MongoDB Assignment

1. Complex Filters & Projections

Q1. List the names and departments of students who have more than 85% attendance and are skilled in both "MongoDB" and "Python".

Query *-- db.student.find({"attendance": { "$gt": 85 },*
"skills": { "$all": ["Python", "MongoDB"] } },
{ "name": 1, "department": 1,"_id": 0})

Output --

```
> db.student.find({"attendance": { "$gt": 85 },
  "skills": { "$all": ["Python", "MongoDB"] } },
  { "name": 1, "department": 1,"_id": 0})
  //Shivam Kumar Maurya 1240258422
< {
    name: 'Mr. Darius Newman',
    department: 'Mechanical'
  }
  {
    name: 'Ronald Trevino',
    department: 'Electrical'
  }
```

Note—Added some new data.

Explanation – This query will return only names and departments of students who have more than 85% attendance & they are skilled in both MongoDB and Python.

Q2. Show all faculty who are teaching more than 2 courses. Display their names and the total number of courses they teach.

Query— *db.faculty.aggregate([{$project: {name: 1, totalCourses: {$size: "$courses"}}}, {$match: {totalCourses: {$gt: 2}}}, {$project: {_id: 0, name: 1, totalCourses: 1}}])*

Output--

```
>_MONGOSH
> db.faculty.aggregate([{$project:{name:1,totalCourses:{$size:"$courses"}}},
  {$match:{totalCourses:{$gt:2}}},{$project:{_id:0,name:1,totalCourses:1}}])
  //Shivam Kumar Maurya 1240258422
< {
    name: 'Charles Newton',
    totalCourses: 3
  }
  {
    name: 'Julia Cole',
    totalCourses: 3
  }
  {
    name: 'Darrell Velasquez',
    totalCourses: 3
  }
  {
    name: 'Michael Poole',
    totalCourses: 3
  }
  {
    name: 'John Duran',
    totalCourses: 3
  }
  {
    name: 'Daniel Allen',
    totalCourses: 3
  }
```

Explanation-- This query will show all faculty who are teaching more than 2 courses and will return only their names & total number of courses.

2. Joins ($lookup) and Aggregations

Q3. Write a query to show each student's name along with the course titles they are enrolled in (use $lookup between enrollments , students, and courses).

Query – *db.enrollment.aggregate([{ $lookup: { from: "student", localField: "student_id", foreignField: "_id", as: "student_info" }},*
{ $unwind: "$student_info" }, { $lookup: { from: "course", localField: "course_id",
foreignField: "_id", as: "course_info" }},

{ $unwind: "$course_info" }, { $project: { _id: 0, student_name: "$student_info.name", course_title: "$course_info.title" }}])

Output—

```
> db.enrollment.aggregate([{ $lookup: { from: "student", localField: "student_id", foreignField:
    "_id", as:  "student_info" }},{ $unwind: "$student_info" }, { $lookup: { from: "course",
    localField: "course_id",   foreignField: "_id", as: "course_info" }},{ $unwind: "$course_info" },
    { $project: { _id: 0, student_name: "$student_info.name",  course_title: "$course_info.title" }}])
  //Shivam Kumar Maurya 1240258422
< {
    student_name: 'Alexandra Bailey',
    course_title: 'Reactive neutral adapter'
  }
  {
    student_name: 'Megan Taylor',
    course_title: 'Sharable bifurcated paradigm'
  }
  {
    student_name: 'Alejandro Hart',
    course_title: 'Focused user-facing paradigm'
  }
  {
    student_name: 'Timothy Sparks',
    course_title: 'Focused user-facing paradigm'
  }
  {
    student_name: 'Juan Morris',
    course_title: 'Balanced asynchronous framework'
  }
```

Explantion – This query will show each student's name along with the course title they are enrolled in with using $lookup between enrollment, student, and course.

Q4. For each course, display the course title, number of students enrolled, and average marks
(use $group).

Query-- *db.enrollment.aggregate([{ $group: { _id: "$course_id", total_students: { $sum: 1 }, avg_marks: { $avg: "$marks" }}},*
{ $lookup: { from: "course", localField: "_id", foreignField: "_id", as: "course_info" }},{ $unwind: "$course_info" },

{ $project: { _id: 0, course_title: "$course_info.title", total_students: 1, avg_marks: { $round: ["$avg_marks", 2] }}}])

Output—

```
> db.enrollment.aggregate(
    [{ $group: { _id: "$course_id", total_students: { $sum: 1 }, avg_marks: { $avg: "$marks" }}},
    { $lookup: { from: "course", localField: "_id", foreignField: "_id", as: "course_info" }},
    { $unwind: "$course_info" },
    { $project: { _id: 0, course_title: "$course_info.title", total_students: 1, avg_marks: { $round:
    ["$avg_marks", 2] }}}])
    //Shivam Kumar Maurya 1240258422

< {
    total_students: 2,
    course_title: 'Configurable scalable data-warehouse',
    avg_marks: 77.5
  }
  {
    total_students: 1,
    course_title: 'Enhanced intangible emulation',
    avg_marks: 78
  }
  {
    total_students: 1,
    course_title: 'Fully-configurable reciprocal installation',
    avg_marks: 53
  }
  {
    total_students: 1,
    course_title: 'Triple-buffered cohesive frame',
    avg_marks: 82
```

Explanation—This questions query will show the course title, number of students enrolled and average marks for each course given in the data.

