MongoDB Practical - CRUD, Logical Operators & Advanced Commands

```
use studentdb

db.students.insertMany([{ roll_no: 1, name: "Sahil", course: "MBA", marks: 85, city: "Pune" }, { roll_no: 2, name: "Riya", course: "MCA", marks: 75, city: "Mumbai" }, { roll_no: 3, name: "Amit", course: "MBA", marks: 65, city: "Nashik" }, { roll_no: 4, name: "Sneha", course: "MBA", marks: 90, city: "Pune" }])

db.students.find()

db.students.insert({ roll_no: 5, name: "Rohan", course: "MCA", marks: 70, city: "pune" })

db.students.updateOne({ name: "Amit" }, { $set: { marks: 75, city: "Pune" } })

db.students.deleteOne({ roll_no: 2 })

db.students.find().pretty()

db.students.find({ $and: [ { course: "MBA" }, { marks: { $gt: 70 } } ] })

db.students.find({ $or: [ { city: "Pune" }, { marks: { $lt: 70 } } ] })

db.students.find({ marks: { $lt: 75 } })

db.students.find({ marks: { $gt: 75 } })

db.students.find().limit(3)

db.students.find().sort({ marks: 1 })

db.students.find({ city: { $ne: "Pune" } })  // Not equal condition

db.students.countDocuments({ course: "MBA" })  // Count number of MBA students

db.students.distinct("city")  // Display unique cities

db.students.find({ marks: { $gte: 70, $lte: 85 } })  // Between 70 and 85 marks

db.students.updateMany({ course: "MBA" }, { $inc: { marks: 5 } })  // Increase marks by 5 for all MBA students

db.students.find({}, { name: 1, marks: 1, _id: 0 })  // Projection: show only name and marks

db.students.aggregate([{ $group: { _id: "$course", avgMarks: { $avg: "$marks" }
```

```
} }])  // Group by course and find average marks
```