ASSIGNMENT – 8

AIM: Basic operations (CRUD operation) on some NoSQL databases like MongoDB, Cassandra Graph, Database (NEO4j).

THEORY:

NoSQL databases represent a departure from traditional relational databases, introducing a flexible approach to data management. These databases are adept at handling unstructured, semi-structured, and structured data, supporting diverse models such as document-oriented, key-value pairs, wide-column stores, and graph databases.

One key advantage of NoSQL databases is their horizontal scalability, allowing for the easy distribution of data across multiple servers. Unlike relational databases, NoSQL databases are schema-less, enabling the insertion of data without a predefined structure and facilitating adaptability to changing data requirements.

MongoDB is a prominent NoSQL database known for its document-oriented architecture, storing data in flexible BSON (Binary JSON) documents. This flexibility allows it to manage unstructured, semi-structured, and structured data efficiently, with each document capable of having a distinct structure. MongoDB excels in scalability, designed to horizontally scale across multiple servers, making it suitable for applications with extensive data and high traffic.

With a robust query language and efficient indexing mechanisms, MongoDB supports complex queries, indexing on any document field for optimized data retrieval. Its aggregation framework enables data transformation and computation on the server side, enhancing the ability to process and analyze data directly within the database.

IMPLEMENTATION:

CREATE OPERATION

1) Launching Mongo server

```
PS C:\Users\aayus> mongosh
Current Mongosh Log ID: 65172a3eccf8174bf1054bcc
Connecting to:          mongodb://127.0.0.1:27017/?directConnection=true&ser
verSelectionTimeoutMS=2000&appName=mongosh+2.0.1
Using MongoDB:          7.0.1
Using Mongosh:          2.0.1

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

------
   The server generated these startup warnings when booting
   2023-09-29T17:10:56.378+05:30: Access control is not enabled for the data
base. Read and write access to data and configuration is unrestricted
------
```

2) Show current databases.

```
test> show dbs
admin   40.00 KiB
config  72.00 KiB
local   88.00 KiB
```

3) Create and select database.

```
aayush> use mit
switched to db mit
```

4) Create collection (collection = table)

```
mit> db.createCollection("students")
{ ok: 1 }
```

5) Insert one record (document)

```
mit> db.students.insertOne({name: "Aayush", age: 20, course:"CSE"})
{
  acknowledged: true,
  insertedId: ObjectId("65172b5accf8174bf1054bcd")
}
```

6) Insert multiple records (documents)

```
mit> db.students.insertMany([{name: "Chirag", age: 20, class : "TY-CSF2"}, {
name: "Anant", age: 20, specialization : "Cyber Security"}])
{
  acknowledged: true,
  insertedIds: {
    '0': ObjectId("65172b80ccf8174bf1054bce"),
    '1': ObjectId("65172b80ccf8174bf1054bcf")
  }
}
```

READ OPERATION

7) Print all entries

```
mit> db.students.find()
[
  {
    _id: ObjectId("65172b5accf8174bf1054bcd"),
    name: 'Aayush',
    age: 20,
    course: 'CSE'
  },
  {
    _id: ObjectId("65172b80ccf8174bf1054bce"),
    name: 'Chirag',
    age: 20,
    class: 'TY-CSF2'
  },
  {
    _id: ObjectId("65172b80ccf8174bf1054bcf"),
    name: 'Anant',
    age: 20,
    specialization: 'Cyber Security'
  }
]
```

#UPDATE OPERATION

8) Updating values

```
mit> db.students.updateOne({'name':"Anant"},{$set:{specialization:"Cyber Sec
urity and Forensic"}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
```

9) Updated values

```
mit> db.students.find()
[
  {
    _id: ObjectId("65172b5accf8174bf1054bcd"),
    name: 'Aayush',
    age: 20,
    course: 'CSE'
  },
  {
    _id: ObjectId("65172b80ccf8174bf1054bce"),
    name: 'Chirag',
    age: 20,
    class: 'TY-CSF2'
  },
  {
    _id: ObjectId("65172b80ccf8174bf1054bcf"),
    name: 'Anant',
    age: 20,
    specialization: 'Cyber Security and Forensic'
  }
]
```

DELETE OPERATION

10) Delete single document

```
mit> db.students.deleteOne({'name':'Chirag'})
{ acknowledged: true, deletedCount: 1 }
```

11) Print data post delete

```
mit> db.students.find()
[
  {
    _id: ObjectId("65172b5accf8174bf1054bcd"),
    name: 'Aayush',
    age: 20,
    course: 'CSE'
  },
  {
    _id: ObjectId("65172b80ccf8174bf1054bcf"),
    name: 'Anant',
    age: 20,
    specialization: 'Cyber Security and Forensic'
  }
]
```

12) Delete collection

```
mit> db.students.drop()
true
```

13) Drop database

```
mit> db.dropDatabase()
{ ok: 1, dropped: 'mit' }
```

CONCLUSION: Hence successfully implemented the CRUD operations in a NoSQL database – MongoDB.