
⭐ ⭐ ⭐ MONGODB PRACTICAL – CRUD OPERATIONS (FINAL PERFECT ANSWER) ⭐ ⭐ ⭐

🎯 AIM

To design and develop MongoDB queries demonstrating CRUD operations using insertOne(), insertMany(), find(), updateOne(), updateMany(), deleteOne(), deleteMany(), and various logical/comparison operators such as $gt, $lt, $in, $nin, $gte, $lte.

📚 THEORY

✓ What is MongoDB?

MongoDB is a NoSQL, document-oriented database that stores data in BSON (Binary JSON) documents instead of rows and columns.

✓ Why MongoDB?

- **Schema-less**
- **Fast & flexible**
- **Handles big data**
- **JSON-like structure is developer-friendly**

✓ MongoDB Basic Components

SQL Term	MongoDB Term	Meaning
Database	Database	Container of collections
Table	Collection	Container of documents
Row	Document	JSON object
Column	Field	Key-value pair
Primary Key	_id	Auto-generated unique value

✓ CRUD Operations

- **Create → insertOne(), insertMany(), save()**

- **Read → find(), findOne()**
- **Update → updateOne(), updateMany(), $set, $inc**
- **Delete → deleteOne(), deleteMany()**

🟩 FULL CLEAN CODE (WRITE IN NOTEBOOK)

(NO explanation here — pure clean professional code)

```
show dbs;
use studentdb;
db.createCollection("student");
show collections;

db.student.insertOne({name:"Shital", age:19, city:"Sinnar"});

db.student.insertMany([
 {name:"Alok", age:20, city:"Nashik"},
 {name:"Sanju", age:18, city:"Nagar"},
 {name:"Anjali", age:19, city:"Niphad"},
 {name:"Sakshi", age:19, city:"Nashik"},
 {name:"Harshada", age:20, city:"Nashik"}
]);

db.student.find();
db.student.find().limit(3);

db.student.find({age:{$gt:19}}, {name:1, age:1});
db.student.findOne({name:"Sanju"});

db.student.updateOne(
```

```
{name:"Anjali"},
{$set:{email:"anjali@gmail.com"}}
);

db.student.updateMany(
{age:{$gt:19}},
{$set:{city:"Pune"}}
);

db.student.deleteOne({age:19});
db.student.deleteMany({age:20});

db.student.find({age:{$lte:18}});
db.student.find({age:{$gte:19}});
db.student.find({age:{$in:[18,19,20]}});
db.student.find({age:{$nin:[19,20]}});

db.student.find().pretty();
db.student.count();
db.student.distinct("city");
db.student.find().sort({name:1});
db.student.drop();
db.dropDatabase();
```

🟦 **CODE WITH PROPER STEP-BY-STEP EXPLANATION**

(This is what you SAY to external examiner)

🔵 1️⃣ **show dbs**

✓ Code

```
show dbs;
```

✓ Explanation

1. Lists all databases stored in MongoDB server.
2. Helps confirm whether database already exists.
3. Empty databases are not shown.

🔵 2️⃣ use studentdb

✓ Code

```
use studentdb;
```

✓ Explanation

1. Switches to database studentdb.
2. If database doesn't exist → it is created automatically.
3. Equivalent to SQL: CREATE DATABASE + USE database.

🔵 3️⃣ Create Collection

✓ Code

```
db.createCollection("student");
```

✓ Explanation

1. Creates a new collection named "student".
2. Collection = SQL Table but schema-less.
3. Stores JSON-like documents.

🔵 4️⃣ INSERT OPERATIONS (CREATE)

🔹 A) insertOne()

✓ Code

```
db.student.insertOne({name:"Shital", age:19, city:"Sinnar"});
```

✓ Explanation

1. Inserts one document.
2. MongoDB auto-generates _id.
3. Used for single record insertion.

🔹 B) insertMany()

✓ Code

```
db.student.insertMany([
{name:"Alok", age:20, city:"Nashik"},
{name:"Sanju", age:18, city:"Nagar"},
{name:"Anjali", age:19, city:"Niphad"},
{name:"Sakshi", age:19, city:"Nashik"},
{name:"Harshada", age:20, city:"Nashik"}
]);
```

