



## Worksheet 10

**Student Name:** Japneet Singh

**Branch:** CSE

**Semester:** 5th

**Subject Name:** ADBMS

**UID:** 23BCS10195

**Section/Group:** KRG 2-A

**Date of Performance:** 30/10/2025

**Subject Code:** 23CSP-333

### **1. Aim: Perform CRUD Operations using No-SQL Database: MongoDB**

### **2. Objective:**

- Create and manage databases and collections.
- Perform **CRUD operations** — Create, Read, Update, and Delete documents.
- Use **aggregation** for grouping and summarizing data.
- Understand how to modify, query, and manage collections efficiently in MongoDB.
- Gain practical experience with real-world data handling through command-line operations.

### **3. Code:**

```
// 1. Show all databases
```

```
show dbs
```

```
// 2. Create or switch to your database
```

```
use db_shivanshu
```

```
// 3. Check current database
```

```
db
```

```
// 4. Create a collection (optional)
```

```
db.createCollection("movies")
```

```
// 5. Insert Operations
```

```
db.movies.insertOne({
```

```
  title: "Inception",
```

```
  director: "Christopher Nolan",
```

```
  year: 2010,
```

```
  rating: 8.8,
```



```
genre: "Sci-Fi",  
features: ["Mind-bending", "Action"],  
added_by: "Shivanshu"  
})
```

```
db.movies.insertMany([  
  
  {  
  
    title: "Interstellar",  
    director: "Christopher Nolan",  
    year: 2014,  
    rating: 8.6,  
    genre: "Sci-Fi",  
    features: ["Space", "Drama"],  
    added_by: "Shivanshu"  
  },
```

```
  {  
  
    title: "Avatar",  
    director: "James Cameron",  
    year: 2009,  
    rating: 7.8,  
    genre: "Adventure",  
    features: ["3D", "Action"],  
    added_by: "Shivanshu"  
  },
```

```
  {  
  
    title: "The Dark Knight",
```



```
    director: "Christopher Nolan",
    year: 2008,
    rating: 9.0,
    genre: "Action",
    features: ["Batman", "Crime"],
    added_by: "Shivanshu"
  }
])

// 6. Show all collections

show collections

// 7. Read Operations

db.movies.find().pretty()

db.movies.findOne()

db.movies.find({}, { title: 1, year: 1, _id: 0 })

db.movies.find({ genre: "Action" })

db.movies.find({ "details.language": "English" })

// 8. Update Operations

db.movies.updateOne(
  { title: "Avatar" },
  { $set: { rating: 8.1, genre: "Sci-Fi Adventure" } }
)

db.movies.updateOne(
  { title: "Inception" },
  { $push: { features: "Thriller" } }
)
```

```
db.movies.updateOne(  
  { title: "Inception" },  
  { $pull: { features: "Action" } }  
)
```

```
db.movies.updateMany(  
  { director: "Christopher Nolan" },  
  { $set: { language: "English" } }  
)
```

```
db.movies.updateMany({}, { $unset: { added_by: "" } })
```

```
db.movies.updateMany({}, { $set: { color: "Full HD" } })
```

```
db.movies.updateOne(  
  { title: "Tenet" },  
  { $set: { director: "Christopher Nolan", year: 2020, rating: 7.5 } },  
  { upsert: true }  
)
```

// 9. Delete Operations

```
db.movies.deleteOne({ title: "Avatar" })
```

```
db.movies.deleteMany({ director: "Christopher Nolan" })
```

```
db.movies.deleteMany({})
```

// 10. Grouping (Aggregation)

```
db.movies.aggregate([  
  { $group: { _id: "$director", total_movies: { $sum: 1 } } }  
)
```

```
db.movies.aggregate([
```

```
{ $group: { _id: "$genre", total_movies: { $sum: 1 } } }
```

```
)
```

```
// 11. Drop collection
```

```
db.movies.drop()
```

```
// 12. Drop database
```

```
db.dropDatabase()
```

