Lab-12

→ PART-A :

1. Display distinct city.

→ db.Student.distinct("CITY");

2. Display city wise count of number of students.

→ db.Student.aggregate([{ $group: { _id: "$CITY", count: { $sum: 1 } }}]);

3. Display sum of fees in your collection.

→ db.Student.aggregate([{ $group : { _id : null, totalFees : { $sum: "$FEES" }}}]);

4. Display average of fees in your document.

→ db.Student.aggregate([{ $group : { _id : null, avgFees : { $avg : "$FEES" }}}]);

5. Display maximum and minimum fees of your document.

```
→ db.Student.aggregate([
        { $group : { _id : null, maxFees : { $max : "$FEES" }, minFees : { $min : "$FEES" }}}]);
```

6. Display city wise total fees in your collection.

→ db.Student.aggregate([{ $group : { _id : "$CITY", totalFees : { $sum : "$FEES" }}}]);

7. Display gender wise maximum fees in your collection.

→ db.Student.aggregate([{ $group : { _id : "$GENDER", maxFees : { $max : "$FEES" }}}]);

8. Display city wise maximum and minimum fees.

```
→ db.Student.aggregate([
        { $group : { _id : "$CITY", maxFees : { $max : "$FEES" }, minFees : { $min : "$FEES" }}}]);
```

9. Display count of persons lives in Baroda city in your collection.

→ db.Student.countDocuments({ CITY: "Baroda" });

10. Display average fees of Rajkot city.

```
→ db.Student.aggregate([
        { $match : { CITY : "Baroda" } },
        { $group : { _id : "$CITY", count : { $sum : 1 }}}
  ]);
```

11. Count the number of male and female students in each Department

```
→ db.Student.aggregate([
 { $group: { _id: { DEPARTMENT: "$DEPARTMENT", GENDER: "$GENDER" }, count: { $sum: 1 }}}]);
```

12. Find the total Fees collected from each Department.

→ db.Student.aggregate([{ $group : { _id : "$DEPARTMENT", totalFees : { $sum : "$FEES" }}}]);

13. Find the minimum Fees paid by male and female students in each City.

```
→ db.Student.aggregate([
        { $group : { _id : { CITY : "$CITY", GENDER : "$GENDER" }, minFees : { $min : "$FEES" }}}
  ]);
```

14. Sort students by Fees in descending order and return the top 5.

→ db.Student.aggregate([{ $sort : { FEES : -1 }}, { $limit : 5 }]);

15. Group students by City and calculate the average Fees for each city, only including cities with more than 1 student.

```
→ db.Student.aggregate([
        { $group : { _id : "$CITY", avgFees : { $avg : "$FEES" }, count : { $sum : 1 }}},
        { $match : { count : { $gt : 1 }}},
        { $project : { _id : 1, avgFees : 1 }}
  ]);
```

16. Filter students from CE or Mechanical department, then calculate the total Fees.

→
```
db.Student.aggregate([
        { $match : { DEPARTMENT : { $in : ["CE", "Mechanical"] } } },
        { $group : { _id : null, totalFees : { $sum : "$FEES" } } }
  ]);
```

17. Count the number of male and female students in each Department.

→
```
db.Student.aggregate([
 { $group: { _id: { DEPARTMENT: "$DEPARTMENT", GENDER: "$GENDER" }, count: { $sum: 1 } } }
]);
```

18. Filter students from Rajkot, then group by Department and find the average Fees for each department.

→
```
db.Student.aggregate([
        { $match : { CITY : "Rajkot" } },
        { $group : { _id : "$DEPARTMENT", avgFees : { $avg : "$FEES" } } }
  ]);
```

19. Group by Sem and calculate both the total and average Fees, then sort by total fees in descending order.

→
```
db.Student.aggregate([
        { $group : { _id : "$SEM", totalFees : { $sum : "$FEES" }, avgFees : { $avg : "$FEES" } } },
        { $sort : { totalFees : -1 } }
  ]);
```

20. Find the top 3 cities with the highest total Fees collected by summing up all students' fees in those cities.

→
```
db.Student.aggregate([
        { $group : { _id : "$CITY", totalFees : { $sum : "$FEES" } } },
        { $sort : { totalFees : -1 } },
        { $limit : 3 }
  ]);
```