**Experiment 6:**

Execute Aggregation Pipeline and its operations (pipeline must contain $match, $group, $sort, $project, $skip etc.

db.students.insertMany([

{

\_id: 67, name: "Amit", age: 24, department: "AIML", score: 85, joined: new Date("2023-01-10")

},

{

\_id: 68, name: "Sneha", age: 23, department: "CSE", score: 92, joined: new Date("2023-02-15")

},

{

\_id: 69, name: "Rohan", age: 22, department: "AIML", score: 78, joined: new Date("2023-03-20")

},

{

\_id: 70, name: "Priya", age: 25, department: "CSE", score: 88, joined: new Date("2023-04-05")

},

{

\_id: 71, name: "Rahul", age: 23, department: "AIML", score: 91, joined: new Date("2023-05-12")

}

])

db.students.aggregate([

// $match → Filter students who have a score greater than 80

{ $match: { score: { $gt: 80 } } },

// $group → Group by department and find the average score

{

$group: {

\_id: "$department",

avgScore: { $avg: "$score" },

totalStudents: { $sum: 1 }

}

},

// $sort → Sort by average score in descending order

{ $sort: { avgScore: -1 } },

// $project → Show only required fields

{

$project: {

\_id: 0,

department: "$\_id",

avgScore: 1,

totalStudents: 1

}

},

// $skip → Skip the first result

{ $skip: 1 }

])