Lab 6: MongoDB Querying

CSL303: Database Management Systems

Objective

To gain hands-on experience with the MongoDB query language. This lab covers data manipulation (insertion, updates, deletion) and a wide range of data retrieval techniques using the `find` command and the aggregation framework.

Setup & Data Model

This lab requires a running MongoDB instance. You will be working with a university database consisting of three collections: `instructors`, `courses`, and `students`.

Step 1: Populate the Database

Connect to your MongoDB instance using the `mongosh` shell. Create a new database named `university` and populate the collections by running the following commands.

```
// Switch to or create the university database
use university;

// Populate the 'instructors' collection
db.instructors.insertMany([
  { _id: 1, name: "Dr. Turing", department: "CS", rank: "Professor" },
  { _id: 2, name: "Dr. Codd", department: "CS", rank: "Professor" },
  { _id: 3, name: "Dr. Ohm", department: "EE", rank: "Associate Professor" },
  { _id: 4, name: "Dr. Curie", department: "Physics", rank: "Professor" }
]);

// Populate the 'courses' collection
db.courses.insertMany([
  { _id: "CSL101", title: "Intro to Programming", credits: 3, instructor_id: 1 },
  { _id: "CSL303", title: "Databases", credits: 4, instructor_id: 2 },
  { _id: "EEL201", title: "Digital Circuits", credits: 3, instructor_id: 3 },
  { _id: "PHL100", title: "Intro to Physics", credits: 4, instructor_id: 4 },
  { _id: "CSL211", title: "Data Structures", credits: 4, instructor_id: 1 }
]);

// Populate the 'students' collection
db.students.insertMany([
  { _id: 101, name: "Alice", major: "CS", gpa: 3.9, courses_enrolled: [
                        { cid: "CSL101", grade: "A" }, { cid: "CSL303", grade: "A" } ] },
  { _id: 102, name: "Bob", major: "EE", gpa: 3.2, courses_enrolled: [
                        { cid: "EEL201", grade: "B" } ] },
  { _id: 103, name: "Charlie", major: "CS", gpa: 3.5, courses_enrolled: [
                        { cid: "CSL101", grade: "B" }, { cid: "CSL211", grade: "A" } ] },
  { _id: 104, name: "David", major: "Physics", gpa: 2.8, courses_enrolled: [
                        { cid: "PHL100", grade: "C" }, { cid: "CSL101", grade: "C" } ] },
  { _id: 105, name: "Eve", major: "CS", gpa: 4.0, courses_enrolled: [] }
]);
```

Instructions for Submission

You must submit a single JSON file named `lab4_submission.json`. The file should contain a single JSON array of objects. Each object must have two keys: `"question_number"` and `"query"`. The value for the `"query"` key should be your complete MongoDB command as a single string.

Submission Template:

```
[
  {
    "question_number": 1,
    "query": "db.students.insertOne({ _id: 106, name: \"Frank\", major: \"EE\", gpa: 3.1, courses_en
  },
  {
    "question_number": 2,
    "query": "db.courses.find({ credits: 4 });"
  }
]
```

Exercises

Part 1: Insert Operations

1. Insert a single new student with the following details: `_id: 106`, `name: "Frank"`, `major: "EE"`, `gpa: 3.1`, and an empty `courses_enrolled` array.
2. Insert two new courses at once:
   - `_id: "MEL110"`, `title: "Thermodynamics"`, `credits: 3`, `instructor_id: 4`.
   - `_id: "BIO101"`, `title: "Intro to Biology"`, `credits: 3`, `instructor_id: 4`.

Part 2: Find Queries

3. Find all students majoring in "CS".
4. Find all courses that are worth 4 credits, but only display their titles and instructor IDs. Do not show the `_id` field.
5. Find all students with a GPA greater than or equal to 3.5.
6. Find all instructors who are either a "Professor" or an "Associate Professor". Use the `$in` operator.
7. Find the names and majors of all students who are enrolled in the "Databases" course (`CSL303`).
8. Find the names of all students who have received a grade of "A" in any course.
9. Find all students who are enrolled in exactly two courses. Use the `$size` operator.
10. Find all courses that do not have an instructor assigned (assume this means the `instructor_id` field is missing or null). Use the `$exists` operator.
11. Find all students who are enrolled in both "CSL101" and "CSL211". Use the `$all` operator.

Part 3: Update Operations

12. The university has decided to rename the "CS" major to "Computer Science". Update all students in the "CS" major to their new major name. Use `updateMany`.
13. Alice (`_id: 101`) has just completed another course. Add a new course object `{ cid: "CSL211", grade: "A" }` to her `courses_enrolled` array. Use the `$push` operator.

14. Increase the GPA of all students in the "EE" major by 0.2 points. Use the `$inc` operator.

15. Change the grade for student Bob (`_id: 102`) in course "EEL201" to "A". (Hint: Use the arrayFilters option with the update command).

Part 4: Delete Operations

16. The "Thermodynamics" course (`MEL110`) has been cancelled. Delete it from the `courses` collection.

17. A student, David (`_id: 104`), has dropped out. Delete him from the `students` collection.

18. Alice (`_id: 101`) has decided to drop "CSL101". Remove only that course from her `courses_enrolled` array. Use the `$pull` operator.

Part 5: Aggregation Framework

19. Calculate the average GPA for each major. The output should have two fields: `_id` (the major) and `avgGpa`.

20. Count the number of courses taught by each instructor. The output should show the instructor's ID and the number of courses they teach, named `courseCount`.