3. Grouping, Sorting, and Limiting

Q5. Find the top 3 students with the highest average marks across all enrolled courses.

Query – *db.enrollment.aggregate([{$group: {_id: "$student_id", averageMarks: {$avg: "$marks"}}}, {$sort: {averageMarks: -1}}, {$limit: 3}])*

Output--

```
> db.enrollment.aggregate([{$group: {_id: "$student_id",
  averageMarks: {$avg: "$marks"}}}, {$sort:
  {averageMarks: -1}}, {$limit: 3}])
  //Shivam Kumar Maurya 1240258422
< {
    _id: 'S080',
    averageMarks: 100
  }
  {
    _id: 'S046',
    averageMarks: 98
  }
  {
    _id: 'S041',
    averageMarks: 94
  }
```

Explanation -- This Query will return highest average marks of top 3 students in the given collection across all enrolled courses.

Q6. Count how many students are in each department. Display the department with the highest number of students.

Query-- *db.student.aggregate([{$group: {_id: "$department", studentCount: {$sum: 1}}}, {$sort: {studentCount: -1}}, {$limit: 1}])*

Output--

```
> db.student.aggregate([{$group: {_id: "$department", studentCount:
  {$sum: 1}}}, {$sort: {studentCount: -1}}, {$limit: 1}])
  //Shivam Kumar Maurya 1240258422
< {
    _id: 'Electrical',
    studentCount: 23
  }
```

Explanation—Firstly this query count how many students are in each department, one by one and then display that department which is highest number of students.

4. Update, Upsert, and Delete

Q7. Update attendance to 100% for all students who won any "Hackathon".

Query— *db.student.updateMany({ activities: "Hackathon" }, { $set: { attendance: 100 }})*

Output--

```
> db.student.updateMany(
  { activities: "Hackathon" },
  { $set: { attendance: 100 }})
  //Shivam Kumar Maurya 1240258422
< {
    acknowledged: true,
    insertedId: null,
    matchedCount: 0,
    modifiedCount: 0,
    upsertedCount: 0
  }
```

Explanation--This query is use to update attendance of only those students who won any “Hackathon” by changing to 100% .

Q8. Delete all student activity records where the activity year is before 2022.

Query-- *db.activities.deleteMany({ year: { $lt: 2022 } })*

Output--

```
> db.activities.deleteMany({ year: { $lt: 2022 } })
  //Shivam Kumar Maurya 1240258422
< {
    acknowledged: true,
    deletedCount: 0
  }
```

Explanation-- This query will delete all the records where the activity year is before 2022 of collection name activities.

Q9. Upsert a course record for "Data Structures" with ID "C150" and credits 4—if it doesn't exist, insert it; otherwise update its title to "Advanced Data Structures".

Query-- *db.course.updateOne({_id: "C150"}, {$set: {title: "Advanced Data Structures", credits: 4}}, {upsert: true})*

Output--

```
> db.course.updateOne({_id: "C150"}, {$set:
  {title: "Advanced Data Structures", credits: 4}},
  {upsert: true})//Shivam Kumar Maurya 1240258422

< {
    acknowledged: true,
    insertedId: null,
    matchedCount: 1,
    modifiedCount: 0,
    upsertedCount: 0
  }
```

Explanation -- This query will upsert a course record for "Data Structures" with ID "C150" and credits it 4-- if it doesn't exist & insert it otherwise update the title to "Advanced Data Structures".

5.Array & Operator Usage

Q10. Find all students who have "Python" as a skill but not "C++".

Query-- *db.student.find({ $and: [{ skills: "Python" }, { skills: { $ne: "C++" }}]},{_id: 0, name: 1, skills: 1 })*

Output—

```
> db.student.find( //Shivam Kumar Maurya 1240258422
  { $and: [{ skills: "Python" }, { skills: { $ne: "C++" }}]},
  {_id: 0, name: 1, skills: 1 })
< {
    name: 'Kyle Hale',
    skills: [
      'Python',
      'Java'
    ]
  }
  {
    name: 'Cody Whitehead',
    skills: [
      'JavaScript',
      'Python'
    ]
  }
  {
    name: 'Thomas Jackson',
    skills: [
      'Python',
      'AutoCAD'
    ]
  }
```

Explanation– This query will give output of all students name & in skills who have “Python” but not “C++”.

Q11. Return names of students who participated in "Seminar" and "Hackathon" both.