## 4. Output:

```
[shivanshuranjansingh@Shivanshu-MacBook ~ % mongosh
Current Mongosh Log ID: 690399ad55d128fad2120e1b
Connecting to:      mongodb://127.0.0.1:27017/?directConnection=true&serverSelectionTimeoutMS=2000&appName=mongosh+2.5.8
Using MongoDB:      7.0.25
Using Mongosh:       2.5.8

For mongosh info see: https://www.mongodb.com/docs/mongodb-shell/

-----
The server generated these startup warnings when booting
2025-10-30T22:06:06.549+05:30: Access control is not enabled for the database. Read and write access to data and configuration is unrestricted
-----

test> show dbs
|
| admin          48.00 KiB
| config         108.00 KiB
| db_shivanshu   60.00 KiB
| local          72.00 KiB
| test> use db_shivanshu
|
| switched to db db_shivanshu
db_shivanshu> db.createCollection("movies")
|
| { ok: 1 }
db_shivanshu>
```

```
db_shivanshu> db.movies.insertOne({
  title: "Inception",
  director: "Christopher Nolan",
  year: 2010,
  rating: 8.8,
  genre: "Sci-Fi",
  features: ["Mind-bending", "Action"],
  added_by: "Shivanshu"
})
{
  acknowledged: true,
  insertedId: ObjectId('690399f655d128fad2120e1c')
}
db_shivanshu> db.movies.insertMany([
  {
    title: "Interstellar",
    director: "Christopher Nolan",
    year: 2014,
    rating: 8.6,
    genre: "Sci-Fi",
    features: ["Space", "Drama"],
    added_by: "Shivanshu"
  },
  {
    title: "Avatar",
    director: "James Cameron",
    year: 2009,
    rating: 7.8,
    genre: "Adventure",
    features: ["3D", "Action"],
    added_by: "Shivanshu"
  },
  {
    title: "The Dark Knight",
    director: "Christopher Nolan",
    year: 2008,
    rating: 9.0,
    genre: "Action",
    features: ["Batman", "Crime"],
    added_by: "Shivanshu"
  }
])
{
  acknowledged: true,
  insertedIds: {
    '0': ObjectId('690399fe55d128fad2120e1d'),
    '1': ObjectId('690399fe55d128fad2120e1e'),
    '2': ObjectId('690399fe55d128fad2120e1f')
  }
}
db_shivanshu>
```

```
(db_shivanshu> db.movies.find()
[
  {
    _id: ObjectId('690399f655d128fad2120e1c'),
    title: 'Inception',
    director: 'Christopher Nolan',
    year: 2010,
    rating: 8.8,
    genre: 'Sci-Fi',
    features: [ 'Mind-bending', 'Action' ],
    added_by: 'Shivanshu'
  },
  {
    _id: ObjectId('690399fe55d128fad2120e1d'),
    title: 'Interstellar',
    director: 'Christopher Nolan',
    year: 2014,
    rating: 8.6,
    genre: 'Sci-Fi',
    features: [ 'Space', 'Drama' ],
    added_by: 'Shivanshu'
  },
  {
    _id: ObjectId('690399fe55d128fad2120e1e'),
    title: 'Avatar',
    director: 'James Cameron',
    year: 2009,
    rating: 7.8,
    genre: 'Adventure',
    features: [ '3D', 'Action' ],
    added_by: 'Shivanshu'
  },
  {
    _id: ObjectId('690399fe55d128fad2120e1f'),
    title: 'The Dark Knight',
    director: 'Christopher Nolan',
    year: 2008,
    rating: 9,
    genre: 'Action',
    features: [ 'Batman', 'Crime' ],
    added_by: 'Shivanshu'
  }
]
db_shivanshu> db.movies.findOne()
{
  _id: ObjectId('690399f655d128fad2120e1c'),
  title: 'Inception',
  director: 'Christopher Nolan',
  year: 2010,
  rating: 8.8,
  genre: 'Sci-Fi',
  features: [ 'Mind-bending', 'Action' ],
  added_by: 'Shivanshu'
}
```

```
db_shivanshu> db.movies.find({}, { title: 1, year: 1, _id: 0 })
[
  [
    { title: 'Inception', year: 2010 },
    { title: 'Interstellar', year: 2014 },
    { title: 'Avatar', year: 2009 },
    { title: 'The Dark Knight', year: 2008 }
  ]
]
db_shivanshu> db.movies.find({ genre: "Action" })
[
  [
    {
      _id: ObjectId('690399fe55d128fad2120e1f'),
