

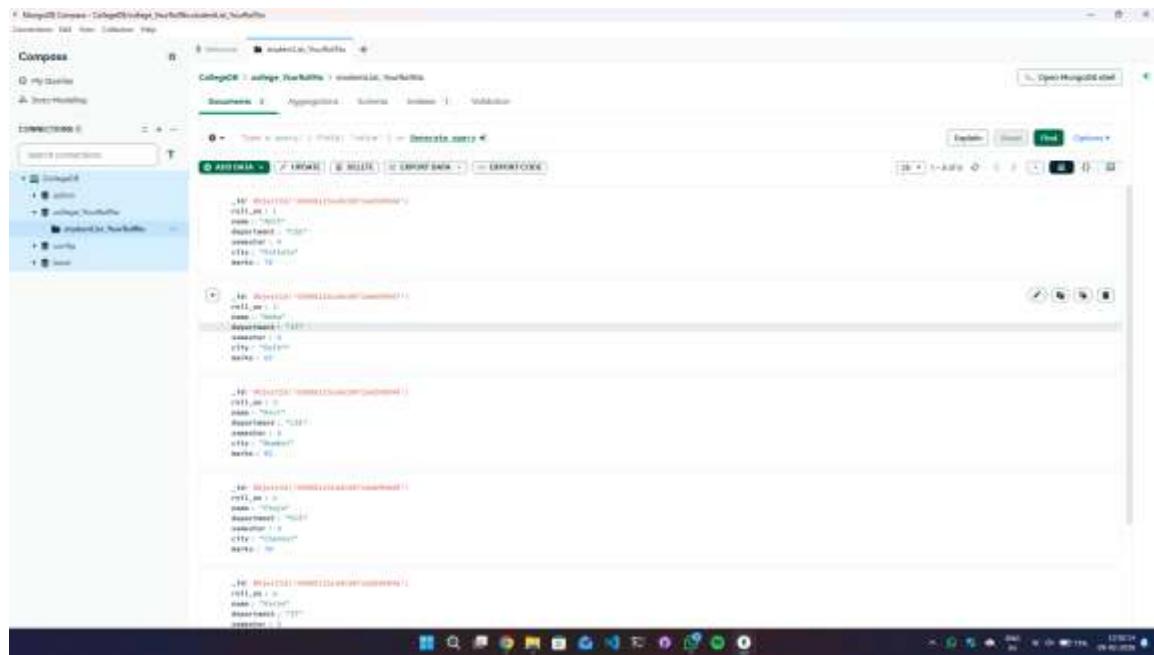
LAB ASSIGNMENT – 4

Name: DISHA BISWAS
Programme: B.Tech CSE(DA)
Roll Number: UG/02/BTCSEDA/2023/004
Registration No: AU/2023/0009796

MongoDB Aggregation Lab Answers

Given table :

Roll No	Name	Department	Semester	City	Marks
1	Amit	CSE	4	Kolkata	78
2	Neha	IT	4	Delhi	65
3	Ravi	CSE	5	Mumbai	82
4	Pooja	ECE	4	Chennai	70
5	Karan	IT	5	Kolkata	55
6	Sneha	CSE	4	Delhi	88



The screenshot shows the MongoDB Compass interface with the 'studentList' collection selected. The aggregation pipeline is defined in the 'Aggregation' tab:

```
[{"$match": {"marks": {"$gt": 70}}}, {"$group": {"_id": "$city", "count": {"$sum": 1}, "avgMarks": {"$avg": "$marks"}, "minMarks": {"$min": "$marks"}, "maxMarks": {"$max": "$marks"}}}
```

The results pane displays the output of the aggregation pipeline:

City	Count	Avg Marks	Min Marks	Max Marks
Kolkata	2	66.5	55	78
Delhi	2	71.5	65	88
Mumbai	1	82.0	82	82
Chennai	1	70.0	70	70

```
db.studentList.aggregate([{$match: {}}, {$group: {_id: "$department", name: "$name", marks: {"$sum": 1}}, $sort: {"marks": -1}}])
```

Q1. Display all student records using aggregation.

```
db.studentList.aggregate([{$match: {}}])
```

```
db.studentList.aggregate([{$match: {}}, {$group: {_id: "$department", name: "$name", marks: {"$sum": 1}}, $sort: {"marks": -1}}])
```

Q2. Display students belonging to CSE department.

```
db.studentList.aggregate([{$match: {department: 'CSE'}}])
```

The screenshot shows the MongoDB Compass interface with the following details:

- Project:** CollegeDB
- Database:** studentList
- Collection:** studentList
- Operations:** Aggregations
- Aggregation Pipeline:**
 - `$_id: { $project: { _id: 0 } }`
 - `$_match: { $and: [{ semester: 4 }] }`
 - `$_sort: { name: 1 }`
 - `$_limit: 10`
- Results:** Displays 10 documents from the collection.

