bbb09a9a63cd20460f659f'), 'Roll': 1013, 'Name': 'Meera', 'Course': 'Data Science', 'Marks': 84, 'City': 'Chennai', 'E
mailID': 'meera111@gmail.com'}
```

12. Exporting data

The student records were exported into a JSON file format for sharing or further analysis.



```
root  [] 14 items
  0
    Roll  1001
    Name  "Aarav"
    Course  "Data Science"
    Marks  88
    City  "Delhi"
    EmailID  "sharmaarav@gmail.com"
  1
    Roll  1002
    Name  "Ananya"
    Course  "Computer Science"
    Marks  92
    City  "Mumbai"
    EmailID  "ananya102@gmail.com"
  2
    Roll  1003
    Name  "Rohan"
    Course  "Data Science"
    Marks  76
    City  "Pune"
    EmailID  "gupta06rohan@gmail.com"
  3
  4
```

Insights

After running the queries and analyzing the dataset, the following insights were gathered:

- Best-performing course:

  The "Data Science" course had some of the highest-performing students, with multiple entries scoring above 80 marks.
- Top students:

  The top 3 students based on marks were:
  - ✓ Roll No. 1008 – Sara – 95 marks
  - ✓ Roll No. 1002 – Ananya – 92 marks
  - ✓ Roll No. 1007 – Vikram – 91 marks
- Students benefiting from updates:

  Students in the "Mechanical" course received a 10-mark bonus, significantly improving their results.
- Data integrity:

  After deleting students with marks less than 40, the dataset reflected only those students who were meeting or exceeding the minimum academic requirements.
- Average performance:

  The aggregation results showed varying average marks across courses, which could help in identifying areas needing academic support.