Query-- *db.activites.aggregate([{ $group: { _id: "$student_id", activityTypes: { $addToSet: "$type" }}},{ $match: { activityTypes: { $all: ["Seminar", "Hackathon"] }}}, { $lookup: { from: "student", localField: "_id", foreignField: "_id",as: "student_info" }},{ $unwind: "$student_info" }, { $project: { _id: 0, name: "$student_info.name" }}])*

Output—

```
> db.activites.aggregate([{ $group: { _id: "$student_id",
  activityTypes: { $addToSet: "$type" }}},{ $match: {
  activityTypes: { $all: ["Seminar", "Hackathon"] }}},
  { $lookup: { from: "student", localField: "_id", foreignField: "_id",
  as: "student_info" }},{ $unwind: "$student_info" },
  { $project: { _id: 0, name: "$student_info.name" }}])
  //Shivam Kumar Maurya 1240258422
< {
    name: 'Adam Solomon'
  }
  {
    name: 'Taylor Webb'
  }
  {
    name: 'Patricia Scott'
  }
  {
    name: 'Carlos Bryant'
  }
  {
    name: 'Lydia Day'
  }
```

Explanation– This query will return names of only those students who participated in “Seminar” and “Hackathon” both.

6.Subdocuments and Nested Conditions

Q12. Find students who scored more than 80 in "Web Development" only if they belong to the "Computer Science" department.

Query— *db.enrollment.find({course_title: "Web Development", marks: {$gt:80},department: "Computer Science"})*

Output—

```
ShivamNow> db.enrollment.find({course_title:"Web Development",marks:{$gt:80},
  department:"Computer Science"}) //Shivam Kymar Maurya 1240258422

ShivamNow>
```

Explanation– This Query will find the students who scored more than 80 in “Web Development” only if they belong to the “Computer Science” department.

7. Advanced Aggregation (Challenge Level)

Q13. For each faculty member, list the names of all students enrolled in their courses along with average marks per student per faculty.

Query-- *db.faculty.aggregate([{ $lookup: { from: "course", localField: "courses", foreignField: "_id", as: "courseInfo" }}, { $unwind: "$courseInfo" }, { $lookup: { from: "enrollment", localField: "courseInfo._id", foreignField: "course_id", as:"enrolledStudents" }}, { $unwind: "$enrolledStudents" }, { $lookup: { from: "student",localField: "enrolledStudents.student_id", foreignField: "_id", as: "studentInfo" }},{ $project: { _id: 0, facultyName: "$name", studentName: { $arrayElemAt:["$studentInfo.name",0] }, marks: "$enrolledStudents.marks" }}, { $group: { _id: { facultyName: "$facultyName", studentName: "$studentName" }, averageMarks: { $avg: "$marks" }}},{ $project: { _id: 0, facultyName: "$_id.facultyName", studentName: "$_id.studentName",averageMarks: 1 }}, { $sort: { facultyName: 1, studentName: 1 }}])*

Output--

```
db.faculty.aggregate( //Shivam Kumar Maurya 1240258422
[{ $lookup: { from: "course", localField: "courses", foreignField: "_id", as: "courseInfo" }},
{ $unwind: "$courseInfo" },  { $lookup: { from: "enrollment", localField: "courseInfo._id",
foreignField: "course_id", as:"enrolledStudents" }}, { $unwind: "$enrolledStudents" },
{ $lookup: { from: "student", localField: "enrolledStudents.student_id", foreignField: "_id",
as: "studentInfo" }},{ $project: { _id: 0, facultyName: "$name", studentName: { $arrayElemAt:
["$studentInfo.name",0] }, marks: "$enrolledStudents.marks" }}, { $group: { _id: {
facultyName: "$facultyName", studentName: "$studentName" },  averageMarks: {  $avg: "$marks" }}},
{ $project: { _id: 0, facultyName: "$_id.facultyName", studentName: "$_id.studentName",
averageMarks: 1 }}, { $sort: { facultyName: 1, studentName: 1 }}])

{
  averageMarks: 90,
  facultyName: 'Alexis Stone',
  studentName: 'Anthony Zavala'
}
{
  averageMarks: 93,
  facultyName: 'Alexis Stone',
  studentName: 'Barbara Jones'
}
{
  averageMarks: 69,
  facultyName: 'Andrew Mcmahon',
  studentName: 'Dr. Michael Griffin Jr.'
}
```

Explanation– This Query will list the names of all students enrolled in their courses along with average marks per students per faculty for each faculty.

Q14. Show the most popular activity type (e.g., Hackathon, Seminar, etc.) by number of student participants.

Query-- *db.activites.aggregate([{$group: {_id: "$type", participantCount: {$sum: 1}}}, {$sort: {participantCount: -1}}, {$limit: 1}, {$project: {_id: 0, activityType: "$_id", participantCount: 1}}])*

Output—

```
db.activites.aggregate([{$group: {_id: "$type", participantCount:
{$sum: 1}}}, {$sort: {participantCount: -1}}, {$limit: 1},
{$project: {_id: 0, activityType: "$_id", participantCount: 1}}])
//Shivam Kumar Maurya 1240258422
{
  participantCount: 35,
  activityType: 'Hackathon'
}
```

Explanation–This query will show most popular activity (e.g., Hackathon, Seminar) by total number of students participated.