✅ MongoDB – Complete Guide for Developers & DevOps

📌 What is MongoDB?

MongoDB is a **NoSQL, document-based database**.
Instead of tables, it stores data in flexible **JSON-like** documents (BSON format).

Example:

json

CopyEdit

{

"name": "John",

"age": 25,

"skills": ["Java", "Spring"]

}

🚀 Why Use MongoDB?

Advantage	Description
🗂️ Schema-less	No fixed table structure – flexible documents
⚡ High Performance	Fast reads/writes for unstructured data
🔁 Easy Scaling	Horizontal scalability with built-in sharding
👨‍💻 Developer Friendly	JSON-style documents, simple queries

🧱 MongoDB Architecture

css

CopyEdit

[Client] ⇄ [MongoDB Server] ⇄ [Database] ⇄ [Collections] ⇄ [Documents]

- **Database**: Logical container (like schema in MySQL)
- **Collection**: Like table (but schema-less)

- **Document**: A single JSON-like object (record)

🔧 Basic Commands

bash

CopyEdit

Start shell

mongo

Create or switch database

use studentdb

Create/Insert document

db.students.insertOne({ name: "Viraj", age: 23 });

Read documents

db.students.find();

Update document

db.students.updateOne({ name: "Viraj" }, { $set: { age: 24 } });

Delete document

db.students.deleteOne({ name: "Viraj" });

📦 Spring Boot + MongoDB Integration

1. Add Dependencies (Maven)

xml

CopyEdit

<dependency>

```
 <groupId>org.springframework.boot</groupId>
 <artifactId>spring-boot-starter-data-mongodb</artifactId>
</dependency>
```

---

2. application.properties

properties

CopyEdit

```
spring.data.mongodb.uri=mongodb://localhost:27017/studentdb
```

---

3. Create Model

java

CopyEdit

```
@Document(collection = "students")
public class Student {
 @Id
 private String id;
 private String name;
 private int age;
}
```

---

4. Create Repository

java

CopyEdit

```
public interface StudentRepository extends MongoRepository<Student, String> {
   List<Student> findByName(String name);
}
```

✅ **Advantages vs SQL**

Feature	MongoDB	MySQL
Structure	Document-based	Table-based
Schema	Flexible (schema-less)	Fixed schema
Best Use	Fast dev, semi-structured data	Structured, transactional
Scaling	Horizontal	Vertical

🐬 MongoDB with Docker

bash

CopyEdit

```
docker run -d -p 27017:27017 --name mongo \
 -e MONGO_INITDB_DATABASE=studentdb \
 mongo
```

Optional: Mongo Express (UI)

yaml

CopyEdit

```
mongo-express:
 image: mongo-express
 restart: always
 ports:
  - 8081:8081
 environment:
  ME_CONFIG_MONGODB_SERVER: mongo
```

Access: http://localhost:8081

💬 Interview Questions (MongoDB)

Q1. What is MongoDB?
A NoSQL document-oriented database.

Q2. What is the difference between MongoDB and SQL DBs?
MongoDB uses flexible JSON-like documents; SQL uses structured tables.

Q3. How is data stored in MongoDB?
In BSON format (Binary JSON).

Q4. What is a collection?
Like a table, but without schema restrictions.

Q5. How does MongoDB scale?
Supports horizontal scaling using sharding.

Q6. How do you connect MongoDB with Spring Boot?
Using spring-boot-starter-data-mongodb and @Document, MongoRepository.

✅ **Summary**

Must Know Area	Covered
MongoDB Basics	✅
CRUD Operations	✅
Spring Boot Integration	✅
Docker Setup	✅
Interview Questions	✅

🧪 MongoDB Practice Tasks (With Sample Queries)

🔹 Task 1: Create a Database and Collection

Objective: Set up a studentdb database and a students collection.

bash

CopyEdit

use studentdb

db.createCollection("students")

🔹 Task 2: Insert Multiple Documents

Objective: Insert sample student records.

js

CopyEdit

db.students.insertMany([

 { name: "Viraj", age: 22, dept: "CS", grade: "A" },

 { name: "Arya", age: 21, dept: "IT", grade: "B" },

 { name: "Kiran", age: 23, dept: "CS", grade: "C" }

])

🔹 Task 3: Read All Documents

js

CopyEdit

db.students.find()

🔹 Task 4: Filter Documents

Objective: Find all CS department students.

js

CopyEdit

db.students.find({ dept: "CS" })

🔹 Task 5: Projection (Only Select Fields)

Objective: Show only name and grade.

js

CopyEdit

db.students.find({}, { name: 1, grade: 1, _id: 0 })

🔹 Task 6: Update a Field

Objective: Update grade of a specific student.

js

CopyEdit

db.students.updateOne({ name: "Arya" }, { $set: { grade: "A+" } })

🔹 Task 7: Delete a Document

Objective: Delete student with grade C.

js

CopyEdit

db.students.deleteOne({ grade: "C" })

🔹 Task 8: Count Records

Objective: How many students are in CS?

js

CopyEdit

db.students.countDocuments({ dept: "CS" })

🔹 **Task 9: Sort Results**

Objective: Sort students by age (descending).

js

CopyEdit

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

🔹 **Task 10: Create Index**

Objective: Index on name field for faster search.

js

CopyEdit

```
db.students.createIndex({ name: 1 })
```

🔹 **Task 11: Aggregation Example**

Objective: Group by department and count students.

js

CopyEdit

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

🔹 **Task 12: Check if a Field Exists**

Objective: Find documents where grade exists.

js

CopyEdit

```
db.students.find({ grade: { $exists: true } })
```

🔹 **Task 13: Array Query**

If student has multiple skills:

js

CopyEdit

db.students.insertOne({ name: "Dev", skills: ["Java", "MongoDB", "Spring"] })

// Find all with MongoDB skill

db.students.find({ skills: "MongoDB" })

✅ **Tip for Interviews**

✅ Practice these queries using **MongoDB Compass UI** or mongo shell
✅ Explain **why** you use $set, $group, $exists, etc.
✅ Know difference between .find() and .aggregate()