

day 6 Assignment

Name – Soham Sarkar

College Name – MCKV INSTITUTE OF ENGINEERING



June 19, 2025

**1.What is MongoDB? How is it different from a traditional RDBMS like MySQL?** *(Explain in terms of schema, data format, scalability, and use cases.)*

**MongoDB** is a **NoSQL** database that stores data in a **document-oriented** format (using **BSON**, a binary form of JSON).

| **Feature** | **MongoDB (NoSQL)** | **MySQL (RDBMS)** |
| --- | --- | --- |
| **Schema** | Schema-less (flexible document structure) | Strict schema (tables with defined columns) |
| **Data Format** | JSON-like documents (BSON) | Tables with rows and columns |
| **Scalability** | Horizontally scalable (via sharding) | Vertically scalable |
| **Use Cases** | Real-time analytics, IoT, fast prototyping | Banking, ERP, applications with structured data |

**Differences with RDBMS like MySQL:**

**2.What are collections and documents in MongoDB? How do they relate to tables and rows in SQL databases?**

| SQL Term | MongoDB Equivalent |
| --- | --- |
| Database | Database |
| Table | Collection |
| Row | Document |
| Column | Field (key-value in document) |

* Collection: A group of documents. Like a table in SQL.
* Document: A single record stored in a JSON-like format. Like a row in SQL.

Example:

{

"\_id": 1,

"name": "Alice",

"age": 25,

"dept": "CSE"

}

This document belongs to a collection (e.g., students), similar to a row in a students table.

**3.What is the purpose of \_id in MongoDB documents? Can we customize it? Explain with an example.**

* \_id is the primary key in every MongoDB document.
* It is automatically generated as a unique 12-byte ObjectId if not provided.
* Yes, we can customize it using any unique value (e.g., string, number).

Example:

db.students.insertOne({

\_id: "student\_101",

name: "Bob",

age: 20

});

If you insert another document with the same \_id, you'll get a duplicate key error.

**4.What is the difference between insertOne() and insertMany() methods? Write syntax for both.**

| **Method** | **Description** |
| --- | --- |
| insertOne() | Inserts **a single document** |
| insertMany() | Inserts **multiple documents** at once |

**insertOne() Syntax:**

db.students.insertOne({

name: "Alice",

age: 22,

dept: "CSE"

});

**insertMany() Syntax:**

db.students.insertMany([

{ name: "Bob", age: 21, dept: "ECE" },

{ name: "Charlie", age: 23, dept: "CIVIL" }

]);

**5.Explain the difference between find() and findOne() with examples.**

| Method | Purpose |
| --- | --- |
| find() | Returns a cursor to multiple documents |
| findOne() | Returns the first matching document |

find() Example:

db.students.find({ dept: "CSE" });

* Returns all documents in students where dept is "CSE".

findOne() Example:

db.students.findOne({ dept: "CSE" });

* Returns only the first document it finds with dept as "CSE".

**Part B: Practical / Query-Based Questions**

**For Creaing Collecions (Student,Users,Producs)**

**Student Collection**

db.createCollection("students")

db.students.insertMany([

{

student\_id: 101,

name: "Aditi Sharma",

marks: 92,

city: "Mumbai"

},

{

student\_id: 102,

name: "Rohit Verma",

marks: 78,

city: "Delhi"

}

]);

**Product Collection**

db.products.insertMany([

{

product\_id: 201,

name: "Wireless Mouse",

price: 499,

category: "Electronics"

},

{

product\_id: 202,

name: "Notebook",

price: 80,

category: "Stationery"

},

{

product\_id: 203,

name: "Bluetooth Speaker",

price: 1500,

category: "Electronics"

}

]);

**6.Write a MongoDB query to find all documents in the students collection where the marks is greater than 80.**

db.students.find({ marks: { $gt: 80 } });

**7.Write a query to update the name of a student whose student\_id is 102.**

db.students.updateOne(

{ student\_id: 102 },

{ $set: { name: "Updated Name" } }

);

**8.Write a query to delete all records from the products collection where the price is less than 100.**

db.products.deleteMany({ price: { $lt: 100 } });

**9.Create a new document in the users collection with fields: name, email, age, and city.**db.users.insertOne({

name: "Soham Sarkar",

email: "soham@example.com",

age: 22,

city: "Kolkata"

});

**10.Write a MongoDB aggregation query to count how many students are from each city in the students collection.**

db.students.aggregate([

{

$group: {

\_id: "$city",

totalStudents: { $sum: 1 }

}

}

]);