Lab-13 : Aggregate Method

Aggregation is used to analyze and summarize data instead of simply displaying records.

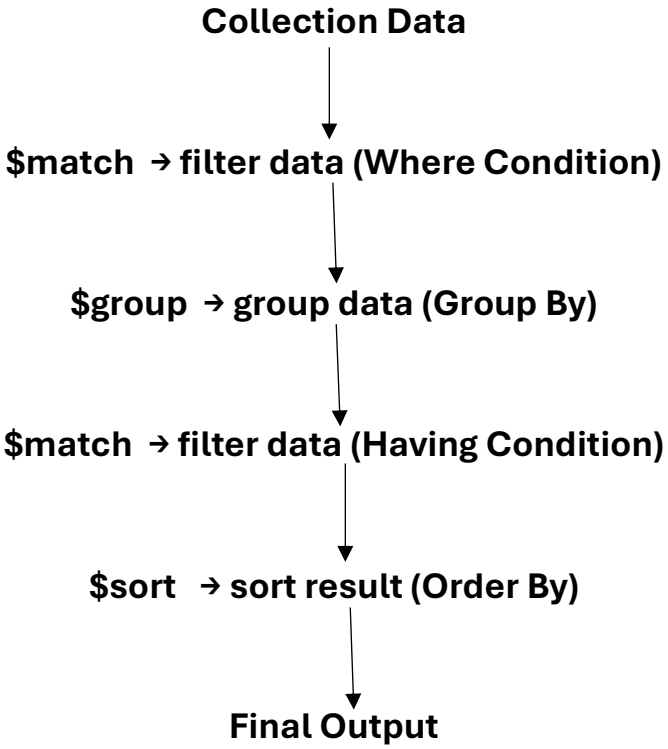
Aggregate Syntax:

db.<collection_name>.aggregate(pipeline , options)

Aggregation Pipeline (array of aggregation stages)

db.<collection_name>.aggregate([{ <stage1> }, { <stage2> }, ...])	db.<collection_name>.aggregate ([{ $match: {...} } { $group: {...} } { $match: {...} } { $sort: {...} } {$limit:{...}}])

Pipeline Flow:



Aggregation pipeline processes data stage by stage, and output of one stage becomes input of next stage.

Examples:

1) Count Total Students

db.Student.find().count() → **without aggregation**

OR (Using Aggregate)

```
db.Student.aggregate([
        { $group: { _id: null, total: { $sum: 1 } } }
])
```

SELECT COUNT(*) FROM Student

Here, **_id** defines grouping field.

_id: null means **no grouping**, so all documents become **one group**.

_id: "$field" → group by field	_id: null → group everything into one group.

---

2) City wise count of number of students

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

"$CITY" → field name

---

3) Distinct city and includes only city field

```
db.Student.aggregate( [
   { $group : { _id : "$CITY" } },
   { $project : { _id : 0 , City : "$_id"}}
] )
```

4) City wise maximum and minimum fees

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

---

5) Count of persons lives in Baroda city

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

OR

```
db.Student.aggregate([
    { $match: { CITY: "Baroda" } },
    { $count:  "Persons"}
])
```

---

6) Count the number of male and female students in each Department. (group by Department & Gender both)

```
db.Student.aggregate(
[
  { $group:
      {
      _id: { Department: "$DEPARTMENT", Gender: "$GENDER" },
       Count: { $sum: 1 }
      }
   }
])
```

7) **Group students by City and calculate the average Fees for each city, only including cities with avg fees more than 12000.**

```
db.Student.aggregate ( [
  { $group: {_id: "$CITY" , avgFees : { $avg : "$FEES" } } },
  { $match: { avgFees : { $gt  :12000 } } }
] )
```

---

8) **Top 3 cities with the highest total Fees collected by summing up all students' fees in those cities.**

(Which top 3 cities collected the most total fees from students?)

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