॥वसुधैव कुटुम्बकम्॥

FULL STACK DEVLOPMENT

ASSIGNMENT - 8

Prepared by:

Apoorv Gupta

21070126018 (AIML A1)

In this assignment, learn about mongodb compass and mongodb shell to utilise poteintial of no sql databases. Learn commands of mongosh and implement the task provided as per reference

Description

This experiment provides a comprehensive overview of working with a NoSQL database, specifically MongoDB, to create a database, import data from a JSON file, and execute various NoSQL queries.

The key steps involved in this experiment are:

1. **Install MongoDB and set up the necessary environment.**
2. **Create a new database named "AIML" in MongoDB.**
3. **Create a collection named "employee" within the AIML database.**
4. **Import employee records from a JSON file into the employee collection.**
5. **Perform various NoSQL queries on the employee data, including:**
   - **Inserting a single document**
   - **Inserting multiple documents**
   - **Updating a single document**
   - **Updating multiple documents**
   - **Retrieving employees by ID**
   - **Retrieving employees assigned to a specific project**
   - **Finding employees who have worked more than 30 hours**
   - **Finding employees older than 40 using the $gt operator**
   - **Sorting the employee data by age and hours worked**
   - **Performing complex queries using logical operators like $and, $or, and $not**
   - **Demonstrating the use of indexes to optimize queries**
   - **The experiment provides a step-by-step guide on how to set up the MongoDB environment, import data, and execute various NoSQL queries. It covers a wide range of query types, from simple to more complex, to help the user understand the capabilities of MongoDB and NoSQL databases.**

By completing this experiment, the user will gain practical experience in working with a NoSQL database, performing data manipulation and querying, and understanding the advantages of using a document-oriented database like MongoDB for certain types of applications.

1. Create Database: Create a MongoDB database named AIML.

```
test> use AIML
switched to db AIML
AIML>
```

2. Create Collection: Create a collection named employee within the AIML database.

```
AIML> db.createCollection("employee")
{ ok: 1 }
```

3. Import Employee Records: Import employee records from a JSON file into the employee collection. Save the following data in a json file and use it for import in Mongodb.

```
C:\Users\erapo\Desktop\Assignment_SEM - 6\FSD>mongoimport --db AIML --collection employee --file "C:\Users\erapo\Desk
top\Assignment_SEM - 6\FSD\employee_data.json" --jsonArray
```

```
AIML> db.employee.countDocuments()
23
```

4. insertOne: Inserts a single document into the collection.

```
AIML> db.employee.insertOne({ "Id": 21, "Name": "John Doe", "Project_id": 2, "Hrs_worked": 35 })
{
  acknowledged: true,
  insertedId: ObjectId("6618302d09d0b432f0a13c16")
}
```

5. insertMany: Inserts multiple documents into the collection.

```
AIML> db.employee.insertMany([{ "Id": 22, "Name": "Jane Smith", "Project_id": 1, "Hrs_worked": 28 }, { "Id": 23, "Name": "Alice Johnson", "Project_id": 3, "Hrs_worked": 42 }])
{
  acknowledged: true,
  insertedIds: {
    '0': ObjectId("661830d809d0b432f0a13c17"),
    '1': ObjectId("661830d809d0b432f0a13c18")
  }
}
```

6. updateOne: Updates a single document that matches the filter.

```
AIML> db.employee.updateOne({ "Id": 21 }, { $set: { "Hrs_worked": 40 } } )
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
```

7. updateMany: Updates multiple documents that match the filter.

```
AIML> db.employee.updateMany({ "Hrs_worked": { $gt: 30 } }, { $set: { "Overtime": true } } )
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 2,
  modifiedCount: 2,
  upsertedCount: 0
}
```

8. find an employee by their ID.

```
AIML> db.employee.find({"Id" : 21})
[
  {
    _id: ObjectId("6618302d09d0b432f0a13c16"),
    Id: 21,
    Name: 'John Doe',
    Project_id: 2,
    Hrs_worked: 40,
    Overtime: true
  }
]
```

9. How would you retrieve all employees who are assigned to a specific project ID?

```
AIML> db.employee.find({ Project_id: 2 })
[
  {
    _id: ObjectId("6618302d09d0b432f0a13c16"),
    Id: 21,
    Name: 'John Doe',
    Project_id: 2,
    Hrs_worked: 40,
    Overtime: true
  }
]
```