Q3. Display students of semester 4.

```
db.studentList.aggregate([{ $match: { semester: 4 } }])
```

The screenshot shows the MongoDB Compass interface with the following details:

- Project:** CollegeDB
- Database:** studentList
- Collection:** studentList
- Operations:** Aggregations
- Aggregation Pipeline:**
 - `$_id: { $project: { _id: 0 } }`
 - `$_project: { name: 1, marks: 1 }`
 - `$_sort: { name: 1 }`
 - `$_limit: 10`
- Results:** Displays 10 documents from the collection.

Q4. Display only name and marks of all students.

```
db.studentList.aggregate([{ $project: { _id: 0, name: 1, marks: 1 } }])
```

The screenshot shows the MongoDB Compass interface with an aggregation pipeline. The pipeline consists of one stage:

```
db.studentList.aggregate([{"$group": {"_id": null, "averageMarks": {"$avg": "$marks" }}}])
```

The results pane displays the output of the aggregation:

_id	averageMarks
null	77.5

Q5. Find the average marks of all students.

```
db.studentList.aggregate([{"$group": {"_id": null, "averageMarks": {"$avg": "$marks" }}}])
```

The screenshot shows the MongoDB Compass interface with an aggregation pipeline. The pipeline consists of one stage:

```
db.studentList.aggregate([{"$group": {"_id": "$department", "totalMarks": {"$sum": "$marks" }}}])
```

The results pane displays the output of the aggregation:

_id	totalMarks
IT	100
CSE	100
ECE	100
MCA	100

Q6. Find total marks scored department-wise.

```
db.studentList.aggregate([{"$group": {"_id": "$department", "totalMarks": {"$sum": "$marks" }}}])
```

The screenshot shows the MongoDB Compass interface with the 'Aggregation' tab selected. The pipeline consists of two stages:

```
[{"$group": {"_id": "$department", "averageMarks": {"$avg": "$marks"}}}
```

Q7. Find average marks department-wise.

```
db.studentList.aggregate([{"$group": {"_id": "$department", "averageMarks": {"$avg": "$marks"}}}])
```

The screenshot shows the MongoDB Compass interface with the 'Aggregation' tab selected. The pipeline consists of two stages:

```
[{"$group": {"_id": "$department", "maxMarks": {"$max": "$marks"}}}]
```

Q8. Find maximum marks scored in each department.

```
db.studentList.aggregate([{"$group": {"_id": "$department", "maxMarks": {"$max": "$marks"}}}])
```

The screenshot shows the MongoDB Compass interface with the 'Aggregation' tab selected. The pipeline consists of three stages:

```
[{"$group": {"_id": "$department", "count": {"$sum": 1}}}]
```

Q9. Count number of students in each department.

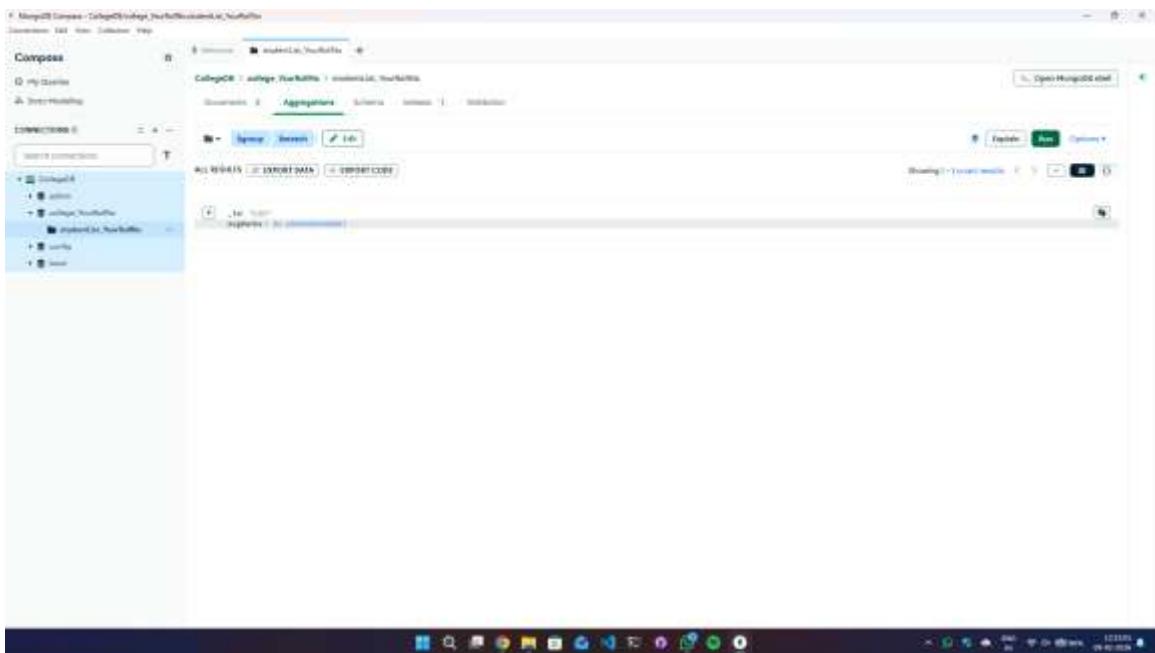
```
db.studentList.aggregate([{"$group": {"_id": "$department", "count": {"$sum": 1}}}])
```