      title: 'The Dark Knight',
      director: 'Christopher Nolan',
      year: 2008,
      rating: 9,
      genre: 'Action',
      features: [ 'Batman', 'Crime' ],
      added_by: 'Shivanshu'
    }
  ]
]
db_shivanshu> db.movies.find({ "details.language": "English" })
[
  [
  ]
]
db_shivanshu> █
```

```
db_shivanshu> db.movies.updateOne(
  | { title: "Avatar" },
  | { $set: { rating: 8.1, genre: "Sci-Fi Adventure" } }
  | )
[
  [
    {
      acknowledged: true,
      insertedId: null,
      matchedCount: 1,
      modifiedCount: 1,
      upsertedCount: 0
    }
  ]
]
db_shivanshu> db.movies.updateOne(
  | { title: "Inception" },
  | { $push: { features: "Thriller" } }
  | )
[
  [
    {
      acknowledged: true,
      insertedId: null,
      matchedCount: 1,
      modifiedCount: 1,
      upsertedCount: 0
    }
  ]
]
db_shivanshu> db.movies.updateOne(
  | { title: "Inception" },
  | { $pull: { features: "Action" } }
  | )
[
  [
    {
      acknowledged: true,
      insertedId: null,
      matchedCount: 1,
      modifiedCount: 1,
      upsertedCount: 0
    }
  ]
]
db_shivanshu> db.movies.updateMany(
  | { director: "Christopher Nolan" },
  | { $set: { language: "English" } }
  | )
[
  [
    {
      acknowledged: true,
      insertedId: null,
      matchedCount: 3,
      modifiedCount: 3,
      upsertedCount: 0
    }
  ]
]
db_shivanshu> █
```

```
db_shivanshu> db.movies.updateMany({}, { $unset: { added_by: "" } })
[
  [
    {
      acknowledged: true,
      insertedId: null,
      matchedCount: 4,
      modifiedCount: 4,
      upsertedCount: 0
    }
  ]
]
db_shivanshu> db.movies.updateMany({}, { $set: { color: "Full HD" } })
[
  [
    {
      acknowledged: true,
      insertedId: null,
      matchedCount: 4,
      modifiedCount: 4,
      upsertedCount: 0
    }
  ]
]
db_shivanshu> db.movies.updateOne(
  | { title: "Tenet" },
  | { $set: { director: "Christopher Nolan", year: 2020, rating: 7.5 } },
  | { upsert: true }
  | )
[
  [
    {
      acknowledged: true,
      insertedId: ObjectId('69039ac0e681e6eb85a7e326'),
      matchedCount: 0,
      modifiedCount: 0,
      upsertedCount: 1
    }
  ]
]
db_shivanshu> █
```

```
db_shivanshu> db.movies.deleteOne({ title: "Avatar" })
[
  { acknowledged: true, deletedCount: 1 }
]
db_shivanshu> db.movies.deleteMany({ director: "Christopher Nolan" })
[
  { acknowledged: true, deletedCount: 4 }
]
db_shivanshu> db.movies.deleteMany({})
[
  { acknowledged: true, deletedCount: 0 }
]
db_shivanshu>
```

```
db_shivanshu> db.movies.aggregate([
  { $group: { _id: "$director", total_movies: { $sum: 1 } } }
])
[
  { _id: "Christopher Nolan", total_movies: 4 }
]

db_shivanshu> db.movies.aggregate([
  { $group: { _id: "$genre", total_movies: { $sum: 1 } } }
])
[
  { _id: "Action", total_movies: 10 },
  { _id: "Drama", total_movies: 8 },
  { _id: "Comedy", total_movies: 5 },
  { _id: "Thriller", total_movies: 3 },
  { _id: "Science Fiction", total_movies: 2 }
]

db_shivanshu>
```

```
db_shivanshu> db.movies.drop()
true
db_shivanshu> db.dropDatabase()
{ ok: 1, dropped: 'db_shivanshu' }
db_shivanshu>
```

## 5. Learning Outcomes:

- Understand the structure and working of MongoDB as a NoSQL database.
- Create and manage databases and collections using MongoDB commands.
- Perform CRUD (Create, Read, Update, Delete) operations on documents.
- Apply aggregation operations for data grouping and analysis.
- Gain practical skills in handling and modifying data using MongoDB shell.