10. Write a query to find employees who have worked more than 30 hours.

```
AIML> db.employee.find({ Hrs_worked: { $gt: 30 } })
[
  {
    _id: ObjectId("6618302d09d0b432f0a13c16"),
    Id: 21,
    Name: 'John Doe',
    Project_id: 2,
    Hrs_worked: 40,
    Overtime: true
  },
  {
    _id: ObjectId("661830d809d0b432f0a13c18"),
    Id: 23,
    Name: 'Alice Johnson',
    Project_id: 3,
    Hrs_worked: 42,
    Overtime: true
  }
]
```

11. Can you demonstrate how to use the $gt operator to find employees who are older than 40?

```
AIML> db.employee.find({ Age: { $gt: 40 } })
[
  {
    _id: ObjectId('66197f6b02ded03bfd16c9b5'),
    Id: 1,
    Name: 'Charlie Moore',
    Age: 44,
    Gender: 'Male',
    Project_id: 4,
    Hrs_worked: 12
  },
  {
    _id: ObjectId('66197f6b02ded03bfd16c9b9'),
    Id: 5,
    Name: 'Bob Davis',
    Age: 41,
    Gender: 'Female',
    Project_id: 4,
    Hrs_worked: 16
  },
  {
    _id: ObjectId('66197f6b02ded03bfd16c9ba'),
    Id: 6,
    Name: 'Hannah Davis',
    Age: 57,
    Gender: 'Male',
    Project_id: 2,
    Hrs_worked: 28
  },
```

12. Explain the purpose of sorting in MongoDB queries.

Sorting in MongoDB queries is the best way to arrange the retrieved documents in a specific order based on your chosen criteria. This allows you to efficiently present data in a user-friendly way, such as displaying products by price (ascending or descending) or chronologically ordering blog posts. By utilising sort options, you can significantly enhance the organisation and accessibility of your query results.

13. Sort the Employee table based on Age in Ascending order and display.

```
AIML> db.employee.find().sort({ Age: 1 })
  {
    _id: ObjectId('66197f6b02ded03bfd16c9c2'),
    Id: 14,
    Name: 'Isaac Wilson',
    Age: 21,
    Gender: 'Female',
    Project_id: 3,
    Hrs_worked: 32
  },
  {
    _id: ObjectId('66197f6b02ded03bfd16c9c5'),
    Id: 17,
    Name: 'David Miller',
    Age: 25,
    Gender: 'Female',
    Project_id: 4,
    Hrs_worked: 32
  },
  {
    _id: ObjectId('66197f6b02ded03bfd16c9c4'),
    Id: 16,
    Name: 'Alice Moore',
    Age: 29,
    Gender: 'Male',
    Project_id: 1,
    Hrs_worked: 39
  },
```

14. Sort the Employee table based on Hrs_worked in Descending order and display.

```
AIML> db.employee.find().sort({ Hrs_worked: -1 })
[
  {
    _id: ObjectId('661830d809d0b432f0a13c18'),
    Id: 23,
    Name: 'Alice Johnson',
    Project_id: 3,
    Hrs_worked: 42,
    Overtime: true
  },
  {
    _id: ObjectId('6618302d09d0b432f0a13c16'),
    Id: 21,
    Name: 'John Doe',
    Project_id: 2,
    Hrs_worked: 40,
    Overtime: true
  },
  {
    _id: ObjectId('66197f6b02ded03bfd16c9c4'),
    Id: 16,
    Name: 'Alice Moore',
    Age: 29,
    Gender: 'Male',
    Project_id: 1,
    Hrs_worked: 39
  },
  {
    _id: ObjectId('66197f6b02ded03bfd16c9b7'),
    Id: 3,
    Name: 'Hannah Brown',
    Age: 33,
    Gender: 'Female',
    Project_id: 2,
    Hrs_worked: 36
  },
```

15. Find Employee whose age is greater then 30 and Has_Worked greater then 20.

```
AIML> db.employee.find({ Age: { $gt: 30 }, Hrs_worked: { $gt: 20 } })
[
  {
    _id: ObjectId('66197f6b02ded03bfd16c9b7'),
    Id: 3,
    Name: 'Hannah Brown',
    Age: 33,
    Gender: 'Female',
    Project_id: 2,
    Hrs_worked: 36
  },
  {
    _id: ObjectId('66197f6b02ded03bfd16c9b8'),
    Id: 4,
    Name: 'Hannah Davis',
    Age: 31,
    Gender: 'Female',
    Project_id: 2,
    Hrs_worked: 22
  },
  {
    _id: ObjectId('66197f6b02ded03bfd16c9ba'),
    Id: 6,
    Name: 'Hannah Davis',
    Age: 57,
    Gender: 'Male',
    Project_id: 2,
    Hrs_worked: 28
  },
```