✓ Explanation

1. Inserts multiple documents at once.
2. Efficient for bulk insertion.
3. Each document gets own _id.

🔵 5️⃣ READ OPERATIONS (READ)

🔹 A) Display all documents

✓ Code

```
db.student.find();
```

✓ Explanation

1. Returns all documents from the collection.
2. Equivalent to SQL: SELECT * FROM student.

🔹 B) Limit the output

✓ Code

```
db.student.find().limit(3);
```

✓ Explanation

1. Shows only first 3 documents.
2. Used for pagination/testing.

🔹 C) Condition + Projection

✓ Code

```
db.student.find({age:{$gt:19}}, {name:1, age:1});
```

✓ Explanation

1. $gt → greater than operator.
2. Retrieves documents where age > 19.
3. Projection shows only name & age fields.

🔹 D) findOne()

✓ Code

```
db.student.findOne({name:"Sanju"});
```

✓ Explanation

1. Returns first matching document.
2. Faster than find() when only one match expected.

🔵 6️⃣ UPDATE OPERATIONS (UPDATE)

🔹 A) updateOne()

✓ Code

```
db.student.updateOne(
 {name:"Anjali"},
```

{$set:{email:"anjali@gmail.com"}}

);

✓ Explanation

1. Finds first document with name = "Anjali".
2. $set adds/updates specific field.
3. Does not modify other fields.

🔹 B) updateMany()

✓ Code

db.student.updateMany(

{age:{$gt:19}},

{$set:{city:"Pune"}}

);

✓ Explanation

1. Updates all records where age > 19.
2. $set modifies only the city field.
3. Used for bulk updates.

🔵 7️⃣ DELETE OPERATIONS (DELETE)

🔹 A) deleteOne()

✓ Code

db.student.deleteOne({age:19});

✓ Explanation

1. Deletes first matching document.
2. Used when duplicates exist.

🔹 B) deleteMany()

✓ Code

```
db.student.deleteMany({age:20});
```

✓ Explanation

1. Deletes all matching documents.
2. Used for mass deletion.

🔵 8️⃣ Comparison Operators

✓ $lte (less than or equal)

```
db.student.find({age:{$lte:18}});
```

✓ $gte (greater than or equal)

```
db.student.find({age:{$gte:19}});
```

✓ $in

```
db.student.find({age:{$in:[18,19,20]}});
```

✓ $nin

```
db.student.find({age:{$nin:[19,20]}});
```

✓ Explanation

- $in → match any value from list
- $nin → exclude values
- $gt, $gte, $lte, $lt → comparison operators

🟧 EXTRA IMPORTANT COMMANDS (VERY IMPORTANT FOR EXAM)

✓ Pretty format

```
db.student.find().pretty();
```

✓ Count documents

```
db.student.count();
```

✓ Show only unique values

```
db.student.distinct("city");
```

✓ Sorting

db.student.find().sort({name:1}); // ascending

db.student.find().sort({age:-1}); // descending

✓ Show collections

show collections;

✓ Drop collection

db.student.drop();

✓ Drop database

db.dropDatabase();

🟩 CONCLUSION

In this practical, CRUD operations were successfully performed on MongoDB using various commands like insertOne(), insertMany(), find(), updateOne(), updateMany(), deleteOne(), deleteMany(), and MongoDB operators such as $gt, $lt, $gte, $lte, $in, and $nin. Additional commands like pretty(), count(), sort(), and distinct() were also demonstrated.

🎤 VIVA QUESTIONS WITH ANSWERS

1. What is MongoDB?

MongoDB is a NoSQL document-oriented database storing data in BSON documents.

2. What is a document?

A JSON-like record inside MongoDB.

3. What is a collection?

A group of documents (similar to a SQL table).

4. What is _id?

A unique auto-generated primary key.

5. What does CRUD stand for?

Create, Read, Update, Delete.

6. Why use insertMany()?

To insert multiple documents at once.

7. What does $set do?

Adds or updates a field.

8. What is the difference between updateOne() and updateMany()?

- **updateOne() updates first match**
- **updateMany() updates all matches**

9. What does $gt mean?

Greater than.

10. What does $in mean?

Matches any value from the given array.

11. How to show all collections?

show collections

12. What is pretty()?

Formats output in readable style.
