### 1. Find All Employees

javascript

Copy code

db.employees.find({})

### 2. Find Employees in the IT Department

javascript

Copy code

db.employees.find({ department: "IT" })

### 3. Find Employees in the Finance Department with Salary Greater than 85000

javascript

Copy code

db.employees.find({ department: "Finance", salary: { $gt: 85000 } })

### 4. Count the Number of Employees in Each Department

javascript

Copy code

db.employees.aggregate([

{ $group: { \_id: "$department", employeeCount: { $sum: 1 } } }

])

### 5. Calculate the Average Salary in Each Department

javascript

Copy code

db.employees.aggregate([

{ $group: { \_id: "$department", averageSalary: { $avg: "$salary" } } }

])

### 6. Find Employees Hired After a Certain Date

For example, if the certain date is "2020-01-01":

javascript

Copy code

db.employees.find({ hire\_date: { $gt: ISODate("2020-01-01") } })

### 7. Update the Salary of All Employees in the IT Department by Adding 50000

javascript

Copy code

db.employees.updateMany(

{ department: "IT" },

{ $inc: { salary: 50000 } }

)

### 8. Delete an Employee Record by employee\_id = 6

javascript

Copy code

db.employees.deleteOne({ employee\_id: 6 })

### 9. Find the Highest Salary in Each Department

javascript

Copy code

db.employees.aggregate([

{ $group: { \_id: "$department", maxSalary: { $max: "$salary" } } }

])

### 10. Count the Number of Employees in Each Department with More Than 1 Employee

javascript

Copy code

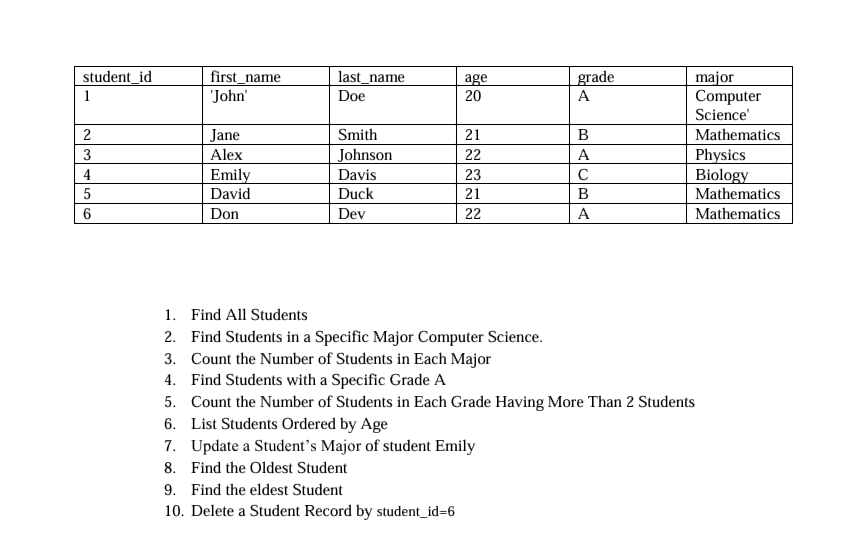
db.employees.aggregate([

{ $group: { \_id: "$department", employeeCount: { $sum: 1 } } },

{ $match: { employeeCount: { $gt: 1 } } }

])

**13. Mongodb**

****

### 1. Find All Students

javascript

Copy code

db.students.find({})

### 2. Find Students in a Specific Major (e.g., "Computer Science")

javascript

Copy code

db.students.find({ major: "Computer Science" })

### 3. Count the Number of Students in Each Major

javascript

Copy code

db.students.aggregate([

{ $group: { \_id: "$major", count: { $sum: 1 } } }

])

### 4. Find Students with a Specific Grade (e.g., "A")

javascript

Copy code

db.students.find({ grade: "A" })

### 5. Count the Number of Students in Each Grade Having More Than 2 Students

javascript

Copy code

db.students.aggregate([

{ $group: { \_id: "$grade", count: { $sum: 1 } } },

{ $match: { count: { $gt: 2 } } }

])

### 6. List Students Ordered by Age

javascript

Copy code

db.students.find({}).sort({ age: 1 }) // Ascending order by age

To list in descending order:

javascript

Copy code

db.students.find({}).sort({ age: -1 }) // Descending order by age

### 7. Update a Student’s Major (e.g., Change Major of Student with first\_name: "Emily" to "Physics")

javascript

Copy code

db.students.updateOne(

{ first\_name: "Emily" },

{ $set: { major: "Physics" } }

)

### 8. Find the Oldest Student

javascript

Copy code

db.students.find().sort({ age: -1 }).limit(1)