16. Find Employee whose Gender is Male or Has_Worked greater then 25.

```
AIML> db.employee.find({ $or: [{ Gender: "Male" }, { Hrs_worked: { $gt: 25 } }] })
[
  {
    _id: ObjectId('6618302d09d0b432f0a13c16'),
    Id: 21,
    Name: 'John Doe',
    Project_id: 2,
    Hrs_worked: 40,
    Overtime: true
  },
  {
    _id: ObjectId('661830d809d0b432f0a13c17'),
    Id: 22,
    Name: 'Jane Smith',
    Project_id: 1,
    Hrs_worked: 28
  },
  {
    _id: ObjectId('661830d809d0b432f0a13c18'),
    Id: 23,
    Name: 'Alice Johnson',
    Project_id: 3,
    Hrs_worked: 42,
    Overtime: true
  },
```

17. Find Employee whose Project_id is not 3.

```
AIML> db.employee.find({ Project_id: { $ne: 3 } })
[
  {
    _id: ObjectId('6618302d09d0b432f0a13c16'),
    Id: 21,
    Name: 'John Doe',
    Project_id: 2,
    Hrs_worked: 40,
    Overtime: true
  },
  {
    _id: ObjectId('661830d809d0b432f0a13c17'),
    Id: 22,
    Name: 'Jane Smith',
    Project_id: 1,
    Hrs_worked: 28
  },
  {
    _id: ObjectId('66197f6b02ded03bfd16c9b5'),
    Id: 1,
    Name: 'Charlie Moore',
    Age: 44,
    Gender: 'Male',
    Project_id: 4,
    Hrs_worked: 12
  },
```

18. Write a MongoDB query to find all employees who are between the ages of 25 and 35.

```
AIML> db.employee.find({ Age: { $gte: 25, $lte: 35 } })
[
  {
    _id: ObjectId('66197f6b02ded03bfd16c9b7'),
    Id: 3,
    Name: 'Hannah Brown',
    Age: 33,
    Gender: 'Female',
    Project_id: 2,
    Hrs_worked: 36
  },
  {
    _id: ObjectId('66197f6b02ded03bfd16c9b8'),
    Id: 4,
    Name: 'Hannah Davis',
    Age: 31,
    Gender: 'Female',
    Project_id: 2,
    Hrs_worked: 22
  },
```

19. How would you retrieve employees who have worked between 20 and 30 hours?

```
AIML> db.employee.find({Hrs_worked : {$gte: 20, $lte : 30}})
[
  {
    _id: ObjectId('661830d809d0b432f0a13c17'),
    Id: 22,
    Name: 'Jane Smith',
    Project_id: 1,
    Hrs_worked: 28
  },
  {
    _id: ObjectId('66197f6b02ded03bfd16c9b8'),
    Id: 4,
    Name: 'Hannah Davis',
    Age: 31,
    Gender: 'Female',
    Project_id: 2,
    Hrs_worked: 22
  },
  {
    _id: ObjectId('66197f6b02ded03bfd16c9ba'),
    Id: 6,
    Name: 'Hannah Davis',
    Age: 57,
    Gender: 'Male',
    Project_id: 2,
    Hrs_worked: 28
  },
```

20. Write a query to find employees who are either working on Project 1 or Project 2.

```
AIML> db.employee.find({ Project_id: { $in: [1, 2] } })
[
  {
    _id: ObjectId('6618302d09d0b432f0a13c16'),
    Id: 21,
    Name: 'John Doe',
    Project_id: 2,
    Hrs_worked: 40,
    Overtime: true
  },
  {
    _id: ObjectId('661830d809d0b432f0a13c17'),
    Id: 22,
    Name: 'Jane Smith',
    Project_id: 1,
    Hrs_worked: 28
  },
```

About Me

Hi, I am Apoorv
I study at symbiosis pune, branch as AIML.

PRN: 21070126018
Batch: 2021-2025
Division: AIML-A1