The screenshot shows the MongoDB Compass interface with the 'Aggregation' tab selected. The pipeline consists of three stages:

```
[{"$group": {"_id": "$department", "averageMarks": {"$avg": "$marks"}}, {"$match": {"averageMarks": {"$gt": 70}}}]
```

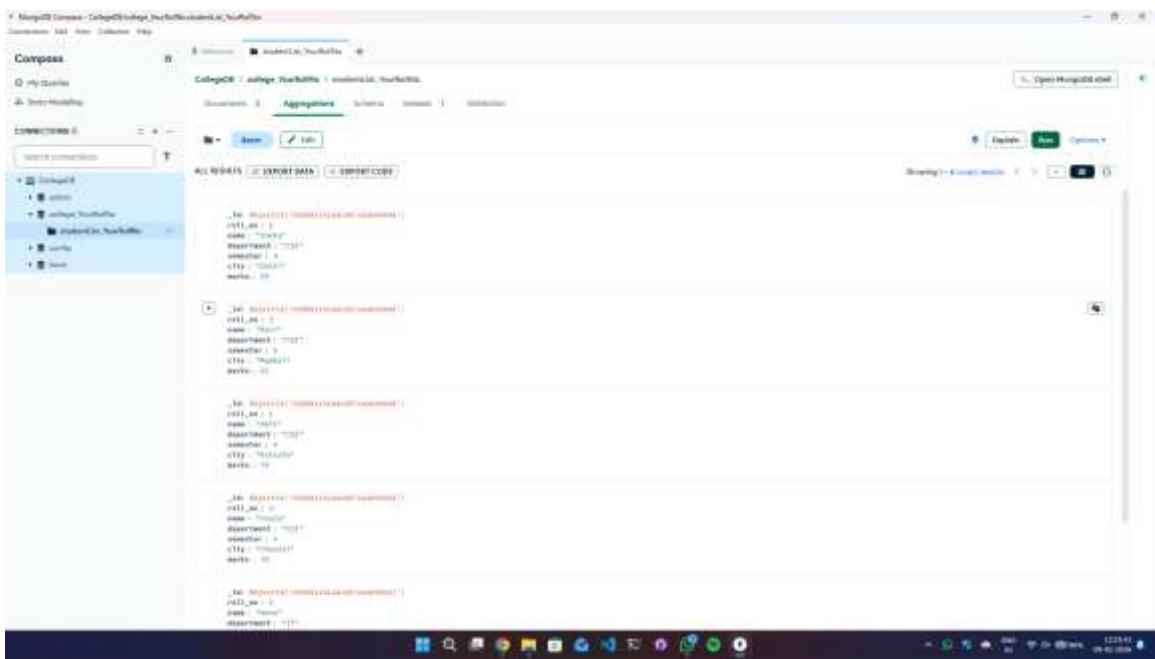
Q10. Display departments having average marks greater than 70.

```
db.studentList.aggregate([{"$group": {"_id": "$department", "averageMarks": {"$avg": "$marks"}}, {"$match": {"averageMarks": {"$gt": 70}}}}])
```



Q11. Sort students by marks in descending order.

```
db.studentList.aggregate([{$sort: { marks: -1 }}])
```



Q12. Display top 3 students based on marks.

```
db.studentList.aggregate([{$sort: { marks: -1 }}, {$limit: 3 }])
```

The screenshot shows the MongoDB Compass interface with the 'Aggregation' tab selected. The pipeline consists of two stages:

```
{$group: {_id: '$department', averageMarks: { $avg: '$marks' }}}, {$sort: { averageMarks: -1 }}
```

Q13. Display department-wise average marks sorted in descending order.

```
db.studentList.aggregate([{$group: {_id: '$department', averageMarks: { $avg: '$marks' }}}, {$sort: { averageMarks: -1 }}])
```

The screenshot shows the MongoDB Compass interface with the 'Aggregation' tab selected. The pipeline consists of one stage:

```
[$match: { city: { $in: ['Delhi', 'Kolkata'] } }]
```

Q14. Display students belonging to Delhi or Kolkata.

```
db.studentList.aggregate([{$match: { city: { $in: ['Delhi', 'Kolkata'] } }}])
```

