Assignment No. 1

Aim: Consider Hospital/Company database schema and implement all MongoDB methods on it.

Pre-Requisites: DBMS, MongoDB

Objective: The objective is to design and implement a robust MongoDB database schema for a Hospital or Company, allowing efficient storage and retrieval of information related to patients, employees, departments, and various interactions within the organization. The schema should support common functionalities like managing patient records, employee details, departmental structures, appointments, and more.

Outcomes: After learning this concept students will be able to,

1.The expected outcome is a well-structured MongoDB database that facilitates seamless data management within the Hospital or Company. This includes the ability to add, update, and retrieve information about patients, employees, departments, appointments, and other relevant entities.

2.The implemented schema should support scalability, data integrity, and optimal performance for a variety of queries and operations.

Theory:

In a hospital database schema, the design is crucial for effective management of patient data, appointments, medical staff, and various other aspects. Here's a theoretical overview of the key entities and their relationships in a hospital database schema:

Patients:

- Patients are individuals receiving medical care.
- Attributes include name, age, gender, address, contact details, admission date, and discharge date.
- Unique patient identifier: _id

Doctors and Medical Staff:

- Medical staff includes doctors, nurses, and other healthcare professionals.
- Attributes include name, position, contact details, and specialization for doctors.
- Unique staff identifier: _id

Departments:

- Represents different medical departments within the hospital (e.g., cardiology, pediatrics, etc.).
- Attributes include name and description.
- Unique department identifier: _id

Appointments:

- Records scheduled appointments between patients and doctors.
- Attributes include patientId, doctorId, appointment date, reason, and status.
- Unique appointment identifier: _id

Below are some commonly used MongoDB methods with examples:

1. Insert Documents:

insertOne(document) - Inserts a single document into the collection.

```
db.collection.insertOne({ name: "John Doe", age: 25, city: "New York" });
```

insertMany([document1, document2, ...]) - Inserts multiple documents into the collection.

```
db.collection.insertMany([
  { name: "Alice", age: 30, city: "San Francisco" },
  { name: "Bob", age: 28, city: "Los Angeles" }
```

]);

2. Update Documents:

updateOne(filter, update) - Updates a single document matching the filter.

db.collection.updateOne({ name: "John Doe" }, { $set: { age: 26 } });

updateMany(filter, update) - Updates multiple documents matching the filter.

db.collection.updateMany({ city: "New York" }, { $set: { city: "Chicago" } });

3. Delete Documents:

deleteOne(filter) - Deletes a single document matching the filter.

db.collection.deleteOne({ name: "Alice" });

deleteMany(filter) - Deletes multiple documents matching the filter.

db.collection.deleteMany({ age: { $lt: 30 } });

4. Query Documents:

find(filter) - Retrieves documents based on the specified filter.

db.collection.find({ age: { $gte: 25 } });

findOne(filter) - Retrieves the first document matching the filter.

db.collection.findOne({ name: "John Doe" });

5. Aggregation Framework:

aggregate(pipeline) - Performs aggregation operations on the collection data.

```
db.collection.aggregate([
  { $group: { _id: "$city", averageAge: { $avg: "$age" } } }
]);
```

6. Indexing:

createIndex(keys) - Creates an index on one or more fields.

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

getIndexes() - Retrieves the indexes on the collection.

```
db.collection.getIndexes();
```

7. Text Search:

createIndex({ "$**": "text" }) - Creates a text index for full-text search.

```
db.collection.createIndex({ "$**": "text" });
```

find({ $text: { $search: "keyword" } }) - Performs a text search.

```
db.collection.find({ $text: { $search: "MongoDB" } });
```

Also Add your implemented methods in lab here

Conclusion:

Hence, we have successfully studied Hospital/Company database schema and implement all MongoDB methods on it.

Frequently Asked Questions:

1. What is the purpose of the insertOne method in MongoDB?
2. How does the updateOne method work in MongoDB?
3. When should I use deleteOne and deleteMany in MongoDB?
4. What is the significance of the aggregate method in MongoDB?
5. What is the difference between find and findOne in MongoDB?