### 9. Find the Eldest Student

The "eldest" typically means the oldest, so this would be the same as the previous query:

javascript

Copy code

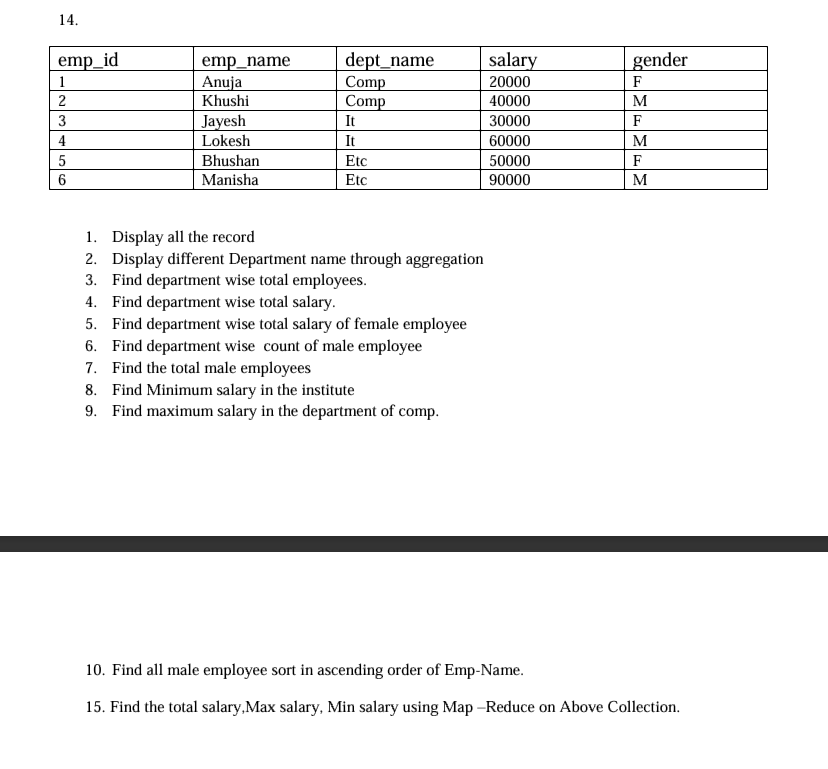
db.students.find().sort({ age: -1 }).limit(1)

### 10. Delete a Student Record by student\_id=6

javascript

Copy code

db.students.deleteOne({ student\_id: 6 })

****

### 1. Display all records

javascript

Copy code

db.employees.find({})

### 2. Display different department names through aggregation

javascript

Copy code

db.employees.distinct("dept\_name")

### 3. Find department-wise total employees

javascript

Copy code

db.employees.aggregate([

{ $group: { \_id: "$dept\_name", totalEmployees: { $sum: 1 } } }

])

### 4. Find department-wise total salary

javascript

Copy code

db.employees.aggregate([

{ $group: { \_id: "$dept\_name", totalSalary: { $sum: "$salary" } } }

])

### 5. Find department-wise total salary of female employees

javascript

Copy code

db.employees.aggregate([

{ $match: { gender: "F" } },

{ $group: { \_id: "$dept\_name", totalFemaleSalary: { $sum: "$salary" } } }

])

### 6. Find department-wise count of male employees

javascript

Copy code

db.employees.aggregate([

{ $match: { gender: "M" } },

{ $group: { \_id: "$dept\_name", maleCount: { $sum: 1 } } }

])

### 7. Find the total male employees

javascript

Copy code

db.employees.countDocuments({ gender: "M" })

### 8. Find minimum salary in the institute

javascript

Copy code

db.employees.aggregate([

{ $group: { \_id: null, minSalary: { $min: "$salary" } } }

])

### 9. Find maximum salary in the department of "Comp"

javascript

Copy code

db.employees.aggregate([

{ $match: { dept\_name: "Comp" } },

{ $group: { \_id: null, maxSalary: { $max: "$salary" } } }

])

### 10. Find all male employees sorted in ascending order of emp\_name

javascript

Copy code

db.employees.find({ gender: "M" }).sort({ emp\_name: 1 })

### 15. Find the total salary, max salary, and min salary using Map-Reduce

javascript

Copy code

db.employees.mapReduce(

function() { emit("salaryStats", { total: this.salary, max: this.salary, min: this.salary }); },

function(key, values) {

return {

total: Array.sum(values.map(v => v.total)),

max: Math.max.apply(Math, values.map(v => v.max)),

min: Math.min.apply(Math, values.map(v => v.min))

};

},

{ out: "salary\_stats" }

)

After running the map-reduce, the results can be viewed in the salary\_stats collection:

javascript

Copy code

db.salary\_stats.find()