MongoDB Course: Beginner to Advanced

Introduction to MongoDB

- **What is MongoDB?**
  - Overview of NoSQL databases.
  - Features of MongoDB: Document-oriented, schema-free, scalability.
  - Use cases of MongoDB in real-world applications.
- **Installing MongoDB**
  - Installation steps for Windows, macOS, and Linux.
  - Setting up MongoDB server and Compass.
  - Verifying the installation.
- **MongoDB Ecosystem**
  - Overview of MongoDB Atlas, Compass, Shell, and Drivers.

Section 1: Getting Started with MongoDB

1.1 Understanding MongoDB Basics

- Introduction to databases, collections, and documents.
- Key differences between SQL and NoSQL.
- JSON-like structure in MongoDB.

1.2 MongoDB CRUD Operations

- **Creating Documents:**

```
db.students.insertOne({
  name: "John Doe",
  age: 15,
  grade: "10th",
});
```

```
db.students.insertMany([
  { name: "Jane Doe", age: 16, grade: "11th" },
  { name: "Sam Smith", age: 14, grade: "9th" },
]);
```

- **Reading Documents:**

```
db.students.find();
db.students.find({ grade: "10th" });
```

- **Updating Documents:**

```
db.students.updateOne({ name: "John Doe" }, { $set: { age: 16 }
});
```

- **Deleting Documents:**

```
db.students.deleteOne({ name: "John Doe" });
```

1.3 Querying Data

- Using comparison operators: `$eq`, `$gt`, `$lt`, `$in`.
- Logical operators: `$and`, `$or`, `$not`.
- Examples:

```
db.students.find({ age: { $gt: 15 } });
db.students.find({ $and: [{ grade: "10th" }, { age: { $lt: 16 }
}] });
```

Section 2: Intermediate MongoDB

2.1 Data Modeling in MongoDB

- Understanding schema design.
- Embedded documents vs. references.

- Best practices for designing collections.

2.2 Indexing

- Importance of indexes.
- Creating indexes:

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

- Viewing indexes:

```
db.students.getIndexes();
```

- Optimizing queries with indexes.

2.3 Aggregation Framework

- Stages in an aggregation pipeline: `$match`, `$group`, `$sort`, `$project`.
- Example:

```
db.students.aggregate([
  { $match: { grade: "10th" } },
  { $group: { _id: "$grade", total: { $sum: 1 } } },
  { $sort: { total: -1 } },
]);
```

- Use cases for aggregations.

2.4 Working with Arrays

- Querying array fields:

```
db.students.find({ subjects: { $in: ["Math", "Science"] } });
```

- Updating arrays:

```
db.students.updateOne(
  { name: "Jane Doe" },
```

```
  { $push: { subjects: "English" } }
);
```

- Array operators: `$size`, `$elemMatch`.

Section 3: Advanced MongoDB

3.1 Transactions

- Overview of ACID transactions in MongoDB.
- Example of a session-based transaction:

```
const session = db.getMongo().startSession();
session.startTransaction();

try {
  const studentsCollection =
session.getDatabase("school").students;
  studentsCollection.updateOne(
    { name: "John Doe" },
    { $set: { grade: "11th" } }
  );

  session.commitTransaction();
} catch (error) {
  session.abortTransaction();
} finally {
  session.endSession();
}
```

3.2 Sharding and Replication

- Overview of sharding for horizontal scaling.
- Setting up replica sets for high availability.
- Commands to initialize a replica set:

```
rs.initiate();
rs.add("node2:27017");
rs.add("node3:27017");
```

3.3 Working with MongoDB Atlas

- Creating a cluster in MongoDB Atlas.
- Connecting your application to MongoDB Atlas.
- Monitoring and scaling with Atlas tools.

3.4 Advanced Query Optimization

- Using the `explain()` method:

```
db.students.find({ grade: "10th" }).explain("executionStats");
```

- Optimizing aggregation pipelines.
- Best practices for query performance.

Section 4: MongoDB Administration

4.1 User Management

- Creating users with roles:

```
db.createUser({
  user: "admin",
  pwd: "password",
  roles: [{ role: "readWrite", db: "school" }],
});
```

- Assigning and revoking roles.

4.2 Backup and Restore

- Using `mongodump` for backup:

```
mongodump --db school --out /backup/school
```

- Using `mongorestore` for restoration:

```
mongorestore --db school /backup/school
```

4.3 Monitoring and Performance

- Monitoring with `mongostat` and `mongotop`.
- Using Compass and Atlas for monitoring.

Section 5: Real-World Projects

5.1 Building a Blogging Platform

- Designing collections for users, posts, and comments.
- Writing queries for user activity and post management.

5.2 Inventory Management System

- Designing collections for products, categories, and stock levels.
- Using aggregation pipelines for inventory reports.

Conclusion

- Recap of MongoDB features and capabilities.
- Best practices for MongoDB development.
- Resources for advanced learning.

Appendix

- Common MongoDB commands.
- Troubleshooting common errors.